МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

До захисту допускається
Завідувач кафедри
_____Скарга-Бандурова І.С.
«____»_____ 2018 р.

ДИПЛОМНИЙ ПРОЕКТ (РОБОТА) БАКАЛАВРА

ПОЯСНЮВАЛЬНА ЗАПИСКА

НА ТЕМУ:

Розробка моделей та дослідження ефективності структурних методів компенсації впливу команд переходів в ядрах сучасних процесорів

Освітньо-кваліфікаційний рівень "бакалавр"
Напрям 6.050102– "Комп'ютерна інженерія"

Керівник проекту:	_____ (підпис)	доц. Недзельский Д.О. (ініціали, прізвище)
Консультант з охорони праці:	_____ (підпис)	ст.викл.Критська Я. О. (ініціали, прізвище)
Здобувач вищої освіти:	_____ (підпис)	Костиря Р.Г. (ініціали, прізвище)
Група:		КІ-14бд

Сєвєродонецьк 2018

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет Інформаційних технологій та електроніки
Кафедра Комп'ютерних наук та інженерії
Освітньо-кваліфікаційний рівень бакалавр
Напрям підготовки 6.050102 Комп'ютерна інженерія
(шифр і назва)
Спеціальність _____
(шифр і назва)

ЗАТВЕРДЖУЮ:

Завідувач кафедри _____
_____ І.С. Скарга-Бандурова
«_____» _____2018 р.

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) БАКАЛАВРА

Костиря Роман Геннадійович
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка моделей та дослідження ефективності структурних методів компенсації впливу команд переходів в ядрах сучасних процесорів
керівник проекту (роботи) Недзельский Д. О., к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від "14" 05 2018р. №
2. Термін подання студентом роботи _____
3. Вихідні дані до роботи матеріали переддипломної практики

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Огляд об'єкту дослідження. Розробка моделей, отримання аналітичних результатів. Розробка програми перевірки. Дослідження отриманих результатів. Охорона праці. Висновки.

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

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	ст.викл. кафедри КНІ Критська Я.О.		

7. Дата видачі завдання _____

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Огляд літератури з теми ДП і постановка задачі	14.05.18-19.05.18	
2	Дослідження матеріалів	20.05.18-25.05.18	
3	Розрахунки	26.05.18-02.06.18	
4	Дослідження результатів	03.06.18-06.06.18	
5	Розробка розділу охорона праці	07.06.18-09.06.18	
6	Оформлення електронних плакатів	10.06.18-12.06.18	
7	Оформлення пояснювальної записки	13.06.18-15.06.18	

Здобувач вищої освіти _____ Костиря Р.Г.
(підпис) (прізвище та ініціали)

Керівник _____ Недзельский Д.О.
(підпис) (прізвище та ініціали)

Р Е Ф Е Р А Т

Пояснювальна записка до дипломного проекту (роботи) бакалавра: 83с., 7 рис., 11 табл., 20 бібліографічних джерел посилань , 2 додатки.

Об'єкт розробки: структури ядер сучасних процессорів.

Мета роботи: розробка моделей та дослідження ефективності структурних методів компенсації впливу команд переходів в ядрах сучасних процесорів

В проекті виконано:

1. Розглянуті та проаналізовані структури ядер сучасних процесорів; типи перешкод, причини їх виникнення та методи боротьби з ними; методи підвищення продуктивності процесорів.
2. Розроблені моделі ядер процесора без блоку передбачення переходів та з блоком передбачення переходів.
3. Отримані аналітичні вирази для коефіцієнтів використання пристроїв.
4. Проведено дослідження структур ядра:
   без блоку передбачення переходів;
   з блоком передбачень і без буфера продешифрованих команд;
   з блоком передбачень і з буфером продешифрованих команд;
5. Проведено порівняння ефективності різних структурних рішень по компенсації негативного впливу команд переходів на продуктивність ядра. Встановлено що, структура ядра з блоком передбачення переходів і з блоком продешифрованих команд є найбільш ефективною.

Практичне значення, галузь застосування роботи: Результати дослідження можуть бути корисні при оптимальному виборі структури ядра з урахуванням співвідношення як показників ефективності, так і апаратних витрат на реалізацію.

Ключові слова: ЯДРО, ПРОЦЕСОР, КОНВЕЄР, ПЕРЕШКОДИ, БЛОК ПЕРЕДБАЧЕННЯ, КОЕФІЦІЄНТ ВИКОРИСТАННЯ, ЕФЕКТИВНІСТЬ

Умови одержання дипломного проекту: СНУ ім. В. Даля, пр. Центральний 59-А,. м. Сєвєродонецьк, 93400.

ЗМІСТ

ВСТУП

Ця робота присвячена розробці моделей та методик дослідження аналітичними методами ефективності структурних методів компенсації впливу команд умовних переходів на продуктивність ядер конвеєрних процесорів.

Для підвищення продуктивності ядер сучасних процесорів широко використовується конвеєрний принцип. Суть його полягає в тому, що процес виконання команди розбивається на n етапів (ступенів), кожен з яких виконує окремий набір функцій і передає результат наступному етапу (ступені), після чого приступає до обробки наступної команди. Як правило, ступені конвеєра розділені буферами у вигляді одного або декількох регістрів.

В ідеальному конвеєрі час роботи всіх ступенів однаковий і відсутні «перешкоди» роботі конвеєра.

В реальних конвеєрах не вдається забезпечити однакові часи роботи всіх ступенів, а також уникнути впливу «перешкод».

Це пояснюється перш за все випадковим характером процесів обробки і виконання команд в ядрі. Наприклад, неможливо точно передбачити час звернення в підсистему пам'яті в разі, якщо в ядрі реалізовані кеш-пам'яті. Часи звернення до кеш-пам'яті верхніх рівнів, не кажучи вже про час звернення до оперативної пам'яті, значно перевищують часи виконання операцій в обчислювальних функціональних пристроях. Час виконання операції ділення чисел істотно більше в порівнянні з іншими обчислювальними операціями т.д.

Крім того, плавному перебігу конвеєра команд в ядрі заважають різні перешкоди, відомі як структурні конфлікти, залежності за даними, залежності з управління.

Структурні конфлікти, це суто проблеми структури ядра процесора. Вони не мають відношення до типу виконуваних програм. Методи вирішенення структурних конфліктів досить прості - розмножити загальні ресурси.

Залежності за даними та по управлінню визначаються властивостями виконуваних програм.

Команда з номером j є залежною за даними від команди з номером i (j більше i), якщо результат команди з номером i є операндом для команди з номером j.

Основними методами компенсації негативного впливу на продуктивність конвеєра залежностей за даними є використання технологій перейменування регістрів і позачергового виконання команд.

Команди умовного переходу за певною ознакою, яка виробляється, як правило, попередньою командою, не можна виконувати до тих пір, поки не з'явиться необхідна ознака, так як пристрій вибірки команд не знає адреси наступної команди.

1 АНАЛІЗ ТЕОРЕТИЧНОЇ ІНФОРМАЦІЇ

1.1 Структура ядер сучасних процесорів

1.1.1 Ядро процесора

Ядро процесора - це його основна частина, яка містить всі функціональні блоки і здійснює виконання всіх логічних і арифметичних операцій.

На рис. 1.1 наведена структурна схема пристрою ядра процесора. Як видно на рисунку, кожне ядро процесора складається з декількох функціональних блоків:

1. блоку вибірки інструкцій;
2. блоків декодування інструкцій;
3. блоків вибірки даних;
4. керуючого блоку;
5. блоків виконання інструкцій;
6. блоків збереження результатів;
7. блоки роботи з перериваннями;
8. ПЗУ, що містить мікрокод;
9. набору регістрів;
10. лічильника команд.

Блок вибірки інструкцій здійснює зчитування інструкцій за адресою, вказаною в лічильнику команд. Зазвичай, за такт він зчитує кілька інструкцій. Кількість зчитувальних інструкцій обумовлена кількістю блоків декодування, так як необхідно на кожному такті роботи максимально завантажити блоки декодування. Для того щоб блок вибірки інструкцій працював в оптимальному режимі, в ядрі процесора є блок передбачення переходів.

Блок передбачення переходів намагається визначити, яка послідовність команд буде виконуватися після здійснення переходу. Це необхідно, щоб після умовного переходу максимально навантажити конвеєр ядра процесора.

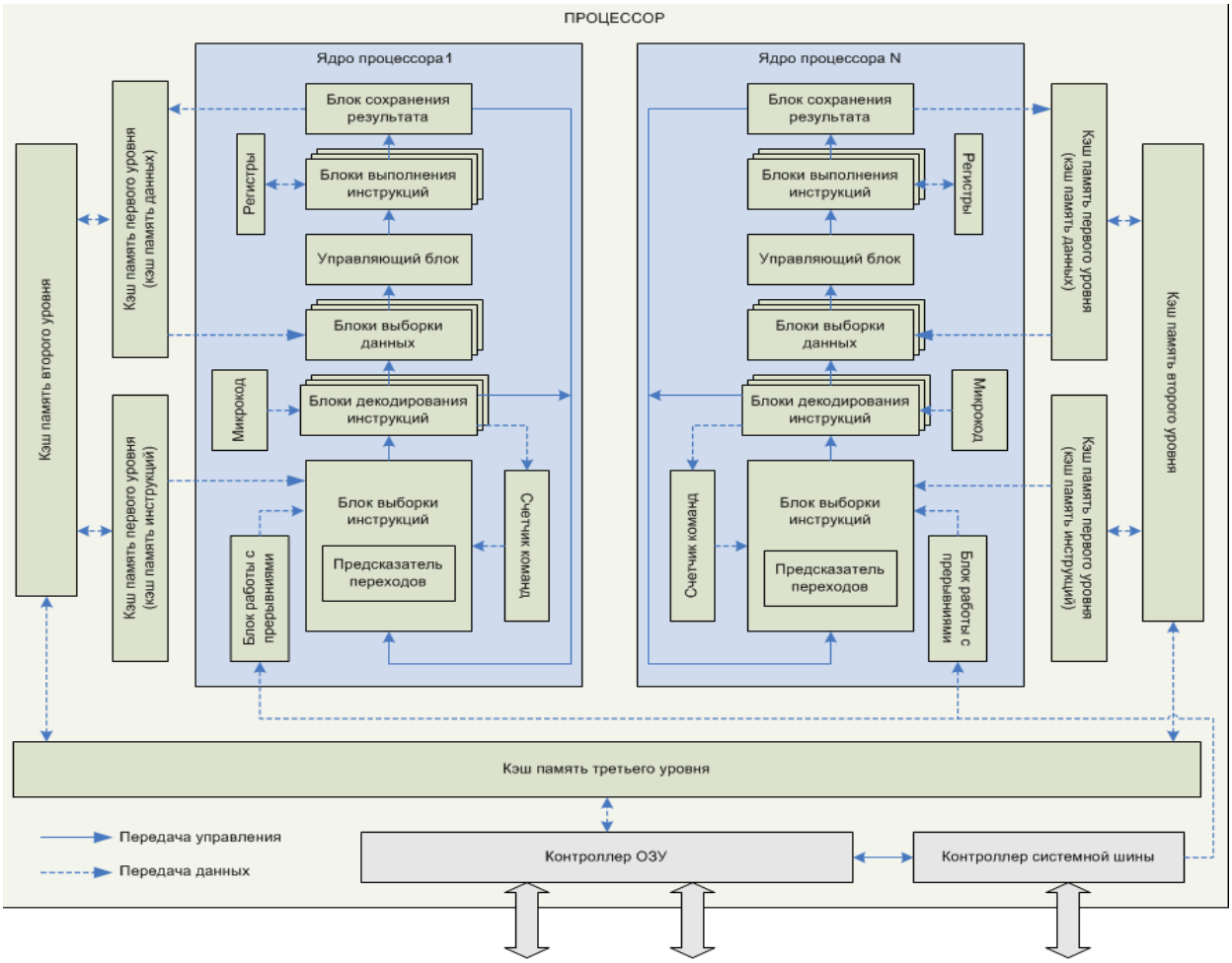


Рисунок 1.1 - Структура сучасних процесорів

Блоки декодування, як зрозуміло з назви, - це блоки, які займаються декодуванням інструкцій, тобто визначають, що треба зробити ядру процесора, і які додаткові дані потрібні для виконання інструкції. Завдання це для більшості сучасних комерційних процесорів, побудованих на базі концепції CISC, - дуже складна. Справа в тому, що довжина інструкцій і кількість операндів - нефіксовані, і це сильно ускладнює життя розробникам ядер процесорів і робить процес декодування нетривіальним завданням.

Часто окремі складні команди доводиться замінювати мікрокодом - серією простих інструкцій, в сукупності виконуючих ту саму дію, що і одна складна інструкція. Набір мікрокоду прошитий в ПЗУ, вбудованому в ядро процесора. До того ж мікрокод спрощує розробку ядра процесора, так як відпадає потреба в створенні складних блоків ядра для виконання окремих команд, так і виправити мікрокод набагато простіше, ніж усунути помилку в функціонуванні блоку.

В сучасних процесорах, зазвичай, буває 2-4 блоки декодування інструкцій.

Блоки вибірки даних здійснюють вибірку даних з КЕШ-пам'яті або ОЗУ, необхідних для виконання поточних інструкцій. Зазвичай, кожне процесорне ядро містить кілька блоків вибірки даних. Наприклад, в процесорах Intel Core використовується по два блоки вибірки даних для кожного ядра.

Керуючий блок на підставі декодованих інструкцій управляє роботою блоків виконання інструкцій, розподіляє навантаження між ними, забезпечує своєчасне і правильне виконання інструкцій. Це один з найбільш важливих блоків ядра процесора.

Блоки виконання інструкцій включають в себе кілька різнотипних блоків:

ALU - арифметично логічний пристрій;

FPU - пристрій з виконання операцій з плаваючою точкою;

Блоки для обробки розширення наборів інструкцій. Додаткові інструкції використовуються для прискорення обробки потоків даних, шифрування і дешифрування, кодування відео і так далі. Для цього в ядро процесора вводять додаткові регістри та набори логіки. На даний момент найбільш популярними розширеннями наборів інструкцій є:

MMX (Multimedia Extensions) - набір інструкцій, розроблений компанією Intel, для прискорення кодування і декодування потокових аудіо і відео-даних;

SSE (Streaming SIMD Extensions) - набір інструкцій, розроблений компанією Intel, для виконання однієї і тієї ж послідовності операцій над безліччю даних з розпаралелюванням обчислювального процесу. Набори команд постійно вдосконалюються, і на даний момент є версії: SSE, SSE2, SSE3, SSSE3, SSE4.

ATA (Application Targeted Accelerator) - набір інструкцій, розроблений компанією Intel, для прискорення роботи спеціалізованого програмного забезпечення і зниження енергоспоживання при роботі з такими програмами. Ці інструкції можуть використовуватися, наприклад, при розрахунку контрольних сум або пошуку даних.

3DNow - набір інструкцій, розроблений компанією AMD, для розширення можливостей набору інструкцій MMX;

AES (Advanced Encryption Standard) - набір інструкцій, розроблений компанією Intel, для прискорення роботи додатків, що використовують шифрування даних за однойменним алгоритмом.

Блок збереження результатів забезпечує запис результату виконання інструкції в ОЗУ за адресою, вказаною в оброблюваній інструкції.

Блок роботи з перериваннями. Робота з перериваннями - одна з найважливіших задач процесора, що дозволяє йому своєчасно реагувати на події, переривати хід роботи програми і виконувати необхідні дії. Завдяки наявності переривань, процесор здатний до псевдопараллельної роботи, тобто до, так званої, багатозадачності.

Обробка переривань відбувається наступним чином. Процесор перед початком кожного циклу роботи перевіряє наявність запиту на переривання. Якщо є переривання для обробки, процесор зберігає в стек адресу інструкції, яку він повинен був виконати, і дані, отримані після виконання останньої інструкції, і переходить до виконання функції обробки переривання.

Після закінчення виконання функції обробки переривання, з стека зчитуються збережені в ньому дані, і процесор відновлює виконання відновленої задачі.

Регістри - надшвидка оперативна пам'ять (доступ до регістрів в кілька разів швидше доступу до КЕШ-пам'яті), що входить до складу процесора, для тимчасового зберігання проміжних результатів виконання інструкцій. Регістри процесора діляться на два типи: регістри загального призначення і спеціальні регістри.

Регістри загального призначення використовуються при виконанні арифметичних і логічних операцій, або специфічних операцій додаткових наборів інструкцій (MMX, SSE і т.д.).

Регістри спеціального призначення містять системні дані, необхідні для роботи процесора. До таких регістрів належать, наприклад, регістри керування, регістри системних адрес, регістри налагодження і т.д. Доступ до цих регістрів жорстко регламентований.

Лічильник команд - регістр, що містить адресу команди, яку ядро процесора почне виконувати на наступному такті роботи.

1.2 Методи підвищення продуктивності ядер процесорів

1.2.1 Оцінка продуктивності ядра процесора

Однією з оцінок продуктивності комп'ютера є кількість команд, які виконуються в одиницю часу (за одну секунду)

$$Продуктивність = \frac{Nкоманд}{Tвиконання} ,$$

де:

N команд - кількість виконаних команд;

T виконання - час виконання N команд.

Як відомо, процесор працює на певній частоті, яка є однією з її найважливіших технічних характеристик. За кожен такт, тобто проміжок часу, зворотний частоті, ядро процесора виконує певну кількість команд. Тому замість кількості команд програми, які виконуються за одиницю часу, зручніше розглядати кількість команд програми, які виконуються за один такт ядра процесора (Instruction Per Clock, IPC). IPC залежить від структури процесора, технології виробництва, яка визначає мінімальні розміри використовуваних транзисторів та їх швидкодію, від оптимізації програми до архітектури процесора.

Переписавши вираз для продуктивності комп'ютера у вигляді добутку кількості команд, які виконуються за один такт процесора, на кількість тактів процесора за одиницю часу (частота процесора, *F),* отримаємо:

$$Продуктивність = \frac{Nкоманд}{Kтактів} * \frac{Kтактів}{Tвиконання} = IPC * F ,$$

де: N команд - кількість виконаних команд;

T виконання - час виконання N команд;

Ктактів - час виконання N команд в тактах;

F - частота;

IPC - кількість виконаних за такт команд.

Як видно, продуктивність комп'ютера прямо пропорційна як частоті, так і кількості команд, які виконуються за один такт. З цієї формули також випливає,

що існує два принципово різних підходи до збільшення продуктивності комп'ютера. Перший з них полягає в збільшенні частоти, а другий - у збільшенні кількості команд, які виконуються за такт (IPC).

