DOI: https://doi.org/10.33216/2222-3428-2021-21-3

УДК 681.3.07

Недзельський Д.О., Сафонова С.О., Барабрук Л.В.

АНАЛІТИЧНЕ ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ ЯДЕР ПРОЦЕСОРІВ ПРИ НАЯВНОСТІ «ПЕРЕШКОД» З ВИКОРИСТАННЯМ ТЕХНОЛОГІЇ HYPER THREADING

В статті досліджені питання ефективності ядер сучасних процесорів з використанням технології Hyper Threading та без її використання при виконанні програм з урахуванням структурних особливостей ядра процесора та комплексному впливі «перешкод» на роботу конвеєра ядра. Для дослідження обрані широко поширені і наочні програми: «Множення матриць», «Рішення диференціальних рівнянь в приватних похідних методом сіток», «Швидке перетворення Фур'є». В досліджуваних програмах: виділялося ядро - ділянка програми, що забезпечує основний внесок під час виконання програми; розроблялися на умовному асемблері програми ядер; з'ясовувалися інформаційно залежні команди та команди редукції, якщо вони є; формувалися групи команд, які виконують інформаційно залежні ділянки команд ядра програми, та визначалася їх кількість; визначалися ймовірності появи кожної групи команд та часи виконання кожної групи. Розроблено модель ядра процесора у вигляді програми послідовних обчислень. В моделі, інформаційно залежні групи команд, виконувалися універсальним функціональним пристроєм послідовно, згідно з алгоритмом виконання програми. Далі визначалися: середній час виконання ядра програми; середні часи використання окремих спеціалізованих функціональних пристроїв таких як: кеш-пам'ять першого, другого і третього рівнів; пристрої множення, додавання; коефіцієнт навантаження універсального функціонального пристрою; коефіцієнт використання пристрою управління (ПУ) моделі в залежності від значення різних параметрів програми і ядра процесора; коефіцієнти використання спеціалізованих функціональних пристроїв; експериментально перевірялася достовірність теоретичних результатів. При виконанні програми без використання технології Hyper Threading ефективність використання ядра процесора не перевищує 25% і є великий запас продуктивності, як обчислювальних ресурсів, так і ресурсів підсистеми пам'яті. При виконанні двох програм з використанням технології Hyper Threading ефективність ядра істотно збільшується в залежності від розмірів оброблюваних масивів і типів програм, що підтверджує доцільність використання технології Hyper Threading. Продемонстровані деякі причини відсутності ефективності технології Hyper Threading.

Ключові слова: *процесор, ядро, конвеєр команд, «перешкоди», продуктивність, ефективність, технологія Hyper Threading.*

Вступ. Найчастіше ефективність ядер процесорів досліджувалася експериментальним шляхом за допомогою програмних тестів. Отримані дані про часи виконання тестів дозволяли визначити відношення реальної ефективності до пікової ефективності (ф.1).

$$\text{Ефективність} = \text{Ефективність реал} / \text{Ефективність пікова} \quad (1)$$

Проте тести не можуть відповісти на питання чому реальна продуктивність значно менше теоретичної продуктивності і якими шляхами можна збільшити реальну продуктивність.

Аналітично ефективність ядер сучасних процесорів досліджувалася в роботах [1,2] за умови, що програми виконуються в ідеальних умовах при відсутності «перешкод». Були отримані оцінки ефективності механізмів управління, окремих функціональних пристроїв в залежності від параметрів ядер, частот появи команд в програмах.

В [3] досліджувалася аналітична однопоточна ефективність ядра процесора при наявності деяких «перешкод», таких як інформаційні залежності, операції редукції, команди переходів. Однак в цій роботі не порушувалися питання комплексного впливу на ефективність ядер процесорів як «перешкод», так і структурних характеристик таких як параметри кеш-пам'ятей всіх видів, оперативної пам'яті, обчислювальних компонент.

