МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Дніпровський національний університет залізничного транспорту імені академіка В. Лазаряна

Кафедра «Електронні обчислювальні машини»

«ДО ЗАХИСТУ»

Завідувач кафедри

_____ _____
(підпис) (ПІБ)

«____» _____ 20____р.

ДИПЛОМНА РОБОТА

на здобуття освітнього ступеня «магістр»

Галузь знань 12 Інформаційні технології
(шифр) (назва)

Спеціальність 123 Комп'ютерна інженерія
(код) (повна назва)

Тема Реалізація і дослідження RISC-процесора з використанням ПЛІС

Theme Implementation and research of a RISC processor using FPGA

Керівник дипломного проекту	_____ (посада)	_____ (підпис)	Шаповалов В. О. (ПІБ)
Консультант розділу з БЖД	_____ (посада)	_____ (підпис)	Музикін М. І. (ПІБ)
Нормоконтролер	_____ (посада)	_____ (підпис)	Шаповалов В. О. (ПІБ)
Студент групи	КС1921 (група)	_____ (підпис)	Новиков А. О. (ПІБ)
Student	Novykov Artem (family name)		

Дніпро
2020

Дніпровський національний університет залізничного транспорту

імені академіка В. Лазаряна

Факультет ___Комп'ютерних технологій і систем__ кафедра _____ЕОМ_____
Спеціальність __Комп'ютерна інженерія_____

«ЗАТВЕРДЖУЮ»
Завідувач кафедри

_____
(підпис)
«___»_____20__ р.

ЗАВДАННЯ

до дипломної роботи на здобуття освітнього ступеня _магістр_____
(освітнього ступеня)

студента групи _КС1921__ ____Новикова Артема Олександровича_____
(номер групи) (ПІБ)

1 Тема дипломної роботи ___Реалізація і дослідження RISC-процесора з використанням ПЛІС_____

затверджена наказом по університету від «_17_»__01__2020_ р. № _57__.

2 Термін подання студентом закінченої роботи_11.12.2020 р._____

3 Вихідні дані до дипломної роботи_____
Необхідно реалізувати багатоядерний процесор з підтримкою таких груп операцій: арифметичні, логічні, операції роботи з пам'яттю та портом вводу-виводу, умовного і безумовного переходу та матричні операції

4 Зміст пояснювальної записки (перелік питань до розробки) _____
Вступ та постановка завдання
1. Процесори RISC архітектури з багатоядерністю та підтримкою матричних операцій
2. Вибір характеристик процесору й системи команд
3. Побудова функціональної схеми процесору
4. Реалізація на VHDL та дослідження процесору
5. Охорона праці та безпека в надзвичайних ситуаціях
Висновки

5 Перелік креслень (демонстраційного матеріалу) _____
Машинний цикл процесору
Види багатопроцесорних систем
Функціональна схема процесору
Система команд процесору
Часові діаграми роботи процесору

6 Розділи та консультанти

Розділ	Консультант	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці в надзвичайних ситуаціях	Музикін М. І.		

КАЛЕНДАРНИЙ ПЛАН

Назва розділу	Термін виконання	Обсяг розділу, %
Вступ та постановка завдання	25.01.2020 р.	2%
Процесори RISC архітектури з багатоядерністю та підтримкою матричних операцій	14.02.2020 р.	8%
Вибір характеристик процесору й системи команд	23.03.2020 р.	15%
Побудова функціональної схеми процесору	29.04.2020 р.	15%
Реалізація на VHDL та дослідження процесору	20.10.2020 р.	40%
Охорона праці та безпека в надзвичайних ситуаціях	14.11.2020 р.	10%
Висновки	27.11.2020 р.	5%
Оформлення диплому, підготовка демонстраційних матеріалів та доповіді	11.12.2020 р.	5%

Дата видачі завдання: «___»_____ 20___ р.

Керівник дипломної роботи _____ _Шаповалов В. О.__
(підпис) (ПІБ)

Завдання прийняв до виконання _____ _Новиков А. О._____
(підпис) (ПІБ)

РЕФЕРАТ

Дипломна робота складається з 125 сторінок, 5 частин, 56 рисунків, 2 таблиць, 10 додатків й 18 літературних джерел.

У дипломній роботі розроблюється RISC-процесор з чотирма обчислювальними ядрами та підтримкою матричних операцій.

Метою даної роботи є дослідження існуючих рішень та реалізація власного проекту процесору на мові VHDL, й спрощення програмування процесору шляхом створення пакету з переліком констант – мнемонічними кодами операцій, назвами регістрів тощо.

У ході виконання роботи були виконані наступні кроки: досліджені існуючі реалізації учбових процесорів; визначені характеристики процесору (тип і розрядність операндів, шин адрес і даних); складена система команд процесору; побудована функціональна схема процесору; реалізований проект процесору на мові VHDL; наведені відомості про охорону праці та безпеку життєдіяльності.

Розроблений процесор можна застосовувати як будь-який універсальний обчислювач – для наукових чи економічних обчислень, автоматизації виробничих процесів, обміну інформацією з периферійними пристроями через порт вводу-виводу тощо.

Оскільки робота з процесором відбувалася шляхом його моделювання, то використання (і, можливо, покупка) ПЛІС не знадобилася. При роботі над проектом використовувався власний персональний комп’ютер, та САПР фірми Xilinx.

Ключові слова: ПРОЦЕСОР, RISC, ПЛІС, БАГАТОЯДЕРНІСТЬ, МАТРИЧНІ ОПЕРАЦІЇ, VHDL, ПРИСКОРЕННЯ ОБЧИСЛЕНЬ

ABSTRACT

Graduate work consists of 125 pages, 5 parts, 56 figures, 2 tables, 10 appendices and 18 references.

Graduate work develops a RISC-processor with four computing cores and support for matrix operations.

The purpose of this work is to study existing solutions and implement my own processor design in VHDL, and simplify CPU programming by creating a package with a list of constants - mnemonic operation codes, register names and more.

During the work the following steps were performed: the existing implementations of educational processors were studied; certain characteristics of the processor (type and bit operands, address and data buses); developed system of processor instructions; the functional scheme of the processor is built; implemented VHDL processor project; information on labor protection and life safety is given.

The developed processor can be used as any universal computer - for scientific or economic calculations, automation of production processes, exchange of information with peripheral devices through the I / O port and more.

Because the CPU was modeled, the use (and possibly purchase) of a FPGA was not required. The project used its my personal computer and Xilinx CAD.

Keywords: PROCESSOR, RISC, FPGA, MULTICORE, MATRIX OPERATIONS, VHDL, CALCULATION ACCELERATION

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

АЛП – арифметико-логічний пристрій;

КА – кінцевий автомат;

ОП – оперативна пам’ять;

ПВВ – порт вводу-виводу;

ПП – постійна пам’ять;

УГП – умовне графічне позначення;

ВСТУП ТА ПОСТАНОВКА ЗАВДАННЯ

У сучасному світі процесори використовуються повсюди – від мікроконтролерів, що керують технікою, смартфонів та персональних комп’ютерів до суперкомп’ютерів, що проводять величезні об’єми обчислень. В залежності від області застосування, процесори можуть мати різну обчислювальну потужність, підтримувати різний об’єм і тип постійної та оперативної пам’яті, сопроцесори для прискорення спеціалізованих обчислень тощо.

Існують учбові проекти процесорів від розробників зі всього світу. Деякі з них досить прості, деякі мають конвеєрну систему виконання команд, підтримку апаратних переривань, кеш-пам’ять та інші покращення.

Метою дипломної роботи є ознайомлення з архітектурою учбових процесорів, визначення характеристик та реалізація власного RISC-процесору. У якості особливостей процесор буде мати кілька обчислювальних ядер (що дає можливість одночасного виконання декількох програм), та мати прискорення виконання однотипних операцій над множиною чисел.

Через розповсюдженість використання процесорів у багатьох сферах, а також через переваги прискореної обробки масивів чисел тема дипломної роботи є актуальною. До того ж більшість учбових проектів не передбачають апаратної можливості одночасного виконання кількох незалежних програм, хоча це збільшує сумарну обчислювальну потужність процесору та спрощує роботу програміста у деяких задачах.

Реалізація та дослідження процесору проводились в САПР фірми Xilinx, яка має наступні можливості: синтез схем для ПЛІС на основі VHDL-коду, моделювання розроблених пристроїв, відлагодження проекту, генерація bitstream-файлу для використання розробленого пристрою на ПЛІС тощо.

1 ПРОЦЕСОРИ RISC АРХІТЕКТУРИ З БАГАТОЯДЕРНІСТЮ ТА ПІДТРИМКОЮ МАТРИЧНИХ ОПЕРАЦІЙ

1.1 Функціонування універсальних цифрових обчислювальних пристроїв

Сучасні комп'ютери є універсальними цифровими обчислювальними пристроями, і мають спільні базові принципи роботи.

Кожен комп'ютер має пам'ять (оперативну та постійну), обчислювальні блоки (арифметично-логічний пристрій та інші) й керуючий пристрій.

Постійна пам'ять (ПП) є енергонезалежною, й зберігає програми для обчислення та необхідні вхідні дані.

Оперативна пам'ять (ОП) є енергозалежною, але набагато швидша за постійну пам'ять. У ній зберігаються дані, необхідні для поточних обчислень.

Обчислювальні блоки виконують цифрові обчислення. Універсальні комп'ютери завжди мають арифметико-логічний пристрій для виконання базових арифметичних та логічних операцій. Також бувають пристрої для обчислень з плаваючою комою (підтримка нецілих чисел), цифрові сигнальні процесори (для обробки аналогових сигналів) та інші спеціалізовані блоки.

Керуючий пристрій організує обчислювальний процес, керуючи іншими блоками системи.

Робота процесору – це повторюване виконання так званого «машинного циклу». Машинний цикл складається з декількох стадій: вибірка команди з пам'яті, дешифровка команди, вибірка операндів, операція з операндами, запис результатів [1, 2].

За організацією роботи з пам'яттю розрізняють Гарвардську та Прістонську архітектуру (архітектуру фон Неймана) [3].

Процесори Гарвардської архітектури мають окремі комунікації для постійної і оперативної пам'яті (шина адреси, шина даних, шина управління). Це дозволяє одночасно працювати з ОП і ПП, але дає більш складну апаратну структуру (майже в два рази більші апаратні затрати на шини).

Процесори Прістонської архітектури (фон Неймана) мають спільні комунікації до ОП і ПП. Одночасно працювати з ними неможливо, але апаратна реалізація даної

схеми більш проста. Неможливість одночасної роботи з ОП і ПП є вузьким місцем Прістонської архітектури.

Існують модифікації Гарвардської архітектури: в процесорі шини адрес і даних оперативної і постійної пам’яті окремі, але при роботі із зовнішніми інтерфейсами вони об’єднані. У мікроконтролерах застосовується ще більш спрощена архітектура: оперативна і постійна пам’ять розміщені у одному адресному просторі, і тип пам’яті визначається по старшій частині адреси.

1.2 Принципи RISC архітектури

Всі цифрові процесори загального призначення розділяють на CISC і RISC-процесори. CISC (Complex Instruction Set Computer) – комп’ютер з комплексною (складною) системою команд. RISC (Reduced Instruction Set Computer) – комп’ютер з системою команд зменшеної складності.

Архітектура RISC-процесорів заснована на тому, що вони мають простий, можливо навіть надлишковий формат команд. Це забезпечує нескладну й, відповідно, швидку дешифрацію команд й адрес, можливість вибірки і дешифровки декількох команд одночасно. [1]

Принципи побудови RISC-архітектури [2]:

Одна команда за один такт

Найпростіша команда повинна виконуватися за один такт. Цей принцип виглядає суперечливим з організацією машинного циклу (він складається з декількох стадій), але наступний принцип (конвеєризації) дозволяє досягти виконання однієї операції за один такт.

Конвеєризація

Виконання кожної команди проходить декілька стадій (вибірка команди, дешифровка команди, вибірка операндів, операція з операндами, запис результатів). Якщо сусідні команди не залежать одна від одної, то ці стадії слід виконувати у конвеєрі команд – одну стадію за такт.

Регістрова пам’ять

Операції над операндами виконуються у регістровій пам’яті. Компілятор так складає розклад операцій, щоб проміжні результати довше затримувались у регістрах

і не переписувалися до ОП (оперативної пам’яті), щоб мінімізувати кількість обмінів між регістрами та ОП.

Простий формат команд

Простий, хоч навіть і надлишковий формат команд, включаючи код операції, поля адрес, безпосередніх операндів, забезпечує нескладну і, відповідно, швидку дешифровку команд і адрес. За рахунок цього мінімізуються як час тактового інтервалу, так і апаратні затрати на схему керування. Всі команди у RISC-процесорах мають однакову довжину й мінімум форматів команд.

Оскільки у RISC процесорів довжина машинної команди постійна, то етап вибірки операндів не залежить від результатів дешифровки команди, і може виконуватись під час вибірки команди.

1.3 Особливості багатоядерних систем

При побудові багатопроцесорних систем є декілька особливостей.

По-перше, оперативна пам’ять у кожного процесору може бути власна (рисунок 1.1), або загальна пам’ять для всіх процесорів (рисунок 1.2) [4]. Це залежить від призначення системи – для спеціалізованих складних обчислень (наприклад наукових) у кожного процесору є власна оперативна пам’ять, такі системи використовуються в суперкомп’ютерах. У користувацьких пристроях (ПК, смартфони) декілька процесорів (процесорних ядер) використовують одну спільну пам’ять, отримуючи переваги багатопроцесорності (виконується більше інструкцій на одиницю часу).

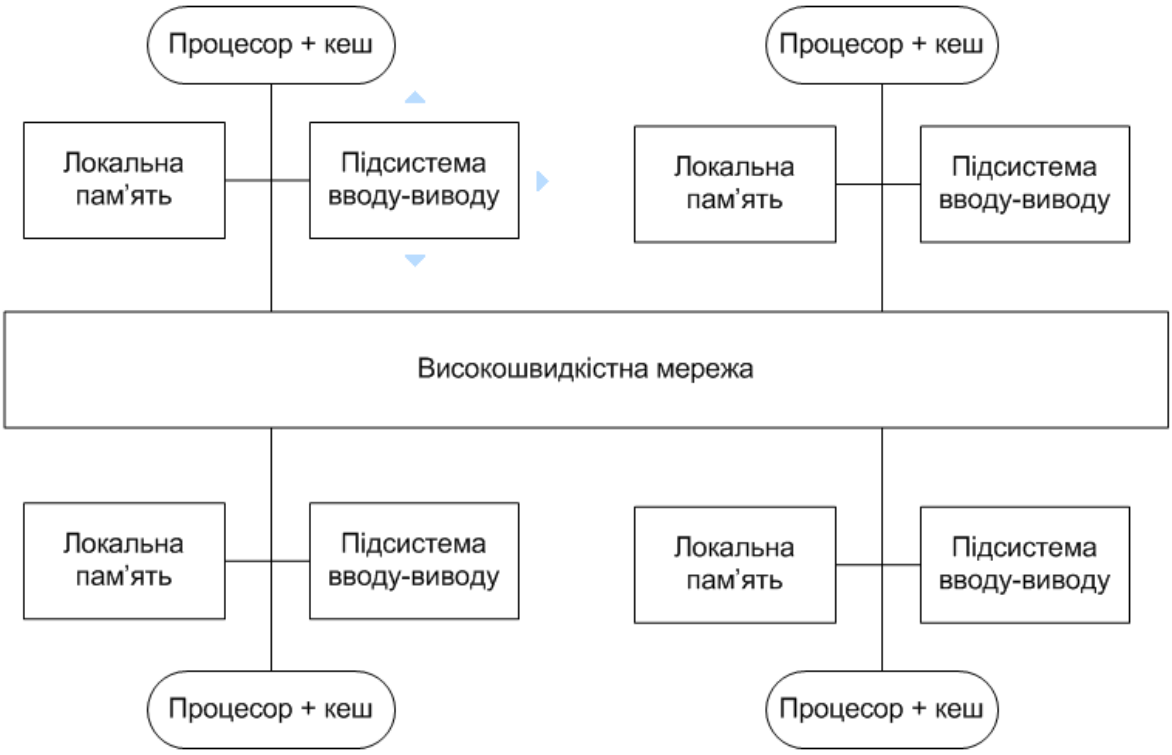

Рисунок 1.1 – Типова архітектура машини із розподіленою пам'яттю

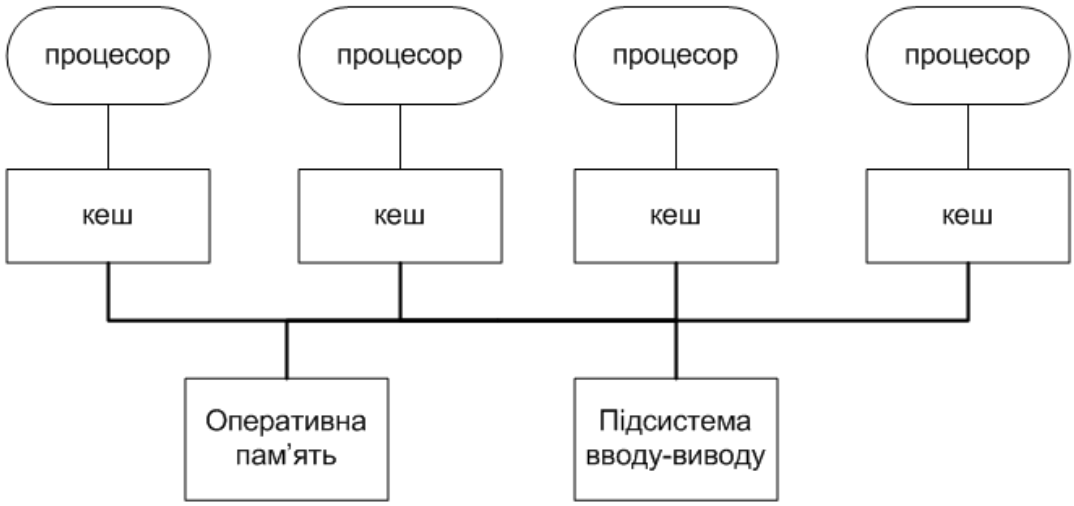

Рисунок 1.2 – Типова архітектура мультипроцесорної системи із загальною пам'яттю

У кожного процесорного ядра є власний кеш, для скорочення часу доступу до ОП. При цьому виникає проблема когерентності – якщо декілька ядер працюють з ділянкою пам'яті, що є у їх кешах, може виникнути ситуація коли інформація в комірці змінена в одному кеші, і залишилася без змін в іншому (рисунок 1.3).

Оскільки це може привести до помилок, використовуються додаткові механізми для когерентності (однорідності) кеша.

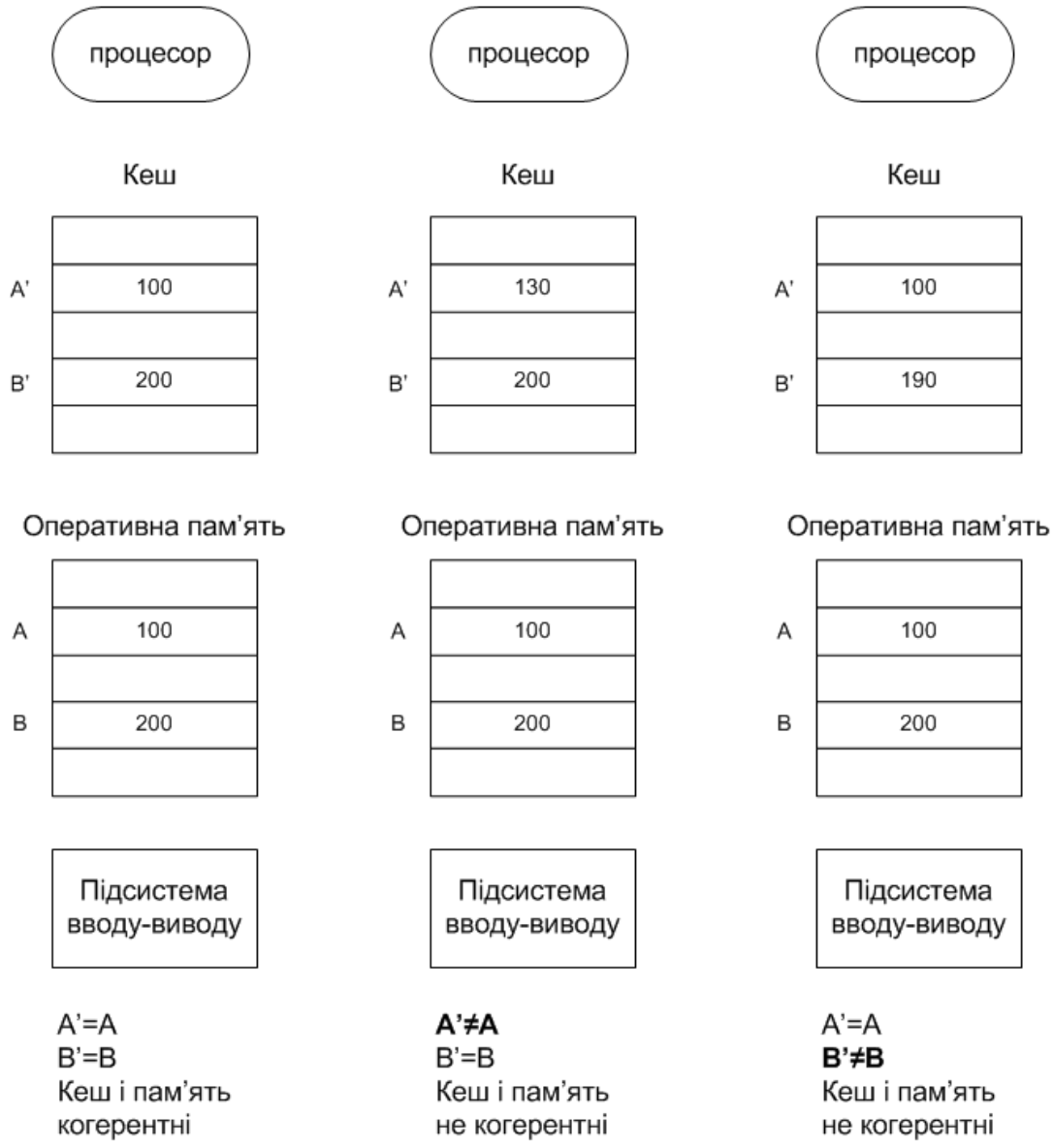

Рисунок 1.3 – Ілюстрація проблеми когерентності кеш-пам'яті

1.4 Операції над матрицями

У якості особливості, наряду з багатоядерністю буде реалізована можливість матричних обчислень на проектованому мікропроцесорі, тому у цьому розділі будуть розглянуті деякі матричні операції. [5]

Сума двох матриць A і B – матриця, що має такі ж розміри як вихідні матриці, і кожен елемент якої рівний сумі відповідних елементів вихідних матриць.