В ідеалі слід прагнути до використання обох підходів одночасно, тобто, продуктивність комп'ютера збільшувати як за рахунок збільшення частоти, так і збільшення кількості команд, які виконуються за один такт IPC. IPC і максимально можлива частота комп'ютера взаємопов'язані.

Перед розробниками комп'ютерів постає питання «Чому віддати перевагу»?

1.2.2 Про збільшення частоти

Найбільш прямий шлях до збільшення продуктивності комп'ютера, це збільшення частоти. Якби, наприклад, вдалося всі затримки в комп'ютері скоротити в **к** раз, то це призвело б до збільшення швидкодії в таке ж число раз. В останні роки були досягнуті величезні успіхи в створенні швидкодіючої елементної бази та відповідних методів монтажу. Очікується подальший прогрес, заснований на використанні нових технологій і зниження розмірів пристроїв. Цей шлях, однак, має ряд обмежень:

1. Для певного рівня технології забезпечується певний рівень швидкодії елементної бази: як тільки він виявився досягнутим, подальше збільшення швидкодії супроводжується величезними витратами аж до досягнення того порогу, за яким уже немає технологій, що забезпечують більшу швидкодію.

2. Більш швидкодіючі елементи зазвичай мають меншу щільність монтажу, що, в свою чергу, вимагають більш довгих сполучних зв'язків між компонентами і платами і, отже, призводить до збільшення затримок (за рахунок з'єднань) і зменшення виграшу в продуктивності.

3. Більш швидкодіючі елементи зазвичай розсіюють більше тепла. Тому потрібні спеціальні заходи щодо відведення тепла, що ще більше знижує щільність монтажу і, отже, швидкодію. Для того щоб уникнути додаткових витрат, затримок за рахунок з'єднань і збільшення розсіювання тепла, доцільно, мабуть, застосовувати швидкодіючі елементи не скрізь, а тільки в тих частинах, які відповідають «вузьким місцям». Однак шлях збільшення швидкодії елементів має свої обмеження і може наступити момент, коли стане необхідним більш доцільно використовувати для збільшення швидкодії інші способи.

До деякого часу домінуючим способом збільшення продуктивності комп'ютера було збільшення частоти. Продуктивність комп'ютерів буквально ототожнювалася з їх частотою, а структура створювалася саме з розрахунком забезпечення її максимально можливого збільшення. Прикладом цього може служити структура NetBurst, покладена в основу процесорів сімейства Pentium 4, особливістю якої був супердовгий конвеєр, що дозволяло нарощувати частоту.

1.2.3 Негативні наслідки збільшення частоти

За п'ять років випуску процесорів сімейства Pentium 4 їх тактова частота була збільшена більш ніж в три рази. Стартувавши з позначки трохи більше 1 ГГц, за п'ять років вона досягла 3,8 ГГц.

Здавалося б, якщо збільшення частоти являє собою досить ефективний засіб для збільшення продуктивності процесорів, що заважає і далі рухатися в цьому ж напрямку?

Справа в тому, що збільшення частоти процесора призводить до зростання його енергоспоживання і, як наслідок, до підвищення тепловиділення.

Залежність споживаної процесором потужності від його частоти можна представити наступною формулою:

$$Потужність = CU^2F$$

де: F - тактова частота;
U - напруга живлення;
C - динамічна ємність.

Емпіричним шляхом встановлено, що при збільшенні частоти на 20% продуктивність процесора зростає приблизно на 13%. Справа в тому, що, незважаючи на теоретичну прямолінійну залежність між продуктивністю процесора і його частотою, в реальності вона не є строго пропорційною. При цьому споживана процесором потужність зростає на 73%. При зменшенні частоти процесора на 20% продуктивність зменшується на 13%, а споживана потужність - на 49%. Цей приклад наочно демонструє, що збільшення частоти

призводить до явного дисбалансу між приростом продуктивності і споживаною потужністю.

У гонитві за продуктивністю, яка, як уже зазначалося, в процесорах сімейства Pentium 4 забезпечувалася головним чином збільшенням частоти, останні версії процесорів на структурі NetBurst досягли рівня енергоспоживання в 130 Вт. Здавалося б, нічого страшного - адже це відповідає одній-двом лампам розжарювання, і будь-яка люстра в квартирі споживає більше електроенергії. Однак проблема полягає в тому, що ця потужність виділяється процесором у вигляді тепла в дуже обмеженому просторі, що призводить до його нагрівання. І якщо для лампи розжарювання такий нагрів не критичний, то для процесора все не так просто. Процесори настільних персональних комп'ютерів можуть зберігати нормальну працездатність аж до температури 70 ° С. Це означає, що при такому високому енергоспоживанні необхідно забезпечити ефективне відведення тепла, для чого використовуються вентилятори (кулери). Проблема, однак, полягає в тому, що лише деякі потужні пристрої дозволяють охолоджувати процесори з більш високим енергоспоживанням, причому ці вентилятори досить гучні.

З урахуванням того, що енергоспоживання процесорів сімейства Pentium 4 вже досягло свого критичного значення, подальше зростання їх продуктивності стало неможливим в рамках існуючої структури NetBurst. Вона повністю вичерпала свої потенційні можливості і створила свого роду тупикову ситуацію в подальшому розвитку процесорів.

У 2000 р. фірма Intel обіцяла досягти в своїх процесорах до 2010 г. рубежу в 10 Ггц. Мрії залишилися мріями, а те, що мало статися, сталося. Зараз про частоту процесорів як про засіб збільшення продуктивності вже не говорять.

Тому основне завдання, яке ставиться при конструюванні сучасних процесорів, - досягнення не просто максимально можливої продуктивності будь-якими засобами, а високого рівня продуктивності при забезпеченні енергоспоживання на прийнятному рівні. У зв'язку з цим доречно розглянути ще одну важливу характеристику процесора - енергетичну ефективність.

1.2.4 Енергетична ефективність ядра процесора

Якщо абсолютну продуктивність ядра процесора прийнято вимірювати в кількості програмних команд, які виконуються в одиницю часу (Instruction Per Second, IPS), то енергетичну ефективність ядер процесорів можна характеризувати величиною, чисельно рівною середній кількості використаної енергії, що припадає на одну виконану команду (Energy Per Instruction, EPI). EPI вимірюється в джоулях (Дж) і визначається за такою формулою:

$$EPI = \frac{Енергія(Дж)}{Ккоманд}.$$

EPI неважко пов'язати і з іншою часто використовуваною характеристикою енергетичної ефективності ядра процесора, іноді званою оптимізованою продуктивністю ядра процесора і яка визначається як продуктивність ядра процесора в розрахунку на кожен ват споживаної потужності (Performance Per Watt):

$$\frac{Продуктивність}{Потужність} = \frac{Ккоманд / Час}{Енергія / Час} = \frac{Ккоманд}{Енергія} = \frac{1}{EPI}$$

Відповідно

$$EPI = \frac{Потужність}{Продуктисть}.$$

Таким чином, енергетичну ефективність ядра процесора, EPI, можна трактувати як споживану потужність в розрахунку на одиницю продуктивності, якщо остання вимірюється в кількості команд, які виконуються за одиницю часу.

Енергетична ефективність процесора, так само як і оптимізована продуктивність, залежить від співвідношення двох величин: швидкості виконання ядром процесора команд і споживаної потужності. При цьому ні енергетична ефективність ядра процесора, ні оптимізована продуктивність не залежать від часу виконання команд. Саме тому і EPI, і оптимізована

продуктивність є зручними величинами для оцінки енергетичної ефективності ядра процесора в умовах, коли основним критерієм для порівняння ядер процесорів служить продуктивність в заданому діапазоні енергоспоживання.

Енергетична ефективність ядер процесорів залежить від таких факторів, як конструкція ядра процесора, технологічний процес виробництва і напруга живлення.

Користувачеві хотілося б мати процесор, який, з одного боку, був би високопродуктивним, а з іншого - мав би низьке значенням EPI, тобто був би енергоефективним.

1.2.5 Інші способи збільшення продуктивності

Зменшення часу реалізації основних операцій. Для того щоб зменшити час виконання операцій, необхідно використовувати алгоритми, які приводили б до швидкодіючих комбінаційних схем і вимагали невеликого числа циклів.

Скорочення витрат часу при зверненнях до підсистеми пам'яті. Звичайні підходи тут складаються, по-перше, в розширенні шляхів доступу за рахунок розбиття оперативної пам'яті на модулі, звернення до яких може здійснюватися одночасно; по-друге, в застосуванні багаторівневої надшвидкодіючої кеш-пам'яті і, нарешті, в збільшенні числа внутрішніх регістрів в процесорі.

Використання всіх перерахованих способів тісно пов'язане з організацією ядер процесорів комп'ютерів.

Зменшення тривалості виконання однієї команди за рахунок тимчасового перекриття різних її фаз. Наприклад, обчислення адреси, за якою потрібно записати результат, може бути виконано одночасно з самою операцією. Цей підхід вимагає, зрозуміло, додаткового обладнання, оскільки модулі оперативної пам'яті не можуть бути одночасно задіяні в суміщених фазах. Збільшення швидкодії, яке можна при цьому досягти, залежить від формату (складу) команди, оскільки саме він визначає наявність незалежних фаз.

Конвейеризація. Конвеєрна обробка передбачає роздільне виконання деякої операції в кілька етапів (кілька ступенів) з передачею даних від одного етапу до наступного. Продуктивність при цьому зростає завдяки тому, що одночасно на різних ступенях конвеєра виконуються декілька операцій.

Конвейеризація ефективна тільки тоді, коли завантаження конвеєра близьке до повного, а швидкість подачі нових операндів відповідає максимальній продуктивності конвеєра. Якщо відбувається затримка, то паралельно буде виконуватися менше операцій і сумарна продуктивність знизиться.

Суперскалярність. Суть цього методу полягає в тому, щоб генерувати протягом відрізка часу одночасно декілька команд. Цей підхід відрізняється від того, який реалізований в звичайному послідовному ядру процесора, коли команди виконуються строго послідовно одна за одною. Паралельний підхід призводить до різних варіантів структури в залежності від способу, за яким здійснюється завдання черговості проходження команд і керування їх виконанням. Розпаралелювання дозволяє значно збільшити продуктивність ядер процесорівв при вирішенні широкого класу прикладних завдань.

Використання конвеєрного принципу та інших структурних методів дозволило істотно підвищити продуктивність ядер процесорів. Однак можливості традиційних структурних методів прискорення виконання команд в одному ядрі процесора на сучасному етапі практично вичерпані.

Отже, єдиний напрямок, що веде до подальшого підвищення продуктивності - це більш повне використання паралелізму при обробці даних.

При послідовному виконанні всіх етапів чергової команди час її виконання дорівнює сумі часів виконання всіх послідовних етапів.

Дуже часто така швидкодія комп'ютера не задовольняє споживача.

1.3 Типи перешкод

1.3.1 Структурні конфлікти

Структурні конфлікти виникають в тому випадку, коли апаратні засоби ядра процесора не можуть підтримувати всі можливі комбінації команд в режимі одночасного виконання з суміщенням.

Причиною структурних конфліктів може бути не повністю конвеєрна структура ядра процесора, при якій деякі ступені окремих команд виконуються більше одного такту.

Нехай етап виконання (i + 1)-ї команди займає 3 такти. Тоді діаграма роботи конвеєра буде мати вигляд, представлений в таблиці 1.1 .

Таблиця 1.1 - Конвеєрна обробка при затримці (i + 1)-ї команди на етапі ВП

команда	**такт**								
	1	2	3	4	5	6	7	8	9
i	ВК	ДК	ЧО	ВП	ЗР				
i +1		ВК	ДК	ЧО	ВП	ВП	ВП	ЗР	
i +2			ВК	ДК	ЧО			ВП	ЗР
i +3				ВК	ДК	ЧО			ВП
i + 4					ВК	ДК	ЧО		

В цьому випадку в роботі конвеєра виникають так звані "бульбашки" (порожні клітинки в таблиці) в обробці (i + 2) -ї команди і наступних за нею починаючи з такту 6, які знижують продуктивність ядра процесора.

Якщо якийсь блок конвеєра вносить затримку, то гальмується робота всього конвеєра. Утворена при цьому "бульбашка" повинна пройти від місця свого виникнення до самого кінця конвеєра (якщо, наприклад, виникла затримка на ступені зчитування команди, то в наступному такті блок декодування від нього нічого не отримає, а через 3 такти, відповідно, блок збереження результатів нічого не отримає від блоку виконання). Таким чином, швидкість конвеєра визначається швидкістю самої повільної його ступені.

Цієї ситуації можна було б уникнути двома способами.

Перший передбачає збільшення часу такту до такої величини, яка дозволила б всі етапи будь-якої команди виконувати за один такт. Однак при цьому істотно знижується ефект конвеєрної обробки, так як всі етапи всіх команд будуть виконуватися значно довше, в той час як зазвичай декількох тактів вимагає виконання лише окремих етапів дуже невеликої кількості команд.

Другий спосіб передбачає використання таких апаратних рішень, які дозволили б значно знизити витрати часу на виконання дії, що приводить до появи "бульбашок" (наприклад, використовувати матричні схеми множення). Але це призведе до ускладнення схеми ядра процесора і скорочення на кристалі місця для реалізації інших, функціонально більш важливих вузлів. Так як представлена в табл. 1.1 ситуація виникає при реалізації команд, що відносно рідко зустрічаються в програмі, то зазвичай розробники ядер

процесорів шукають компроміс між збільшенням тривалості такту і ускладненням того чи іншого пристрою ядра процесора.

1.3.2 Конфлікти за даними

Конфлікти за даними виникають у випадках, коли виконання однієї команди залежить від результату виконання попередньої команди.

Під час обговорення цих конфліктів будемо припускати, що команда і передує команді j.

Існує кілька типів конфліктів за даними.

Конфлікти типу RAW (Read After Write - читання після запису)

Команда з номером j намагається прочитати операнд перш, ніж команда з номером і запише на це місце свій результат. При цьому команда з номером j може отримати некоректне старе значення операнда.

Проілюструємо цей тип конфлікту на прикладі виконання наступних команд

i) ADD R 1, R 0; (R 1 = R 1 + R 0)

i + 1 = j) SUB R2, R1; (R2 = R2-R1).

Команда з номером і змінить стан регістра R1 в такті 5. Але команда з номером i + 1 повинна прочитати значення операнда R1 в такті 4. Якщо не прийняти спеціальних заходів, то з регістра R1 буде прочитано значення, яке було в ньому до виконання команди з номером і.

Конфлікти типу **RAW** обумовлені саме конвеєрної організацією обробки команд. Вони називаються істинними взаємозалежностями.

Зменшення впливу конфлікту типу **RAW** забезпечується методом, який називається пересиланням чи просуванням даних, обходом, іноді закороткою.

В цьому випадку результати, отримані на виходах виконавчих пристроїв, крім входів приймача результату передаються також на входи всіх виконавчих пристроїв ядра процесора.

Якщо пристрій керування виявляє, що отриманий будь-якої командою результат потрібен одній з наступних команд в якості операнда, то він відразу

ж, паралельно із записом в приймач результату, передається на вхід виконавчого пристрою для використання за допомогою такої команди.

Головною причиною двох інших типів конфліктів за даними є можливість неупорядкованого виконання команд в сучасних ядрах процесорів, тобто, виконання команд не в тому порядку, в якому вони записані в програмі (помилкові взаємозалежності).

Конфлікти типу WAR (Write After Read - запис після читання). Команда з номером j намагається записати результат в приймач, перш ніж він считається звідти командою з номером i. При цьому команда з номером i може отримати некоректне нове значення операнда:

i) ADD R1, R0 (R1 = R1 + R0);

i + 1 = j) SUB R0, R2 (R0 = R0-R2).

Цей конфлікт виникне в разі, якщо команда з номером j внаслідок неупорядкованого виконання завершиться раніше, ніж команда з номером i прочитає старий вміст регістра R0.

Конфлікти типу WAW (Write After Write - запис після запису). Команда з номером j намагається записати результат в приймач, перш ніж в цей же приймач буде записаний результат виконання команди з номером i, тобто запис закінчується в неналежному порядку, залишаючи в приймальнику результату значення, записане командою з номером i:

i) ADD R 1, R 0 (R 1 = R 1 + R 0);
. . .
j) SUB R 1, R 2 (R 1 = R 1 R 2).

Усунення конфліктів за даними типів **WAR** і **WAW** досягається шляхом відмови від неврегульованого виконання команд, але частіше за все шляхом введення буфера відновлення послідовності команд.

Частина конфліктів за даними може бути знята спеціальною методикою планування компілятора. В найпростішому випадку компілятор просто планує розподіл команд в базовому блоці.

Базовий блок **це** собою лінійна ділянка послідовності команд програми з одним входом і одним виходом, в якій відсутні внутрішні команди переходу. Оскільки в такому блоці кожна команда буде виконуватися, якщо виконується перша з них, - можна побудувати граф залежностей цих команд і впорядкувати їх так, щоб мінімізувати призупинення конвеєра. Ця техніка називається плануванням завантаження конвеєра або плануванням потоку команд.

Наприклад, для оператора A = B + C компілятор, швидше за все, згенерує наступну послідовність команд:

1. Rb = B .
2. Rc = C .
3. Ra = Rb + Rc .
4. A = Ra .

Очевидно, виконання 3-ої команди повинно бути припинено до тих пір, поки не стануть доступними операнди B і C, що надходять з підсистеми пам'яті.

Для цього найпростішого прикладу компілятор ніяк не може поліпшити ситуацію, проте в ряді більш загальних випадків він може реорганізувати послідовність команд так, щоб уникнути призупинень конвеєра.

Нехай, наприклад, є послідовність операторів:

A = B + C;
D = E - F.

В цьому випадку компілятор може згенерувати наступну послідовність команд, виконання якої не призведе до призупинення конвеєра:

1. Rb = B .
2. Rc = C .
3. Re = E .
4. Ra = Rb + Rc .
5. Rf = F .
6. A = Ra .
7. Rd = Re - Rf .
8. D = Rd .

Зауважимо, що використання різних регістрів для першого і другого компілюємого оператора було достатньо важливим для реалізації такого правильного планування. Зокрема, якщо змінна E була б завантажена в той же самий регістр, що і B або C, таке планування не було б коректним. У загальному випадку планування конвеєра може вимагати збільшеної кількості регістрів.

Для простих конвеєрів стратегія планування на основі базових блоків цілком задовільна, проте коли конвейеризація стає більш інтенсивною і дійсні затримки конвеєра ростуть, потрібні більш складні алгоритми планування.

Найчастіше залежність за даними не є необхідною - просто так уже повелося у програмістів: чим менше змінних (і регістрів в програмах на асемблері) використовує програма - тим краще. В результаті часто виходить, що вся програма використовує один-два регістра з залежністю за даними мало не в кожній парі команд.