У роботах [4,5] досліджувалася віртуальна двопоточна ефективність (при реалізації технології Hyper Treading) експериментальним шляхом, виконуючи різні тести. Встановлено, що в багатьох випадках віртуальна двопоточність забезпечує збільшення ефективності до 30%. У той же час, при виконанні деяких програм ефективність або не збільшується, або навіть погіршується. У всіх експериментальних дослідженнях шляхом виконання програм і фіксації часів їх виконання не вказувалися причини, що пояснили б, чому отримані

конкретні результати. У випадках відсутності або навіть негативного ефекту це пояснювалося тим, що виконуючі дві програми конкурували за однакові ресурси без вказівки цих ресурсів і ступеня їх використання кожною програмою окремо.

Метою даної статті є дослідження реальних значень ефективності ядер процесорів при виконанні ними однієї програми без технології HT і двох програм з використанням технології HT з урахуванням основних особливостей структури ядер і структур виконуваних програм при комплексному впливі «перешкод» на ефективність ядер процесорів.

Основний зміст роботи
Методика досліджень

Для дослідження комплексного впливу «перешкод» (інформаційних залежностей, операцій редукції, команд переходів) на ефективність ядер процесорів при виконанні програм як без використання технології HT, так і з використанням технології HT були обрані широко поширені і наочні програми: «Множення матриць», «Рішення диференціальних рівнянь в приватних похідних методом сіток», «Швидке перетворення Фур'є».

У досліджуваних програмах:

- виділялося ядро - ділянка програми, що забезпечує основний внесок під час виконання програми;
- розроблялися на умовному асемблері програми ядер;
- з'ясовувалися інформаційно залежні команди і команди редукції, якщо вони є;
- формувалися групи команд, що виконують інформаційно залежні ділянки команд ядра, і їх кількість;
- визначалися ймовірності появи кожної групи команд в програмі ядра;
- визначалися часи виконання кожної групи команд в програмі ядра;
- розроблялася модель ядра процесора, що виконує ядро програми;
- в моделі інформаційно залежні групи команд виконувалися універсальним функціональним пристроєм послідовно згідно з алгоритмом виконання програми.

Далі визначалися:

- середній час виконання ядра програми;
- середні часи використання окремих спеціалізованих функціональних пристроїв таких як: кеш-пам'яті першого, другого і третього рівнів; пристрої множення; пристрої складання;
- коефіцієнт навантаження універсального функціонального пристрою;
- коефіцієнт використання пристрою управління (ПУ) моделі в залежності від значення різних параметрів програми і ядра процесора;
- коефіцієнти використання спеціалізованих функціональних пристроїв;
- експериментально перевірялася достовірність теоретичних результатів.

Модель ядра процесора при виконанні програми

Використовувалася двофазна спрощена модель ядра процесора, що складається з пристрою управління (ПУ), підсистеми виконання груп команд і буфера між ними.

ПУ читає програму, дешифрує команди і поміщає чергову групу команд в буфер дешифрованих груп команд підсистеми виконання. Імовірність генерації і-ї групи команд Wi.

Передбачається, що процес генерації груп команд ПУ найпростіший з показовим законом розподілу часу між згенерованими групами команд. Інтенсивність генерації груп команд визначається формулою

$$\lambda_{ПУ}=1/тген, \quad (2)$$

де тген - математичне очікування часу генерації групи команд.

Час підготовки чергової групи команд ПУ залежить тільки від архітектурних і структурних особливостей ядра процесора.

ПУ переходить до генерації нової групи команд тільки після завершення генерації попередньої групи команд.

Якщо буфер груп команд заповнений, то ПУ припиняє генерацію груп команд.

Підсистема виконання груп команд складається з універсального функціонального пристрою (ФП), який може виконувати будь-яку групу команд з продуктивністю еквівалентною продуктивності підсистеми виконання реального ядра процесора.