Різниця двох матриць A і B – матриця, що має такі ж розміри як вихідні матриці, і кожен елемент якої рівний різниці відповідних елементів вихідних матриць. Положення матриць відносно знаку «мінус» має значення, як і при звичайному відніманню.

Добуток матриці і числа – матриця, що має розмір вихідної матриці, і кожен елемент якої є добутком відповідного елементу вихідної матриці і числа, на яку множиться матриця.

Добутком матриці $Am \times n = (a_{ij})$ на матрицю $Bn \times k = (b_{ij})$ називається матриця $Cm \times k = (c_{ij})$, для якої кожен елемент c_{ij} рівний сумі добутків відповідних елементів і-ої строки матриці A на елементи j-го стовпця матриці B.

Для обчислення суми і різниці матриць, а також для будь-яких попарних операцій буде реалізована машинна команда, яка: приймає у якості аргументів інформацію про діапазони адрес регістрів вхідних і вихідних даних, а також код функції арифметично-логічного пристрою (АЛП). Над парою аргументів із вхідних матриць буде виконана операція, задана кодом функції АЛП, і записана у відповідне місце третьої матриці.

Добуток двох матриць пропонується реалізувати програмно при необхідності.

2 ВИБІР ХАРАКТЕРИСТИК ПРОЦЕСОРУ Й СИСТЕМИ КОМАНД

2.1 Основні характеристики процесору

Процесор має декілька основних характеристик та, в залежності від його особливостей, додаткові, що є нетиповими.

До основних характеристик належать: розрядність операндів, розрядність регістрів, розрядність адрес, розрядність шин даних.

Розрядність операндів впливає на точність обчислень: чим більше розрядність, тим більший є доступний діапазон для цілих чисел, та діапазон і точність для чисел із плаваючою комою (дробів).

Розрядність регістрів завжди відповідає розрядності операндів, бо вони призначені для їх зберігання.

Розрядність адреси впливає на максимальний об’єм пам’яті, що можна адресувати. Мінімальною адресованою одиницею пам’яті у розроблюваному процесору є одне слово. Розрядність слова постійної пам’яті (ПП) – 21 біт, розрядність слова оперативної пам’яті (ОП) – 32 біти. Кількість доступної пам’яті обчислюється як 2^n слів, де n – розрядність адреси.

Були обрані такі характеристики процесору:

Розрядність операндів (і регістрів): 32 біт;

Розрядність адреси: 32 біт;

Кількість ядер процесору: 4.

При реалізації процесорів у вигляді реальних пристроїв розрядність шин даних визначають з практичних міркувань – щоб розрядність була достатньою, але не надмірною (не потребувала забагато апаратних ресурсів).

Розрядність шини даних постійної пам’яті 21 біти (по довжині машинної команди), для оперативної пам’яті – 32 біти.

Адресний простір обчислювальної системи поділений на: область ПП, область ОП, та порт вводу-виводу PORT0. Для ПП два старші біти адреси – «00», для ОП – «01». Для PORT0 два старші біти – «10», всі інші біти – нульові. Такий розподіл дозволяє за потребою додавати нові пристрої до системи с мінімальними змінами у проекті – до керуючої шини і шини даних PORT0 можна додати шину адреси,

приєднати до цих шин нові пристрої, і генерувати дозволяючі сигнали до пристроїв аналізуючи шину адреси.

При використанні команд переходу вказується адреса ПП без старшої частини.

Оскільки для реалізації на ПЛІС по 2^{30} машинних слів ПП і ОП недостатньо ресурсів, то при моделюванні їх об’єми знижені до 2^{16} слів.

2.2 Система команд процесору

Команди процесору поділяються на кілька типів: обчислювальні операції, робота з пам’яттю та портом вводу-виводу, операції переходу.

Обчислювальні операції використовують АЛП для арифметичних і логічних обчислень.

Операції роботи з пам’яттю служать для обміну інформацією між регістрами процесору, ОП і портами вводу-виводу.

Операції переходу керують процесом обчислень – змінюють адресу для виконання програми в залежності від деяких умов чи безумовно.

Для побудови системи команд процесору, а саме їх найменування та функціоналу використовувалися джерела [6, 7].

Регістр PC – Program Counter, вказує на адресу у постійній пам’яті що зараз виконується.

R0 – регістр, всі біти якого мають нульове значення; захищений від запису.

R1 – регістр, всі біти якого мають одиничне значення; захищений від запису.

R2 – регістр прапорів.

R3 – вказівник на вершину стеку.

R4 – зберігає старшу частину результату при операції множення

R5 – регістр для отримання безпосередніх значень від програми.

R6 – R9 – регістри для передачі аргументів машинній операції MATR.

Регістри R5-R255 є базовими регістрами, деякі з них мають додаткове призначення.

Структура регістру прапорів зображена на рисунку 2.1.

7	6	5	4	3	2	1	0
			I	O	C	N	Z

Рисунок 2.1 – Структура регістру прапорів

Z – прапор нульового результату АЛП;

N – прапор від’ємного результату АЛП;

C – прапор переносу АЛП;

O – прапор переповнення АЛП;

I – прапор-індикатор, що визначає чи обробляється зараз переривання.

Для АЛП код функції займає 4 біти, відповідність назви функції і коду приведено у таблиці 2.1

Таблиця 2.1 – Функції арифметично-логічного пристрою

Назва функції	Код
ADD	0000
ADDC	0001
SUB	0010
SUBC	0011
MUL	0100
DIV	0101
AND	0110
OR	0111
XOR	1000
SRL	1001
SLL	1010
SRA	1011
SLA	1100

Код операції процесору складається з 5 біт. Назва, кодування операції і її опис приведені у таблиці 2.2

Позначення у табл. 2.2:

- n – номер команди;
- f – код функції АЛП;
- K – константа;
- d – регістр призначення (перший з аргументів);

- r – регістр другого аргументу.

Таблиця 2.2 – Система команд процесору

№	Команда	Код	Опис
1	NOP	00000	-
2	LD	00001 dddddddd rrrrrrrr	Rd = (Rr)
3	LDIL	00010 KKKKKKKKKKKKKKKK	$R5_L$ = K
4	LDIH	00011 KKKKKKKKKKKKKKKK	$R5_H$ = K
5	ST	00100 ddddddd rrrrrrrr	(Rd) = Rr
6	MOV	00101 dddddddd rrrrrrrr	Rd = Rr
7	LJMP	00110 dddddddd	PC = Rd
8	JMP	00111 KKKKKKKKKKKKKKKK	PC_L = K
9	BREQ	01000 KKKKKKKKKKKKKKKK	PC_L = K при прапорі Z=1
10	BRHI	01001 KKKKKKKKKKKKKKKK	PC_L = K при прапорі Z=0 та N=0
11	BRLO	01010 KKKKKKKKKKKKKKKK	PC_L = K при прапорі Z=0 та N=1
12	CALL	01011 KKKKKKKKKKKKKKKK	PC_L = K зі збереженням наступного PC і регістру прапорів у стеку повернень
13	RET	01100 KKKKKKKKKKKKKKKK	Відновлення PC_L і регістру прапорів за інформацією у стеку повернень
14	RETI	01101 KKKKKKKKKKKKKKKK	Знімає прапор I та виконує дії, аналогічні RET
15	PUSH	01110 rrrrrrrr	R3--; (R3)=Rr
16	POP	01111 dddddddd	Rd=(R3); R3++
17	ADD	10000 dddddddd rrrrrrrr	Rd = Rd + Rr
18	ADDC	10001 dddddddd rrrrrrrr	Rd = Rd + Rr + C
19	SUB	10010 dddddddd rrrrrrrr	Rd = Rd - Rr
20	SUBC	10011 dddddddd rrrrrrrr	Rd = Rd - Rr - C
21	MUL	10100 dddddddd rrrrrrrr	Rd = Rd * Rr
22	DIV	10101 dddddddd rrrrrrrr	Rd = Rd / Rr
23	AND	10110 dddddddd rrrrrrrr	Rd = Rd and Rr
24	OR	10111 dddddddd rrrrrrrr	Rd = Rd or Rr
25	XOR	11000 dddddddd rrrrrrrr	Rd = Rd xor Rr
26	SRL	11001 dddddddd KKKKKKKK	Rd = Rd srl K
27	SLL	11010	Rd = Rd sll K
28	SRA	11011	Rd = Rd sra K
29	SLA	11100	Rd = Rd sla K
30	MATR	11101 ffff	(R6)=(R7) func (R8)

Порівняння двох машинних слів пропонується виконувати відніманням одного слова від іншого, і подальшим умовним переходом з використанням команд BREQ, BRHI, BRLO.

3 ПОБУДОВА ФУНКЦІОНАЛЬНОЇ СХЕМИ ПРОЦЕСОРУ

3.1 Умовні графічні позначення розроблюваних елементів

Для побудови функціональної схеми спочатку необхідно визначити умовні графічні позначення елементів, з яких буде складатися функціональна схема обчислювальної системи, процесору та ядра процесору.

Деталі функціонування даних елементів будуть наведені у розділі 4.

Для всіх елементів цього розділу виходи, що під’єднані до постійної пам’яті, оперативної пам’яті чи порту вводу-виводу, можуть працювати зі станом високого імпедансу.

На вхід core_n елементів CU і CORE подаються постійні сигнали від джерела живлення та/або нулю схеми; вхід установлюється таким, щоб кодувати номер обчислювального ядра процесору.

Умовне графічне позначення АЛП наведено на рисунку 3.1.

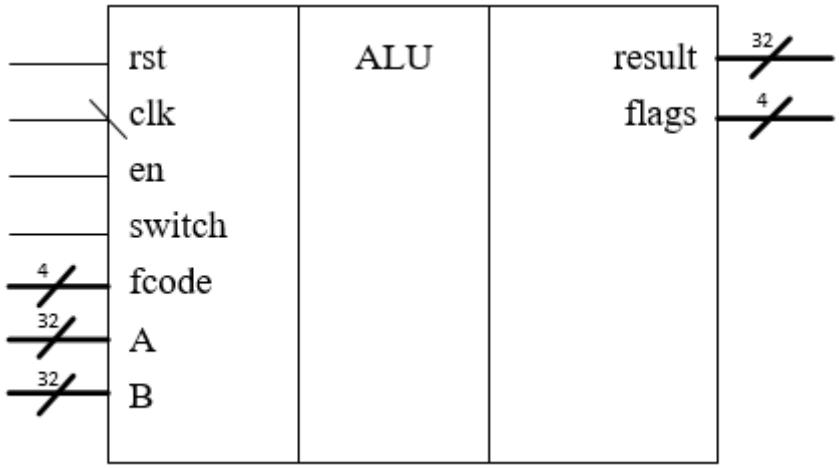

Рисунок 3.1 – Умовне графічне позначення АЛП

Умовне графічне позначення ОП наведено на рисунку 3.2.

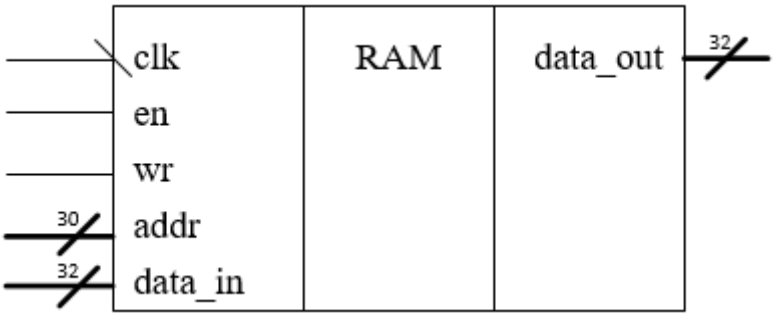

Рисунок 3.2 – Умовне графічне позначення ОП

Умовне графічне позначення ПП наведено на рисунку 3.3.

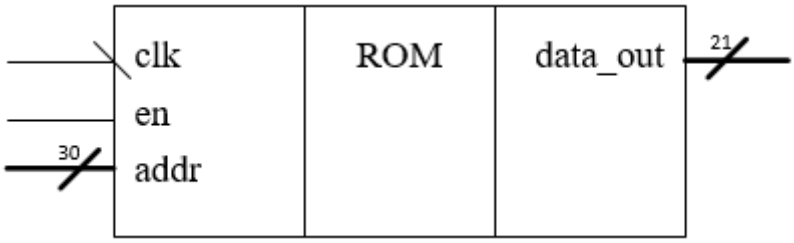

Рисунок 3.3 – Умовне графічне позначення ПП

Умовне графічне позначення порту вводу-виводу наведено на рисунку 3.4.

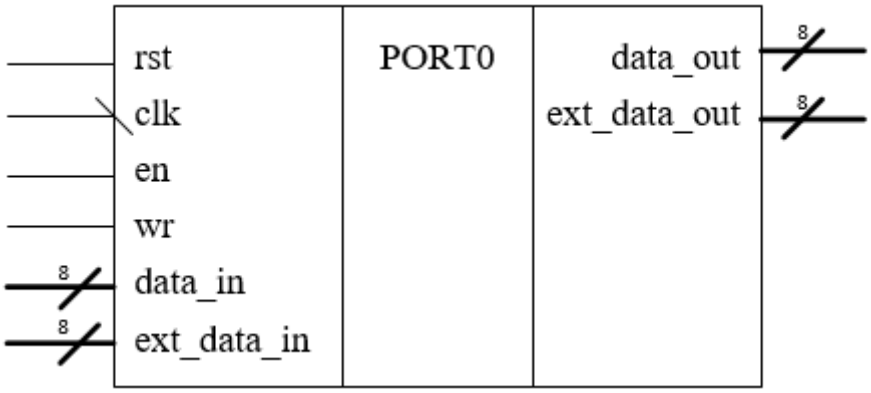

Рисунок 3.4 – Умовне графічне позначення пристрою вводу-виводу PORT0

Умовне графічне позначення блоку контролю до розподілених ресурсів наведено на рисунку 3.5.

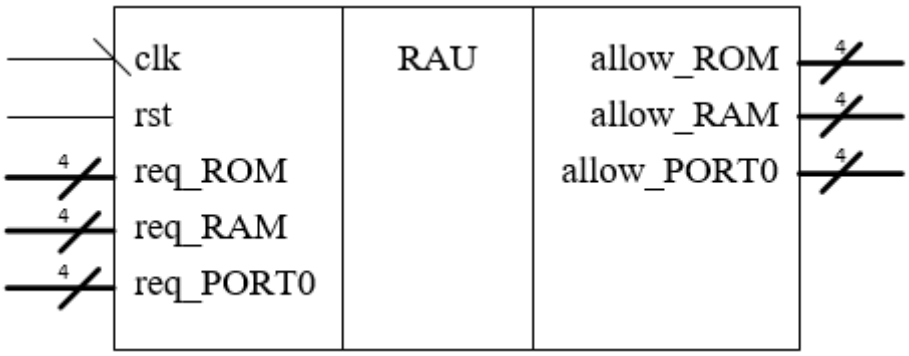

Рисунок 3.5 – Умовне графічне позначення RAU

Умовне графічне позначення контролюючого пристрою наведено на рисунку 3.6.

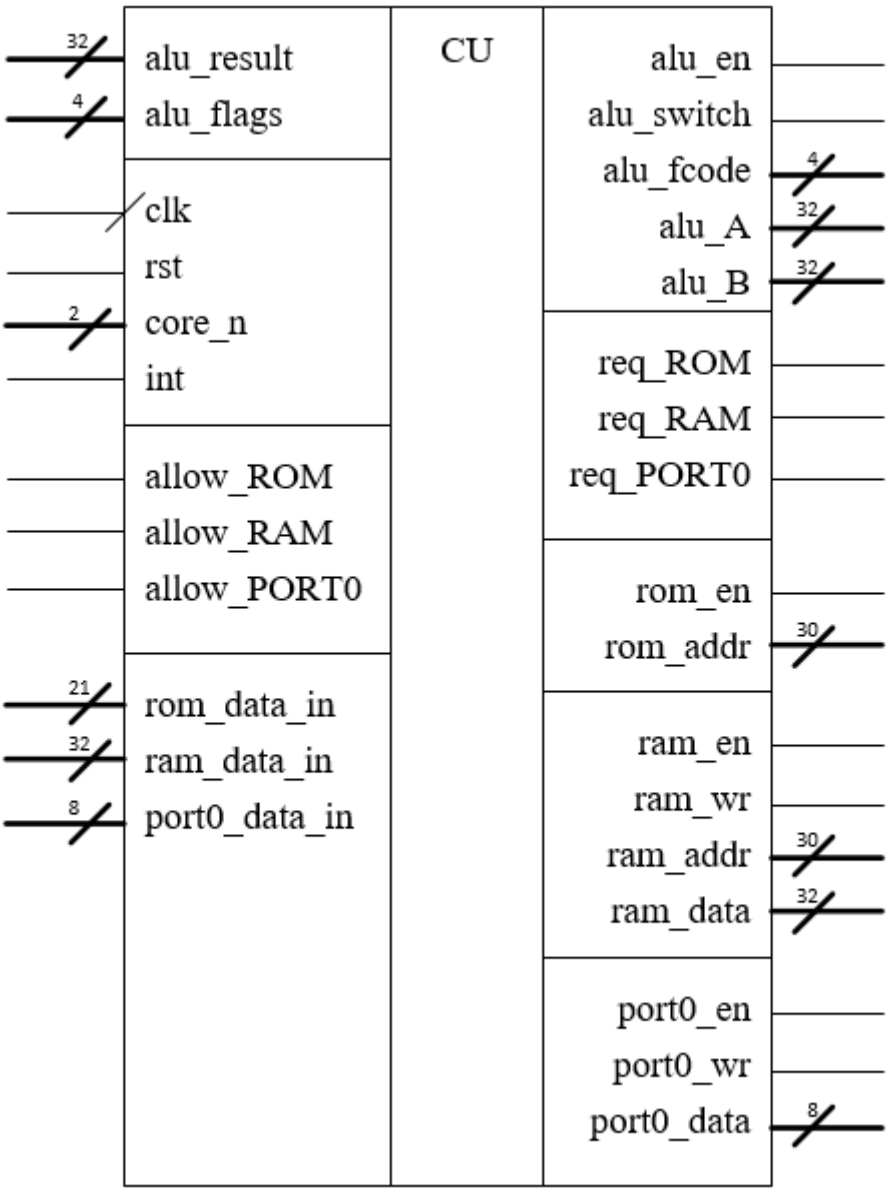

Рисунок 3.6 – Умовне графічне позначення КП

Умовне графічне позначення ядра процесору зображено на рисунку 3.7.