1.3.3 Конфлікти з управління

Конфлікти з управління виникають при виконанні команд переходів та інших команд, що змінюють значення лічильника команд (регістра номера команди).

Суть конфліктів цієї групи найбільш зручно проілюструвати на прикладі команд умовного переходу. Нехай (i + 1)-а команда є командою умовного переходу, яка формує адресу наступної команди в залежності від результату виконання і-ї команди.

Команда з номером і завершить своє виконання в такті 5. У той же час команда умовного переходу вже в такті 3 повинна прочитати необхідні їй ознаки, щоб правильно сформувати адресу наступної команди. Якщо конвеєр має велику довжину (наприклад, 20 ступенів), то проміжок часу між формуванням ознаки результату і тактом, де він аналізується, може бути ще більше.

Найпростіший спосіб вирішення цієї ситуації - використання так званого методу вичікування. Він полягає в блокуванні (заморожуванні) операцій в конвеєрі шляхом блокування виконання будь-якої команди, наступної за командою умовного переходу, до тих пір, поки не стане відомим напрямок

переходу. Привабливість такого рішення полягає в його простоті. Головний недолік - різке зменшення переваг конвеєрної обробки. В інженерних задачах приблизно кожна шоста команда є командою умовного переходу, тому призупинення конвеєра при виконанні команд переходів до визначення справжнього напрямку переходу істотно позначаться на продуктивності процесора.

Можна трохи поліпшити цю ситуацію, використовуючи схему "затриманих переходів". При цьому на стадії компіляції компілятор таким чином створює одержувану об'єктну програму, щоб зробити команди, які йдуть за командою переходу, дійсними і корисними (рис. 1.2).

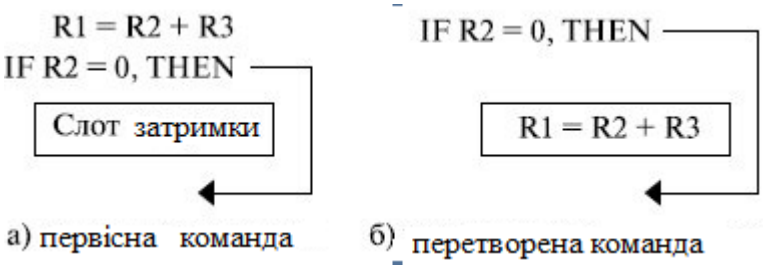


Рисунок 1.2 - Організація затриманого переходу

Слот затримки (часовий інтервал) заповнюється незалежною командою, яка знаходиться перед командою умовного переходу. Ця команда виконується за той час, поки ядро процесора формує справжню умову, за якимю має бути визначено напрям виконання програми. Однією з основних труднощів в цьому підході є визначення точного часу виконання команди, що вноситься в слот затримки. Апаратура повинна гарантувати реальне виконання цих команд перед виконанням власне переходу.

1.4 Висновок

З наявної інформації можна зробити висновок, що структура ядер сучасних процесорів дуже складна в своїй організації. Через це підвищувати їх продуктивність стає все складніше. Так як метод нарощування частоти процесора вже вичерпав свої можливості, то його використання стало недоцільним і неефективним.

На даний момент для збільшення продуктивності ядер сучасних процесорів використовується конвейеризація та суперскалярність. Ці методи

дозволяють збільшити продуктивність ядра процесора, не збільшуючи при цьому енергетичну ефективність в 2 і більше разів, що дозволяє уникнути перегріву і, як наслідок, виходу процесора з ладу.

Процессори з конвеєрною організацією можуть налічувати 20 і більше ступенів. Залежно від типів виконуваних команд в ядрах таких процесорів часто можуть виникнути конфлікти з управління. Для запобігання конфліктів даного типу часто використовується блок передбачення переходів, (так як конфлікти з управління виникають при появі команди умовного переходу), який дозволяє уникнути істотних простоїв.

В даній роботі буде досліджена модель ядра процесора з як з використанням блоку передбачення переходів так і без блоку передбачення переходів. При цьому буде також проведений аналіз отриманих результатів і зроблені висновки щодо доцільності застосування отриманих даних.

2 ВИБІР МЕТОДУ ДОСЛІДЖЕННЯ

2.1 Методи дослідження обчислювальних систем

2.1.1 Аналітичні методи

При використанні аналітичних методів будується математична модель об'єкта, яка представляє фізичні властивості об'єкта у вигляді математичних об'єктів і відношень (математичних операцій над величинами), наприклад, у вигляді диференційних або інтегральних рівнянь. Математична модель будується на основі понять, символіки і методів деякої теорії, наприклад, теорії масового обслуговування, яка визначає клас математичних аналогій, тобто основоположних математичних моделей. При аналітичному підході необхідні залежності виводяться з математичної моделі послідовним застосуванням математичних правил. Перешкодою при цьому може бути нерозв'язність рівнянь в аналітичній формі, відсутність первісних для підінтегральних функцій і т.п., з чим доводиться стикатися досить часто. Тому лише за певних властивостях моделі можна отримати рішення в явній аналітичній формі. Незважаючи на обмежені можливості аналітичного підходу, рішення, отримані в явній аналітичній формі, мають велику пізнавальну цінність і знаходять результативне застосування при вирішенні широкого класу теоретичних і прикладних завдань.

При неможливості отримання рішення рівнянь в аналітичному вигляді, для їх вирішення можна застосовувати чисельні методи. Результат застосування чисельних методів - таблиці (графіки) залежностей, які розкривають властивості об'єкта. Використання чисельних методів для вирішення рівнянь дозволяє вирішувати значно ширше коло завдань. Негативна властивість використання чисельних методів розв'язання рівнянь - приватний характер результатів, які не розкривають залежності, а лише визначають її в окремих апріорно призначених точках.

2.1.2 Чисельні методи

Чисельні методи грунтуються на побудові кінцевої послідовності дій над числами, що приводить до отримання необхідних результатів. При наявності

математичної моделі досліджуваного об'єкта застосування чисельних методів зводиться до заміни математичних операцій і відносин відповідними операціями над числами: заміні інтегралів сумами, похідних різницевими схемами, нескінченних сум кінцевими і т.д. В результаті цього будується алгоритм, який дозволяє точно або з допустимою похибкою визначити значення необхідних величин. Алгоритм реалізується комп’ютером. Результат застосування чисельних методів - таблиці (графіки) залежностей, які розкривають властивості об'єкта. Чисельні методи в порівнянні з аналітичними дозволяють вирішувати значно ширше коло завдань.

2.1.3 Імітаційні методи

Характер процесів, властивих досліджуваному об'єкту і які підлягають відображенню в моделі, може бути настільки складним, що побудова математичної моделі перетворюється в важку задачу, а аналіз моделі навіть з використанням чисельних методів розв'язання рівнянь може виявитися нерезультативним через трудомісткість або нестійкість алгоритмів щодо похибок апроксимації та округлення.

Відтворення в моделі динаміки складних просторово-часових відносин між об'єктами, які утворюють систему, всього різноманіття її зв'язків із середовищем, діючих в системі законів керування, адаптивних властивостей і рис цілеспрямованної поведінки важко здійснити тільки математичними засобами. При дослідженні таких систем широке застосування знаходять моделі, які представляють собою змістовний опис об'єктів дослідження в формі алгоритмів. В описах відображаються як структура досліджуваних систем, що досягається ототоджненням елементів систем з відповідними елементами алгоритмів, так і процеси функціонування систем у часі, що подаються в логіко-математичній формі. При цьому описи об'єктів дослідження мають алгоритмічний характер, а самі моделі суть програми для комп’ютерів. Моделі такого типу називають імітаційними або алгоритмічними.

Головна особливість даного підходу до моделювання заключається в тому, що використовувані для побудови моделі алгоритмічні мови, є більш гнучкими і доступними засобами опису складних систем, ніж мова математичних функціональних співвідношень. Завдяки цьому в імітаційних моделях складних систем знаходять відображення багато деталей їх структури і

функцій, які вимушено опускаються або мимоволі втрачаються в математично строгих моделях. Властива імітаційним моделям реалістичність ґрунтується на використанні для їх побудови всіх наявних уявлень про об'єкт дослідження як теоретичного, так і евристичного характеру.

Імітаційне моделювання це процес отримання статистичних даних про процеси, які відбуваються в моделюємій системі. Для отримання результатів статистичні дані, отримані при дослідженні моделі, обробляються і класифікуються з використанням методів математичної статистики. Позитивна властивість імітаційного моделювання - універсальність, яка гарантує принципову можливість аналізу систем будь-якого ступеня складності з будь-яким ступенем деталізації досліджуваних процесів. Негативна властивість імітаційного моделювання - приватний характер результатів, який не розкриває залежності, а лише визначає їх в окремих апріорно призначених точках.

2.1.4 Експериментальні методи

Експериментальні методи базуються на вимірюванні характеристик процесів, які відбуваються в реальних системах, і обробці результатів вимірювання з метою виявлення необхідних залежностей. Експериментальні дослідження є джерелом найбільш достовірної інформації, проте отримувані при цьому результати носять приватний характер. Ці методи - найбільш достовірне джерело відомостей про функціонування комп'ютерних систем.

При проведенні досліджень навіть вузького кола питань виникає необхідність у використанні декількох методів вирішення однієї і тієї ж задачі. Рішення задач аналітичними методами завжди передбачає використання припущень, прийнятих або на етапі побудови концептуальної моделі, або на етапі побудови математичної моделі, тобто. виведення результуючих залежностей. Отримувані при цьому залежності можуть бути використані для вирішення прикладних завдань тільки після оцінки методичної похибки, яка вноситься прийнятими припущеннями. Оцінити похибку аналітичними методами - завдання зазвичай більш складне, ніж вивід залежностей. Тому для оцінки методичної похибки широко використовується метод імітаційного моделювання, що дозволяє отримати чисельне рішення з будь-якою заданою точністю. У такому ж аспекті використовуються і експериментальні методи аналізу комп'ютерних систем.

2.2 Моделі ядра при виявленні «перешкоди»

2.2.1 Модель ядра з негайним блокуванням конвеєра

У разі, коли виявлено чергову «перешкоду» роботі конвеєра (Пристрій керування ПК згенерував команду умовного переходу) припиняється генерація нових команд в ПК (тобто ПК блокується) до тих пір, поки «перешкоду» не усунено. Для команд умовного переходу це означає, що повинні виконатися всі команди, які передували їм. Будемо вважати, що результат для команди умовного переходу готує попередня їм команда обробки, яка або знаходиться ще в буфері, або виконується в функціональному пристрої (ФП).

Після усунення «перешкоди» (а це означає, що в підсистемі виконання операцій немає команд - буфер порожній і ФП закінчив виконання останньої команди). ПК розблокується і продовжує генерацію нових команд.

Підхід «Блокування конвеєра відразу при виявленні перешкоди» забезпечує правильну послідовність виконання програми при мінімальних витратах апаратури, та й при мінімальній продуктивності конвеєра.

Як відомо, будь-яка модель об'єкта - це спрощене уявлення основних особливостей цього об'єкта.

Вимоги до моделі - вона повинна бути максимально простою, щоб забезпечити її дослідження аналітичними методами, і при цьому відображати основні, суттєві для дослідження риси об'єкта.

Однією з істотних особливостей сучасних ядер процесорів є використання конвеєрного принципу обробки і виконання команд. Ступінь конвейеризации коливається в широких межах і може досягати 10 і більше.

Побудова моделі, яка відображає всі ступені конвеєра, недоцільно з наступних причин.

При наявності більше 2 ступенів рівняння, які описують поведінку моделі стають дуже громіздкими і вирішити їх неможливо.

В той же час відображати всі ступені конвеєра для нашого дослідження немає необхідності. Досить розглянути функціонування моделі ядра з 2 ступенями конвеєра. Одна ступінь конвеєра буде включати в себе всі ступені по формуванню адреси чергової команди, читання команди з підсистеми пам'яті, просуванню прочитаної команди до дешифратора команд, дешифрування команд. Інша ступінь буде включати в себе всі ступені з виконання команд.

Для цілей дослідження байдуже чи є функціональний пристрій з виконання команд конвеєрним чи комбінаційним. У будь-якому випадку для отримання очікуваного результату необхідно дочекатися виконання всіх команд, які згенеровані першим ступенем конвеєра.

При цьому не має значення чи виконувались ці команди строго по порядку в універсальному функціональному пристрої, або було використано позачергове виконання команд в спеціалізованих функціональних пристроях, так як результати виконання команд записуються в архітектурні регістри строго в порядку передбаченому програмою, тобто послідовно в порядку номерів команд програми. Структура зі спеціалізованими функціональними пристроями може бути замінена універсальним функціональним пристроєм з еквівалентної продуктивністю.

Структура моделі ядра представлена на рис. 2.1.

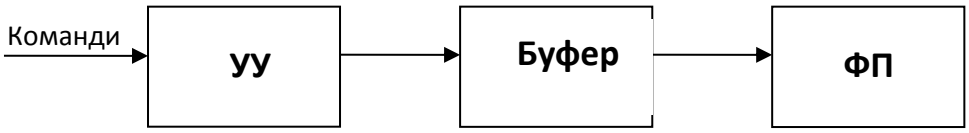


Рисунок 2.1 - Структурна схема моделі ядра процесора

Були зроблені такіі припущення:

- конвеєр обробки і виконання команд є двоступінчастим;
- перша ступінь генерує потік команд (заявок). На її вхід надходить програма, яка складається з нескінченного числа команд наступних типів:
- команд обробки і читання з підсистеми пам'яті з ймовірністю W_0;
- команд умовного переходу (УП) - з ймовірністю $W_П$;
- порядок проходження команд в програмі випадковий з вірогідністю появи команд w_O , $w_П$ за типами команд, відповідно;
- команди умовних переходів (і інших переходів) виконуються в пристрої генерації (тобто першим ступенем конвеєра);
- команди обробки, команди звернення до підсистеми пам'яті (як читання, так і запису) виконуються послідовно в другим ступенем в універсальному функціональному пристрої;
- між ступенями є буфер на **n** заявок;
- потік заявок, який згенерував ПК, найпростіший з експоненціальним законом розподілу;

- якщо буфер заявок заповнений, то ПК блокується, тобто призупиняється;
- ФП виконує операції відповідно з кодом команди з інтенсивністю μ . Закон розподілу часів виконання операцій експоненціальний.
- дисципліна вибору заявок на виконання з буфера - FIFO ;
- при відсутності підготовлених до виконання заявок ФП простоює;
- якщо буфер порожній і ПК згенерувало чергову заявку, то вона миттєво надходить в ФП.

Розглянемо всі стани, в яких може перебувати система. Ідентифікувати стан будемо за допомогою індексу i **(0≤ i ≤ n +1)** , який відображає кількість згенерованих ПК заявок. Заявки можуть перебувати - одна на виконанні в ФП, (i -1) - в буфері.

ПК і ФП можуть або працювати, або простоювати.

Стан a_0 - буфер - порожній, ФП простоює (чекає заявок). ПК генерує чергову команду. Якщо чергова команда з ймовірністю W_O це команда обробки, то система переходить в стан **a_1** . Якщо чергова команда будь-яка інша команда (з ймовірністю $1 - W_O$), то система залишається в стані **a_0** .

Стан a_i $(1 \le i \le n)$ - в буфері (i -1) заявка, ФП виконує чергову команду обробки. ПК генерує чергову команду. Якщо чергова команда з ймовірністю W_O це команда обробки, то система переходить в стан **a_{i+1}** .

Якщо чергова команда з ймовірністю $W_П$ це команда умовного переходу, то система переходить в стан **b_i** .

Якщо ПК не закінчить генерувати чергову заявку, а ФП закінчив виконувати чергову заявку, то система переходить в стан **a_{i-1}** .

Стан a_{n+1} - в буфері **n** заявок. ФП виконує чергову команду обробки. ПК блоковано через заповнення буфера.

Коли ФП закінчить виконувати чергову заявку, система переходить в стан **a_n** .

Стан b_i $(1 \le i \le n)$ - в буфері (i -1) заявка, ФП виконує чергову команду обробки. ПК блоковано через генерації команди умовного переходу - очікує отримання умови з підсистеми обробки.

Коли ФП закінчує виконувати чергову заявку, система переходить в стан **a_{i-1}** .

Граф станів системи при негайному блокуванні конвеєра при виявленні «перешкоди» наведено на рис. 2.2.

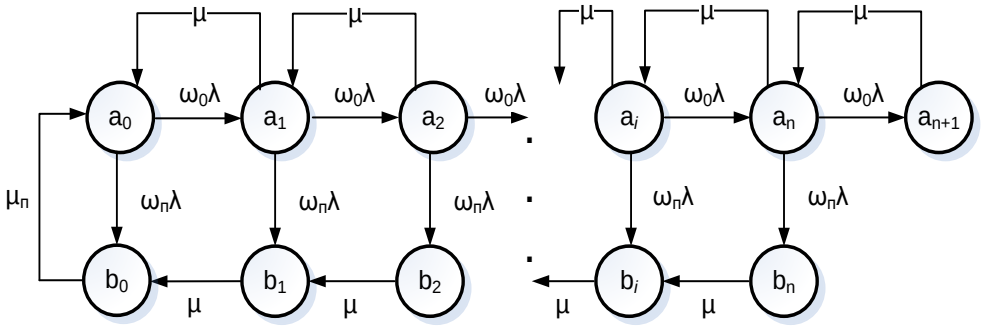


Рисунок 2.2 - Граф станів системи при блокуванні конвеєра відразу при виявленні «перешкоди»

$\omega_n = 1 - \omega_o$;

ω_o - частота появи команд обробки;

ω_n - частота появи команд переходів;

μ - інтенсивність виконання команд обробки;

μ_n - інтенсивність виконання команд переходів;

λ - інтенсивність генерації команд;

n - розмір буфера.

У кожному стані система може перебувати з ймовірністю

$$P_{a0}, P_{a1}, \ldots, P_{au-1}, P_{b1}, \ldots, P_{bn}$$

$$P_{a0} + P_{a1} + P_{a2} + \ldots + P_{au} + P_{b1} + \ldots + P_{bn} = 1$$

За графу станів системи складають рівняння балансу для кожного стану.

$$P_{a0} \cdot (\lambda + \mu) = P_{b0} \cdot \mu_n + P_{a1} \cdot \mu$$

$$P_{a1} \cdot (\lambda + \mu) = P_{a0} \cdot \omega_0 \cdot \lambda + P_{a2} \cdot \mu$$

$$P_{a2} \cdot (\lambda + \mu) = P_{a1} \cdot \omega_0 \cdot \lambda + P_{a3} \cdot \mu$$

.....

$$P_{ai} \cdot (\lambda + \mu) = P_{a(i-1)} \cdot \omega_0 \cdot \lambda + P_{a(i+1)} \cdot \mu, \ (1 \le i \le n)$$

.....

$$\begin{cases} P_{a(n+1)} \cdot \mu = P_{an} \cdot \omega_0 \cdot \lambda \\ P_{b0} \cdot \mu_n = P_{a0} \cdot \omega_n \cdot \lambda + P_{b1} \cdot \mu \\ P_{b1} \cdot \mu = P_{a1} \cdot \omega_n \cdot \lambda + P_{b2} \cdot \mu \\ P_{b2} \cdot \mu = P_{a2} \cdot \omega_n \cdot \lambda + P_{b3} \cdot \mu \end{cases}$$

.....

$$P_{bi} \cdot \mu = P_{ai} \cdot \omega_n \cdot \lambda + P_{b(i+1)} \cdot \mu \,, \; (1 \le i \le n-1)$$

.....

$$P_{bn} \cdot \mu = P_{an} \cdot \omega_n \cdot \lambda$$

ПК працює у всіх станах $\mathbf{a}_i$ ($0 \le i \le n$).