Передбачається, що універсальний ФП виконує групи команд з інтенсивністю

$$\mu_{ФП}=1/t_{ФП}, \quad (3)$$

де $t_{ФП}$ визначається за формулою

$$t_{ФП} = \sum_{1}^{n} \omega_{і групи} * t_{і групи} \quad (4)$$

Наприклад, при виконанні програми «Множення матриць» група команд, що складається:

- з команд читання 2-х операндів з кеш-пам'яті даних першого рівня (L1D), команди множення прочитаних операндів і команди нагромаджуючого додавання результатів множення виконується з інтенсивністю

$$\mu_1 = 1/t_{дод}; \quad (5)$$

- з команд читання операнда з кеш-пам'яті даних першого рівня (L1D), команди читання операнда з оперативної пам'яті, команди множення прочитаних операндів і команди нагромаджуючого додавання результатів групи виконується з інтенсивністю

$$\mu_{ОП} = 1/t_{ОП}, \quad (6)$$

де $t_{ОП}$- час читання операнда з оперативної пам'яті;

- з команд читання операнда з кеш-пам'яті даних першого рівня L1D, команди читання операнда з кеш-пам'яті другого рівня L2, команди множення прочитаних операндів і команди нагромаджуючого додавання результатів множення, виконується з інтенсивністю, що визначається часом читання операнда з кеш-пам'яті другого рівня L2, формула (7)

$$\mu_{L2} = 1/t_{L2}; \quad (7)$$

- з команд читання операнда з кеш-пам'яті даних першого рівня L1D, команди читання операнда з кеш-пам'яті третього рівня L3, команди множення прочитаних операндів і команди нагромаджуючого додавання результатів множення (тип групи L1; L3; Множ.; Дод.) виконується з інтенсивністю, що визначається часом читання операнда з кеш-пам'яті третього рівня L3

$$\mu_{L3} = 1/t_{L3}. \quad (8)$$

Передбачається також, що:

- якщо ФП вільний, то чергова згенерована група команд з ПУ відразу надходить на виконання в ФП;
- якщо ж ФП зайнято виконанням групи команд, то чергова згенерована група команд з ПУ надходить в буфер груп команд.

ФП вибирає групи команд з буфера груп команд згідно дисципліни FIFO.

У будь-який момент часу ФП виконує тільки одну групу команд.

ФП переходить до виконання нової групи команд тільки після завершення виконання попередньої групи команд.

Якщо ФП вільний і в буфері груп команд немає заявок, то ФП простоює.

Потік виконаних в ФП груп команд найпростіший, закон розподілу показовий з інтенсивністю

$$\mu_{ФП} = 1/t_{ФП}, \quad (9)$$

де $t_{ФП}$ - середній час виконання групи команд.

Структурну схему еквівалентної спрощеної моделі надано на рис. 1.

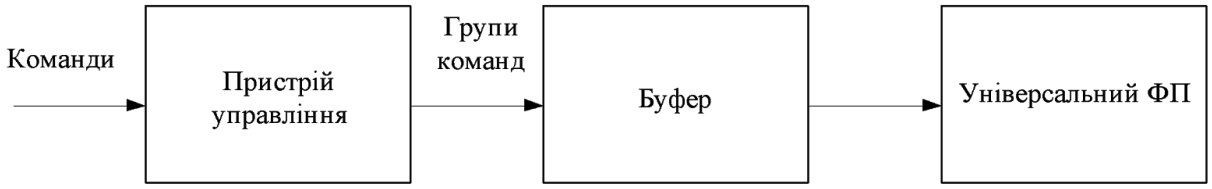

Рисунок 1 - Спрощена структурна схема моделі

Склавши і вирішивши систему рівнянь для ймовірностей перебування моделі в різних станах можна отримати вирази для коефіцієнтів використання компонентів моделі:

- **коефіцієнт використання універсального ФП:**

для $\rho \neq 1$

$$ККД_{ФП}=\rho*(1-\rho^{n+1})/(1-\rho^{n+2}) \quad (10)$$

для $\rho=1$

$$ККД_{ФП}=(n+1)/(n+2), \quad (11)$$

- **коефіцієнт використання ПУ:**

для $\rho \neq 1$

$$ККД_{ПУ}=(1-\rho^{n+1})/(1-\rho^{n+2}), \quad (12)$$

для $\rho=1$

$$ККД_{ПУ}=(n+1)/(n+2), \quad (13)$$

де - ρ це коефіцієнт навантаження універсального ФП;
n - розмірність буфера.

При $n \to \infty$ і $\rho>1$ $ККД_{ФП}=1$, $ККД_{ПУ}=1$.