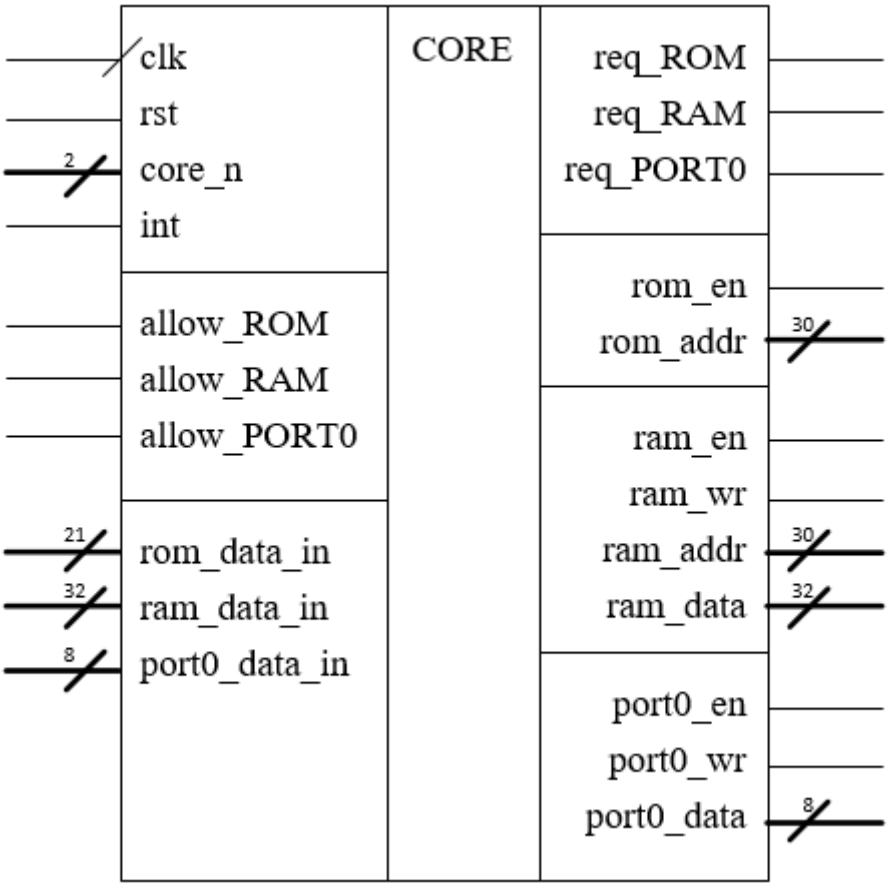

Рисунок 3.7 – Умовне графічне позначення ядра процесору

Умовне графічне позначення процесору зображено на рисунку 3.8.

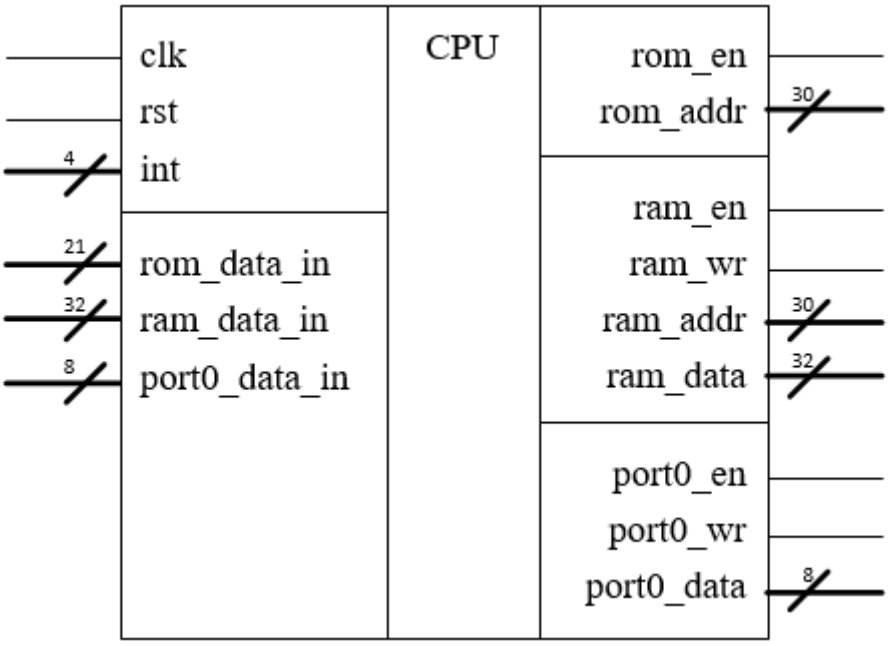

Рисунок 3.8 – Умовне графічне позначення процесору

3.2 Функціональна схема процесору та обчислювальної системи

Процесор складається з чотирьох ядер та блоку контролю доступу до розподілених ресурсів. Кожне ядро може виконувати окрему програму, яку читає з постійної пам’яті. Для обчислень у ядер є арифметико-логічні пристрої.

У обчислювальній системі є ресурси, що розподіляються між усіма ядрами – постійна пам’ять, оперативна пам’ять та порт вводу-виводу. Спроба одночасного використання одного ресурсу різними ядрами призведе до змішування їх сигналів, тому блок контролю доступу до ресурсів приймає сигнали-запити до ресурсів від ядер, да видає їм сигнали-дозволи.

Коли ядро не має доступу до ресурсу, то на виходах, що керують цим ресурсом установлюється стан високого імпедансу – це діє як розрив на лінії; виходи ядра не будуть змінювати стан сигналів від ядра що має доступ.

Функціональна схема процесору зображена на рисунку 3.9.

Функціональна схема ядра процесору зображена на рисунку 3.10.

Функціональна схема обчислювальної системи, у якій працює процесор зображена на рисунку 3.11.

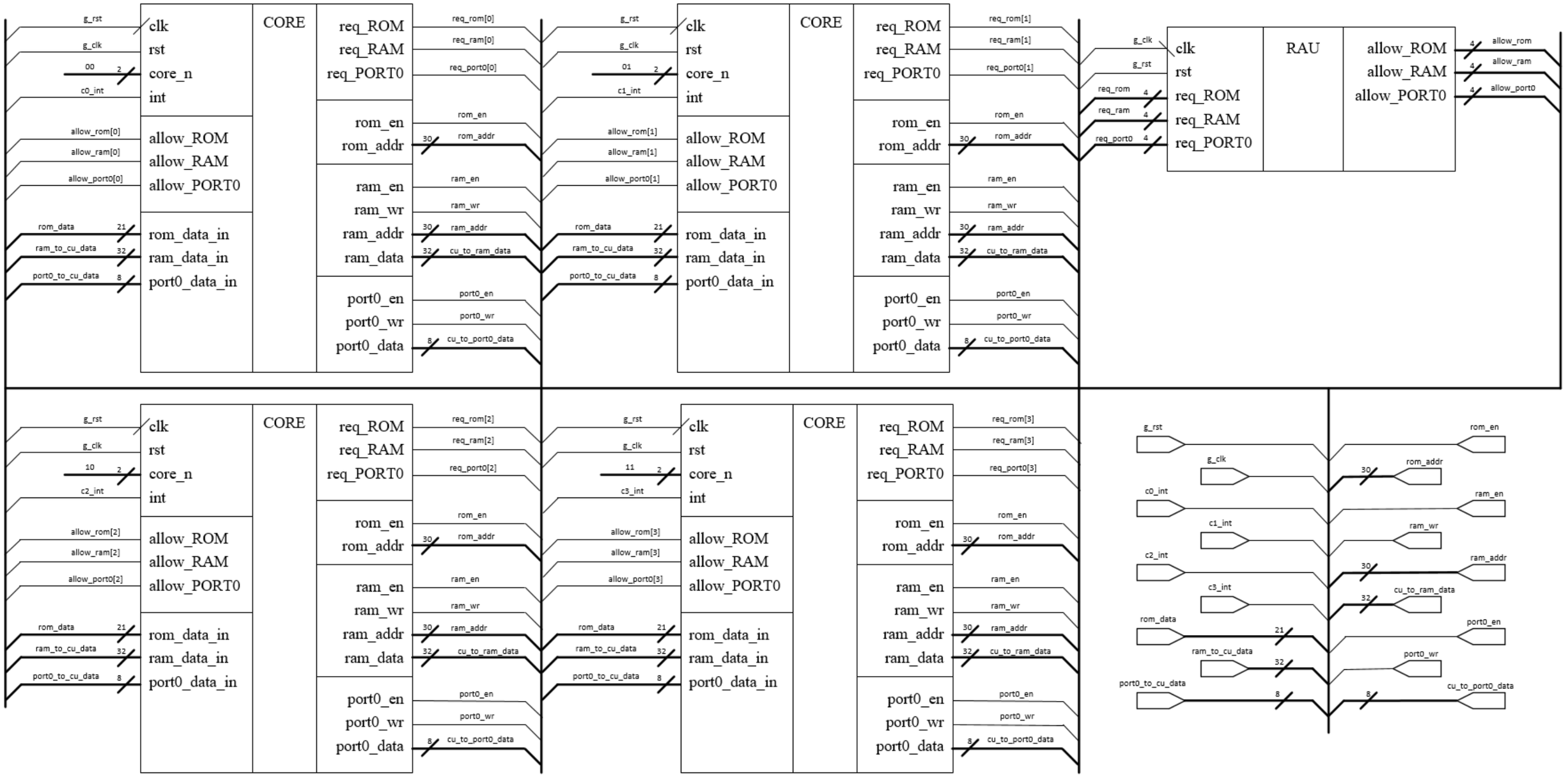

Рисунок 3.9 – Функціональна схема процесору

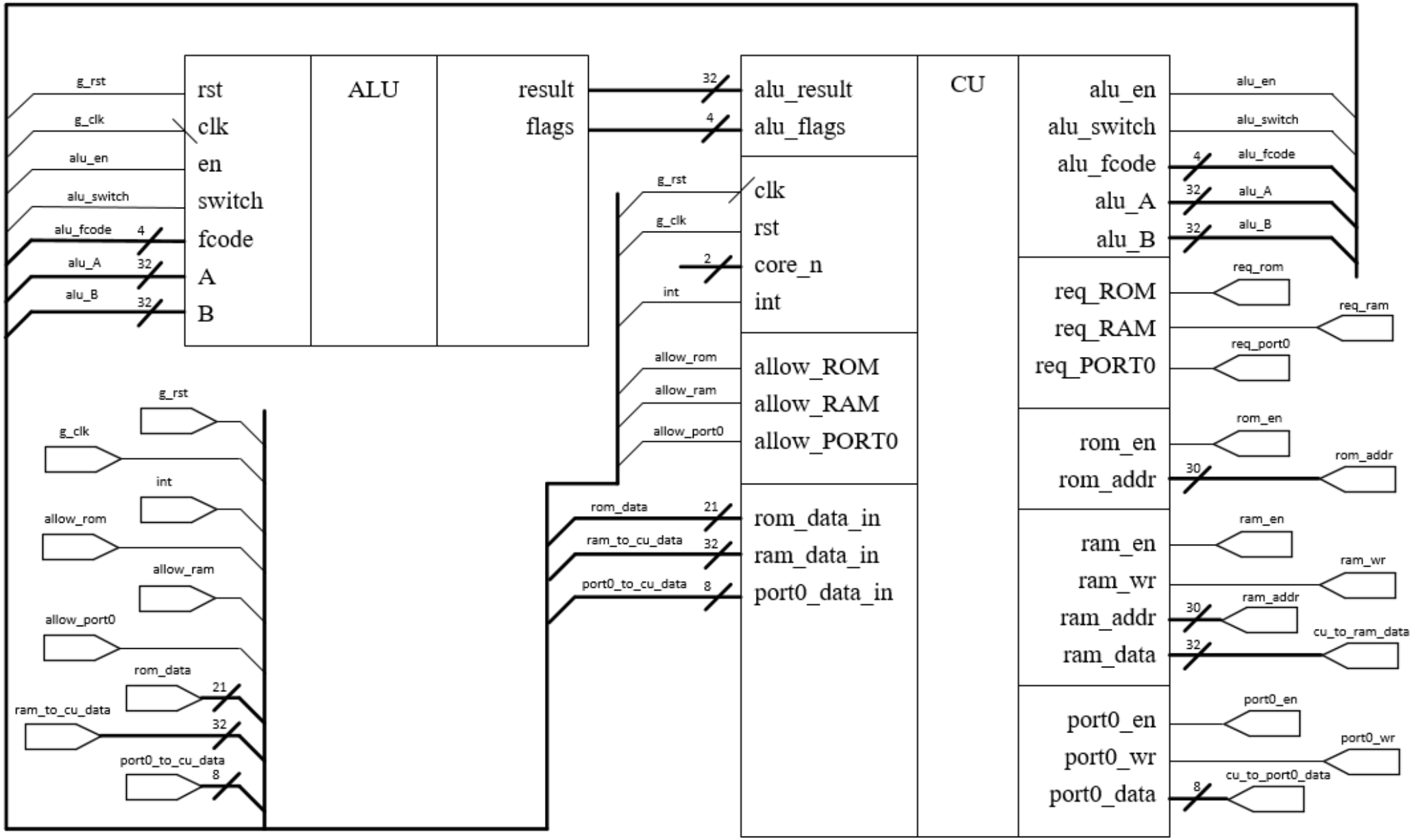

Рисунок 3.10 – Функціональна схема ядра процесору

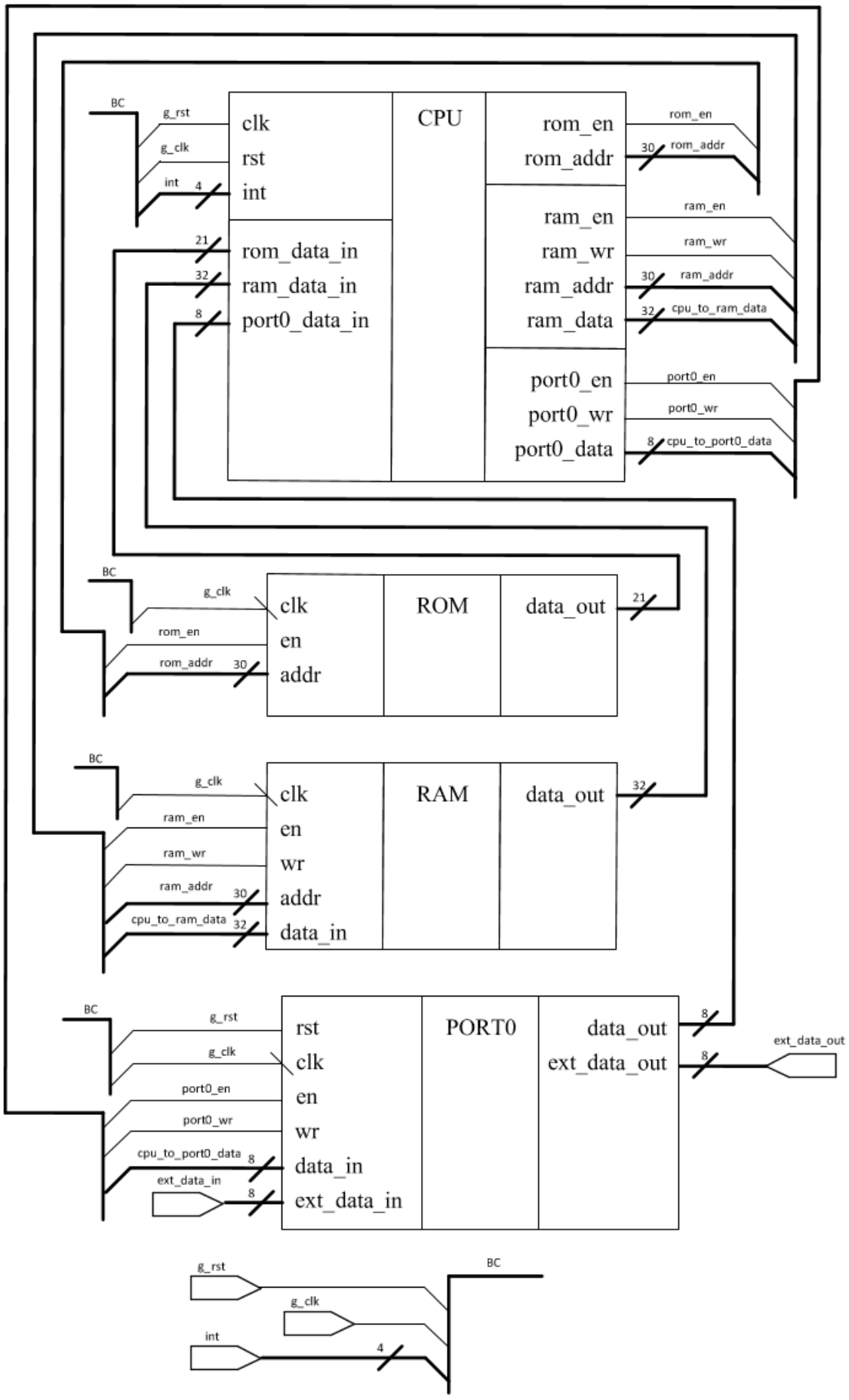

Рисунок 3.11 – Функціональна схема обчислювальної системи

4 РЕАЛІЗАЦІЯ НА VHDL ТА ДОСЛІДЖЕННЯ ПРОЦЕСОРУ

4.1 Особливості реалізації проекту

Процесор розроблювався та реалізовувався у безкоштовній версії САПР Xilinx Vivado.

Спочатку були розроблені окремі структурні блоки процесору та обчислювальної системи, що відповідають розробленим у розділі 3 умовним графічним позначенням елементів. Наступним кроком окремі блоки були об’єднані структурним стилем мови VHDL в обчислювальну систему, що містить процесор, постійну та оперативну пам'ять, й порт вводу-виводу.

Для спрощення написання проекту був створений пакет, що містить константи – розрядність шин адрес і даних, розрядність і кількість регістрів, коди функцій арифметико-логічного пристрою та операцій процесору, і коди регістрів для спрощення написання програм для розробленого процесору.

Програми для процесору зберігаються у постійній пам’яті. Вміст постійної пам’яті описаний у її VHDL-коді, з вказанням адреси комірки (у десятковій системі числення), мнемоніки машинної операції, позначень регістрів та безпосередніх значень.

Можливості мови VHDL і САПР Xilinx Vivado дозволили використовувати мову асемблеру без розробки додаткових засобів. Створені константи, що мають ім’я – мнемонічне позначення команди, і значення – код команди. Також задані константи виду R0 – R255, що містять коди відповідних регістрів.

Приклад написаної програми:

```
32 => OP_LDIL & x"0000",
33 => OP_LDIH & x"8000",
34 => OP_MOV & R10 & R5,
35 => OP_XOR & R5 & R5,
36 => OP_LDIL & x"0002",
37 => OP_MOV & R11 & R5,
38 => OP_LDIL & x"0003",
39 => OP_MUL & R5 & R11,
40 => OP_ST & R10 & R5,
41 => OP_JMP & x"0027",
```

4.2 Реалізація арифметично-логічного пристрою

Реалізація АЛП наведена у додатку А.

Для позначення АЛП у коді та на схемам застосовується скорочення ALU – arithmetic logic unit.

Арифметично-логічний пристрій працює з цілими числами зі знаком і виконує такі операції:

– арифметичні: додавання, додавання з переносом, віднімання, віднімання з переносом, множення, ділення;

– логічні: та, або, виключне або, та операції логічного й арифметичного зсуву.

– для вищеназваних операцій КП може прискорювати виконання програми при масовій обробці чисел однією функцією АЛП, деталі у п. 4.6.

Оскільки АЛП має в своєму складі елементи пам’яті, то перед використанням потрібно виконати скидання його стану за сигналом reset (rst).

АЛП спрацьовує при спадаючому фронту сигналу clk за умови позитивного сигналу en (enable). Код операції задається вхідним сигналом alu_fcode, операнди – сигналами A і B.

При виконанні операцій з урахуванням прапору переносу контролюючий пристрій (КП) модифікує код операції таким чином:

– при нульовому прапору переносу операція ADDC (або SUBC) на КП замінюється на ADD (або SUB);

– при одиничному прапору переносу команда ADDC (або SUBC) посилається на АЛП, де при виконанні операції враховується додатковий одиничний операнд.

Вихідний вектор result надає результат виконання операції, повністю або його половину (для операції довжина результату дорівнює подвоєній довжині операндів). Для отримання старшої частини результату при множенні КП використовує додатковий такт процесору.

Вихідний вектор flags складається з декількох інформаційних біт, що звітують про властивості результату. При тестуванні АЛП, прапори мають таке значення:

– перший прапор – прапор переповнення overflow. При роботі із числами зі знаком, одиничне значення цього прапору означає, що розрядності АЛП не вистачило для збереження результату.