ПК простоює (блокований): через очікування умови переходу - в станах $\mathbf{b}_i$ ($1 \le i \le n$) і через заповнення буфера - в стані $\mathbf{a}_{n+1}$.

Коефіцієнт завантаження ПК дорівнює сумі ймовірностей всіх станів, в яких ПК не блокуваний.

Коефіцієнт простою ПК через очікування умови переходу дорівнює сумі вірогідності всіх станів $\mathbf{b}_i$ ($1 \le i \le n$).

Коефіцієнт завантаження ФП дорівнює сумі ймовірностей всіх станів, в яких ФП не простоює.

2.2.2 Модель ядра з блоком передбачення переходів

При появі залежності по управлінню (згенерована команда умовного переходу) з певною ймовірністю блок передбачення прогнозує вибір потрібного напряму. Пристрій вибірки команд не блокується, а продовжує вибірку і обробку команд за передбаченою адресою.

Обрані за передбаченою адресою команди позначаються як умовно виконувані. Результати цих команд не заносяться в архітектурні регістри до тих пір, поки не стане відомо справжнє значення ознаки переходу. Якщо блок передбачення правильно визначив напрямок переходу, то всі умовно виконувані команди оголошуються безумовно виконуваними і результати

виконаних команд в порядку, передбаченим програмістом, записуються в архітектурні регістри.

Якщо блок передбачення неправильно визначив напрямок переходу, то всі умовно виконувані команди анулюються.

Існує кілька варіантів структурних рішень по реалізації технології передбачення.

2.2.2.1 В ядрі відсутній спеціальний буфер продешифрованних команд

В цьому варіанті формується адреса першої команди нового напряму і відправляється в підсистему пам'яті (як правило в кеш-пам'ять команд) - читається перша команда нового напряму і просувається по конвеєру до дешифратора команд. Після отримання цієї команди дешифратором вона готується до виконання і записується у відповідний буфер.

Час затримки в роботі дешифратора команд при переході до обробки першої команди нового напряму

$T_2 = t_{ФА} + t_{ЧТ} + t_{ЗАП\ КОН}$,

де:

$t_{ФА}$ - час формування адреси першої команди нового напряму;

$t_{ЧТ}$ - час читання першої команди нового напряму,

$t_{ЗАП\ КОН}$ - час заповнення конвеєра до дешифратора команд.

2.2.2.2 В ядрі є спеціальний буфер продешифрованних команд

В цьому варіанті також формується адреса першої команди нового напряму, але вона відправляється в спеціальний буфер продешифрованних команд, готових до виконання.

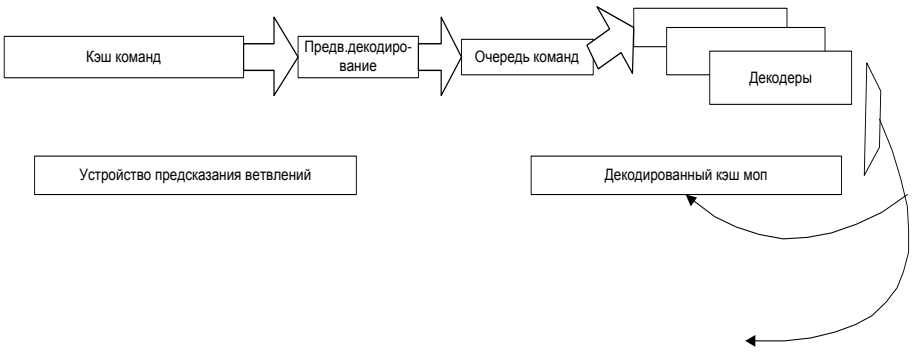


Рисунок 2.3. - Структура ядра з буфером продешифрованних команд

Кількість продешифрованих команд в цьому спеціальному буфері продешифрованних команд в ядрах деяких процесорів змінюється у великих межах. Наприклад, в ядрах процесорів фірми Intel зі структурою Sandy Bridge це 1500 продешифрованних внутрішніх команд ядра.

Тепер перша (і наступні команди нового напряму) видаються на виконання в функціональний пристрій з цього швидкого спеціального буфера продешифрованних команд.

Затримка складе

$T_З = t_{ФА}$,

де $t_{ФА}$ - час формування адреси першої команди нової гілки.

Втрати часу на читання першої команди нової гілки або з підсистеми пам'яті, або зі спеціального буфера команд, а також на заповнення конвеєра до дешифратора команд і дешифрування команд відсутні.

2.2.3 Модель ядра процесора с блоком передбачення переходів

Всі припущення, зроблені при розробці моделі з негайним блокуванням конвеєра залишаються в силі.

Структура моделі ядра аналогічна представленій рис. 2.1 і представлена на рис. 2.4.

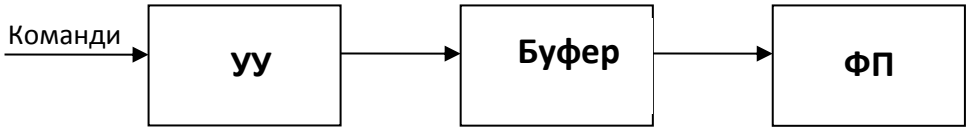


Рисунок 2.4 - Структурна схема моделі ядра процесора

Граф станів моделі системи наведено на рис. 2.5.

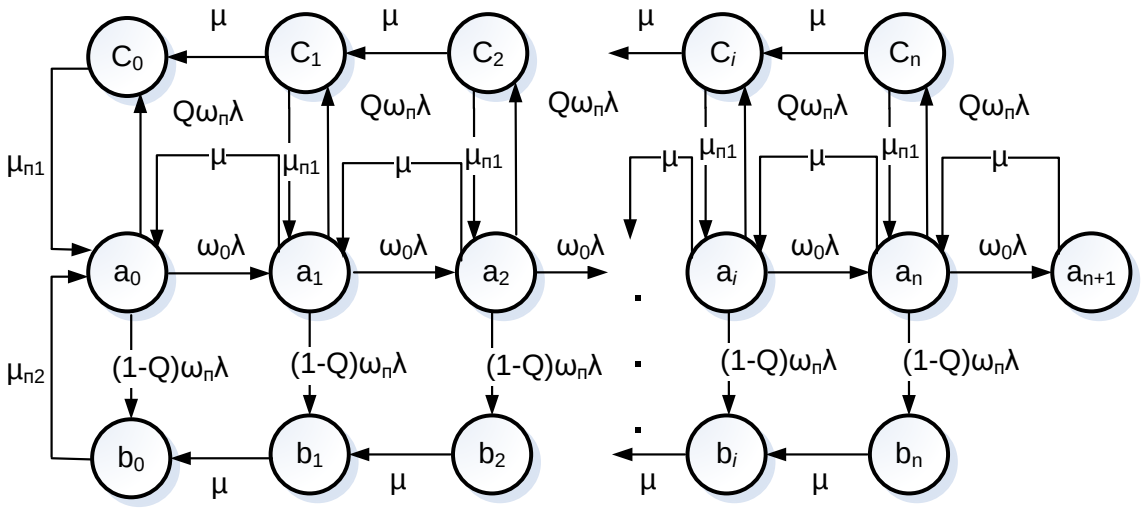


Рисунок 2.5 - Граф станів моделі системи з блоком передбачень

ω_0 - частота команд обробки

$\omega_n = 1 - \omega_0$ - частота команд переходу

μ - інтенсивність виконання команд обробки;

μ_n - інтенсивність виконання команд переходу;

λ - інтенсивність генерації команд;

n - розмір буфера;

q - ймовірність правильного напрямку переходу. ПК не блокується і продовжує генерацію команд за цим напрямком;

(1- q) - ймовірність помилкового передбачення напрямки переходу. ПК блокується.

Розглянемо всі стани, в яких може перебувати система. Заявки можуть перебувати - одна на виконанні в ФП,(i -1) - в буфері.

ПК і ФП можуть або працювати, або простоювати.

В стані $\mathbf{a_i}$ $(0 \le i \le n)$ ПК працює, тобто генерує чергові заявки.

В стані $\mathbf{a_{n+1}}$ ПК блоковано через заповнення буфера заявок.

В станах $\mathbf{c_i}$ $(0 \le i \le n+1)$ ПК виконує перехід до першої команди нової правильно передбаченої гілки. Дії ПК залежать від реалізованого структурного рішення по компенсації негативного впливу команд умовного переходу. Математичне очікування часу виконання переходу $t_{пер1}$. Інтенсивність переходів в стані $\mathbf{a_i}$ $(0 \le i \le n)$

$\mu_{n1} = 1 / t_{пер1}$.

В станах $\mathbf{b_i}$ $(1 \le i \le n)$ ПК блокований, так як він очікує отримання дозволу (ознаки) переходу.

В стані **b 0** ПК виконує перехід до першої команди нової неправильно передбаченої гілки. Формується адреса переходу, який надходить в підсистему пам'яті (спочатку в кеш команд першого рівня). Віртуальна адреса перетворюється в фізичну адресу, потім здійснюється читання блоку командної інформації з кешу команд. Прочитаний блок надходить в конвеєр. В конвеєрі відбувається підготовка блоку командної інформації для дешифрування.

Математичне очікування часу виконання переходу $t_{Пер2}$.

Інтенсивність переходу в стан **a 0** - $\mu_{n2} = 1 / t_{Пер2}$.

ФП простоює в станах **a 0, b 0 і c 0** через відсутність готових до виконання заявок.

В кожному з таких станів система може перебувати з певною ймовірністю.

За графу станів системи складаються рівняння балансу для кожного стану

$$P_{a0}\lambda = P_{b0} * \mu_{n2} + P_{a1} * \mu + P_{C0} * \mu_{n1}$$

$$P_{ai}(\lambda + \mu) = P_{ai-1} * \omega_0 * \lambda + P_{a(i+1)} * \mu + P_{Ci} * \mu_{n1}, \qquad (1 \le i \le n)$$

$$P_{an+1} * \mu = P_{an} * \omega_0 * \lambda$$

$$P_{b0}\mu_{n2} = (1-Q)P_{a0} * \omega_n * \lambda + P_{b1} * \mu$$

$$P_{bi}\mu = (1-Q)P_{ai} * \omega_n * \lambda + P_{b(i+1)} * \mu, \qquad (1 \le i \le n-1)$$

$$P_{bn}\mu = (1-Q)P_{an} * \omega_n * \lambda$$

$$P_{C0}\mu_{n1} = Q * P_{a0} * \omega_n * \lambda + P_{C1} * \mu$$

$$P_{Ci}(\mu + \mu_{n1}) = Q * P_{ai} * \omega_n * \lambda + P_{c(i+1)} * \mu, \quad (1 \le i \le n-1)$$

$$P_{Cn}(\mu + \mu_{n1}) = Q * P_{an} * \omega_n * \lambda$$

Як показники ефективності використані коефіцієнти завантаження ПК і ФП.

Коефіцієнт завантаження ПК дорівнює сумі ймовірностей всіх станів, в яких ПК не блокований.

Коефіцієнт завантаження ФП дорівнює сумі ймовірностей всіх станів, в яких ФП не простоює.

2.3 Рішення системи рівнянь

З одноманітного визначення ймовірностей станів P_{ai} (и = 1 ... n +1), (и = 2 ... n) і (и = 2 ... n) випливає, що:

$$P_{ai} = R^{i-1} * P_{a1} \qquad (2 \le i \le n+1)$$

$$P_{bi} = R^{i-1} * P_{b1} \qquad (2 \le i \le n)$$

$$P_{ci} = R^{i-1} * P_{c1} \qquad (2 \le i \le n),$$

але

$$P_{a1} \ne R * P_{a0}$$

$$P_{b1} \ne R * P_{b0}$$

$$P_{c1} \ne R * P_{c0}.$$

З рівнянь для ймовірностей станів C_{ai} і B_{ai} можна визначити:

$$P_{ci} = \frac{q\omega_n \rho}{A_1 + 1} P_{a1} \qquad (1 \le i \le n)$$

$$P_{bi} = \frac{(1-q) * \omega_n * \rho}{1-R} P_{a1} \quad (1 \le i \le n) \;;$$

де: $A_1 = \frac{\mu_{n1}}{\mu} = \frac{t_{вып}}{t_{пер1}}$, $\rho = \frac{\lambda}{\mu} = \frac{t_{вып}}{t_{ген}}$

З рівняння

$$P_{ai} \cdot (\lambda + \mu) = P_{ai-1} \cdot \omega_o \cdot \lambda + P_{ai+1} \cdot \mu + P_{ci} \cdot \mu_{n1}$$

розділивши всі члени на μ і підставивши значення

$$P_{ai} = RP_{ai-1}$$

$$P_{ai+1} = R^2 P_{ai-1}$$

$$P_{ci} = \frac{q\omega_n\rho}{A_1+1} RP_{ai-1}$$

отримаємо

$$P_{ai-1} * \{R^2 - [(\rho+1) - \frac{A_1}{A_1+1} * q * \omega_n * \rho] * R + \omega_o * \rho\} = 0$$

$P_{ai-1} \neq 0$, тоді

$$R^2 - R * [(\rho+1) - \frac{A_1}{A_1+1} * q * \omega_n * \rho] + \omega_o * \rho = 0$$

З двох коренів рівняння умову задачі задовольняє корінь

$$R = \frac{\left[(\rho+1) - \frac{A_1}{A_1+1} * q * \omega_n * \rho\right] - \sqrt{\left[(1+\rho) - \frac{A_1}{A_1+1} * q * \omega_n * \rho\right]^2 - 4 * \omega_o * \rho}}{2}$$

Ймовірності станів P_{a1}, P_{b0} , P_{c0}, P_{a0} можна виразити в такому вигляді

$$P_{a1} = \frac{\omega_o * \rho}{1 + \frac{\omega_П * q + \rho}{1 + A_1} + \frac{\omega_П * (1-q) * \rho}{1 - R}} * P_{a0} = F * P_{a0}$$

$$P_{b0} = \frac{\omega_П * (1-q) * \rho}{A_2} [1 + \frac{\omega_o * \rho}{(1-R) * (1 + \omega_П * \rho * (\frac{q}{1+A_1} + \frac{1-q}{1-R})}] * P_{a0}$$

$$P_{b0} = \frac{\omega_П * (1-q) * \rho}{A_2 (1-R)} \{1 - R + F\} * P_{a0}$$

$$F = \frac{\omega_o * \rho}{1 + \omega_П * \rho * (\frac{q}{1 + A_1} + \frac{1 - q}{1 - R})}$$

$$P_{c0} = \frac{\omega_П * q * \rho}{A_1} [1 + \frac{\omega_o * \rho}{(1 + A_1) * (1 + \omega_П * \rho * (\frac{q}{1 + A_1} + \frac{1 - q}{1 - R})}] * P_{a0}$$

$$P_{c0} = \frac{\omega_П * q * \rho}{A_1 (1 + A_1)} [1 + A_1 + F] * P_{a0}$$

де $$A_2 = \frac{\mu_{n2}}{\mu} = \frac{t_{вып}}{t_{пер2}}.$$

$$P_{a0} = \frac{1}{1 + \frac{F}{1 - R} + \omega_П * \rho * \left\{ q \left[\frac{1}{A_1} + \frac{F}{A_1 (1 + A_1)} + \frac{F}{(1 + A_1)(1 - R)} \right] + (1 - q) * \left[\frac{1}{A_2} + \frac{F}{(1 + A_2)(1 - R)} + \frac{F}{(1 - R)^2} \right] \right\}}$$

$$P_{a0} = \frac{1 - R}{1 - R + F + \frac{\omega_П * q * \rho}{A_1 (1 + A_1)} * [1 + A_1 + F)(1 - R) + A_1 F] + \frac{\omega_П * (1 - q) * \rho}{A_2 * (1 - R)} * \left[(1 - R + F)(1 - R) + A_2 F \right]}$$

2.3.1 Коефіцієнт завантаження функціонального пристрою

$$E_{фу} = 1 - (P_{a0} + P_{b0} + P_{c0})$$

$$E_{ФУ} = \frac{1 + \omega_П * \rho * \left[\frac{1 - q}{1 - R} + \frac{q}{1 + A_1} \right]}{1 + \frac{1 - R}{F} + \omega_П * \rho * \left\{ q * \left[\frac{1 - R}{A_1 F} + \frac{1 - R}{A_1 (1 + A_1)} + \frac{1}{1 + A_1} \right] + (1 - q) * \left[\frac{1}{A_2} + \frac{1 - R}{A_2 F} + \frac{1}{1 - R} \right] \right\}}$$

Після спрощення отримаємо

$$E_{фУ} = \frac{\omega_0 * \rho}{1 - R + F + \frac{\omega_П * q * \rho}{A_1(1 + A_1)}[1 + A_1 + F)(1 - R) + A_1 F] + \frac{\omega_П * (1 - q) * \rho}{A_2(1 - R)}\left[(1 - R + F) * (1 - R) + A_2 F\right]}$$

2.3.2 Коефіцієнт завантаження пристрою керування

$$H_{уУ} = \sum_{i=0}^{n} P_{ai} = P_{a0} + \sum_{i=1}^{n} P_{ai} = P_{a0} + \sum_{i=1}^{n} R^{i-1} P_{a1} = P_{a0} + \frac{(1 - R^{n})}{1 - R} P_{a1}$$

$$\lim_{n \to \infty} H_{уУ} = \lim_{n \to \infty} P_{a0} + \lim_{n \to \infty} \frac{(1 - R^{n})}{1 - R} P_{a1} = P_{a0} + \frac{F}{1 - R} P_{a0} = P_{a0} * (1 + \frac{F}{1 - R})$$

Так як R <1, то при великих значеннях **n** (n > 16) значенням R n можна знехтувати.

$$H_{уУ} = \frac{(1 - R) * (1 + \frac{F}{1 - R})}{1 - R + F + \frac{\omega_П * q * \rho}{A_1(1 + A_1)}[1 + A_1 + F)(1 - R) + A_1 F] + \frac{\omega_П * (1 - q) * \rho}{A_2(1 - R)}\left[(1 - R + F)(1 - R) + A_2 F\right]}$$

$$H_{уУ} = \frac{1 - R + F}{1 - R + F + \frac{\omega_П * q * \rho}{A_1(1 + A_1)}[1 + A_1 + F)(1 - R) + A_1 F] + \frac{\omega_П * (1 - q) * \rho}{A_2(1 - R)}\left[(1 - R + F)(1 - R) + A_2 F\right]}$$

$$R_c = \frac{E_{фУ}}{H_{уУ}} = \frac{\omega_0 * \rho}{1 - R + F}$$

Підставивши замість F його значення отримаємо

$$R_c = \frac{E_{фУ}}{H_{уУ}} = \frac{\omega_0 * \rho}{1 - R + \frac{\omega_o * \rho}{1 + \omega_П * \rho * (\frac{q}{1 + A_1} + \frac{1 - q}{1 - R})}}$$

Після перетворень

$$R_c = \frac{E_{ФУ}}{H_{УУ}} = \frac{1 + \omega_П * \rho * (\frac{q}{1 + A_1} + \frac{1 - q}{1 - R})}{1 + (1 - R) * [1 + \omega_П * \rho * (\frac{q}{1 + A_1} + \frac{1 - q}{1 - R})]}$$

Програма рішення системи рівнянь методом Гаусса приведена в додатку А.