Середній час виконання груп команд універсальним ФП:

$$t_{ФПі} = \sum_{1}^{m} w_{ігрупи} * t_{ігрупи} \quad (14)$$

Ймовірності генерації груп команд визначаються конкретним видом виконуваної програми і параметрами підсистеми пам'яті.

Коефіцієнти використання окремих спеціалізованих функціональних пристроїв (таких як: кеш-пам'яті першого, другого і третього рівнів; пристрої множення; пристрою додавання) визначалися як

$$ККД_{ФП}=\text{Корисний час зайнятості пристрою}/\text{Загальний час виконання ядра системи} \quad (15)$$

Результати досліджень

У таблицях 1, 2 і 3 наведені теоретичні та експериментальні результати виконання програм «Множення матриць», «Рішення ДР в приватних похідних», «Швидке перетворення Фур'є», відповідно.

Таблиця 1 - Результати дослідження програми «Множення матриць»

Розмір матриць	Розрядність даних	Кіл. пото-ків	ККД ПУ %	ККД ОП	ККД Сум %	ККД L2 %	ККД L3 %	Теоретичний коеф. прискорення	Експериментальний коеф. прискорення
64	32	1	22.3	100	0.25	-	-	-	-
	64	1	20.7	100	0.25	-	-	-	-
	32	2	40	100	0.5	-	-	1	1
	64	2	33	100	0.5	-	-	1	1
128	32	1	21	47	0.25	16	-	-	-
	64	1	18.2	45	0.25	27	-	-	-
	32	2	40	42	0.5	15	-	1.6	1.6
	64	2	27	37	0.5	27	-	1.5	1.4
512	32	1	16.5	9.9	0.25	-	36	-	-
	64	1	12.3	7.7	0.25	-	55	-	-
	32	2	24	13.4	0.5	-	53.7	1.4	1.4
	64	2	12.3	9.8	0.5	-	70.5	1.3	1.3
214	32	1	0.4	100	0.25	-	-	-	-
	64	1	0.4	100	0.25	-	-	-	-
	32	2	0.4	100	0.25	-	-	1	1
	64	2	0.4	100	0.25	-	-	1	1

Таблиця 2 - Результати дослідження програми «Рішення ДР в приватних похідних». Всі дані 32-х розрядні. Кількість ітерацій - 128.

Розмір сітки	Кіл. потоків	ККД ПУ %	ККД ОП %	ККД сум %	ККД $_{L2}$ %	ККД $_{L3}$ %	Теор. коеф. прискорення	Експ. коеф. прискорення
64	1	16.4	1.54	98.5	-	-	-	-
	2	32.3	3	97	-	-	1.9	1.4
128	1	15.9	1.49	92.6	5.9	-	-	-
	2	24.6	2.7	86.6	10.7	-	1.8	1.4
512	1	15.4	1.34	82.8	-	15.9	-	-
	2	24.6	2.2	70.7	-	26.5	1.7	1.4
ОП	1	5.6	67.4	32.6	-	-	-	-
	2	7.7	80	20		1	1.2	1.3

Таблиця 3 - Експериментальні результати дослідження програми «Швидке перетворення Фур'є»

Розмір	Теор. прискорення	Експер. прискорення
1024	1,34	1,33

Висновки

Максимальне значення коефіцієнта використання ПУ (і всього ядра процесора) змінюється в досить широкому діапазоні - від 1% до 22.3% в залежності від типу програми, розмірності масивів і форматів даних.

Коефіцієнти використання ПУ (і всього ядра процесора), обчислені з еквівалентної моделі ядра процесора, є верхньою межею ефективності ядра процесора.

У кращому випадку, коли ПУ в одному такті генерує 4 команди, коефіцієнт використання ПУ (і всього ядра процесора) не перевищуватиме 25%.

Пропускна здатність обчислювальної компоненти підсистеми виконання команд, що складається з кеш-пам'яті даних першого рівня L1D, помножувача та суматора не залежить від розмірності масивів і форматів даних при виконанні ядра програми «Множення матриць» дорівнює: 0.25 - при виконанні одного потоку; 0.5 - при виконанні 2-х потоків з використанням технології Hyper Threading. Це означає, що обчислювальна компонента підсистеми виконання команд, що складається з кеш-пам'яті даних першого рівня L1D, помножувача та суматора може бути використана для виконання інших потоків.