– другий прапор – прапор переносу чи позики. При роботі з числами без знаку, одиничне значення цього прапору означає, що виник перенос одиниці за розрядну сітку (при додаванні), або позика одиниці з-за розрядної сітки (при відніманні). За допомогою цього прапору можна виконувати додавання та віднімання чисел без знаку із більшою розрядністю, ніж дозволяє АЛП;

– третій прапор – прапор від'ємного результату. Одиничне значення цього прапору означає, що результат обчислень – від'ємне число. Використовується для умовних переходів при порівнянні двох чисел – після віднімання командами BRHI, BRLO можна переходити до виконання інших частин програми, якщо перше число було більшим (BRHI) або меншим (BRLO);

– четвертий прапор – прапор нульового результату. Одиничне значення цього прапору означає, що результатом обчислень є нуль. Використовується для команди умовного переходу BREQ – перехід відбувається якщо операнди віднімання були рівними.

На рисунках 4.1, 4.2 проводиться тестування АЛП при виконанні операції додавання, числа інтерпретуються як беззнакові. При роботі з беззнаковими числами має значення прапор CARRY (другий біт flags) і не має значення прапор OVERFLOW (перший біт flags). При додаванні останніх операндів на рис. 4.2 виникає прапор переносу, тобто розрядності АЛП не вистачило для збереження результату. Програміст може перевіряти наявність переповнення у регістрі прапорів; якщо потрібно, можна додавати чи віднімати беззнакові числа більш високої розрядності ніж 32 з використанням команд ADDC / SUBC.

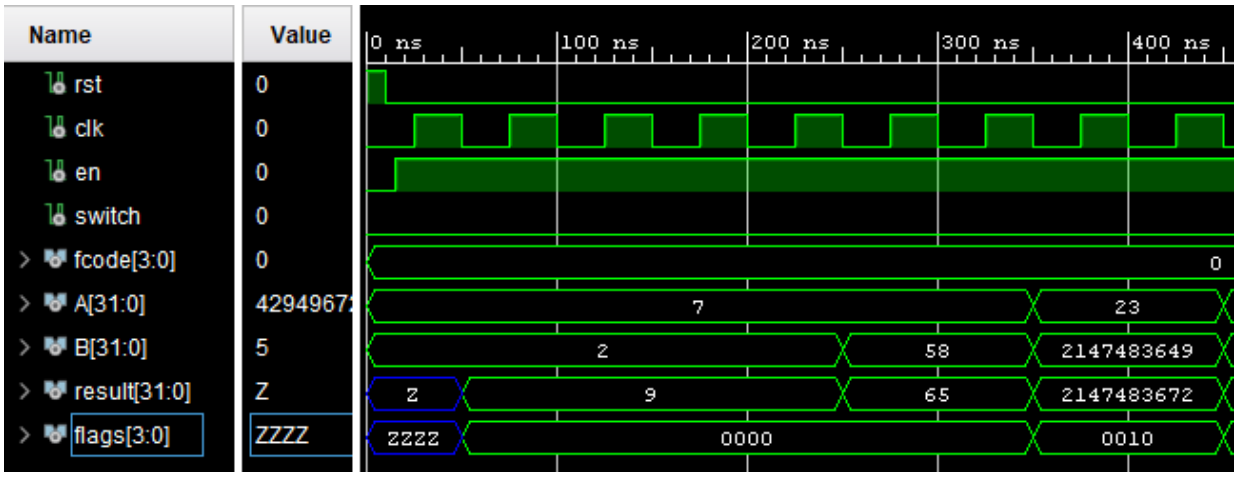

Рисунок 4.1 – Тестування АЛП при операції додавання (числа інтерпретуються як беззнакові), частина 1

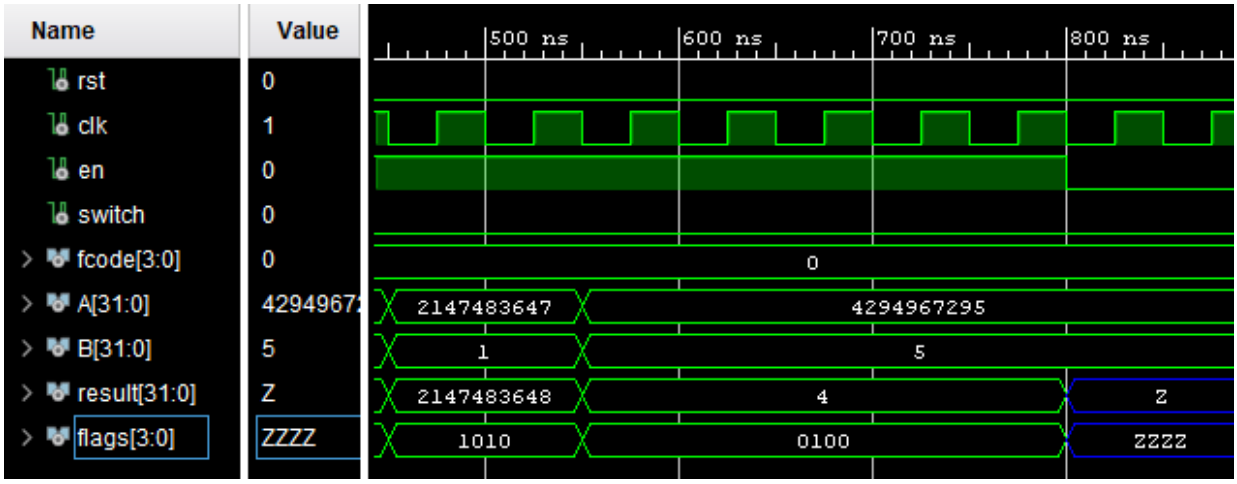

Рисунок 4.2 – Тестування АЛП при операції додавання (числа інтерпретуються як беззнакові), частина 2

На рисунках 4.3, 4.4 проводиться тестування роботи АЛП при виконанні операції додавання, числа інтерпретуються як знакові. При цьому має значення прапор переповнення OVERFLOW (перший біт flags) і не має значення CARRY (другий біт flags). На рис. 4.4 можна бачити некоректний результат, прапор OVERFLOW одиничний через виникнення переповнення.

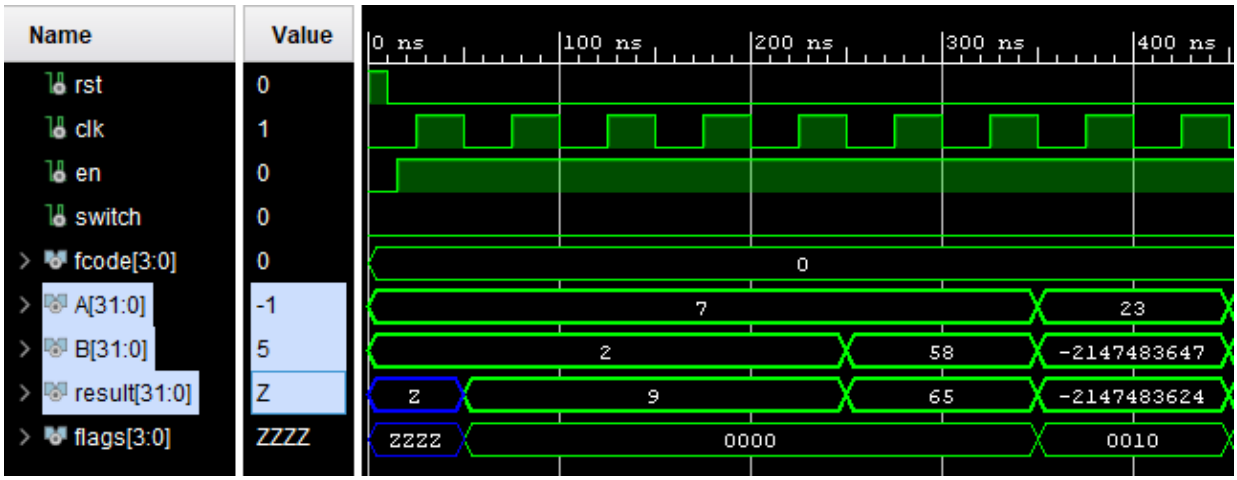

Рисунок 4.3 – Тестування АЛП при операції додавання (числа інтерпретуються як знакові), частина 1

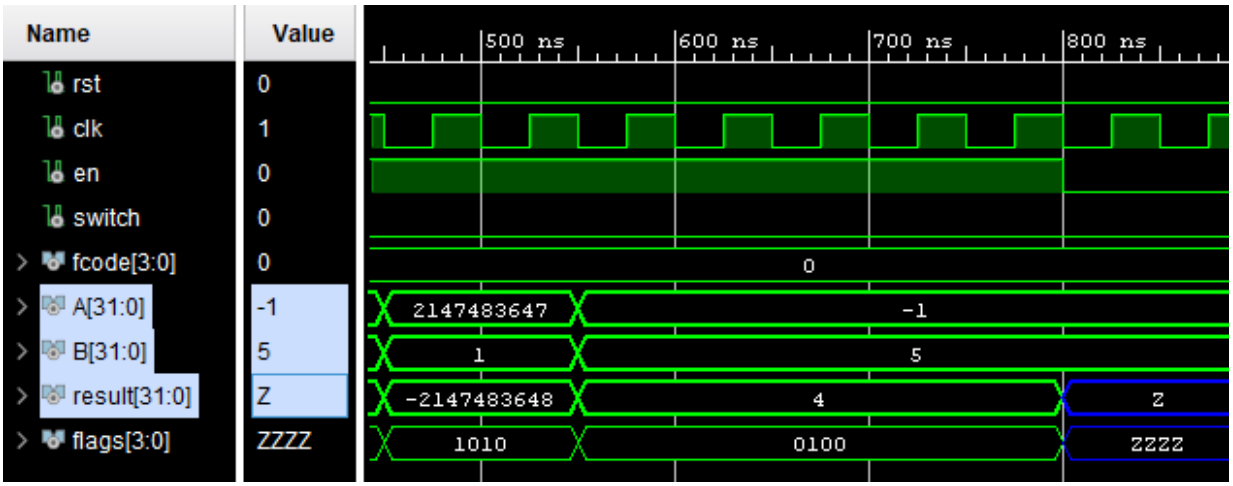

Рисунок 4.4 – Тестування АЛП при операції додавання (числа інтерпретуються як знакові), частина 2

На рисунку 4.5 тестується робота АЛП при виконанні операції віднімання, числа інтерпретуються як беззнакові. При відніманні чисел 23 і 2147483649 виникає позика одиниці з-за розрядної сітки, про що свідчить прапор CARRY (другий біт flags).

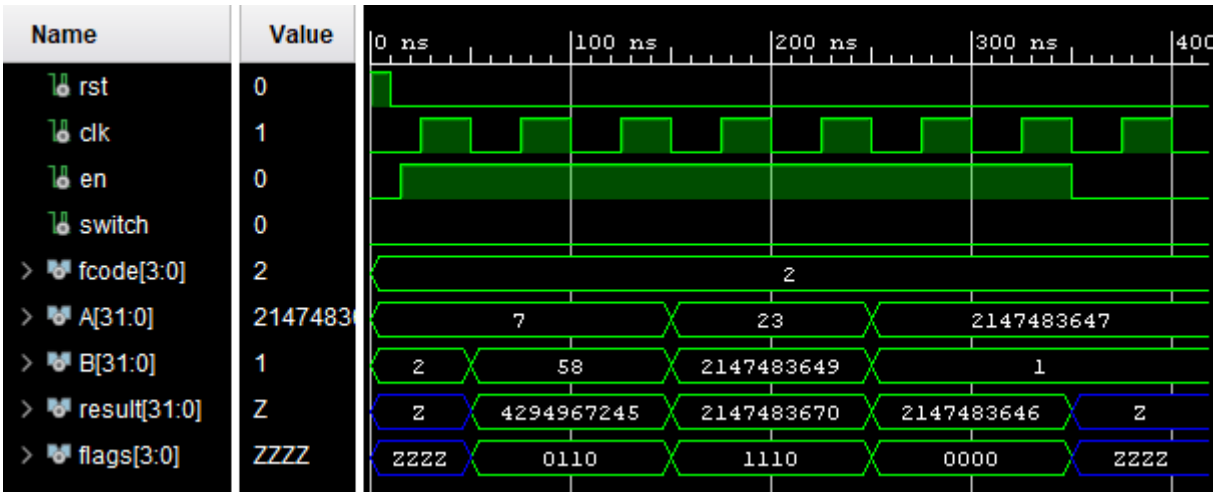

Рисунок 4.5 – Тестування АЛП при операції віднімання (числа інтерпретуються як беззнакові)

На рисунку 4.6 тестується робота АЛП при виконанні операції віднімання, числа інтерпретуються як знакові. При відніманні чисел 23 і -2147483647 виникає переповнення, про що свідчить прапор OVERFLOW (перший біт flags).

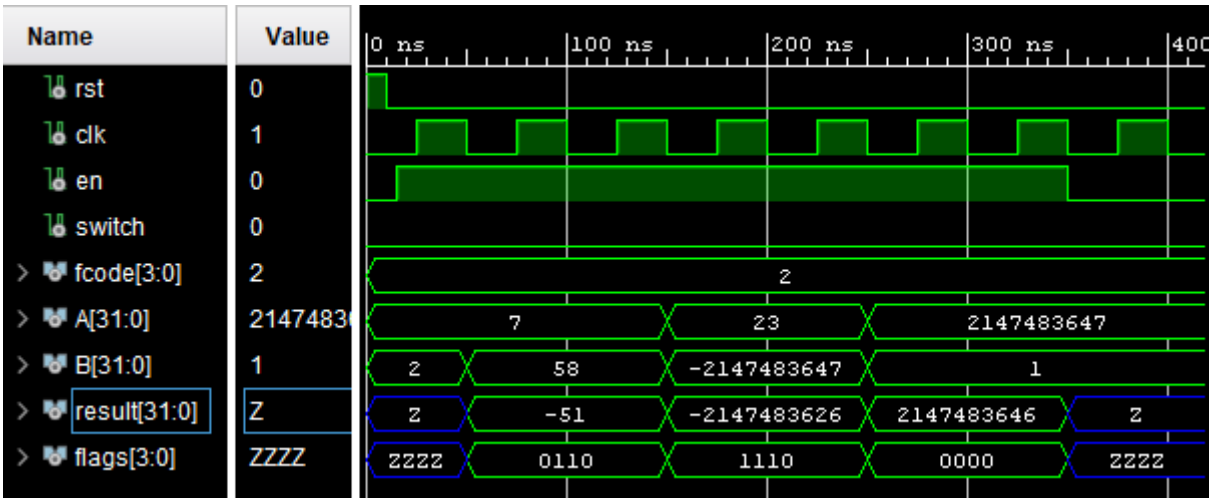

Рисунок 4.6 – Тестування АЛП при операції віднімання (числа інтерпретуються як знакові)

На рисунках 4.7, 4.8 проводиться тестування АЛП при виконанні множення. Для останніх операндів на рис. 4.7 виникає переповнення, про що свідчить прапор OVERFLOW (перший біт flags).

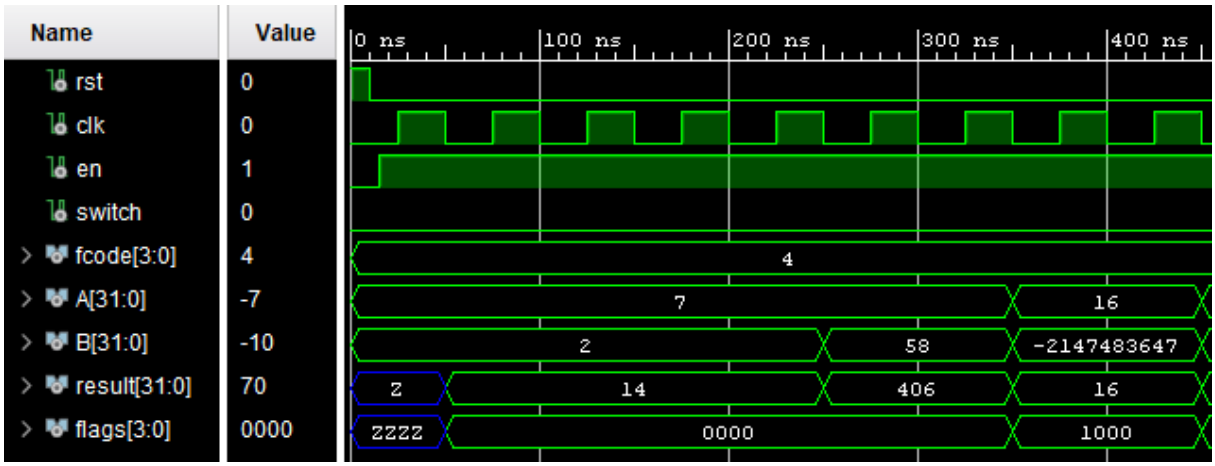

Рисунок 4.7 – Тестування АЛП при операції множення (при множенні останніх операндів виникає переповнення), частина 1

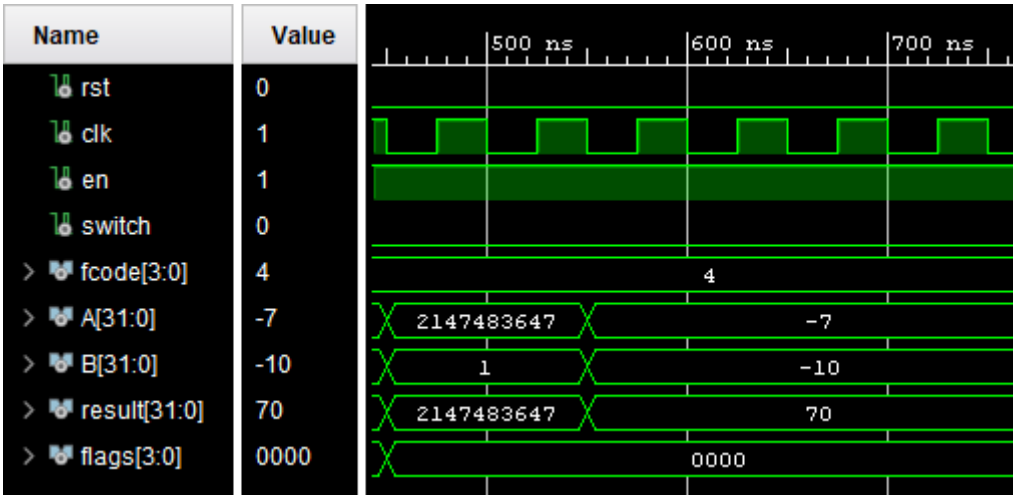

Рисунок 4.8 – Тестування АЛП при операції множення, частина 2

На рисунку 4.9 проводиться тестування роботи АЛП при виконанні ділення.

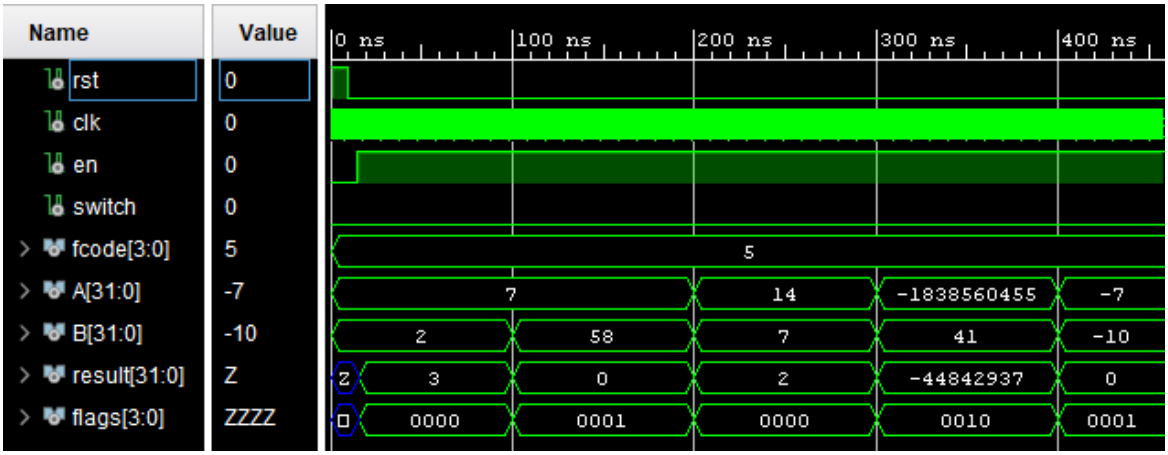

Рисунок 4.9 – Тестування АЛП при операції ділення

На рисунку 4.10 проводиться тестування АЛП при виконанні операції «логічне ТА». У молодших бітах операндів наведені всі можливі комбінації для тестування логічних функцій («00», «01», «10», «11»).