2.4 Висновки

В даному розділі були розглянуті методи дослідження і обрані аналітичний та чисельний методи.

Були розроблені моделі, складені графи станів моделей з блоком передбачення переходів та без блоку передбачення переходів. На основі графів станів моделей з блоком передбачення переходів були складені і вирішені системи рівнянь для знаходження ймовірностей знаходження моделі в різних станах. На основі ймовірностей станів визначені коефіцієнти завантаження функціональних пристроїв і пристроїв керування. Також була розроблена програма, яка реалізує рішення складеної системи рівнянь методом Гаусса.

В подальшій роботі буде проведений аналіз і дослідження отриманих результатів, а також перевірка точності аналітичних результатів за допомогою чисельних методів.

3 ДОСЛІДЖЕННЯ ОТРИМАНИХ РЕЗУЛЬТАТІВ

3.1 Аналіз результатів експериментів з моделлю без блоку передбачення

3.1.1 Дослідження впливу коефіцієнта навантаження і розміру буфера

Вирази, необхідні для дослідження наведені нижче .

$$E_{фУ} = \frac{R*[(1-R^{n+1}) + \frac{\omega_П \rho}{(1-R)}(1-R^n)]}{1-R^{n+2} + \omega_П \rho * \left[\frac{1}{A} + \frac{R(1-R^n)}{(1-R)}\right]}$$

$$H_{уУ} = \frac{1-R^{n+1} + \frac{\omega_П \rho}{A}}{1-R^{n+2} + \omega_П \rho * [\frac{1}{A} + \frac{R(1-R^n)}{1-R}]}$$

$$\lim_{n\to\infty} E_{фУ} == \frac{R*[1+\frac{\omega_П \rho}{1-R}]}{1+\omega_П \rho * \left[\frac{1}{A} + \frac{R}{1-R}\right]}$$

$$\lim_{n\to\infty} H_{уУ} = \frac{1+\frac{\omega_П \rho}{A}}{1+\omega_П \rho * [\frac{1}{A} + \frac{R}{1-R}]}$$

$$R = \frac{(1+\rho) - \sqrt{(1+\rho)^2 - 4*\omega_o * \rho}}{2}$$

$$\rho = \frac{\lambda}{\mu} = \frac{t_{вып}}{t_{ген}}, \qquad A = \frac{\mu_n}{\mu} = \frac{t_{вып}}{t_{пер}}, \qquad \omega_n = 1 - \omega_o$$

Коефіцієнти завантаження (використання) функціонального пристрою і пристрою кевання є функціями багатьох змінних, зокрема:

коефіцієнта навантаження ρ на функціональний пристрій при виконанні програм без команд переходу - $\rho = \frac{\lambda}{\mu} = \frac{t_{вып}}{t_{ген}}$,

коефіцієнта навантаження R на функціональний пристрій при виконанні програм с командами переходу;

частоти команд переходу $\omega_П$;

розміру буфера;

коефіцієнта А, який визначає співвідношення інтенсивностей виконання команд переходу і команд, виконуваних функціональним пристроєм.

Коефіцієнт навантаження ρ визначається як структурними особливостями ядра, так і типом виконуваної в конкретний момент програми. Він може приймати значення в широкому діапазоні.

Значення коефіцієнта навантаження:

ρ = 1 означає, що продуктивності функціонального пристрію і пристрою керування рівні, тобто, що система збалансована;

ρ> 1 означає, що пристрій керування швидше функціонального пристрою і система розбалансована;

ρ <1 означає, що функціональний пристрій швидший пристрою керування і система розбалансована.

Розбалансування системи, як правило, це вплив конкретної програми. Для дослідження доцільно розглянути досить типові значенія ρ з діапазону 0.5; 1; 2.

Зміни частот команд переходів доцільно досліджувати в діапазоні 0,05; 0.10; 0.15; 0.20; 0.25 ; 0.30. Частота команд переходу $\omega_П$ = 0.1 означає, що кожна десята команда в виконуваної програмою є командою переходу.

В таблиці 3.1 наведені значення коефіцієнта R при різних значеннях коефіцієнта ρ і частоти команд переходу.

Таблиця 3.1 - Значення коефіцієнта R при різних значеннях коефіцієнта ρ і частоти команд переходу

ρ		ω П					
		0.05	0.10	0.15	0.20	0.25	0.30
0.5	R	0.4542	0.4146	0 . 3792	0.3469	0.3170	0.2890
1	R	0.7764	0.6838	0.6127	0.5528	0.5	0.4523

2	R	0.9084	0.8292	0.7584	0.6938	0.6340	0.5780

Коефіцієнт навантаження R на функціональний пристрій при виконанні програм с командами переходу при будь-яких значеннях ρ і $\omega_П > 0$ завжди менше 1.

Значення коефіцієнтів використання функціонального пристрою ($E_{ФП}$) в залежності від значення ρ, розміру буфера наведені в таблиці 3.2.

Таблиця 3.2 - Значення коефіцієнта використання функціонального пристрою ($E_{ФП}$) в залежності від значення ρ, розміру буфера n при $\omega_П = 0.20$; A = 1.

ρ		n						
		2	4	8	16	32	64	∞
0.5	$E_{ФП}$	0.3402	0.3463	0.3469	0.3469	0.3469	0.3469	0.3469
1	$E_{ФП}$	0,5334	0,55	0,5527	0,5528	0,5528	0,5528	0,5528
2	$E_{ФП}$	0.6805	0.6926	0.6938	0.6938	0.6938	0.6938	0.6938

Так як при будь-яких значеннях ρ і $\omega_П$ R <1, то вже при розмірі буфера n = 8 значення коефіцієнта використання функціонального пристрою дуже близькі до граничних значень. Похибка при n = 8 не перевищує 0.5%, а при n = 16, 32, 64 значення коефіцієнтів використання практично збігаються з граничними значеннями.

Отже, при проектуванні структури ядра мінімально необхідний розмір буфера повинен бути не менше 8 при різних значеннях коефіцієнта навантаження ρ. Однак, з огляду на те, що ядро можуть виконуватися найрізноманітніші програми (в тому числі і такі, в яких дуже мало команд умовних переходів (тобто з $\omega_П$ близькими до нуля), то, розмір буфера необхідно вибирати з урахуванням найбільш «важкого» режиму роботи. А таким є режим з ρ = 1 і $\omega_П = 0$. Для такого режиму роботи близькі до граничних значень коефіцієнтів використання забезпечує буфер n =32. Такий розмір буфера задовольняє вимогам у всіх режимах роботи ядра.

Всі подальші дослідження проведени в припущенні, що розмір буфера n = 32 і при визначенні коефіцієнта використання пристроїв использова ни граничні значення.

3.1.2 Дослідження впливу частоти команд переходів і коефіцієнта А

Коефіцієнт А, що дорівнює

$$A = \frac{\mu_n}{\mu} = \frac{t_{вып}}{t_{пер}},$$

визначає співвідношення інтенсивностей виконання команд переходу і команд, виконуваних функціональним пристроєм.

Під часом виконання команди переходу мається на увазі час, який витрачається на читання першої команди нової гілки з підсистеми пам'яті і час, необхідний для заповнення конвеєра до дешифратора команд (ПК), тобто до переходу системи в стаціонарний режим генерації команд нової гілки.

В продуктивних ядрах сучасних процесорів в підсистемі пам'яті є кеш-пам'ять команд. Крім того можуть бути і кеш-пам'яті наступних рівнів (другого і третього, спільні для команд і даних). При значних обсягах кеш-пам'яті команд (наприклад, в процесорах фірми Intel по 32 Кбайта) ймовірність «успіхів» дуже велика. Затримка читання першої команди нової гілки становить 4 такти. Після читання команди з кеш-пам'яті команд вона просувається ще по декількох ступенях конвеєра. Кількість цих ступенів визначається як архітектурою, так і структурою ядра. Для ядер процесорів фірми Intel це 2-3 ступені з часом затримки в 1 такт. Отже, час повернення в стаціонарний режим генерації команд ПК становить 6-7 тактів.

Середній час виконання команд в функціональному пристрої (підсистемі виконання команд) залежить як від структури підсистеми обробки (функціонального пристрою), так і від типу виконуваної в конкретний момент часу команди.

В ядрах сучасних процесорів фірми Intel час виконання команд множення і складання з плаваючою точкою 3-4 такти. Час виконання команди ділення значно більший. Середній час звернення в підсистему пам'яті може змінюватися в дуже широких межах залежно від властивостей локальності конкретної програми.

З урахуванням всього вищезгаданого для дослідження впливу коефіцієнта А розглянуті значення А рівні 0.5, 1, 2.

Залежності коефіцієнтів використання пристроїв від частоти команд переходів і коефіцієнта А при $\rho = 1$, $n = 32$ наведені в таблиці 3.3.

Таблиця 3.3 - Залежності коефіцієнтів використання пристроїв від частоти команд переходів і коефіцієнта А при ρ = 1, n = 32.

A	$H_{ПК}$ / $E_{ФП}$	$\omega_п$			
		0,05	0,1	0,15	0,2
0.05	$H_{ПК}$	0.4601	0.3107	0.2360	0.1903
	$E_{ФП}$	0.4370	0.2799	0.2006	0.1525
0.10	$H_{ПК}$	0.5974	0.4512	0.3651	0.3080
	$E_{ФП}$	0.5676	0. 4061	0.3105	0.2464
0.25	$H_{ПК}$	0.7277	0.6188	0.5444	0.4885
	$E_{ФП}$	0.6915	0.5568	0.4626	0.3908
0.5	$H_{ПК}$	0.7881	0.7062	0.6506	0.6071
	$E_{ФП}$	0.7459	0.6355	0.5530	0.4857
1	$H_{ПК}$	0.8239	0.7596	0.7210	0.6911
	$E_{ФП}$	0.7764	0.6838	0.6127	0.5528
2	$H_{ПК}$	0.8343	0.7897	0.7621	0.7423
	$E_{ФП}$	0.7925	0.7108	0.6477	0.5938

При збільшенні частоти команд переходів коефіцієнти використання пристроїв зменшуються і це зменшення може бути значним.

При збільшенні коефіцієнта А коефіцієнти використання пристроїв збільшуються, причому коефіцієнт збільшення зростає зі збільшенням частоти команд переходів. Це явище пояснюється тим, що збільшення коефіцієнта А означає, що зменшується час входження системи в стаціонарний режим, а, відповідно, зменшуються простої як пристрою керування, так і функціонального пристрою.

При ρ=1 незалежно від значення коефіцієнта А коефіцієнт використання функціонального пристрою при наявності команд умовного переходу дорівнює R .

$$E_{ФУ} = R = \frac{(1+\rho) - \sqrt{(1+\rho)^2 - 4*\omega_o * \rho}}{2} .$$

Точність аналітичних рішень була перевірена точності за допомогою чисельного методу. Результати перевірки точності аналітичних результатів чисельним методом приведені в таблиці 3.4.

Таблиця 3.4 - Перевірка точності аналітичних результатів.

A	$H_{ПК}$ / $E_{ФП}$	$\omega_п$					
		0.1			0.2		
		Аналит.	Чисельні.	Δ (%)	Аналит.	Чисельні.	Δ (%)
0.05	$H_{ПК}$	0.3107	0,3109	0.06	0.1903	0.1906	0.16
	$E_{ФП}$	0.2799	0,2798	0.04	0.1525	0.1525	0
0.10	$H_{ПК}$	0.4512	0,4512	0	0.3080	0,3080	0
	$E_{ФП}$	0. 4061	0,4061	0	0.2464	0,2464	0
0.25	$H_{ПК}$	0.6188	0,6187	0.02	0.4885	0,4885	0
	$E_{ФП}$	0.5568	0,5568	0	0.3908	0,3908	0
0.5	$H_{ПК}$	0.7062	0,7061	0.01	0.6071	0,6071	0
	$E_{ФП}$	0.6355	0,6355	0	0.4857	0,4857	0
1	$H_{ПК}$	0.7596	0,7598	0.03	0.6911	0,6910	0.01
	$E_{ФП}$	0.6838	0,6838	0	0.5528	0,5528	0
2	$H_{ПК}$	0.7897	0,7898	0.01	0.7423	0,7423	0
	$E_{ФП}$	0.7108	0,7108	0	0.5938	0,5938	0

Так як похибка в обчисленнях становить не більше 5%, можливо сміливо стверджувати, що розрахунки аналітичним методом досить точні для використання в подальших дослідженнях.

3.1.3 Висновки з аналізу результатів моделі без блоку передбачення

Коефіцієнт навантаження R на функціональний пристрій при виконанні програм с командами переходу при будь-яких значеннях ρ і $\omega_П > 0$ завжди менший 1.

При проектуванні структури ядра мінімально необхідний розмір буфера повинен бути не менше 8 при різних значеннях коефіцієнта навантаження ρ. Однак, з огляду на, що ядро може виконувати найрізноманітніші програми (в тому числі і такі, в яких дуже мало команд умовних переходів (тобто з $\omega_П$ близьким до нуля)), то, розмір буфера необхідно вибирати з урахуванням самого «важкого» режиму роботи. А таким є режим з ρ = 1 і $\omega_П = 0$. Для такого режиму роботи близькі до граничних значень коефіцієнтів використання забезпечує буфер n = 32. Такий розмір буфера задовольняє вимогам всіх режимів роботи ядра.

При збільшенні частоти команд переходів коефіцієнти використання пристроїв зменшуються і це зменшення може бути значним.

3.2 Аналіз результатів моделі з блоком передбачення

Вирази, необхідні для досліджень, наведені нижче.

$$R = \frac{\left[(1+\rho) - \frac{A_1}{A_1+1} * q * \omega_n * \rho\right] - \sqrt{\left[(1+\rho) - \frac{A_1}{A_1+1} * q * \omega_n * \rho\right]^2 - 4 * \omega_o * \rho}}{2}$$

$$F = \frac{\omega_o * \rho}{1 + \omega_П * \rho * (\frac{q}{1+A_1} + \frac{1-q}{1-R})}$$

$$E_{фУ} = \frac{\omega_0 * \rho}{1-R+F+\frac{\omega_П q \rho}{A_1(1+A_1)}[1+A_1+F)(1-R)+A_1F] + \frac{\omega_П (1-q)\rho}{A_2(1-R)}\left[(1-R+F)(1-R)+A_2F\right]}$$

$$H_{уУ} = \frac{1-R+F}{1-R+F+\frac{\omega_П q \rho}{A_1(1+A_1)}[1+A_1+F)(1-R)+A_1F] + \frac{\omega_П (1-q)\rho}{A_2(1-R)}\left[(1-R+F)(1-R)+A_2F\right]}$$

$$\rho = \frac{\lambda}{\mu} = \frac{t_{вып}}{t_{ген}}, \quad A_1 = \frac{\mu_{п1}}{\mu} = \frac{t_{вып}}{t_{пер1}}, \quad A_2 = \frac{\mu_{п2}}{\mu} = \frac{t_{вып}}{t_{пер2}},$$

В моделі з блоком передбачення з певною ймовірністю q блок передбачення вдало передбачує адресу першої команди нової гілки. Пристрій керування (дешифратор) блокується тільки на час читання першої команди нової гілки + час заповнення конвеєра до дешифратора.

При невдалому прогнозі адреси першої команди нової гілки пристрій керування блокується на час завершення всіх раніше згенерованих команд, які знаходяться на виконанні в підсистемі обробки (функціональному пристрої) +

час читання першої команди нової гілки + час заповнення конвеєра до дешифратора.

Також як і в разі моделі ядра без блоку передбачення коефіцієнти завантаження (використання) функціонального пристрою і пристрою керування є функціями багатьох змінних, зокрема:

коефіцієнта навантаження ρ на функціональний пристрій при виконанні програм без команд переходу - $\rho = \frac{\lambda}{\mu} = \frac{t_{вып}}{t_{ген}}$,

коефіцієнта навантаження R на функціональний пристрій при виконанні програм с командами переходу;

частоти команд переходу $\omega_П = 1- \omega o$;

розміру буфера n;

коефіцієнтів A 1 і A 2, які визначають співвідношення інтенсивностей виконання команд переходу зі станів Сі і Ві в стан Аі та команд, виконуваних функціональним пристроєм;

ймовірності q вдалих прогнозів гілки переходу.

Коефіцієнт навантаження ρ визначається як структурними особливостями ядра, так і типом виконуваної в конкретний момент програми. Він може приймати значення в широкому діапазоні.

Імовірність вдалих прогнозів переходів в сучасних ядрах процесорів заявляється на рівні 0.97-0.98. Однак, реальні значення q дуже істотно визначаються властивостями виконуваних програм. Тому буде досліджуватися зміна q в діапазоні від 0.8 до 0.97.

3.2.1 Дослідження впливу частоти команд переходів і коефіцієнта A

При дослідженні моделі системи як без блоку передбачення, так і з блоком передбачення було встановлено, що коефіцієнт R завжди менше 1 при будь-яких значеннях параметрів моделі. В цьому не складно переконатися, порівнявши відповідні вирази для коефіцієнта R при $\omega_П > 0$.

В таблиці 3.5 наведені значення коефіцієнта R при $A_1 = A_2 = 1$; q = 0.8 і q = 0.95 при різних значеннях коефіцієнта ρ і частоти команд переходу.