Коефіцієнти прискорення при виконанні 2-х поточних варіантів с використанням технології Hyper Threading більше 1, що доводить доцільність використання цього режиму.

Якщо при виконанні однопотокового варіанту програми без використання технології Hyper Threading коефіцієнт використання критично важливого ресурсу комп'ютера (наприклад оперативної пам'яті) близький до 100%, то використання двопотокового режиму не дасть ефекту.

Експериментальні дані досить добре корелюють з теоретичними даними. Точні значення модельних (теоретичних) і експериментальних даних не збігаються, що і слід було очікувати, так як технічні характеристики реального комп'ютера відрізнялися від характеристик моделі. Однак тенденції результатів однозначно однакові.

Програма обчислення ШПФ (співвідношення команд звернення до підсистему пам'яті і команд обробки даних 1:1, тобто 10 команд звернення до підсистеми пам'яті і 10 команд обробки даних) є прикладом того, що підсистема пам'яті не завжди є гальмом обчислювального процесу, якщо переважна кількість звернень в підсистему пам'яті виконується в швидку кеш-пам'ять даних першого рівня L1D з максимальним поєднанням.

При малих (n <64) розмірностях матриць (програма «Множення матриць»), коли дані повністю розміщуються в кеш-пам'яті даних першого рівня L1D відсутність прискорення пояснюється специфікою програми «Множення матриць», яка виражається в тому, що час запису елемента результату безпосередньо в оперативну пам'ять більше часу обчислення цього елемента результату.

Література

1. Недзельский Д.А. Исследование и анализ эффективности структурных методов компенсации влияния команд переходов на производительность конвейерных ядер процессоров. Луганськ: Вісник Східноукраїнського національного університету ім. В. Даля.- 2013. - №16 (205), Частина 2. - С.174-181.
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.
3. Недзельський Д.О., Сафонова С.О. Про реальну продуктивність та ефективність ядер сучасних процесорів. Наукові вісті Далівського університету. Електронне видання. 2019. № 17. http://dspace.snu.edu.ua:8080/jspui/handle/123456789/3206.
4. Евгений Серов. Как себя чувствует Intel без Hyper-Threading? 14.06.2020/ https://i2hard.ru/publications/23379/
5. Сравнение эффективности Hyper-threading и SMT. 21.08.2018. https://overclockers.ru/blog/Vital-uK/show/21103/

References

1. Nedzelskyi D.A. Research and analysis of the effectiveness of structural methods to compensate for the influence of transient instructions on the performance of pipelined processor cores. Lugansk. Bulletin of the Volodymyr Dahl East Ukrainian National University. - 2013. - No. 16 (205), Part 2. - P.174-181.
2. Nedzelskyi D.A. Investigation of the efficiency of single-core superscalar computing systems. Lugansk. Bulletin of the Volodymyr Dahl East Ukrainian National University. - 2011. - No. 15 (169) Part 2. - p. 133-140.
3. Nedzelskyi D.O., Safonova S.O. About the real productivity and efficiency of the cores of the modern processors. Science of the Dalivsky University. Electronic edition. 2019. No. 17. http://dspace.snu.edu.ua:8080/jspui/handle/123456789/3206.
4. Evgeny Serov. How does Intel feel without Hyper-Threading? 06/14/2020 / https://i2hard.ru/publications/23379/
5. Comparison of the efficiency of Hyper-threading and SMT. 21.08.2018. https://overclockers.ru/blog/Vital-uK/show/21103/