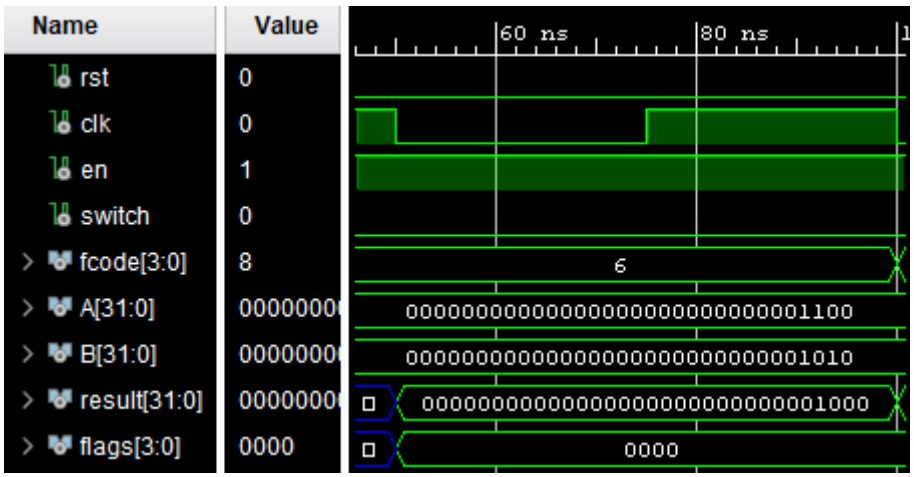

Рисунок 4.10 – Тестування АЛП при операції «логічне ТА»

На рисунку 4.11 проводиться тестування АЛП при виконанні операції «логічне АБО».

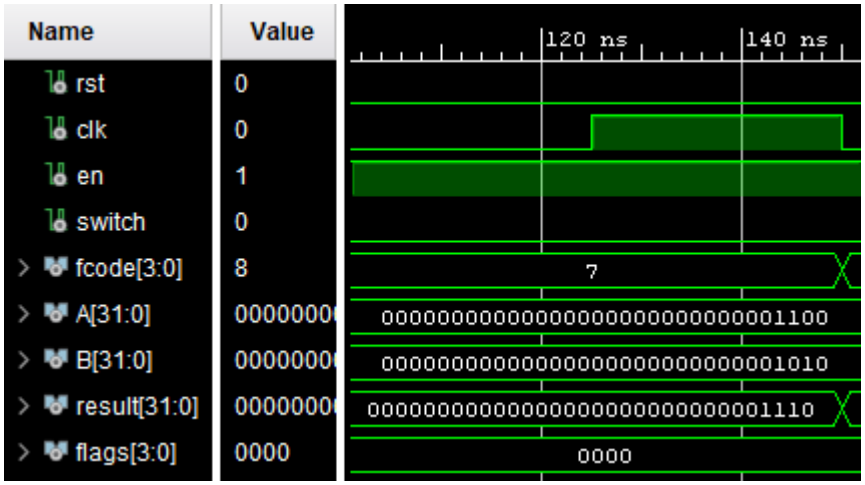

Рисунок 4.11 – Тестування АЛП при операції «логічне АБО»

На рисунку 4.12 проводиться тестування АЛП при виконанні операції «виключне АБО».

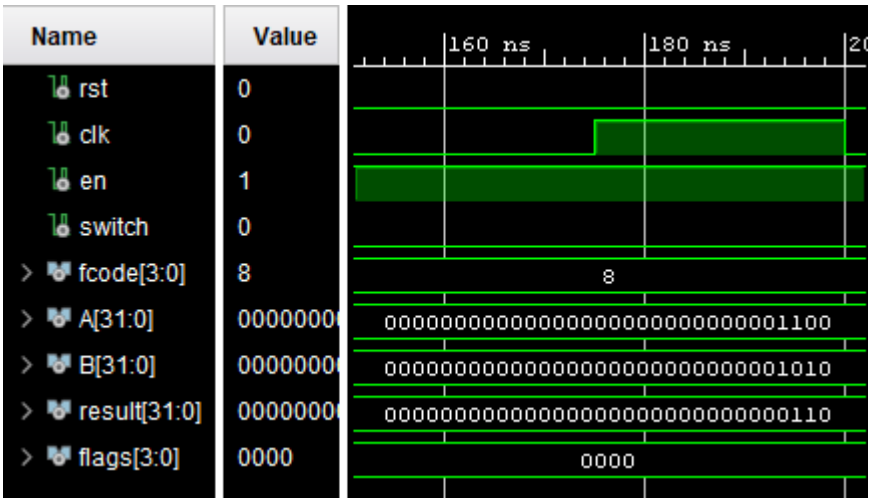

Рисунок 4.12 – Тестування АЛП при операції «виключне АБО»

На рисунку 4.13 проводиться тестування АЛП при виконанні функції «логічний здвиг вліво».

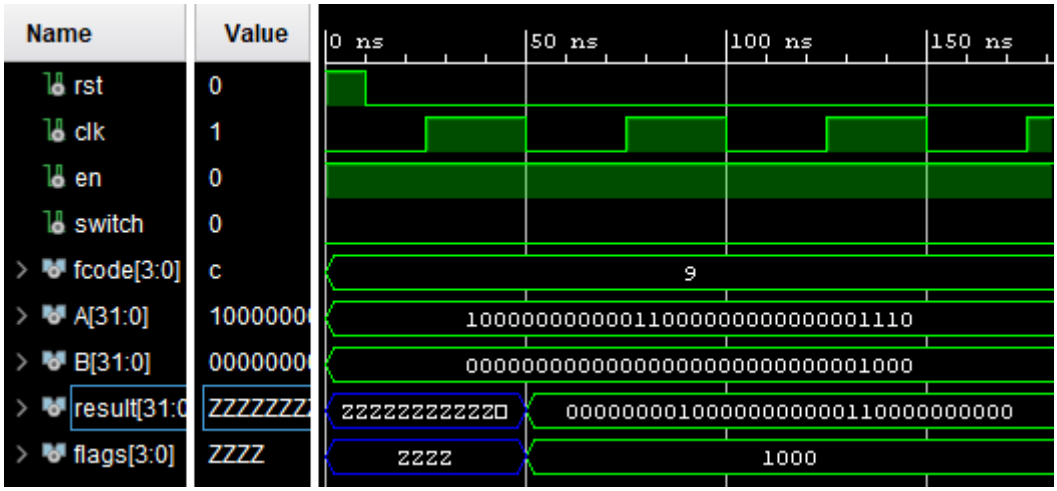

Рисунок 4.13 – Тестування АЛП при операції «логічний здвиг вліво»

На рисунку 4.14 проводиться тестування АЛП при виконанні операції «логічний здвиг вправо».

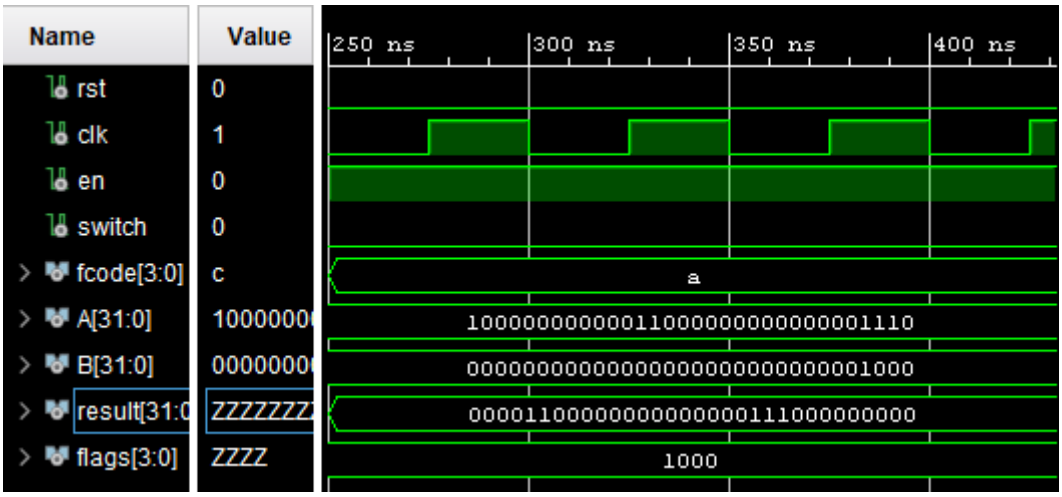

Рисунок 4.14 – Тестування АЛП при операції «логічний здвиг вправо»

На рисунку 4.15 проводиться тестування АЛП при виконанні операції «арифметичний здвиг вліво». Можна побачити, що при арифметичному здвигу знак першого операнду дублюється.

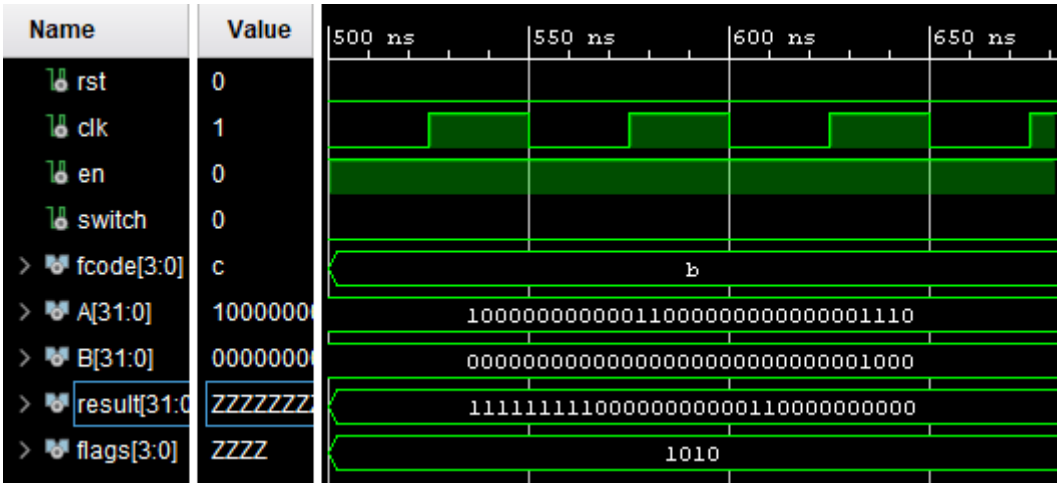

Рисунок 4.15 – Тестування АЛП при операції «арифметичний здвиг вліво»

На рисунку 4.16 проводиться тестування АЛП при виконанні операції «арифметичний здвиг вправо».

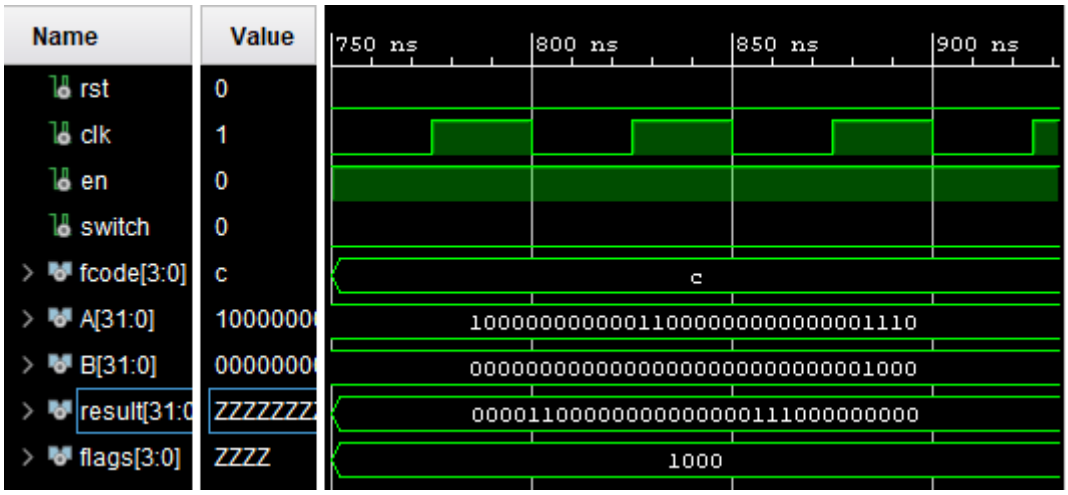

Рисунок 4.16 – Тестування АЛП при операції «арифметичний здвиг вправо»

4.3 Реалізація оперативної і постійної пам’яті

ОП і ПП є пристроями із внутрішнім станом, але через їх властивості сигнал rst не використовується. Скидання пам’яті для ОП при запуску системи займає багато часу і енергії, тож краще відразу працювати з ОП без скидання, враховуючи що комірки що не були використані після запуску, будуть заповнені довільними даними.

Для ПП скидання стану потрібне дуже рідко та займає багато часу, тож реалізується програмним шляхом.

Оперативна пам’ять спрацьовує по спадаючому фронту сигналу clk при позитивному значенні сигналу en. Адреса комірки задається вхідним вектором addr. Якщо сигнал wr позитивний, то у комірку заноситься інформація з вхідного вектору data_in. Незалежно від значення сигналу wr, значення поточної комірки виводиться на вихідний вектор data_out.

Через особливості мови VHDL, при спробі читати не ініціалізовану комірку повертаються не довільні дані, а сигнал U (undefined, невизначено). Для працездатності ОП при моделюванні вона була модифікована таким чином, що при запуску заповнена нульовими значеннями – але в реальній ОП неініціалізовані комірки мають деякі визначені, довільні значення.

ОП у коді та на схемах позначається RAM – random access memory.

Реалізація ОП наведена у додатку Д.

На рисунках 4.17, 4.18 проводиться тестування роботи ОП. Демонструється читання і запис інформації по різним адресам.

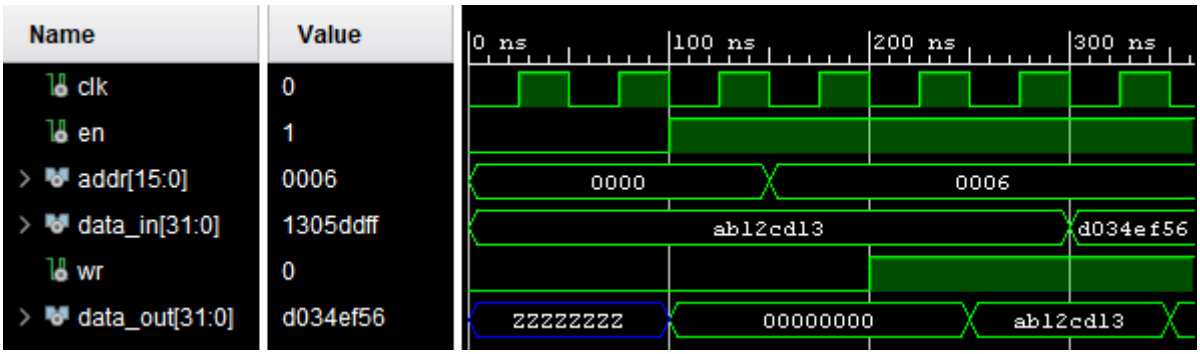

Рисунок 4.17 – Тестування роботи ОП, частина 1

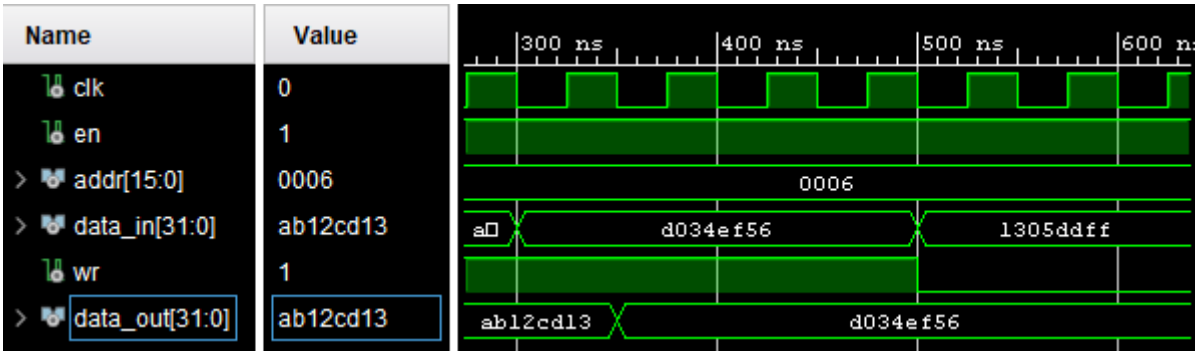

Рисунок 4.18 – Тестування роботи ОП, частина 2

Реалізація постійної пам'яті на VHDL подібна до ОП. Різниця у тому, що ПП доступна тільки до читання, тому відсутні вхідний сигнал wr і сигнал вхідних даних data_in. Внутрішній стан ПП заданий у коді, і не змінюється при роботі пристрою.

ПП у коді та на схемах позначається ROM – read-only memory.

Реалізація ПП наведена у додатку Г.

На рисунку 4.19 проводиться тестування роботи ПП, проводиться читання інформації з різних комірок.

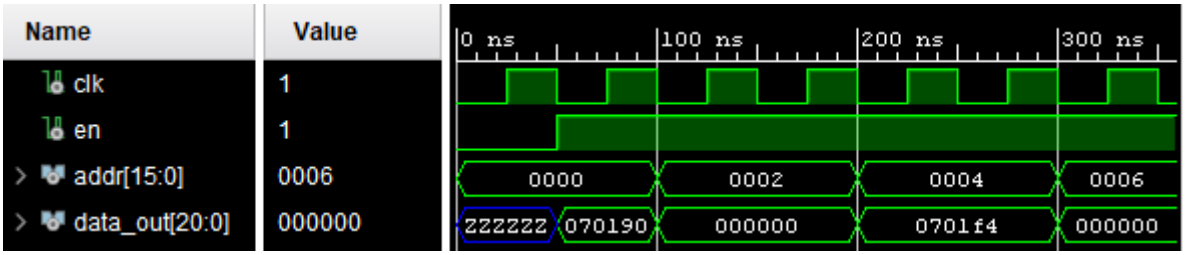

Рисунок 4.19 – Тестування роботи ПП

4.4 Реалізація блоку доступу до розподілених ресурсів

Процесор має чотири ядра, які при роботі конкурують за ресурси – ПП, ОП, PORT0. Блок доступу до розподілених ресурсів був позначений RAU (Resource Access Unit). RAU спрацьовує по спадаючому фронту сигналу clk.

Реалізація RAU наведена у додатку В.

Якщо ядру потрібен доступ до певного ресурсу, він посилає відповідний сигнал до RAU, а саме: req_ROM, req_RAM, req_PORT0. По спадаючому фронту RAU визначає, якому ядру до яких ресурсів надати доступ, і виставляє дозволяючі сигнали: allow_ROM, allow_RAM, allow_PORT0.

При ініціалізації системи сигналом rst черга установлюється таким чином: ядро #0 перше в черзі, ядро #1 – друге, і так далі.

КП потребує декілька тактів при роботі з ПП, ОП та PORT0. Тому RAU після видачі доступу ядру до певного ресурсу буде надавати доступ до тих пір, поки ядро не відмовиться від нього. Обчислювальна система буде працювати коректно і без монопольного захоплення ресурсу, бо КП звільнює ресурс після виконання кожної машинної операції.

Доступ до ресурсу надається тому ядру, яке ближче до початку черги і потребує ресурс, при цьому попереднє ядро вже звільнило ресурс (або у минулому такті ніяке з ядер не потребувала доступ).

На рисунку 4.20 проводиться тестування роботи RAU. Можна побачити, що зміна сигналів дозволу проводиться не по кожному спадаючому фронту тактового сигналу, а лише при відмові ядра від ресурсу.

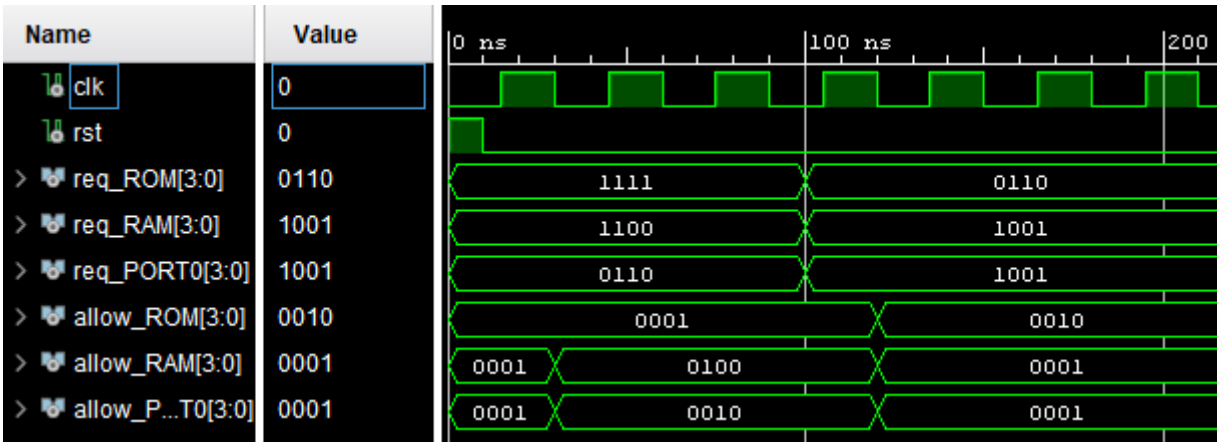

Рисунок 4.20 – Тестування роботи блоку контролю доступу до розподілених ресурсів

4.5 Реалізація порту вводу-виводу

Порт вводу-виводу (ПВВ) призначений для обміну даними із зовнішнім середовищем. У коді та на схемах він позначається PORT0.