Таблиця 3.5 - Значення коефіцієнта R при $A_1 = A_2 = 1$; q = 0.8 і q = 0.95 при різних значеннях коефіцієнта ρ і частоти команд переходу.

q	ρ		ω п			
			0.05	0.10	0.15	0.20
0.8	0.5	R	0.4621	0.4276	0.3956	0.3654
	1	R	0.8165	0.7342	0.6685	0.6113
	2	R	0.9411	0.8842	0.8288	0.7747
0.95	0.5	R	0.4367	0.4302	0.3989	0.3693
	1	R	0.8257	0.7459	0.6815	0. 6250
	2	R	0.9477	0.8959	0.8445	0.7934

Як і для моделі системи без блоку передбачення коефіцієнт навантаження R на функціональний пристрій при виконанні програм с командами переходу при будь-яких значеннях параметрів при $\omega_П > 0$ завжди менше 1 і менше ρ .

Аналіз залежності коефіцієнтів використання функціонального пристрою $E_{ФП}$ та пристрою керування $H_{ПК}$ від розміру буфера показує, що вже при розмірі буфера $n \geq 8$ значення коефіцієнтів використання пристроїв дуже близькі до граничних значень і практично збігаються з ними. Це наслідок того, що коефіцієнт R <1 при будь-яких значеннях змінних і $\lim_{n\to\infty} R^n = 0.$

3.2.2 Дослідження моделі системи з блоком передбачення, коли читання першої команди нової гілки проводиться з кеш-пам'яті команд

В таблиці 3.6 наведені залежності коефіцієнтів використання пристроїв від частоти команд переходів $\omega_п$, коефіцієнтів А, ймовірності вдалих прогнозів q при ρ = 1, n = 32.

Таблиця 3.6 - Залежності коефіцієнтів використання пристроїв від частоти команд переходів, коефіцієнтів А і ймовірності вдалих прогнозів q при ρ = 1, n = 32.

q	$A_1 = A_2$	$\omega_п$	0,05	0,1	0,15	0,2	0,25
	0.125	$H_{ПК}$	0. 7974	0. 6530	0. 5423	0. 4578	0. 3925
		$E_{ФП}$	0.6904	0.5244	0.4086	0.3249	0.2625
		$H_{ПК}$	0. 8665	0. 7741	0. 6925	0. 6218	0. 5611

0.8	0.25	$E_{ФП}$	0.7543	0.6244	0.5228	0.4409	0.3736
	0.5	$H_{ПК}$	0.9039	0.8364	0.7765	0.7233	0.6757
		$E_{ФП}$	0.7949	0.6935	0.6103	0.5382	0.4749
	1	$H_{ПК}$	0.9256	0.8820	0.8437	0.8100	0.7889
		$E_{ФП}$	0.8227	0.7395	0.6703	0.6086	0.5519
	2	$H_{ПК}$	0.9337	0.9049	0.8818	0.8612	0.8431
		$E_{ФП}$	0.8436	0.7711	0.7104	0.6556	0.6044
0.9	0.125	$H_{ПК}$	0. 8288	0. 6852	0. 5702	0. 4807	0. 4111
		$E_{ФП}$	0.7057	0.5389	0.4196	0.3328	0.2680
	0.25	$H_{ПК}$	0. 8932	0. 8053	0. 7229	0. 6497	0. 5859
		$E_{ФП}$	0.7652	0.6362	0.5333	0. 4496	0.3806
	0.5	$H_{ПК}$	0.9249	0.8603	0.8003	0.7454	0.6954
		$E_{ФП}$	0.8026	0.7032	0.6201	0.5474	0.4830
	1	$H_{ПК}$	0.9558	0.9052	0.8691	0.8348	0.8008
		$E_{ФП}$	0.8302	0.7488	0.6803	0.6185	0.5612
	2	$H_{ПК}$	0.9522	0.8505	0.9087	0.8884	0.8697
		$E_{ФП}$	0.8525	0.7819	0.7222	0.6673	0.6156
0.95	0.08	$H_{ПК}$	0. 7839	0. 6022	0. 4721	0. 3785	0. 3123
		$E_{ФП}$	0.6610	0.4688	0.3444	0.2614	0.2037
	0.125	$H_{ПК}$	0.8298	0.6793	0.5 7 77	0.4648	0.3946
		$E_{ФП}$	0.7137	0.5463	0.4253	0.3370	0.2710
	0.25	$H_{ПК}$	0. 9054	0. 8219	0. 7282	0. 6547	0. 5989
		$E_{ФП}$	0.7705	0.6423	0.5389	0.4543	0.3843
	0.5	$H_{ПК}$	0.8142	0.8 13 6	0.8 27 3	0.7576	0.7064
		$E_{ФП}$	0.8066	0.7082	0.6522	0.5522	0.4873
	1	$H_{ПК}$	0.9549	0.9202	0.88 4 8	0.8484	0.6266
		$E_{ФП}$	0.8342	0.7539	0.6857	0.6237	0.5653
	2	$H_{ПК}$	0.9677	0.9454	0.9242	0.9042	0.8848
		$E_{ФП}$	0.8573	0.7879	0.7286	0.6739	0.6220

Була проведена перевірка точності аналітичних результатів за допомогою чисельного методу.

У таблиці 3.7 наведені результати перевірк точності аналітичних рішень в порівнянні з чисельним методом.

Таблиця 3.7 - Перевірка точності аналітичних рішень в порівнянні з чисельним методом.

q	$A_1 = A_2$	$H_{ПК} / E_{ФП}$	$\omega_п$

		/ R	0,05			0,15		
			Аналит.	Чисельні.	Δ (%)	Аналит.	Чисельні.	Δ (%)
0.8	0.125	Н ПК	0. 7974	0,8174	2.5 1	0. 5423	0,5623	3.6 9
		Е ФП	0.6904	0,6904	0	0.4086	0,4086	0
	0.25	Н ПК	0. 8665	0,8865	2.31	0. 6925	0,7125	2.8 9
		Е ФП	0.7543	0,7543	0	0.5228	0,5228	0
	0.5	Н ПК	0.9039	0,9235	2.1 7	0.7 8 65	0,8185	4.07
		Е ФП	0.7949	0,7949	0	0.6103	0,6103	0
	1	Н ПК	0.9256	0,9397	1.52	0.8437	0,8791	4. 20
		Е ФП	0.8227	0,8227	0	0.6703	0,6703	0
	2	Н ПК	0.9337	0,9441	1.11	0.8818	0,9066	2.81
		Е ФП	0.8436	0,8436	0	0.7104	0,7104	0
0.9	0.125	Н ПК	0. 8288	0,8488	2.41	0. 5702	0,5902	3.5 1
		Е ФП	0.7057	0,7050	0. 10	0.4196	0,4194	0.0 5
	0.25	Н ПК	0. 8932	0,9132	2.2 4	0. 7229	0,7429	2.7 7
		Е ФП	0.7652	0,7642	0	0.5333	0,5332	0.0 2
	0.5	Н ПК	0.9249	0,9474	2.43	0.8 1 03	0,8498	4.87
		Е ФП	0.8026	0,8026	0	0.6201	0,6201	0
	1	Н ПК	0.9558	0,9626	0.71	0.8691	0,9105	4.76
		Е ФП	0.8302	0,8302	0	0.6803	0,6803	0
	2	Н ПК	0.9522	0,9671	1.56	0.9087	0,9384	3.27
		Е ФП	0.8525	0,8525	0	0.7222	0,7222	0

Так як похибка в обчисленнях становить не більше 5%, то можемо стверджувати, що аналітичні розрахунки і формули досить точні для використання в подальших дослідженнях.

3.2.2.1 Висновки

Якщо в високопродуктивному ядрі є блок передбачення переходів, а команди нової гілки читаються з кеш-пам'яті команд з ймовірністю дуже близькою до 1, то значення коефіцієнтів A_1 і A_2 значно менші 1 і лежать в діапазоні 0.08 - 0.25.

Чим менше значення коефіцієнтів A_1 і A_2 (а вони рівні, так як в обох випадках читання команд здійснюється з кеш-пам'яті команд), тим нижче коефіцієнти використання пристроїв.

При збільшенні частоти команд переходів $\omega_п$ коефіцієнти використання пристроїв зменшуються, причому істотно.

Зі збільшенням частоти вдалих прогнозів q коефіцієнти використання пристроїв збільшуються не суттєво.

У високопродуктивних ядрах для компенсації негативного впливу команд переходу на продуктивність недостатньо наявності тільки блоку передбачення переходів навіть з хорошими показниками ймовірності вдалих переходів. Необхідні структурні рішення, які забезпечили б часи виконання команд переходу при вдалих прогнозах близькі до часів виконання команд в підсистемі обробки (в функціональному пристрої).

3.2.3 Дослідження моделі з блоком передбачення і буфером продешіфрірованних команд

При вдалому прогнозі переходу команди нової гілки читаються з спеціального буфера, а не з кеш-пам'яті команд. Час читання першої команди нової гілки з кеш-пам'яті команд (затримка) в ядрах процесорів фірми Intel дорівнює 4 тактів. Час читання першої команди нової гілки з буфера продешифрованних команд в може бути зменшеий до 1 - 2 тактів.

З врахуванням часу заповнення конвеєра до дешифратора команд 2-3 такти час виконання команди переходу складе 3-5 тактів.

При читанні першої команди нової гілки з кеш-пам'яті команд час виконання команди переходу з врахуванням часу заповнення конвеєра до дешифратора команд складе 6-7 тактів.

У кращому випадку затримка зменшиться в 2 рази. Це означає, що коефіцієнт A_1 при використанні спеціального буфера прдешифрованих команд збільшиться в 2 рази в порівнянні з читанням команд з кеш-пам'яті команд.

Коефіцієнт A_2 залишається таким же, як і в попередніх дослідженнях, так як при невдалому прогнозі переходу команди нової гілки читаються з кеш-пам'яті команд.

Залежності коефіцієнтів використання пристроїв від частоти команд переходів $\omega_п$, коефіцієнтів A_1, A_2 і ймовірності вдалих прогнозів q при $\rho = 1$, $n = 32$, наведені в таблиці 3.8 .

Таблиця 3.8 - Залежності коефіцієнтів використання пристроїв від частоти команд переходів $\omega_п$, коефіцієнтів A_1, A_2 і ймовірності вдалих прогнозів q при $\rho = 1$, $n = 32$.

q	A_1, A_2	$\omega_п$	0,05	0,1	0,15	0,2
0.8	A_1=0.05 A_2=0.025	$Н_{ПК}$	0.5671	0.3710	0.2678	0.2062
		$Е_{ФП}$	0.5645	0.3580	0.2457	0.1786
	A_1=0.1 A_2=0.05	$Н_{ПК}$	0.7098	0.5307	0.4151	0.3371
		$Е_{ФП}$	0.6746	0.4928	0. 3708	0.2868
	A_1=0.15 A_2=0.075	$Н_{ПК}$	0.7758	0.6221	0.5250	0.4446
		$Е_{ФП}$	0.6542	0.4892	0.3770	0.2966
0.9	A_1=A_2=0.5	$Н_{ПК}$	0.9249	0.8603	0.8003	0.7454
		$Е_{ФП}$	0.8026	0.7032	0.6201	0.5474
	A_1=A_2=1	$Н_{ПК}$	0.9558	0.9052	0.8691	0.8348
		$Е_{ФП}$	0.8302	0.7488	0.6803	0.6185
	A_1=A_2=2	$Н_{ПК}$	0.9522	0.8505	0.9087	0.8884
		$Е_{ФП}$	0.8525	0.7819	0.7222	0.6673
0.95	A_1=A_2=0.5	$Н_{ПК}$	0.9402	0.8716	0.8133	0.7576
		$Е_{ФП}$	0.8066	0.7082	0.6522	0.5522
	A_1=A_2=1	$Н_{ПК}$	0.9549	0.9202	0.8828	0.8484
		$Е_{ФП}$	0.8342	0.7539	0.6857	0.6237
	A_1=A_2=2	$Н_{ПК}$	0.9677	0.9454	0.9242	0.9042
		$Е_{ФП}$	0.8573	0.7879	0.7286	0.6739

Значення коефіцієнта використання функціонального пристрою $Е_{ФП}$ при різних значеннях $\omega_П$; q ; $A_1 = A_2 = 1$, в залежності від значення ρ, розміру буфера n наведені в таблиці 3.9 .

Таблиця 3.9 - Значення коефіцієнта використання функціонального пристрою $Е_{ФП}$ при $\omega_П = 0.20$; $A_1 = A_2 = 1$, q = 0.8 в залежності від значення ρ, розміру буфера n.

	n = 2	n = 4	n = 8	n = 16	n = 32	n = 64
	$\omega_П = 0.20$; $A_1 = A_2 = 1$, q = 0.8					
ρ = 0.5	0.3501	0.3581	0.3588	0.3588	0. 3588	0. 3588
ρ = 1	0.5739	0.6063	0.60 7 7	0.6086	0.6086	0.6086

ρ = 2	0.7673	0. 768 2	0.7 69 7	0.78 2 7	0 .7851	0.7851
	ω П = 0.05; A 1 = A 2 = 1, q = 0.8					
ρ = 0.5	0.4365	0.4561	0.4604	0.4606	0.4606	0.4606
ρ = 1	0.7070	0.7764	0.8150	0.8227	0.8227	0.8227
ρ = 2	0.8976	0.9505	0.9 57 5	0.9597	0.95 97	0.95 9 7
	ω П = 0.20; A 1 = A 2 = 1, q = 0.95					
ρ = 0.5	0.3523	0.3606	0.3613	0.3613	0. 3613	0. 3613
ρ = 1	0.5832	0.6221	0.62 2 5	0. 623 3	0.6237	0.6237
ρ = 2	0.7898	0. 794 0	0.8 0 31	0.81 2 2	0. 8152	0.8152
	ω П = 0.05; A 1 = A 2 = 1, q = 0.95					
ρ = 0.5	0.4374	0.4573	0.4618	0.4619	0.4619	0.4619
ρ = 1	0.7103	0.7831	0.8250	0.834 0	0.8340	0.8340
ρ = 2	0.9043	0.9616	0.9 624	0.9 6 36	0.96 4 0	0.9654

3.3 Порівняння результатів моделей

У таблиці 3.10 наведені значення коефіцієнтів використання функціонального пристрою для 3 моделей (без блоку передбачення, з блоком передбачення і без буфера продешифрованих команд ($L1K$), з блоком передбачення і з буфером продешифрованих команд ($L0$) при однакових значеннях параметрів A_2, q, ω_n, $\rho = 1$, $n = 32$.

Таблиця 3.10 - Порівняння значень коефіцієнтів використання функціонального пристрою для 3 моделей

q	A_1, A_2	ω_n			
		0,05	0,1	0,15	0,2
Модель ядра без блоку передбачення					
0	A_2 =0.05	0.459	0.312	0.241	0.200
Модель ядра ($L1K$) с блоком передбачення та без буфера $L0$					
0.95	A_1 =0.05; A_2 =0.05	0. 588	0.379	0.261	0.190

0.98	A_1=0.05; A_2=0.05	0.597	0.384	0.265	0.192
Модель ядра ($L0$) с блоком передбачення та буфером $L0$					
0.95	A_1=0.15; A_2=0.05	0.709	0.548	0.431	0.345
0.98	A_1=0.15; A_2=0.05	0.726	0.568	0.450	0.361
Відношення коефіцієнтів використання					
0.95	$E_{ФПL0}$/ $E_{ФПL1К}$	1.205	1.446	1.651	1.816
0.98	$E_{ФПL0}$/ $E_{ФПL1К}$	1.216	1.479	1.698	1.880

Отже, варіант системи з з блоком передбачення і з буфером продешифрованих команд (з буфером L 0) є найбільш ефективним (в 1.2 -1.88 разів у порівнянні з варіантом з блоком передбачення, але без буфера L0 продешіфрірованних команд).

3.4 Висновки

На підставі аналізу функціонування конвеєрних суперскалярних ядер сучасних процесорів при наявності «перешкод» плавному перебігу конвеєра підготовки команд у вигляді команд умовних і безумовних переходів (залежностей з управління) розроблена модель функціонування ядра при використанні різних структурних методів компенсації втрат продуктивності.

Вирішені рівняння і отримані аналітичні залежності показників ефективності (коефіцієнтів використання пристроїв) від різних параметрів (розміру буфера, коефіцієнтів перезавантаження конвеєра підготовки команд , частоти команд переходів, ймовірності «вдалих» передбачень напрямків переходів) наступних структур ядра: структура без блоку передбачення; структура з блоком передбачень і без буфера продешифрованих команд; структура з блоком передбачень і з буфером продешифрованих команд.

Проведено порівняння ефективності різних структурних рішень по компенсації негативного впливу команд переходів на продуктивність ядра. У високопродуктивних ядрах для компенсації негативного впливу команд переходу на продуктивність недостатньо наявності тільки блоку передбачення переходів навіть з хорошими показниками ймовірності «успішних» передбачень переходів. Необхідні структурні рішення, які забезпечили б

мінімальні часи перезавантаження конвеєра команд при «успішних» прогнозах переходів близькі до часів генерації команд дешифратором (наприклад, реалізація буфера продешифрованих команд).

Результати дослідження можуть бути корисні при оптимальному виборі структури ядра з урахуванням співвідношення як показників ефективності, так і апаратних витрат на реалізацію.

4 ОХОРОНА ПРАЦІ

В даному розділі проведено аналіз потенційних небезпечних та шкідливих виробничих факторів, причин пожеж. Розглянуті заходи, які дозволяють забезпечити гігієну праці і виробничу санітарію. На підставі аналізу розроблені заходи з техніки безпеки та рекомендації з пожежної профілактики.

Завданням даного проекту бакалавра була розробка мобільного металодетектора та додатку для дистанційного керування ним. Так як процес розробки виконувався у домашніх умовах, то аналіз потенційно небезпечних і шкідливих виробничих чинників виконується для приміщення, де проводились роботи над дипломним проектом.

4.1 Аналіз стану умов праці

Робота над створенням мобільного металодетектора проходить в побутовому приміщенні. Для даної роботи достатньо однієї людини, для якої надано робоче місце зі стаціонарним комп’ютером.

4.1.1 Вимоги до приміщення

Геометричні розміри приміщення зазначені у таблиці 4.1. Для зручності спільної роботи з іншими працівниками (обговорення ідей, з’ясування проблем і т.д.) в кімнаті є диван і журнальний стіл. Задля дотримання визначеного рівня мікроклімату в будівлі встановлено систему опалення та кондиціонування.

Для забезпечення потрібного рівного освітленості кімната має вікно та систему загального рівномірного освітлення, що встановлена на стелі. Для дотримання вимог пожежної безпеки встановлено порошковий вогнегасник та систему автоматичної пожежної сигналізації.