В статье исследованы вопросы эффективности ядер современных процессоров с использованием технологии Hyper Threading и без использования этой технологии при выполнении программ с учетом структурных особенностей ядра процессора и комплексном воздействии «помех» на работу конвейера ядра. Для исследования были выбраны широко распространенные и наглядные программы: «Умножение матриц», «Решение дифференциальных уравнений в частных производных методом сеток», «Быстрое преобразование Фурье». В исследуемых программах: выделялось ядро - участок программы, обеспечивающее основной вклад во время выполнения программы; разрабатывались на условном ассемблере программы ядер; определялись информационно зависимые команды и команды редукции, если они есть; формировались группы команд, которые выполняют информационно зависимые участки команд ядра программы, и определялось их количество; определялись вероятности появления каждой группы команд и время выполнения каждой группы. Разработана модель ядра процессора в виде программы последовательных вычислений. В модели, информационно зависимые группы команд, выполнялись универсальным функциональным устройством последовательно, согласно алгоритму выполнения программы. Далее определялись: среднее время выполнения ядра программы; средние времена использования отдельных специализированных функциональных устройств таких как: кэш-память первого, второго и третьего уровней; устройства умножения, сложения; коэффициент нагрузки универсального функционального устройства; коэффициент использования устройства управления (УУ) модели в зависимости от значения различных параметров программы и ядра процессора; коэффициенты использования специализированных функциональных устройств; экспериментально проверялась достоверность теоретических результатов. При выполнении программы без использования технологии Hyper Threading эффективность использования ядра процессора не превышает 25% и есть большой запас производительности, как вычислительных ресурсов, так и ресурсов подсистемы памяти. При выполнении двух программ с использованием технологии Hyper Threading эффективность ядра существенно увеличивается в зависимости от размеров обрабатываемых массивов и типов программ, что подтверждает целесообразность использования технологии Hyper Threading. Продемонстрированы некоторые причины отсутствия эффективности технологии Hyper Threading. Рис.1, таблиц 3, источников 5.

Ключевые слова: *процессор, ядро, конвейер команд, «помехи», производительность, эффективность, технология Hyper Threading.*

The article investigates the efficiency of modern processor cores with the use of Hyper Threading technology and without its use in programs, taking into account the structural features of the processor core and the complex impact of "interference" on the core pipeline. Widespread and visual programs were selected for the study: "Matrix multiplication", "Solution of differential equations in partial derivatives by the grid method", "Fast Fourier transform". In the studied programs: the core - the site of the program providing the basic contribution during program execution was allocated; developed on a conditional assembler kernel programs; information-dependent commands and reduction teams, if any, were clarified; groups of commands were formed, which perform information-dependent sections of the program core commands, and their number was determined; the probabilities of occurrence of each group of commands and execution times of each group were determined. The model of the processor core in the form of the program of consecutive calculations is developed. In the model, information-dependent groups of commands were executed by a universal functional device sequentially according to the program execution algorithm. Next, the following were determined: the average execution time of the program kernel; average times of use of separate specialized functional devices such as: cache memory of the first, second and third levels; multiplication, addition devices; load factor of the universal functional device; the utilization factor of the control device (DC) of the model depending on the values of various program parameters and the processor core; utilization factors of specialized functional devices; the reliability of theoretical results was experimentally tested. When running a program without the use of Hyper Threading technology, the efficiency of the CPU core does not exceed 25% and there is a large margin of performance for both computing resources and resources of the memory subsystem. When running two programs using Hyper Threading technology, the efficiency of the kernel increases significantly depending on the size of the processed

arrays and types of programs, which confirms the feasibility of using Hyper Threading technology. Some reasons for the ineffectiveness of Hyper Threading technology have been demonstrated. Fig. 1, tables 3, sources 5.
Keywords: *processor, core, command pipeline, "obstacles", performance, efficiency, Hyper Threading technology.*

Недзельський Д.О. – к.т.н., доцент, доцент кафедри комп'ютерних наук та інженерії Східноукраїнського національного університету імені Володимира Даля, e-mail: nedzelsky946@gmail.com

Сафонова С.О. – к.т.н., доцент, доцент кафедри комп'ютерних наук та інженерії Східноукраїнського національного університету імені Володимира Даля, e-mail: safonovasa@ukr.net.

Барбарук Л.В. – ст. викладач кафедри комп'ютерних наук та інженерії Східноукраїнського національного університету імені Володимира Даля, e-mail: barbaruk.angelina@gmail.com