ПВВ має 8-розрядний вихідний сигнал і 8-розрядний вхідний. При позитивному значенню сигналу rst ПВВ установлює всі вихідні порти у нульовий стан. При спадаючому фронту сигналу clk, в залежності від інших вхідних сигналів, ПВВ: читає сигнал із зовнішнього середовища і виставляє його на вихідний внутрішній порт, або читає вхідний сигнал від КП та виставляє його на вихідний зовнішній порт.

Реалізація ПВВ наведена у додатку Е.

На рисунку 4.21 проводиться тестування роботи порту вводу виводу, виконуються операції читання з зовнішнього порту за запису в нього.

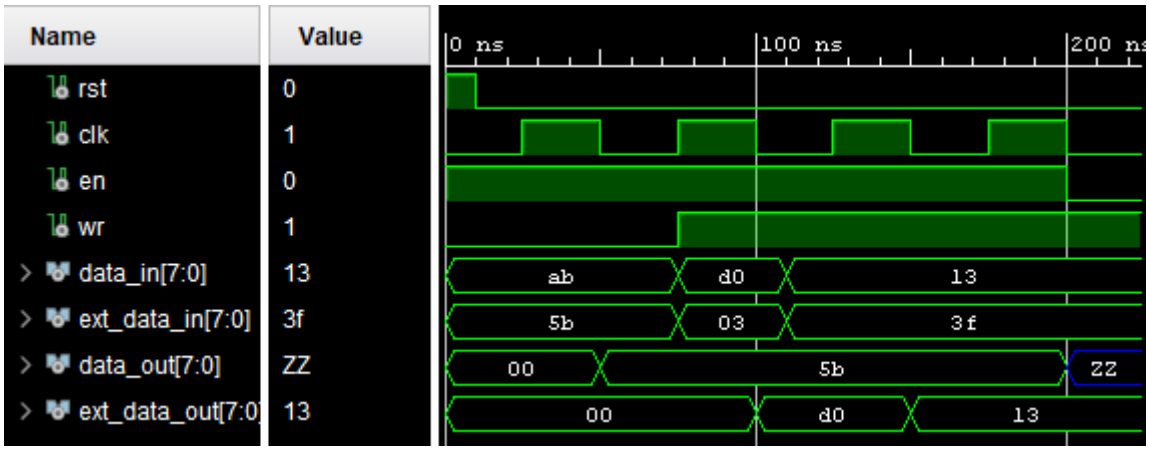

Рисунок 4.21 – Тестування роботи порту вводу-виводу

4.6 Реалізація керуючого пристрою

Керуючий пристрій – частина ядра процесору, яка працює з пам’яттю та портом вводу-виводу для отримання та виводу інформації, та з АЛП для виконання арифметично-логічних операцій. КП позначений у коді та на схемах як CU (control unit).

Реалізація КП наведена у додатку Б.

КП складається з кінцевого автомату, стеку повернень з підпрограм та блоку регістрів.

Блок регістрів зберігає 256 регістрів, всі з яких доступні для читання та майже всі – для запису.

Стек повернень з підпрограм використовується при операціях виклику підпрограми call та при обробці переривань. Об’єм стеку – 128 слів, що дозволяє мати глибину вкладеності підпрограм 64 (одне слово зберігає адресу для повернення, друге –регістр прапорів).

По сигналу rst кінцевий автомат (КА) приймає початковий стан. По зростаючому фронту сигналу clk КА, в залежності від поточного стану і вхідних сигналів, може змінити свій стан і значення вихідних сигналів.

Стани КА мають наступне призначення:

Стан 0 – підготовка до вибірки машинної операції: КА потребує доступ до ПП, доки не отримає його. Після отримання виставляє на вихідні шини інформацію для ПП, необхідну для зчитування машинної операції, включаючи адресу.

Стан 1 – отримання машинної операції від ПП і виконання операції.

Стан 2 – завершення виконання операцій, що взаємодіють із зовнішніми для КП пристроями та перевірка наявності переривання. За наявності переривання і за умови, що переривання не обробляється зараз, починається обробка переривання.

Алгоритм КП при роботі з розподілюваними пристроями:

1. Виставляє сигнал-запит до ресурсу;
2. Очікує до тих пір, доки не отримає дозволюючий сигнал;
3. За один чи декілька тактів виконує необхідні дії (читання чи запис для ПП, ОП та PORT0);
4. Скидає сигнал-запит до ресурсу у нульовий стан.

Алгоритм КП при роботі з арифметично-логічними операціями:

1. Виставляє необхідні дані на виходи, що керують АЛП;
2. Отримує результат та прапори стану АЛП;
3. Якщо виконувалася операція множення, додатковий такт процесору використовується щоб отримати старшу частину результату множення.

Алгоритм КП при виконанні матричної операції:

1. Якщо регістр-лічильник R9 має нульове значення, то подальше виконання алгоритму не потрібне;
2. По адресам з регістрів-вказівників на операнди R7, R8 отримуються операнди і виставляються на АЛП, з операнду машинної команди отримується код функції АЛП і виставляється на вихід alu_fcode;
3. Результат виконання операції АЛП заноситься у регістр, на який вказує регістр R6;
4. Значення регістрів-вказівників на операнди і результат збільшується на 1, значення регістру-лічильнику зменшується на 1;
5. Перейти до пункту 1.

Тестування КП буде проводитися тільки у складі процесору.

4.7 Реалізація RISC-процесору на основі розроблених блоків

На основі розроблених блоків, які описані у підпунктах 4.2 – 4.6, створений процесорний елемент. У коді та на схемах процесор позначений як CPU (central processing unit).

Структурним стилем мови VHDL відповідно до функціональної схеми процесору визначені такі зв’язки:

- АЛП поєднаний з КП, пара цих елементів є ядром процесору;
- КП зв’язаний з блоком контролю доступу до ресурсів (RAU);
- КП також під’єднаний до шин, що спільно використовуються чотирма ядрами процесору у порядку черги. Ці шини ведуть до ПП, ОП та PORT0.

Реалізація процесору наведена у додатку Ж.

4.8 Дослідження процесору

Дослідження процесору буде проводитися шляхом виконання програм, записаних у постійній пам’яті. Нижче наведені фрагменти програм та результати моделювання процесору.

Оскільки при роботі процесору його ядра по черзі отримують доступ до постійної пам’яті та інших ресурсів, показані тільки інформативні частини результатів моделювання.

У кожного ядра є своя адреса ПП для початку виконання програми та обробки переривання. Всі програми для тестування процесору знаходяться у ПП, але зміною команди безумовного переходу задається конкретна програма для виконання.

На рисунках 4.22 – 4.25 зображене тестування програми для циклічного множення чисел. Результати виводяться зо порт вводу-виводу, що позначений як ext_data_out.

Код програми:

```
32 => OP_LDIL & x"0000",
33 => OP_LDIH & x"8000",
34 => OP_MOV & R10 & R5,
35 => OP_XOR & R5 & R5,
36 => OP_LDIL & x"0002",
37 => OP_MOV & R11 & R5,
38 => OP_LDIL & x"0003",
39 => OP_MUL & R5 & R11,
40 => OP_ST & R10 & R5,
41 => OP_JMP & x"0027",
```

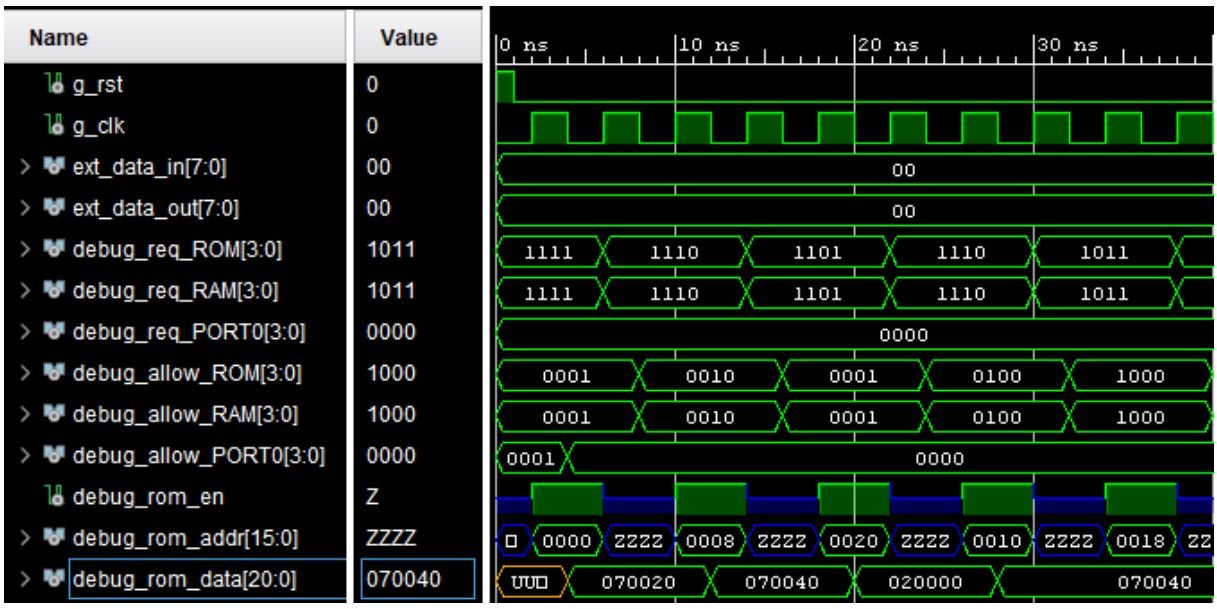

Рисунок 4.22 – Програма для циклічного множення чисел, частина 1

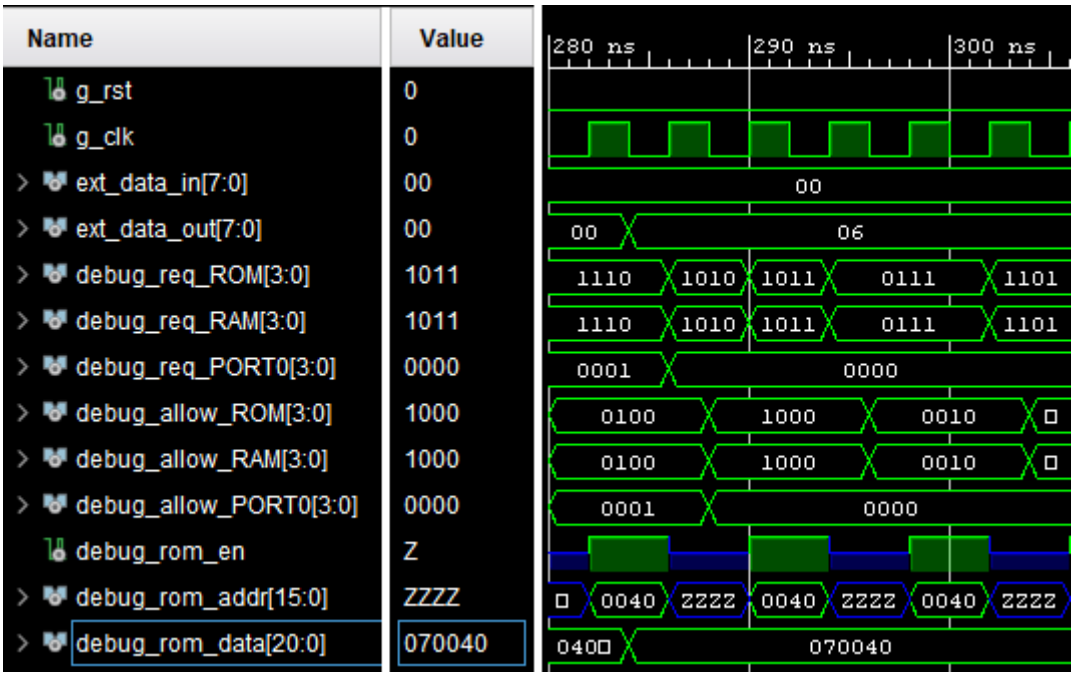

Рисунок 4.23 – Програма для циклічного множення чисел, частина 2

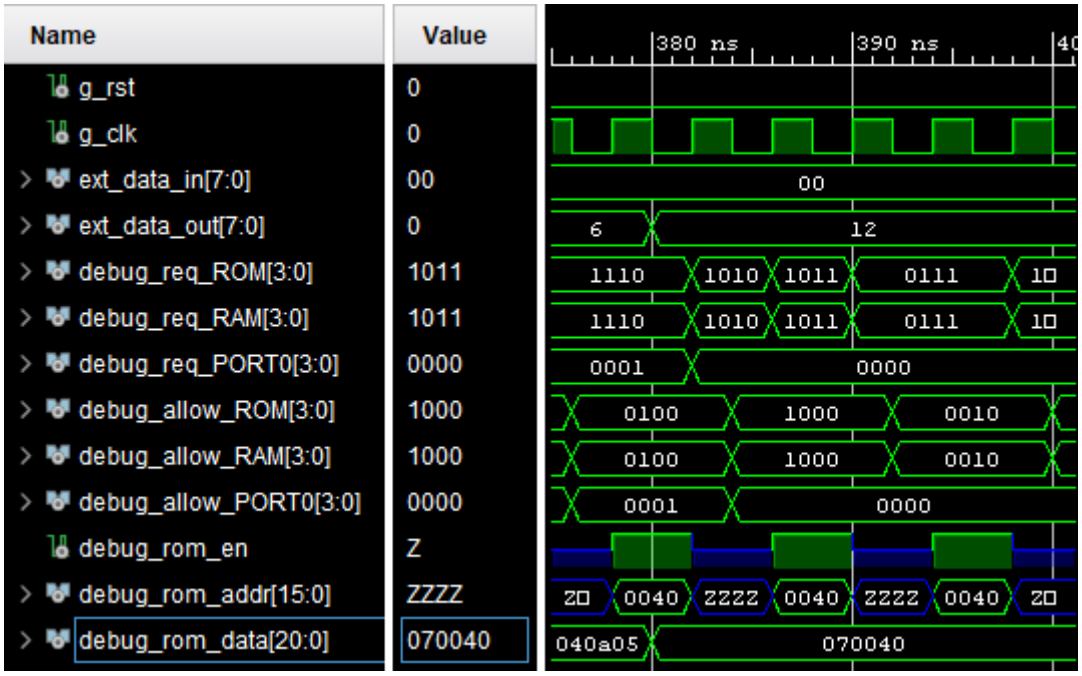

Рисунок 4.24 – Програма для циклічного множення чисел, частина 3

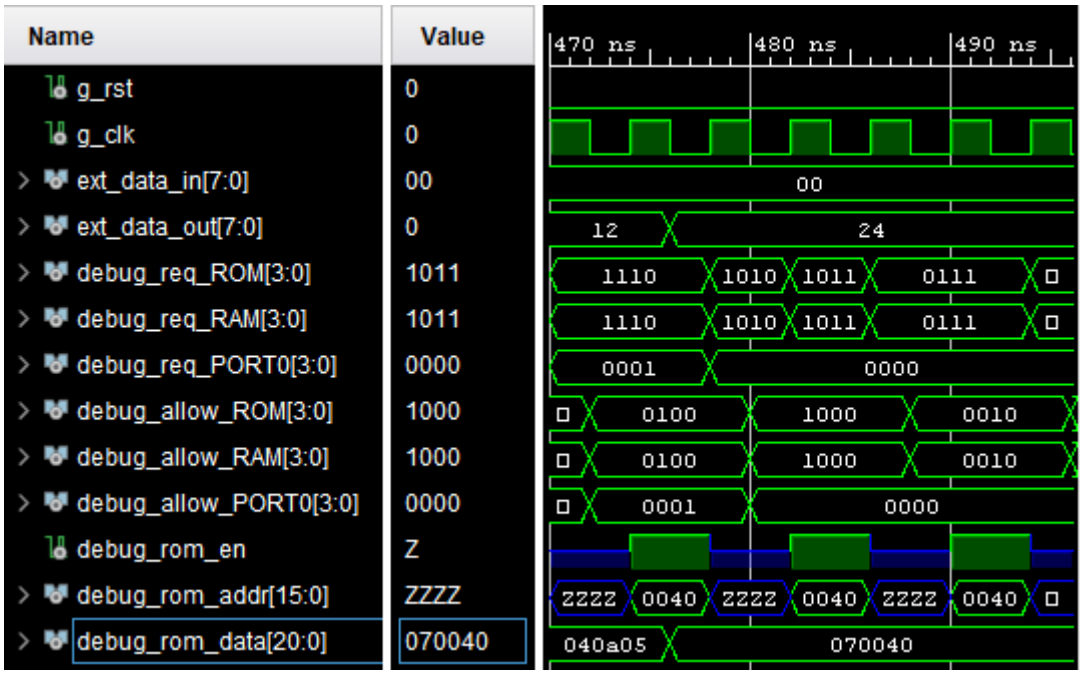

Рисунок 4.25 – Програма для циклічного множення чисел, частина 4

На рисунках 4.26 – 4.31 досліджується робота зі стеком. Стек розміщений у кінці оперативної пам’яті. Оскільки ядра можуть одночасно працювати зі стеком і ОП у них спільна, то у два молодші біти адреси ОП розміщується номер ядра. При додаванні елементу до стеку адреса вершини стеку зменшується на чотири.

На рис. 4.26 – 4.28 інформація заноситься до стеку, на рис. 4.29 – 4.31 інформація, що зчитується зі стеку виводиться на порт вводу-виводу. Заносяться числа 567b, 567e, 5681. Розрядність порту вводу-виводу – 8 біт, тому при виводі числа обрізаються і залишаються молодші розряди: 81, 7e, 7b.

Код програми:

```
50 => OP_LDIL & x"0003",
51 => OP_MOV & R10 & R5,
52 => OP_LDIL & x"5678",
53 => OP_ADD & R5 & R10,
54 => OP_PUSH & R5 & x"00",
55 => OP_ADD & R5 & R10,
56 => OP_PUSH & R5 & x"00",
57 => OP_ADD & R5 & R10,
58 => OP_PUSH & R5 & x"00",
59 => OP_LDIL & x"0000",
60 => OP_LDIH & x"8000",
```