Таблиця 4.1 – Розміри робочого місця

Параметр	Значення
Довжина, м	6
Ширина, м	4
Висота, м	2,5
Площа, $м^2$	24
Об’єм, $м^3$	60

Згідно до санітарних норм мікроклімату виробничих приміщень [13] розмір площі для одного робочого місця оператора персонального комп’ютера має бути не менше 6 кв. м, а об’єм – не менше 20 куб. м. Отже, дане приміщення цілком відповідає зазначеним нормам.

4.1.2 Вимоги до організації робочого місця

При порівнянні відповідності характеристик робочого місця нормативним основні вимоги до організації робочого місця [14] (табл. 5.2) і відповідними фактичними значеннями для робочого місця, констатуємо повну відповідність.

Таблиця 4.2 – Характеристика робочого місця

Найменування параметра	Фактичне значення	Нормативне значення
Висота робочої поверхні, мм	750	680 ÷800
Висота простору для ніг, мм	730	не менше 600
Ширина простору для ніг, мм	660	не менше 500
Глибина простору для ніг, мм	700	не менше 650

Висота поверхні сидіння, мм	470	400 ÷500
Ширина сидіння, мм	400	не менше 400
Глибина сидіння, мм	400	не менше 400
Висота поверхні спинки, мм	600	не менше 300
Ширина опорної поверхні спинки, мм	500	не менше 380
Радіус кривини спинки в горизонтальній площині, мм	400	400
Відстань від очей до екрану дисплея, мм	800	700 ÷800

Робочий стіл на досліджуваному місці також містить достатньо простору для ніг. Крісло, що використовується в якості робочого сидіння, є підйємно-поворотним, має підлокітники і можливість регулювання за висотою і кутом нахилу спинки, також воно м'яке і виконане з екологічної шкіри, що дає можливість працювати у комфорті. Екран монітору знаходиться на відстані 0.8 м, клавіатура має можливість регулювання кута нахилу 5-15°. Отже, за всіма параметрами робоче місце відповідає нормативним вимогам. Приміщення кабінету знаходиться на першому поверсі двох поверхової будівлі і має об’єм 60 м^3, площу – 24 м^2. У цьому кабінеті обладнано одне робоче місце, яке укомплектовано 2 ПК, один з котрих сервер без наявності пристроїв I/O інформації.

Температура в приміщенні протягом року коливається у межах 18–24°C, відносна вологість — близько 50%. Швидкість руху повітря не перевищує 0,2 м/с. Шум у приміщенні знаходиться на рівні 50 дБА. Система вентилювання приміщення — природна, а опалення — централізоване.

Розміщення вікон забезпечує природне освітлення з коефіцієнтом природного освітлення не менше 1,5%, а загальне штучне освітлення, яке здійснюється за допомогою восьми люмінесцентних ламп, забезпечує рівень освітленості не менше 200 Лк.

У кабінеті є електрична мережа з напругою 220 В, яка створює небезпеку ураження електричним струмом. ПК та периферійні пристрої

можуть бути джерелами електромагнітних випромінювань, аерозолів та шкідливих речовин (часток тонеру, оксидів нітрогену та озону).

За ступенем пожежної безпеки приміщення належить до категорії В. Кабінет оснащений переносним вуглекислотним вогнегасником ВВК-5 .

Наявна аптечка для надання долікарської допомоги, а також у кабінеті роблять вологе прибирання та щоденно провітрюють приміщення.

4.1.3 Навантаження та напруженість процесу праці

За фізичним навантаженням робота відноситься до категорії легкі роботи (Іа), її виконують сидячи з періодичним ходінням. Щодо характеру організування виконання дипломної роботи, то він підпадає під нав'язаний режим, оскільки певні розділи роботи необхідно виконати у встановлені конкретні терміни. За ступенем нервово-психічної напруги виконання роботи можна віднести до ІІ – ІІІ ступеня і кваліфікувати як помірно напружений – напружений за умови успішного виконання поставлених завдань.

Під час виконання робіт використовують ПК та периферійні пристрої (лазерні та струменеві), що призводить до навантаження на окремі системи організму. Такі перекоси у напруженні різних систем організму, що трапляються під час роботи з ПК, зокрема, значна напруженість зорового аналізатора і довготривале малорухоме положення перед екраном, не тільки не зменшують загального напруження, а навпаки, призводять до його посилення і появи стресових реакцій.

Найбільшому ризику виникнення різноманітних порушень піддаються: органи зору, м'язово скелетна система, нервово-психічна діяльність, репродуктивна функція у жінок.

Тобто наявне психофізіологічні небезпечні та шкідливі фактори:

а) фізичного перевантаження:

- статичного;

- динамічного;

б) нервово-психічного перевантаження:

- розумового перенапруження;

- монотонності праці;

- перенапруження аналізаторів;

- емоційних перевантажень.

Рекомендовано застосування екранних фільтрів, локальних світлофільтрів (засобів індивідуального захисту очей) та інших засобів захисту, а також інші профілактичні заходи [14].

Роботу за дипломним проектом визнано, таку, що займає 50% часу робочого дня та за восьмигодинної робочої зміни рекомендовано встановити додаткові регламентовані перерви тривалістю 15 хв через кожну годину роботи;

4.2 Виробнича санітарія

На підставі аналізу небезпечних та шкідливих факторів при виробництві (експлуатації), пожежної безпеки можуть бути надалі вирішені питання необхідності забезпечення працюючих достатньою кількістю освітлення, вентиляції повітря, організації заземлення, тощо.

4.2.1 Аналіз небезпечних та шкідливих факторів при розробці виробу

Аналіз небезпечних та шкідливих виробничих факторів виконується у табличній формі (табл. 4.3). Роботу, пов'язану з ЕОП з ВДТ, у тому числі на тих, які мають робочі місця, обладнані ЕОМ з ВДТ і ПП, виконують із забезпеченням виконання , які встановлюють вимоги безпеки до обладнання робочих місць, до роботи із застосуванням ЕОМ з ВДТ і ПП. Переважно роботи за проектами виконують у кабінетах чи інших приміщеннях, де використовують різноманітне електрообладнання, зокрема персональні комп’ютери (ПК) та периферійні пристрої. Основними робочими характеристиками персонального комп'ютера є:

– робоча напруга U = +220В +-5%;

– робочий струм I = 2А;
– споживана потужність P = 600 Вт.

Таблиця 4.3 – Аналіз небезпечних і шкідливих виробничих факторів

Небезпечні і шкідливі виробничі фактори	Джерела факторів (види робіт)	Нормативні документи
Фізичні		
- підвищена температура поверхонь обладнання	експлуатація ЕОМ, серверного обладнання для роботи	[13]
- підвищена або знижена вологість повітря	-//-	[13]
- підвищена або знижена рухливість повітря	-//-	[13]
- підвищений рівень напруги електричної мережі	-//-	[16] [17]
- підвищений рівень статичної електрики	-//-	[115]
- підвищена напруженість електромагнітного поля	-//-	[15]
- недостатність природного світла	порушення умов праці (вимог до приміщень)	[15]
- недостатнє освітлення робочої зони	порушення гігієнічних параметрів виробничого середовища	[16]
Психофізіологічні		
-нервово-психічна перевантаження	Розумова робота над проектом	[11]

		[14]
- фізичні (статичне – сидіння)	порушення умов праці та організації робочого часу	[11]

Робочі місця мають відповідати вимогам Державних санітарних правил і норм роботи з візуальними дисплейними терміналами електронно-обчислювальних машин, затверджених постановою Головного державного санітарного лікаря України від 10.12.98 [14]. За умов роботи з ПК виникають наступні небезпечні та шкідливі чинники: несприятливі мікрокліматичні умови, освітлення, електромагнітні випромінювання, забруднення повітря шкідливими речовинами (джерелом, яких можуть бути: принтер, сканер та інші джерела виділення багатьох хімічних речовин - напр., озону, оксидів азоту та аерозолів високодисперсних частинок тонера), шум, вібрація, електричний струм, електростатичне поле, напруженість трудового процесу та інше.

4.2.2 Пожежна безпека

Небезпека розвитку пожежі на обчислювальному центрі обумовлюється застосуванням розгалужених систем електроживлення ЕОМ, вентиляції і кондиціювання. Небезпека загоряння пов'язана з особливістю комп'ютерів – із значною кількістю щільно розташованих на монтажній платі і блоках електронних вузлів і схем, електричних і комутаційних кабелів, резисторів, конденсаторів, напівпровідникових діодів і транзисторів. Надійна робота окремих елементів і мікросхем в цілому забезпечується тільки в певних інтервалах температури, вологості і при заданих електричних параметрах. При відхиленні реальних умов експлуатації від розрахункових можуть виникнути пожежонебезпечні ситуації.

Висока щільність елементів в електронних схемах призводить до значного підвищення температури окремих вузлів (80…100°*C*). При проходженні електричного струму по провідниках і деталей виділяється тепло, що в умовах їх високої щільності може привести до перегріву, і може

служити причиною запалювання ізоляційних матеріалів. Слабкий опір ізоляційних матеріалів дії температури може викликати порушення ізоляції і привести до короткого замикання між струмоведучими частинами обладнання (шини, електроди). Також ймовірна небезпека внаслідок перевантаження напруги, розрядки зарядів статичної електрики, пошкодження обладнання та електропроводки. Електростатичний розряд виникає під час тертя двох ізольованих матеріалів. Розряд статичної електрики може виникнути під час роботи вентилятора або комп'ютера. Кабельні лінії є найбільш пожежонебезпечними місцем. Наявність пального ізоляційного матеріалу, ймовірних джерел запалювання у вигляді електричних іскор і дуг, розгалуженість і недоступність роблять кабельні лінії місцем найбільш ймовірного виникнення і розвитку пожежі. Для зниження займистості і здатності поширювати полум'я кабелі покривають вогнезахисними покриттями.

Для гасіння пожеж в офісному приміщенні пропонується використовувати порошкові або вуглекислотні вогнегасники, так як вони є універсальними.

Виникнення пожежі можливе, якщо на об'єкті є горючі речовини, окиснювач і джерела запалювання. Вірогідність пожежної небезпеки приймається значною, якщо ймовірна взаємодія цих трьох чинників. Горючими компонентами є: будівельні матеріали для акустичної і естетичної обробки приміщень, перегородки, підлоги, двері, ізоляція силових, сигнальних кабелів і т.д.

Горючими матеріалами в приміщенні, де розташовані ЕОМ, є:

1) поліамід – матеріал корпусу мікросхем, горюча речовина, температура самозаймання 420°*C*,
2) полівінілхлорид – ізоляційний матеріал, горюча речовина, температура запалювання 335°*C*, температура самозаймання 530°*C*,
3) склотекстоліт ДЦ – матеріал друкарських плат, важкогорючий матеріал, показник горючості 1.74, не схильний до температурного самозаймання,
4) пластикат кабельний – матеріал ізоляції кабелів, горючий матеріал, показник горючості більше 2.1,
5) деревина – будівельний і обробний матеріал, з якого виготовлені меблі, горючий матеріал, показник горючості більше 2.1, температура запалювання

255°*C*, температура самозаймання 399°*C*.

Для відводу теплоти від ЕОМ діє потужна система кондиціонування. Тому кисень, як окиснювач процесів горіння, є в будь-якій точці приміщень обчислювального центру.

Простори усередині приміщень в межах, яких можуть утворюватися або знаходиться пожежонебезпечні речовини і матеріали відповідно до [19] відносяться до пожежонебезпечної зони класу П-IIа. Це обумовлено тим, що в приміщенні знаходяться тверді горючі та важкозаймисті речовини та матеріали. Приміщенню, у якому розташоване робоче місце, присвоюється II ступень вогнестійкості.

Потенційними джерелами запалювання можуть бути:

- іскри і дуги короткого замикання;
- електрична іскра при замиканні і розмиканні ланцюгів;
- перегріви від тривалого перевантаження,
- відкритий вогонь і продукти горіння,
- наявність речовин, нагрітих вище за температуру самозаймання,
- розрядна статична електрика.

Причинами можливого загоряння і пожежі можуть бути:

- несправність електроустановки;
- конструктивні недоліки устаткування;
- коротке замикання в електричних мережах;
- запалювання горючих матеріалів, що знаходяться в безпосередній близькості від електроустановки.

Продуктами згорання, що виділяються на пожежі, є: окис вуглецю; сірчистий газ; окис азоту; синильна кислота; акромін; фосген; хлор і ін. При горінні пластмас, окрім звичних продуктів згорання, виділяються різні продукти термічного розкладання: хлорангідридні кислоти, формальдегіди, хлористий водень, синильна кислота, аміак, ацетон та ін.

4.2.3 Електробезпека

На робочому місці виконуються наступні вимоги електробезпеки: ПК, периферійні пристрої та устаткування для обслуговування, електропроводи і

кабелі за виконанням та ступенем захисту відповідають класу зони за ПУЕ (правила улаштування електроустановок), мають апаратуру захисту від струму короткого замикання та інших аварійних режимів. Лінія електромережі для живлення ПК, периферійних пристроїв і устаткування для обслуговування, виконана як окрема групова три- провідна мережа, шляхом прокладання фазового, нульового робочого та нульового захисного провідників. Нульовий захисний провідник використовується для заземлення (занулення) електроприймачів. Штепсельні з'єднання та електророзетки крім контактів фазового та нульового робочого провідників мають спеціальні контакти для підключення нульового захисного провідника. Електромережа штепсельних розеток для живлення персональних ПК, укладено по підлозі поруч зі стінами відповідно до затвердженого плану розміщення обладнання та технічних характеристик обладнання. Металеві труби та гнучкі металеві рукави заземлені. Захисне заземлення включає в себе заземлюючих пристроїв і провідник, який з'єднує заземлюючий пристрій з обладнанням, яке заземлюється - заземлюючий провідник.

4.3 Висновки

В результаті проведеної роботи було зроблено аналіз умов праці, шкідливих та небезпечних чинників, з якими стикається робітник. Було визначено параметри і певні характеристики приміщення для роботи над запропонованим проектом написаному в кваліфікаційній роботі, описано, які заходи потрібно зробити для того, щоб дане приміщення відповідало необхідним нормам і було комфортним і безпечним для робітника.

Приведені рекомендації щодо організації робочого місця, а також важливу інформацію щодо пожежної та електробезпеки. Також були наведені інструкції з охорони праці, техніки безпеки при роботі на комп’ютері.

ЗАКЛЮЧЕННЯ

В роботі розглянуті та проаналізовані структури ядер сучасних процесорів.

Проаналізовані типи перешкод, причини їх виникнення та методи боротьби з ними.

Розглянуті методи підвищення продуктивності процесорів.

Розроблені моделі ядер процесора з блоком передбачення переходів і без блоку передбачення переходів.

Були складені графіки станів ядра процесора з блоком передбачення переходів і без блоку передбачення переходів. На основі графів станів ядра процесора з блоком передбачення переходів була складені та вирішені системи рівнянь для знаходження ймовірностей знаходження моделі в різних станах, а також написана програма, яка вирішує дану систему рівнянь методом Гаусса.

Було проведено дослідження структур ядра з блоком передбачень і без буфера продешифрованних команд , з блоком передбачень і з буфером продешифрованих команд і без блоку передбачення переходів.

Проведено порівняння ефективності різних структурних рішень по компенсації негативного впливу команд переходів на продуктивність ядра.

Результати дослідження можуть бути корисні при оптимальному виборі структури ядра з урахуванням співвідношення як показників ефективності, так і апаратних витрат на реалізацію.

Було встановлено що, структура ядра з блоком передбачення переходів і з блоком продешефрірованних команд є найбільш ефективною.

Література

1. Рязанцев О.І., Недзельський Д.О. «Архітектурна та структура комп'ютерів», Сєвєродонецьк, 2017, 365 сторінок.

2. Паттерсон Д. Архитектура компьютера и проектирование компьютерных систем. 4-е изд., Пер. с англ., СПб.: Питер, 2011. -784 с.: ил.

3. Столлингс В. Структурная организация и архитектура компьютерных систем,5-е изд.; Пер. с англ. – М.: Издательский дом «Вильямс», 2002. – 896 с.: ил.

4. Таненбаум Э. Архитектура компьютера, 6 изд., Пер. с англ., СПб.: Питер, 2013. -816 с.: ил.

5. Хамахер К., Вранешевич З., Заки С. Организация ЭВМ, 5-е изд., - СПб.: Питер, Киев. Издательская группа BHV, 2003. – 848 с.: ил.

6. Орлов С.А., Цилькер Б.Я. Организация ЭВМ и систем. 2-е изд. – СПб.: Питер, 2011. – 688 с.

7. 2. Недзельский Д.А. Исследование эффективности одноядерных суперскалярных вычислительных систем. Луганск. Вісник Східноукраїнського національного університету ім. В. Даля. - 2011. - №15 (169) Ч. 2. - с. 133-140.

8. Грищенко В.И., Ладыженский Ю.В., Юнис Моатаз. Влияние выделенного кэша команд на производительность сетевого процессора. Наукові праці ДонНТУ. Серія "Інформатика, кібернетика та обчислювальна техніка". Випуск 13(185), 2011 с. 85-91.

9. Грищенко В.И. Моделирование маршрутизаторов на многоядерных сетевых процессорах. Наукові праці ДонНТУ. Серія «Інформатика, кібернетика і обчислювальна техніка». – 2010. – Вип. 12(165). – С. 169-177.

10. Клейнрок Л. Вычислительные системы с очередями. -М.: Мир, 1979. - 600с.

11. НПАОП 0.00.-1.28-10 «Правил охорони праці під час експлуатації електронно-обчислювальних машин»;

12. НПАОП 0.00-4.15-98 «Положення про розробку інструкцій з охорони праці»;

13. ДСН 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень» ;

14. ДСанПіН 3.3.2.007-98 «Правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин»;
15. ГОСТ 12.1.044-89 «ССБТ. Вогнестійкість. Номенклатура показників і методи їх визначення»;
16. ГОСТ 12.1.030-81 «Електробезпека. Захисне заземлення, занулення».
17. ГОСТ 13109-97 «Електрична енергія. Сумісність технічних засобів. Норми якості електричної енергії в системах електропостачання загального призначення»;
18. ДБН В.2.5-28:2015 «Державні Будівельні Норми України. Природне і штучне освітлення»;
19. НАПБ Б.03.002-2007 «Норми визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою».
20. ДБН В.2.5-67:2013. «Опалення, вентиляція та кондиціонування.»

Додаток А

Код програми:

```
#include <iostream>
# Include < math . h >
# Include < stdlib . h >
# Include < time . h >
using namespace std ;
int main ()
{
   int i, j, n, m, h = 0, z = 0, s = 0, r = 0, e;
long double s1 = 0, s2 = 0, A1, A2, Wn, Wo, P, Q, w;
   // Введення ключових змінних і створення масиву
   cout << "A1:";
cin >> A1;
cout << "A2:";
cin >> A2;
cout << "Wn:";
cin >> Wn;
cout << "P:";
cin >> P;
cout << "Q:";
   cin >> Q;
cout << "Розмір буфера:";
   cin >> n;
Wo = 1-Wn;
m = n;
n = (n + 2) + ((n + 1) * 2);
e = n + 1;
float ** matrix = new float * [n];
for (i = 0; i <n; i ++)
matrix [i] = new float [e];
// Заповнення матриці
for (i = 0; i <n; i ++) {
                    for (j = 0; j <e; j ++) {
```

```
                matrix [i] [j] = 0;
            }
}
            // Заповнити матрицю правельно
            for (i = 0; i <n; i ++)
            {
                for (j = 0; j <e; j ++)
                {
                    if (i == (n-1)) {
                        matrix [i] [j] = 1;
                    } Else {
                        if (i == 0) {
                            // Заповнення рядка A0
                            matrix [0] [0] = P;
                            matrix [0] [1] = -1;
                            matrix [0] [34] = -A2;
                            matrix [0] [67] = -A1;
                        }
                    }
                    if (i> 0 && i <33) {
                        // Заповнення сегмента Ai
                        if (i == (h + 1)) {
                        h = i;
                        matrix [h] [z] = (-Wo * P);
                        matrix [h] [z + 1] = 1 + P;
                        matrix [h] [z + 2] = -1;
                        matrix [h] [z + 35] = -A2;
                        matrix [h] [z + 67] = -A1;
                        z ++;
                    }
                    } Else if (i == 33) {
                        // Кінцеве заповнення An + 1
                        z = 0;
                        matrix [i] [z + 32] = -Wo * P;
                        matrix [i] [z + 33] = 1;
```

```
} Else if (i == 34) {
                    // Заповнення рядка B0
                    matrix [34] [0] = - ((1-Q) * Wn *
P);

                    matrix [34] [35] = A2;
                    matrix [34] [36] = -1;
                } Else if (i> 34 && i <66) {
                // Заповнення сегмента Bi
                if (i> h) {
                h = i;
                matrix [h] [s + 1] = - ((1-Q) * Wn * P);
                matrix [h] [s + 35] = 1;
                matrix [h] [s + 36] = -1;
                s ++;
}
} Else if (i == 66) {
                // Кінцеве заповнення Bn
                s = 0;
                matrix [i] [s + 32] = - ((1-Q) * Wn * P);
                matrix [i] [s + 66] = 1;
} Else if (i == 67) {
                    // Заповнення рядка C0
                    matrix [67] [0] = - (Q * Wn * P);
                    matrix [67] [68] = -1;
                    matrix [67] [67] = A1;
                } Else if (i> 67 && i <n) {
                // Заповнення сегмента Ci
                if (i> h) {
                h = i;
                matrix [h] [r + 1] = - (Q * Wn * P);
                matrix [h] [r + 69] = -1;
                matrix [h] [r + 68] = 1 + A1;
                r ++;
}
}
```

```
                                }
                        }
// Вивід матриці
cout << "Вихідний масив:" << endl;
   for (i = 0; i <n; i ++)
{
for (j = 0; j <e; j ++)
cout << matrix [i] [j] << "";
cout << endl;
   }
cout << endl;
// Метод Гаусса
// Прямий хід, приведення до верхнетреугольному виду
   float tmp;
int k;
          float * xx = new float [e];
for (i = 0; i <n; i ++)
{
tmp = matrix [i] [i];
for (j = n; j> = i; j--)
matrix [i] [j] / = tmp;
for (j = i + 1; j <n; j ++)
{
tmp = matrix [j] [i];
for (k = n; k> = i; k--)
matrix [j] [k] - = tmp * matrix [i] [k];
        }
}
//Зворотній хід
xx [n-1] = matrix [n-1] [n];
    for (i = n-2; i> = 0; i--)
{
xx [i] = matrix [i] [n];
for (j = i + 1; j <n; j ++) xx [i] - = matrix [i] [j] * xx [j];
      }
```

```
// Виводимо результати
for (i = 0; i <n; i ++)
   {
if (i == 0 || i == 34 || i == 67) {
                  s1 = s1 + xx [i];
                  w = 1-s1;
}
if (i <= m) {
                   s2 = s2 + xx [i];
}
}
cout << "\ n Коефіцієнт завантаження ФП :" << w;
cout << "\ n Коефіцієнт завантаження ПК :" << s2;
cout << endl;
delete [] matrix;
system ( "pause");
   return 0;
}
```

Додаток Б

Міністерство освіти і науки, молоді та спорту України
Технологічний інститут СНУ ім. В.Даля
(м. Сєвєродонецьк)

Розробка моделей та дослідження ефективності структурних методів компенсації впливу команд переходів в ядрах сучасних процесорів

Студент гр. КІ-14бд — Костиря Р.Г.

Керівник проекту — Недзельский Д.О.

Типи перешкод

Структурні конфлікти

В цьому випадку в роботі конвеєра виникають так звані "бульбашки" (порожні клітинки в таблиці) в обробці (i + 2) - ї команди і наступних за нею починаючи з такту 6, які знижують продуктивність ядра процесора.

команда	такт								
	1	2	3	4	5	6	7	8	9
i	ВК	ДК	ЧО	ВП	ЗР				
i +1		ВК	ДК	ЧО	ВП	ВП	ВП	ЗР	
i +2			ВК	ДК	ЧО			ВП	ЗР
i +3				ВК	ДК	ЧО			ВП
i + 4					ВК	ДК	ЧО		

2

Типи перешкод

Конфлікти за даними

Конфлікти за даними виникають у випадках, коли виконання однієї команди залежить від результату виконання попередньої команди.

Частина конфліктів за даними може бути знята спеціальною методикою планування компілятора. В найпростішому випадку компілятор просто планує розподіл команд в базовому блоці.

- Наприклад, є послідовність операторів:
- A = B + C;
- D = E - F.
- В цьому випадку компілятор може згенерувати наступну послідовність команд, виконання якої не призведе до призупинення конвеєра:

```
Rb = B .
Rc = C .
Re = E .
Ra = Rb + Rc .
Rf = F .
A = Ra .
Rd = Re - Rf .
D = Rd .
```

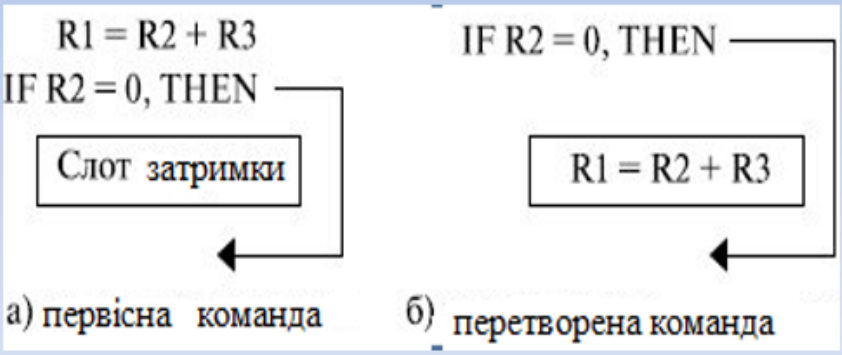
3

Типи перешкод

Конфлікти з управління

Нехай (i + 1)-а команда є командою умовного переходу, яка формує адресу наступної команди в залежності від результату виконання i-ї команди. Команда з номером i завершить своє виконання в такті 5. У той же час команда умовного переходу вже в такті 3 повинна прочитати необхідні їй ознаки, щоб правильно сформувати адресу наступної команди.

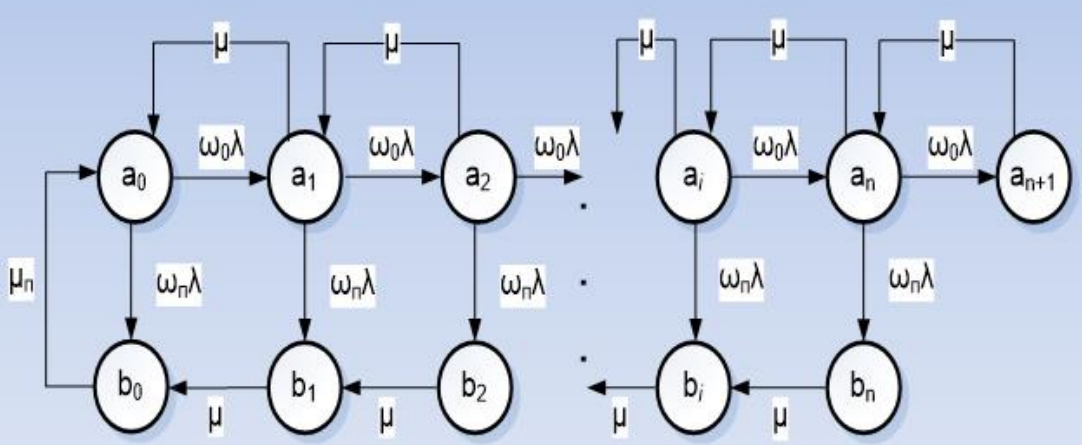
Найпростіший спосіб вирішення цієї ситуації - використання так званого методу вичікування, або затриманих переходів.



4

Моделі ядра при виявленні «перешкоди»

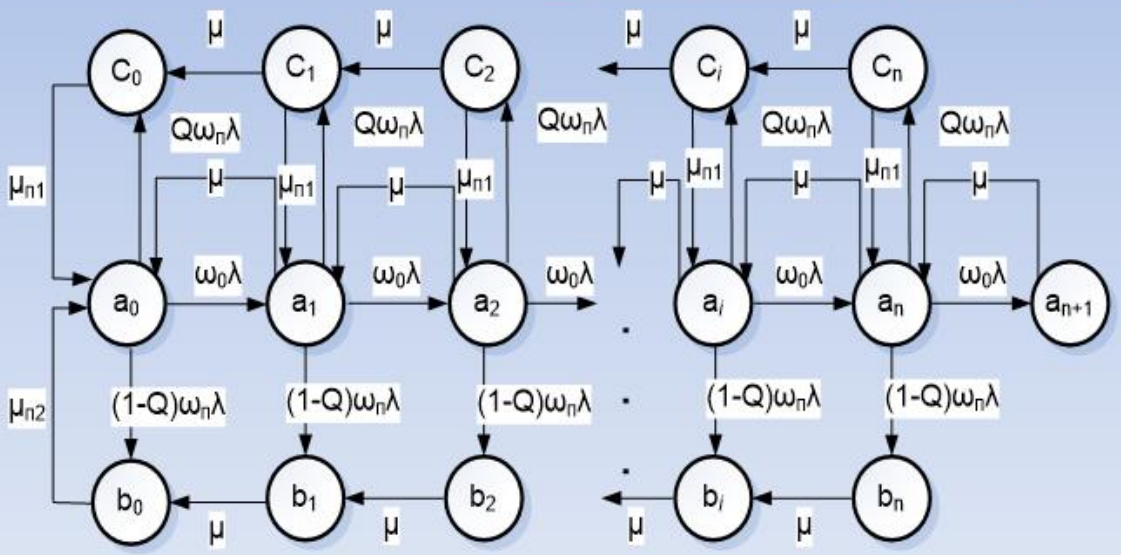
Модель ядра з негайним блокуванням конвеєра



5

Моделі ядра при виявленні «перешкоди»

Модель ядра з блоком передбачення переходів



6

Аналіз результатів експериментів з моделлю без блоку передбачення

Дослідження впливу коефіцієнта навантаження і розміру буфера

ρ		n						
		2	4	8	16	32	64	∞
0.5	$E_{фп}$	0.3402	0.3463	0.3469	0.3469	0.3469	0.3469	0.3469
1	$E_{фп}$	0,5334	0,55	0,5527	0,5528	0,5528	0,5528	0,5528
2	$E_{фп}$	0.6805	0.6926	0.6938	0.6938	0.6938	0.6938	0.6938

7

Аналіз результатів експериментів з моделлю без блоку передбачення

Дослідження впливу частоти команд переходів і коефіцієнта A

A	$H_{пк} / E_{фп}$	$\omega_п$			
		0,05	0,1	0,15	0,2
0.05	$H_{пк}$	0.4601	0.3107	0.2360	0.1903
	$E_{фп}$	0.4370	0.2799	0.2006	0.1525
0.10	$H_{пк}$	0.5974	0.4512	0.3651	0.3080
	$E_{фп}$	0.5676	0. 4061	0.3105	0.2464
0.25	$H_{пк}$	0.7277	0.6188	0.5444	0.4885
	$E_{фп}$	0.6915	0.5568	0.4626	0.3908
0.5	$H_{пк}$	0.7881	0.7062	0.6506	0.6071
	$E_{фп}$	0.7459	0.6355	0.5530	0.4857
1	$H_{пк}$	0.8239	0.7596	0.7210	0.6911
	$E_{фп}$	0.7764	0.6838	0.6127	0.5528
2	$H_{пк}$	0.8343	0.7897	0.7621	0.7423
	$E_{фп}$	0.7925	0.7108	0.6477	0.5938

Перевірка точності аналітичних результатів.

A	$H_{пк}$ / $E_{фп}$	ω_n					
		0.1			0.2		
		Аналит.	Чисельні.	Δ (%)	Аналит.	Чисельні.	Δ (%)
0.05	$H_{пк}$	0.3107	0,3109	0.06	0.1903	0.1906	0.16
	$E_{фп}$	0.2799	0,2798	0.04	0.1525	0.1525	0
0.10	$H_{пк}$	0.4512	0,4512	0	0.3080	0,3080	0
	$E_{фп}$	0. 4061	0,4061	0	0.2464	0,2464	0
0.25	$H_{пк}$	0.6188	0,6187	0.02	0.4885	0,4885	0
	$E_{фп}$	0.5568	0,5568	0	0.3908	0,3908	0
0.5	$H_{пк}$	0.7062	0,7061	0.01	0.6071	0,6071	0
	$E_{фп}$	0.6355	0,6355	0	0.4857	0,4857	0
1	$H_{пк}$	0.7596	0,7598	0.03	0.6911	0,6910	0.01
	$E_{фп}$	0.6838	0,6838	0	0.5528	0,5528	0
2	$H_{пк}$	0.7897	0,7898	0.01	0.7423	0,7423	0
	$E_{фп}$	0.7108	0,7108	0	0.5938	0,5938	0

9

Аналіз результатів моделі з блоком передбачення

- **Дослідження моделі системи з блоком передбачення, коли читання першої команди нової гілки проводиться з кеш-пам'яті команд**

q	$A_1 = A_2$	ω_n	0,05	0,1	0,15	0,2	0,25
0.8	0.125	$H_{пк}$	0. 7974	0. 6530	0. 5423	0. 4578	0. 3925
		$E_{фп}$	0.6904	0.5244	0.4086	0.3249	0.2625
	0.25	$H_{пк}$	0. 8665	0. 7741	0. 6925	0. 6218	0. 5611
		$E_{фп}$	0.7543	0.6244	0.5228	0.4409	0.3736
	0.5	$H_{пк}$	0.9039	0.8364	0.7765	0.7233	0.6757
		$E_{фп}$	0.7949	0.6935	0.6103	0.5382	0.4749
	1	$H_{пк}$	0.9256	0.8820	0.8437	0.8100	0.7889
		$E_{фп}$	0.8227	0.7395	0.6703	0.6086	0.5519
	2	$H_{пк}$	0.9337	0.9049	0.8818	0.8612	0.8431
		$E_{фп}$	0.8436	0.7711	0.7104	0.6556	0.6044
0.9	0.125	$H_{пк}$	0. 8288	0. 6852	0. 5702	0. 4807	0. 4111
		$E_{фп}$	0.7057	0.5389	0.4196	0.3328	0.2680
	0.25	$H_{пк}$	0. 8932	0. 8053	0. 7229	0. 6497	0. 5859
		$E_{фп}$	0.7652	0.6362	0.5333	0. 4496	0.3806
	0.5	$H_{пк}$	0.9249	0.8603	0.8003	0.7454	0.6954
		$E_{фп}$	0.8026	0.7032	0.6201	0.5474	0.4830
	1	$H_{пк}$	0.9558	0.9052	0.8691	0.8348	0.8008
		$E_{фп}$	0.8302	0.7488	0.6803	0.6185	0.5612
	2	$H_{пк}$	0.9522	0.8505	0.9087	0.8884	0.8697
		$E_{фп}$	0.8525	0.7819	0.7222	0.6673	0.6156

10

Перевірка точності аналітичних результатів.

q	$A_1 = A_2$	$H_{пк} / E_{оп} / R$	ω_n					
			0,05			0,15		
			Аналит.	Чисельні.	Δ (%)	Аналит.	Чисельні.	Δ (%)
0.8	0.125	$H_{пк}$	0. 7974	0,8174	2.51	0. 5423	0,5623	3.69
		$E_{оп}$	0.6904	0,6904	0	0.4086	0,4086	0
	0.25	$H_{пк}$	0. 8665	0,8865	2.31	0. 6925	0,7125	2.89
		$E_{оп}$	0.7543	0,7543	0	0.5228	0,5228	0
	0.5	$H_{пк}$	0.9039	0,9235	2.17	0.7 8 65	0,8185	4.07
		$E_{оп}$	0.7949	0,7949	0	0.6103	0,6103	0
	1	$H_{пк}$	0.9256	0,9397	1.52	0.8437	0,8791	4.20
		$E_{оп}$	0.8227	0,8227	0	0.6703	0,6703	0
	2	$H_{пк}$	0.9337	0,9441	1.11	0.8818	0,9066	2.81
		$E_{оп}$	0.8436	0,8436	0	0.7104	0,7104	0
0.9	0.125	$H_{пк}$	0. 8288	0,8488	2.41	0. 5702	0,5902	3.51
		$E_{оп}$	0.7057	0,7050	0.10	0.4196	0,4194	0.05
	0.25	$H_{пк}$	0. 8932	0,9132	2.24	0. 7229	0,7429	2.77
		$E_{оп}$	0.7652	0,7642	0	0.5333	0,5332	0.02
	0.5	$H_{пк}$	0.9249	0,9474	2.43	0.8 1 03	0,8498	4.87
		$E_{оп}$	0.8026	0,8026	0	0.6201	0,6201	0
	1	$H_{пк}$	0.9558	0,9626	0.71	0.8691	0,9105	4.76
		$E_{оп}$	0.8302	0,8302	0	0.6803	0,6803	0
	2	$H_{пк}$	0.9522	0,9671	1.56	0.9087	0,9384	3.27
		$E_{оп}$	0.8525	0,8525	0	0.7222	0,7222	0

11