61 => OP_POP & R11 & x"00",

62 => OP_ST & R5 & R11,

63 => OP_JMP & x"003d",

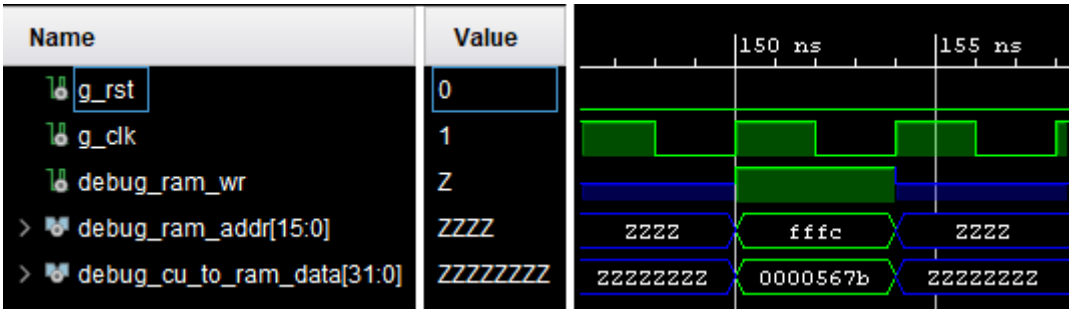

Рисунок 4.26 – Програма для роботи зі стеком, частина 1

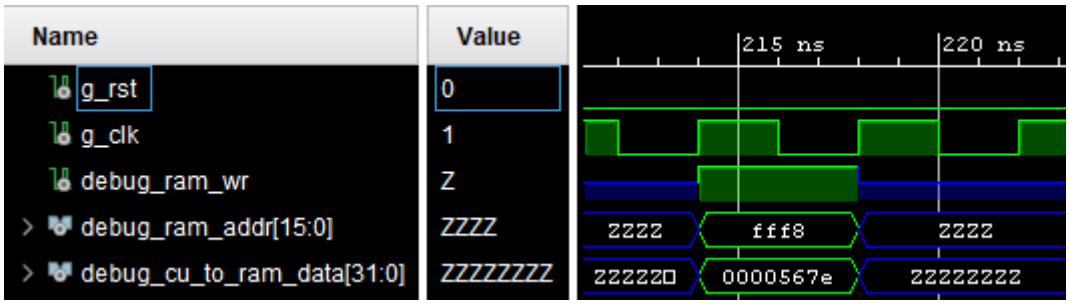

Рисунок 4.27 – Програма для роботи зі стеком, частина 2

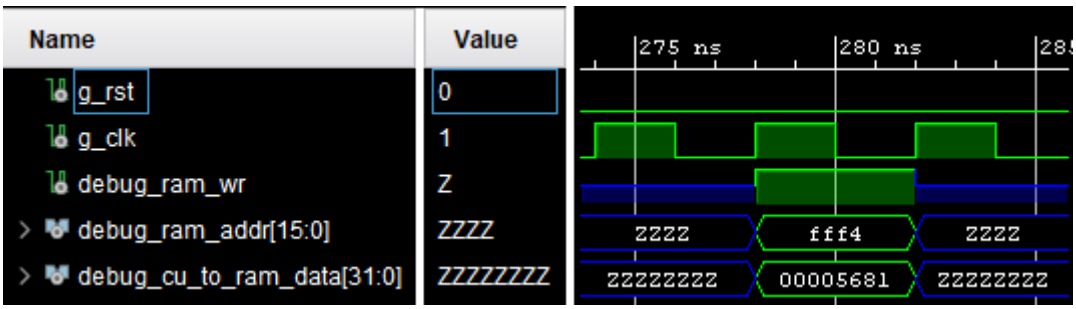

Рисунок 4.28 – Програма для роботи зі стеком, частина 3

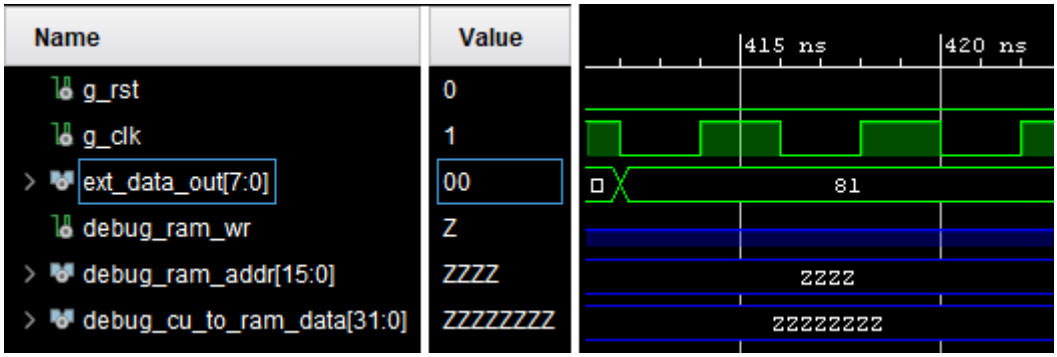

Рисунок 4.29 – Програма для роботи зі стеком, частина 4

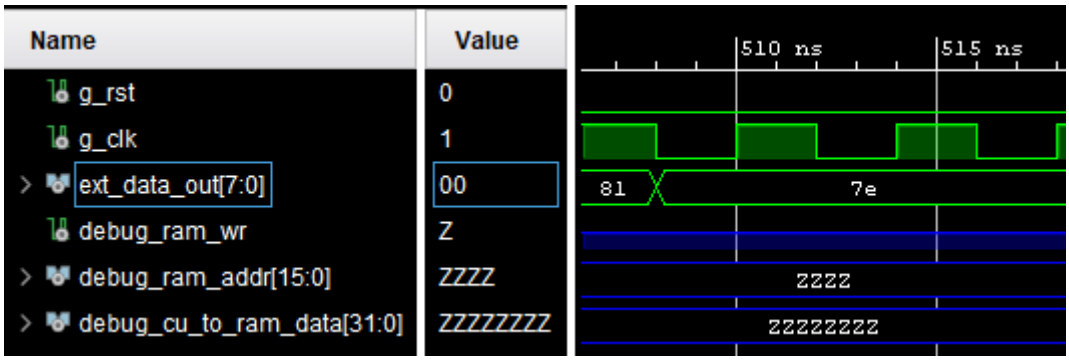

Рисунок 4.30 – Програма для роботи зі стеком, частина 5

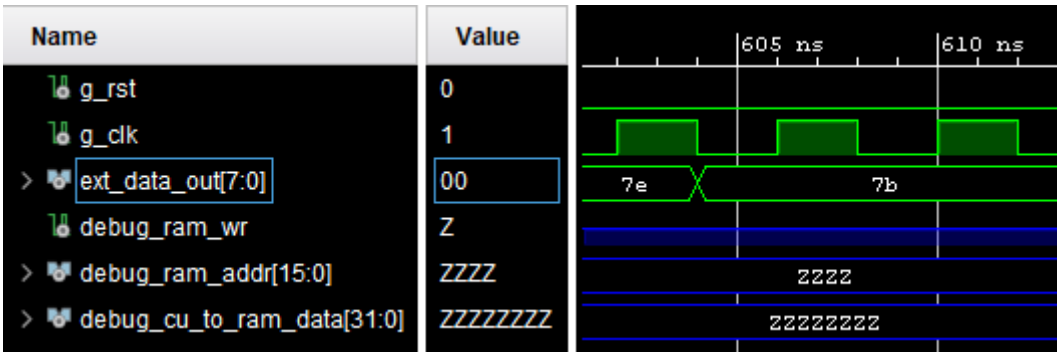

Рисунок 4.31 – Програма для роботи зі стеком, частина 6

На рисунках 4.32 – 4.34 зображене тестування програми із викликом підпрограм. На рис. 4.32 ядро #0 переходить до адреси 0x0041 (початок програми). На рис. 4.33 видно перехід від виконання адреси 0x0045 до 0x00ff, відповідно до інструкції CALL. На рис. 4.34 результат обчислень основної програми і підпрограм виводиться на порт вводу-виводу.

Програма підводить числа 3 і 5 до квадрату, додає до кожного з них 34 і сумує отримані числа. Результат цих обчислень – 102, що відповідає результатам моделювання на рис. 4.34.

Код програми:

```
65 => OP_LDIL & x"0003",
66 => OP_MOV & R20 & R5,
67 => OP_LDIL & x"0005",
68 => OP_MOV & R21 & R5,
69 => OP_CALL & x"00ff",
70 => OP_LDIL & x"0000",
71 => OP_LDIH & x"8000",
```

```
72 => OP_ST & R5 & R20,

255 => OP_MUL & R20 & R20,
256 => OP_MUL & R21 & R21,
257 => OP_CALL & x"012c",
258 => OP_ADD & R20 & R21,
259 => OP_RET & x"0000",

300 => OP_LDIL & x"0022",
301 => OP_ADD & R20 & R5,
302 => OP_ADD & R21 & R5,
303 => OP_RET & x"0000",
```

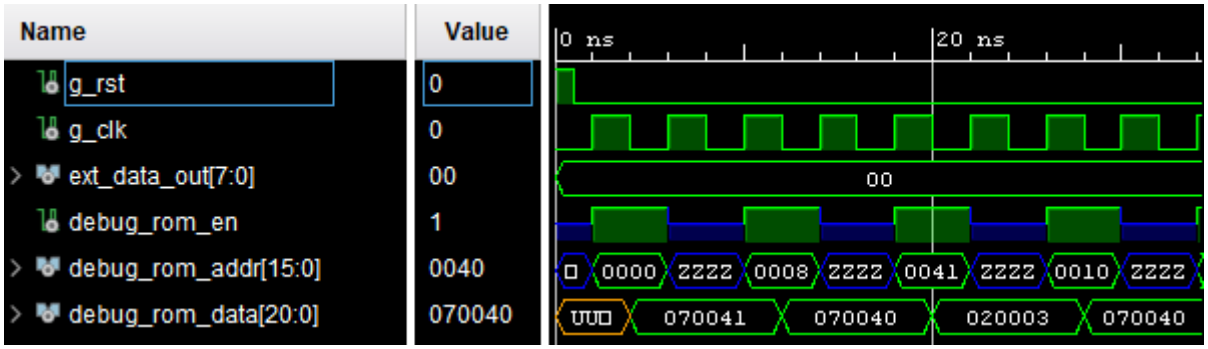

Рисунок 4.32 – Програма для перевірки роботи виклику підпрограм, частина 1

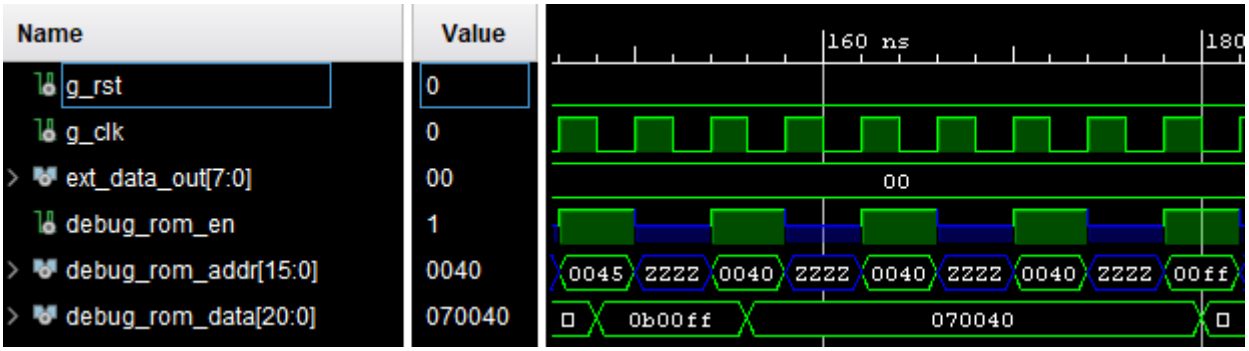

Рисунок 4.33 – Програма для перевірки роботи виклику підпрограм, частина 2

Рисунок 4.34 – Програма для перевірки роботи виклику підпрограм, частина 3

На рисунках 4.35 – 4.37 зображено тестування програми з умовними переходами. Порівняння чисел виконується шляхом їх віднімання і використанням машинних операцій умовного переходу BREQ, BRHI, BRLO.

На рис. 4.35 ядро #0 починає виконувати програму по адресі 0x0064. На рис. 4.36 видно спрацьовування умовного переходу – адреса програми змінилася з 0x0069 на 0x006e.

У програмі порівнюються числа 0x002f та 0x002e. Оскільки перше з них більше, спрацьовує ділянка коду що зберігає значення 0x3333, яке на рис. 4.37 виводиться на порт вводу-виводу (обрізане, через розрядність ПВВ 8 біт).

Код програми:

```
100 => OP_LDIL & x"002f",
101 => OP_MOV & R30 & R5,
102 => OP_LDIL & x"002e",
103 => OP_SUB & R30 & R5,
104 => OP_BREQ & x"006c",
105 => OP_BRHI & x"006e",
106 => OP_LDIL & x"dddd", -- default value, address 0x6a
107 => OP_JMP & x"006f",
108 => OP_LDIL & x"1111", --if equal, address 0x6c
109 => OP_JMP & x"006f",
110 => OP_LDIL & x"3333", -- if higher, address 0x6e
111 => OP_MOV & R31 & R5, -- next section, address 0x6f
112 => OP_LDIL & x"0000",
113 => OP_LDIH & x"8000",
114 => OP_ST & R5 & R31,
```

115 => OP_JMP & x"0040",

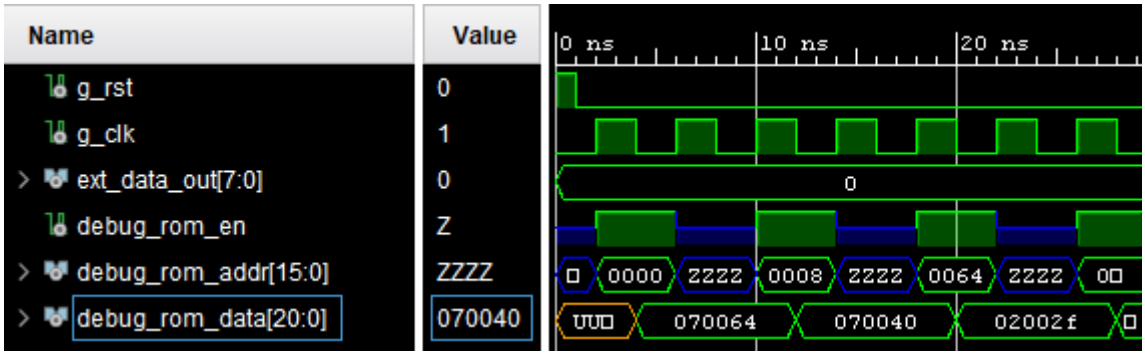

Рисунок 4.35 – Програма для перевірки умовних переходів, частина 1

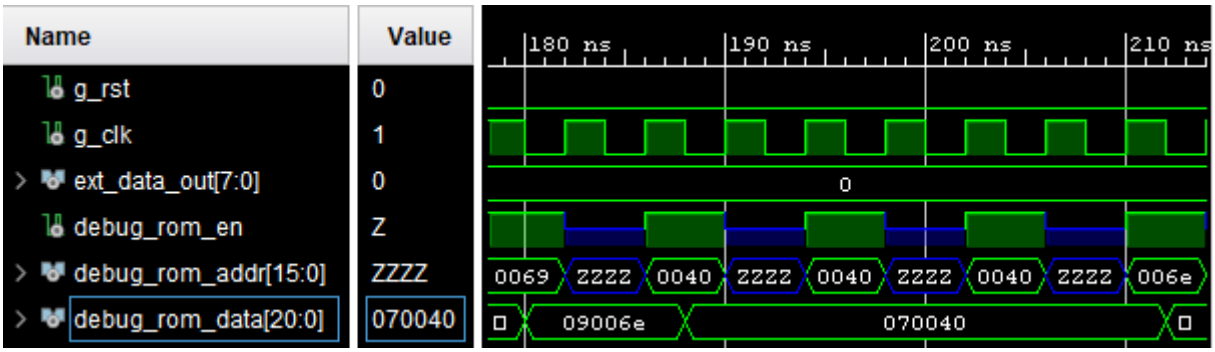

Рисунок 4.36 – Програма для перевірки умовних переходів, частина 2

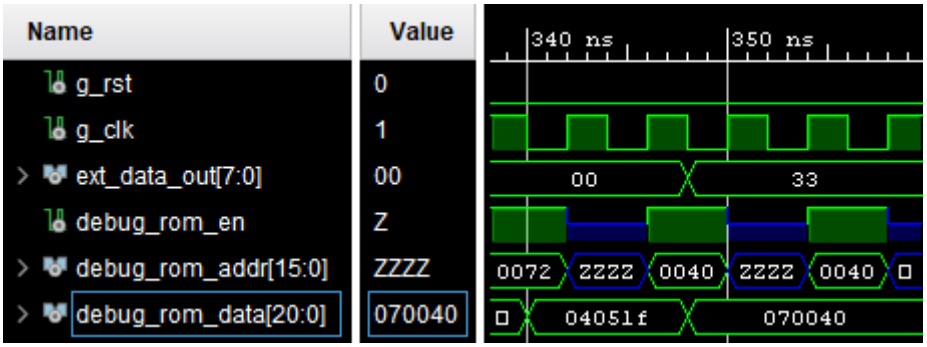

Рисунок 4.37 – Програма для перевірки умовних переходів, частина 3

На рисунках 4.38, 4.39 зображено тестування програми для обробки апаратного переривання.

Основна програма, що виконується процесором починає рахувати числа й виводити їх на ПВВ – 1, 2, 3 і так далі. При отриманні сигналу переривання інформація про наступну адресу для виконання і стан регістру прапорів зберігається у апаратний стек повернень. Далі починається виконання програми обробки

переривання, яка виводить на ПВВ значення 0xfe. Після повернення з програми обробки переривання рахунок чисел продовжується на тому ж місці, де закінчився.

Оскільки при обробці переривання можуть використовуватися регістри, які використовує основна програма, значення цих регістрів зберігається у стек перед виконанням операцій над ними, і відновлюється перед поверненням у основну програму.

Код програми:

```
-- custom code A, 0x190
400 => OP_LDIL & x"0000",
401 => OP_LDIH & x"8000",
402 => OP_MOV & R40 & R5,
403 => OP_XOR & R5 & R5,
404 => OP_LDIL & x"0001",
405 => OP_MOV & R41 & R5,
406 => OP_ST & R40 & R5, -- 0x196
407 => OP_ADD & R5 & R41,
408 => OP_JMP & x"0196",

-- interrupt handler A, 0x1f4
500 => OP_PUSH & R5 & x"00",
501 => OP_PUSH & R40 & x"00",
502 => OP_LDIL & x"0000",
503 => OP_LDIH & x"8000",
504 => OP_MOV & R40 & R5,
505 => OP_XOR & R5 & R5,
506 => OP_LDIL & x"00fe",
507 => OP_ST & R40 & R5,
508 => OP_POP & R40 & x"00",
509 => OP_POP & R5 & x"00",
510 => OP_RETI & x"0000",
```

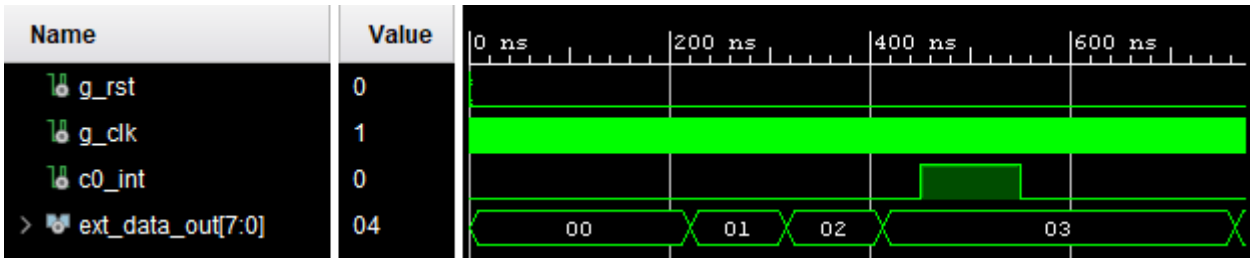

Рисунок 4.38 – Програма для обробки апаратного переривання, частина 1

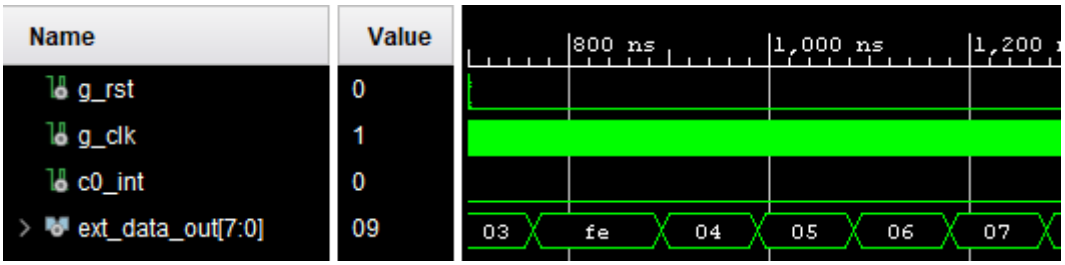

Рисунок 4.39 – Програма для обробки апаратного переривання, частина 2

На рисунках 4.40, 4.41 зображено тестування двох програм для обробки матриць. Над елементами двох матриць з однаковими індексами виконується операція, задано кодом операції АЛП (у даному випадку - множення). Операнди знаходяться у регістрах. Класична програма потребує стільки разів зчитувати машинну операцію множення з ПП, скільки разів потрібно виконати цю операцію.

Для машинної операції MATR, після зчитування команди відбувається виконання операції над парами елементів і збереження результатів без використання ПП, що пришвидшує виконання програми. Початковий адрес першої матриці задається у регістрі R7, початковий адрес другої матриці – у R8. Початковий адрес матриці результатів задається у R6, кількість операцій які необхідно виконати – у R9. Фрагменти програм наведено нижче.

Фрагмент класичної програми:

628 => OP_MOV & R30 & R10,

629 => OP_MOV & R31 & R11,

630 => OP_MOV & R32 & R12,

631 => OP_MOV & R33 & R13,

632 => OP_MOV & R34 & R14,

633 => OP_MOV & R35 & R15,

634 => OP_MOV & R36 & R16,

635 => OP_MUL & R30 & R20,
636 => OP_MUL & R31 & R21,
637 => OP_MUL & R32 & R22,
638 => OP_MUL & R33 & R23,
639 => OP_MUL & R34 & R24,
640 => OP_MUL & R35 & R25,
641 => OP_MUL & R36 & R26,

Фрагмент програми із використанням машинної команди MATR:

728 => OP_LDIL & x"001e",
729 => OP_MOV & R6 & R5,
730 => OP_LDIL & x"000a",
731 => OP_MOV & R7 & R5,
732 => OP_LDIL & x"0014",
733 => OP_MOV & R8 & R5,
734 => OP_LDIL & x"0007",
735 => OP_MOV & R9 & R5,
736 => OP_MATR & b"0100" & x"000",

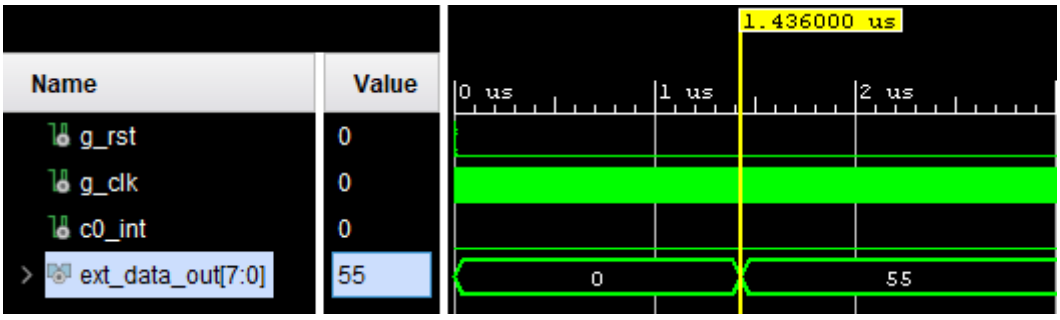

Рисунок 4.40 – Програма для обробки матриці (множення пар елементів) класичним способом

Рисунок 4.41 – Програма для обробки матриці (множення пар елементів) з використанням машинної операції для обробки матриць

5 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

5.1 Вимоги безпеки при виконанні робіт на робочому місці

Виконання дипломної роботи проводиться на електронно-обчислювальній машині, робота з якою пов’язана з певними ризиками.

Вимоги безпеки при виконанні робіт на робочому місці регламентуються: Законом України про охорону праці від 14.10.1992 р. № 49 (чинна редакція від 16.10.2020 р. № 124-IX) [8], Кодексом законів про працю України від 10.12.1971 р. № 322-VIII (чинна редакція від 25.10.2020 р. № 931-IX) [9], Законом України про загальнообов’язкове державне страхування від 23.09.1999 р. № 1105-XIV (чинна редакція від 25.10.2020 р. № 931-IX) [10], наказом Міністерства соціальної політики України про затвердження вимог щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями від 14.02.2018 № 207 [11], наказом Міністерства охорони здоров’я України про затвердження порядків надання домедичної допомоги особам при невідкладних станах від 16.06.2014 № 398 (чинна редакція від 18.01.2019 № z1500-18) [12] та іншими нормативно-правовими актами.

Згідно Закону України про охорону праці [8], охорона праці – це комплекс правових, санітарно-гігієнічних і лікувально-профілактичних заходів і засобів, що спрямовані на збереження життя та здоров’я людини у процесі трудової діяльності. Охорона праці встановлює відносини між працівником і роботодавцем, що регулюють їх права і зобов’язання для створення сприятливих умов праці, що не загрожують життю і здоров’ю працівника. Дія цього Закону поширюється на всіх фізичних та юридичних осіб, що використовують найману працю, та на всіх працюючих.

Державна політика в галузі охорони праці базується на наступних принципах:

– установленню високого пріоритету життя і здоров’я працівників;

– відповідальність роботодавця за створення безпечних і здорових умов праці;

– забезпечення технічного контролю за станом виробництва для надання безпечних та здорових умов праці;

– розв’язання завдань охорони праці на загальнодержавному, регіональному та місцевому рівнях з урахуваннями досягнень у науці та техніці, що сприяють покращенню умов на виробництві;

– соціальний захист працівників при виникненні травм та нещасних випадках;

– адаптація умов виробництва до можливостей працівника з урахуванням його фізичного та психічного стану;

– інформування населення про права працівників у сфері охорони праці;

– використання досвіду інших країн та міжнародних організацій для поліпшення умов і підвищення безпеки праці.

Кодекс законів про працю України [9] регулює трудові відносини працівників і сприяє покращенню стану роботи, умов на виробництві та культурного рівня життя працівників. Також визначаються основні трудові права працівників, гарантується забезпечення громадян працею, визнаються недійсними договори про працю що погіршують становище працівників.

Державою гарантується право громадян України на працю – тобто одержання роботи з оплатою праці не нижче мінімальної зарплати. Створюються умови для підвищення зайнятості населення, спрощується пошук роботи відповідно до здібностей і професійних навиків громадян, та проводиться перепідготовка осіб що змінюють сферу діяльності.

Права та обов’язки роботодавця та працівника регулюються трудовим договором – угодою, за якою працівник зобов’язується виконувати роботу, визначену угодою, а власник підприємства чи уповноважений ним орган чи особа зобов’язується виплачувати працівнику заробітну плату і забезпечити умови праці, необхідні для виконання роботи.

Закон України про загальнообов’язкове державне страхування [10] визначає правові, фінансові та організаційні засади державного соціального страхування, надає гарантії громадянам щодо їх захисту через втрату працездатності, вагітність, нещасні випадки та інші ситуації. Соціальне страхування передбачає внесення працівниками страхових внесків до Фонду соціального страхування України, та виплати страхових

виплат працівникам у випадках, що передбачені законом (травми на підприємстві, профзахворювання тощо).

Фонд соціального страхування України – це некомерційна самоврядна організація, що керує загальнообов'язковим державним соціальним страхуванням від нещасного випадку, у зв'язку з тимчасовою втратою працездатності, проводить збереження страхових внесків, контролює використання коштів та здійснює інші функції згідно із затвердженим статутом.

Роботодавець зобов'язаний:

– у разі настання страхового випадку оплачувати застрахованим особам відповідний вид матеріального забезпечення, страхових виплат та соціальних послуг;

– вести облік страхових внесків і організувати вчасну звітність до Фонду соціального страхування;

– при перевірці діяльності підприємства працівниками Фонду надавати необхідні документи та пояснення;

– інформувати про кожний нещасний випадок та професійне захворювання на підприємстві.

Наказ про затвердження вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями [11] визначає вимоги до робочих місць, оснащених екранними пристроями. Роботодавець повинен забезпечити відповідність робочих місць даним вимогам. Оскільки наказ передбачає тільки мінімальні вимоги, роботодавець може установити більш жорсткі та/або спеціальні умови на робочих місцях. Визначаються технічні й ергономічні вимоги до робочих місць, та організаційні вимоги (обмеження часу роботи з екранними пристроями, відключення пристроїв від електричної мережі після роботи, підтримання мікроклімату, очищення приладів від пилу тощо).

Роботодавець повинен під розписку інформувати працівників про небезпечні та шкідливі фактори, що діють на виробництві. Перерви для відпочинку у працівників проводяться за рахунок тривалості робочої зміни. Проведення медичних оглядів проводиться за рахунок роботодавця, за результатами оглядів роботодавець повинен забезпечити проведення оздоровчих заходів.

Для визначення небезпечних і шкідливих факторів на виробництві, важкості та напруженості трудового процесу роботодавець повинен забезпечити проведення лабораторних досліджень та вжити заходів щодо усунення виявлених ризиків.

Наказ про затвердження порядків надання домедичної допомоги особам при невідкладних станах [12] передбачає перелік документів, що описують порядок надання допомоги при серцевому нападі, при підозрі на пошкодження хребта, травмі голови, отруєнні шкідливими речовинами, при підозрі на інсульт, при падінні з висоти, ураженням струмом тощо.

5.2 Шкідливі виробничі фактори на робочому місці

Робоче місце для виконання дипломної роботи представляє собою комп'ютерний стіл, стілець, та електронну обчислювальну машину з монітором. Кімната освітлюється штучним та природним освітленням, провітрювання проводиться двічі на добу при досить низькій температурі навколишнього середовища, та більшу частину добу – при середній та високих температурах.

Шкідливі фактори, що присутні на робочому місці:

- низький рівень фізичної активності впродовж довгого часу;
- психологічна втома через довготривале розумове навантаження;
- втома зорового апарату через постійне напруження очей, що сфокусовані на одному й тому ж місці;
- випромінювання монітору;
- електромагнітне випромінювання від обладнання;
- шум від роботи обладнання;
- виділення теплоти під час роботи обладнання;
- можливе відбиття штучного чи зовнішнього освітлення від екрану чи клавіатури;
- втома рук під час роботи з клавіатурою та мишкою.

Обмеження шкідливих факторів визначаються: Державними санітарними нормами виробничої загальної та локальної вібрації ДСН 3.3.6.039-99 від 01.12.1999 р. № 39 [13], Санітарними нормами виробничого шуму, ультразвуку та інфразвуку

ДСН 3.3.6.037-99 від 01.12.1999 р. № 37 [14], Санітарними нормами мікроклімату виробничих приміщень ДСН 3.3.6.042-99 від 01.12.1999 р. № 42 [15], ДБН В.2.5-28:2018 «Природне і штучне освітлення» від 01.03.2019 р. [16] та іншими ДСН та ДБН.

Для зниження шкідливого впливу низького рівня фізичної активності рекомендуються короткочасові перерви в роботі для виконання фізичних вправ. Це сприяє покращенню кровообігу і загального самопочуття, а також знижує ймовірність розвитку професійних захворювань програміста-розробника. Рекомендується помірне фізичне навантаження – вправи для шиї, рук, колін, спортивна ходьба з виконанням махів руками, ногами, поворотом тулуба. Можливо також виконання силових вправ – віджимання, присідання, прес.

Для зниження втоми зорового апарату рекомендується перериватися від роботи на виконання вправ для очей – послідовно фокусуватися на деякій дальній точці (наприклад за вікном) і ближній точці, або рухи очима вліво-вправо, вгору-вниз, по часовій стрілці проти, моргання.

Випромінення монітору повинне бути не дуже яскравим і не дуже тусклим, також необхідно освітлювати робоче місце природним і штучним світлом. Детальні вимоги до природного і штучного освітлення наведені у ДБН В.2.5-28:2018 «Природне і штучне освітлення» [16].

Електромагнітне випромінення від комп’ютерного обладнання завжди виділяється під час його роботи. Зазвичай випромінення комп’ютерного обладнання відповідає багатьом міжнародним нормам, про що можна дізнатися із документації до нього. При обладнанні робочого місця слід уникнути закупівлі сумнівної техніки, що може бути підробкою і не відповідати нормам електромагнітного випромінювання. Також, раніше застосовували монітори на базі електро-променевих трубок, що мали більш жорстке випромінення та призводили до іонізації повітря. Вони потребували додаткової вентиляції приміщення, щоб уникати накопичення іонізованого повітря. На сьогодні, такі монітори втратили свою актуальність і застосовуються рідко.

Шум при роботі комп’ютерного обладнання виникає через необхідність активного охолодження деяких деталей (центрального та графічного процесору).

Рівень їх шуму також контролюється міжнародними нормами, що наведені у документації до обладнання. Іноді є можливість збільшити обчислювальну потужність комп'ютеру, збільшивши значення живлення центрального процесору – це призводить до більш високого рівня шуму. Для визначення допустимих рівнів шуму користуються ДСН 3.3.6.037-99 [14].

Виділення теплоти завжди виникає при роботі комп'ютерного обладнання. Однак, звичайні персональні комп'ютери та робочі станції рідко мають потужність більше 600 Ват, тому при дотриманні норм щодо вентиляції житлових/робочих приміщень виділена теплота не буде призводити до дискомфорту працівників.

Для уникнення відблисків світла від монітору чи клавіатури, слід обладнувати робоче місце обладнанням із матовим покриттям, та бажано уникати потрапляння прямих сонячних променів на монітор та клавіатуру. Відблиски заважають роботі користувача комп'ютеру, їх тривалий вплив може шкодити очам через їх велику яскравість.

Втома рук при роботі з клавіатурою виникає через постійне напруження кистей і м'язів рук. Для зниження впливу цього шкідливого фактору слід застосовувати на робочому місці обладнання ергономічної форми та виконувати короткі перерви для виконання гімнастики рук. Загалом, гімнастику рук можна сумістити із загальною гімнастикою, або гімнастикою для очей.

Робоче місце, що використовується для написання дипломної роботи, відповідає нормам мікроклімату, освітленості, шуму та вібрації. Загальна гімнастика, та гімнастика очей і рук виконується у разі втоми відповідної області тіла.

Провітрювання проводиться мінімум два рази на добу. Навколо робочого місця є досить вільного місця, щоб не обмежувати рухів користувача комп'ютеру. Монітор знаходиться напроти обличчя, тож тримати голову постійно нахиленою чи піднятою не потрібно. Клавіатура обладнана ніжками, тому клавіші розміщені більш зручно для рук під деяким кутом.

Тема магістерської роботи – «реалізація та дослідження RISC-процесору з використанням ПЛІС». Реалізація та дослідження процесору проводиться у системі автоматизованого проектування. ПЛІС часто поставляються у вигляді плат без корпусів, з неізольованими контактами – тобто при роботі із ввімкненою платою

можливе ураження електричним струмом. Оскільки реалізація і дослідження роботи процесору проводиться на електронній обчислювальній машині, робота з платою ПЛІС мінімізується – тому зменшується ймовірність ураження електричним струмом при роботі над проектом.

5.3 Дії працівників у надзвичайних ситуаціях

При роботі з електричними приладами взагалі та, зокрема, екранними пристроями, існує ймовірність ураження електричним струмом.

Наказ про затвердження порядків надання домедичної допомоги особам при невідкладних станах [12] визначає порядок надання домедичної допомоги постраждалим при ураженні електричним струмом та блискавкою. Він орієнтований на надання допомоги постраждалим від електротравми не медичними працівниками. Електротравма – це пошкодження тіла, що виникають при ураженні електричним струмом великої сили або ударом блискавки.

Роботодавець зобов’язаний проводити інструктажі з техніки безпеки, на яких працівники можуть дізнатися про ризики на виробництві та рекомендації про план дій при виникненні надзвичайних ситуаціях. Інструктаж проводиться з певною періодичністю, по факту проведення працівники розписуються про ознайомлення з технікою безпеки.

Для надання допомоги застосовується така послідовність дій:

1. Переконатися у відсутності небезпеки для власного життя і здоров’я. При наявності небезпеки слід самому уникнути загрози ураження електричним струмом, а потім перейти до наступного пункту.
2. Якщо постраждалий ще перебуває під дією струму, треба визначити його джерело і вимкнути (або відкинути оголений провід діелектричним матеріалом). Якщо є проблеми з локальним вимикачем, слід відключити струм на щитку електропостачання.
3. Провести огляд постраждалого на наявність свідомості та дихання;
4. Викликати швидку медичну допомогу за номером 103;
5. За відсутності дихання почати проведення серцево-легеневої реанімації. Серцево-легенева реанімація передбачає штучне дихання та непрямий масаж серця.

6. За відсутності свідомості та наявності дихання надати постраждалому більш зручне для дихання положення (лежачи);

7. Накласти на місця опіку чисті, стерильні пов’язки;

8. Забезпечити нагляд за постраждалим до приїзду швидкої допомоги;

9. При погіршенні стану постраждалого повторно зателефонувати до диспетчеру швидкої допомоги.

Інформування працівників щодо надання допомоги при невідкладних станах сприяє зменшенню негативних наслідків при отриманні травм, серцевих нападах, загостренню деяких хронічних захворювань тощо.

ВИСНОВКИ

У дипломній роботі було розроблено та реалізовано на мові VHDL RISC-процесор. Порівняно з аналогами, розроблений процесор має чотири обчислювальні ядра (що рідко зустрічається серед проектів учбових процесорів), та підтримує операцію прискорення виконання однотипних операцій арифметико-логічного пристрою над множиною чисел.

При реалізації на ПЛІС процесор можна використовувати при автоматизації процесів виробництва, для проведення будь-яких обчислень та для зв’язку з периферійними пристроями через порт вводу-виводу.

Науковою новизною даної роботи є реалізація його особливостей (чотири обчислювальні ядра та прискорення однотипних операцій над множиною чисел). Описання та особливості реалізації цих особливостей можуть допомогти у розробці нових, більш оптимізованих або функціональних процесорів. Іншою особливістю є реалізація кінцевого автомату, що має три основні стани, та підстани що використовуються для деяких машинних операцій.

Розроблені елементи, з яких складається процесор та обчислювальна система, були перевірені на працездатність шляхом виконання testbench, з наведенням відповідних результатів.

Для написання програм для процесору засобами мови VHDL та САПР була створена мова асемблеру, що має мнемонічні позначення машинних команд та номерів регістрів.

На розробленому процесорі були виконані програми, написані на мові асемблеру. За їх допомогою досліджена робота процесору і його окремих елементів, та доведена його працездатність. Всі програми знаходяться у постійній пам’яті, при запуску процесору кожне ядро починає виконувати код з певної комірки; адреса комірки визначається на основі номеру ядра процесору. У цій початковій комірці розміщена команда безумовного переходу, що передає управління основній програмі.

Реалізована підтримка апаратних переривань – кожне ядро має вхід переривання, при отриманні одиничного сигналу на цей вхід зберігається контекст процесору та починається обробка переривання. Адреса постійної пам’яті для обробки переривання визначається на основі номеру ядра процесору. По даній адресі

знаходиться команда безумовного переходу до основної програми обробки переривання.

Розробка і моделювання процесору проводилися у САПР від фірми Xilinx. При моделюванні можна побачити значення входів і виходів обчислювальної системи з процесором, а також значення debug-виходів. На debug-виходи виводяться деякі внутрішні сигнали системи, що дозволяють дослідити деталі роботи процесору та обчислювальної системи.

Проект можна покращувати далі, наприклад: додати кеш команд і даних, замінити підтримку одного порту вводу-виводу на більш універсальну підсистему зв’язку із зовнішнім світом, дослідити використовувані набори машинних команд для різних галузей і покращити систему команд тощо.

У проекті проводилося моделювання для ПЛІС від фірми Xilinx сімейства Kintex UltraScale – xcku035-fbva676-3-e.

Результати дипломної роботи апробовані на конференціях [17, 18].

ПЕРЕЛІК ДЖЕРЕЛ

1. Сергиенко А. М., Корнейчук В. И. Микропроцессорные устройства на программируемых логических ИС : Киев : ЧП «Корнейчук», 2005. 108 с.

2. Сергиенко А. М. VHDL для проектирования вычислительных устройств : Киев : ЧП «Корнейчук», ООО «ТИД «ДС», 2003. 208 с.

3. What’s the difference between Von-Neumann and Harvard architectures? URL: https://www.microcontrollertips.com/difference-between-von-neumann-and-harvard-architectures/ (дата звернення: 08.12.2020).

4. Bryan S. Multicore Processors – A Necessity : ProQuest Discovery Guides, 2008. 14 с.

5. Булдигін В. В., Алєксєєва І. В., Гайдей В. О., Диховичний О. О., Коновалова Н. Р., Федорова Л. Б. Лінійна алгебра та аналітична геометрія : навч. посібник / за ред. проф. В. В. Булдигіна. Київ : ТВіМС, 2011. 224 с.

6. Вычислительный модуль 16-разрядного RISC-микропроцессора. URL: https://kanyevsky.kpi.ua/ru/полезные-ip-cores/вычислительный-модуль-16-разрядного-risc-м/ (дата звернення: 08.12.2020).

7. Система команд 8-разрядных RISC микроконтроллеров семейства AVR. URL: http://mega-avr.com.ua/sistema-komand-8-razryadnyx-risc-mikrokontrollerov-semejstva-avr/ (дата звернення: 08.12.2020).

8. Про охорону праці : Закон України від 14.10.1992 р. № 49. Чинна редакція від 16.10.2020 р. № 124-IX.

9. Кодекс законів про працю України : Закон від 10.12.1971 р. № 322-VIII. Чинна редакція від 25.10.2020 р. № 931-IX.

10. Про загальнообов’язкове державне страхування : Закон України від 23.09.1999 р. № 1105-XIV. Чинна редакція від 25.10.2020 р. № 931-IX.

11. НПАОП 0.00-7.15-18 «Вимоги щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями» : затверджені наказом Міністерства соціальної політики України від 14.02.2018 р. № 207. Поточна редакція є чинною.

12. Про затвердження порядків надання домедичної допомоги особам при невідкладним станах : наказ Міністерства охорони здоров'я України від 16.06.2014 р. № 398. Чинна редакція від 18.01.2019 № z1500-18.

13. ДСН 3.3.6.039-99 «Державні санітарні норми виробничої загальної та локальної вібрації»: затв. постановою Головного державного санітарного лікаря України від 01.12.1999 р. № 39. Поточна редакція є чинною.

14. ДСН 3.3.6.037-99 «Санітарні норми виробничого шуму, ультразвуку та інфразвуку»: затв. постановою Головного державного санітарного лікаря України від 01.12.1999 р. № 37. Поточна редакція є чинною.

15. ДСН 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень»: затв. постановою Головного державного санітарного лікаря України від 01.12.1999 р. № 42. Поточна редакція є чинною.

16. ДБН В.2.5-28:2018 «Природне і штучне освітлення»: наказ Міністерства регіонального розвитку, будівництва та житлово-комунального господарства України від 03.10.2018 р. № 264. *Укрархбудінформ*. 2018. 132 с. Поточна редакція є чинною.

17. Новиков А. О. Реалізація та дослідження RISC-процесора з використанням ПЛІС / А. О. Новиков // Всеукраїнська конференція студентів та молодих вчених. – 2020. – С. 6.

18. Новиков А. О. Реалізація та дослідження RISC-процесора з використанням ПЛІС / А. О. Новиков // XIV Міжнародна науково-практична конференція. – 2020.