УДК 004.232, 004.274, 004.318

М.О. Березовський
Національний університет “Львівська політехніка”,
кафедра спеціалізованих комп’ютерних систем

СПОСІБ ВИКОНАННЯ КІЛЬКОХ ПОТОКІВ КОМАНД НА ОДНОМУ ДИНАМІЧНО РЕКОНФІГУРОВАНОМУ ПРОЦЕСОРНОМУ ЯДРІ

Описано спосіб підвищення продуктивності мікропроцесора з динамічно реконфігурованим ядром під час роботи у багатозадачному середовищі за допомогою розподілу апаратних ресурсів такого ядра на кілька віртуальних процесорних ядер безпосередньо під час виконання програм.

Вказаний спосіб може бути використаний у багатозадачних системах, зокрема в системах реального часу.

Ключові слова: паралельне виконання, динамічно реконфігуроване ядро.

Described way to improve performance microprocessor with a dynamically reconfigurable core for multitasking environment by distributing hardware resources of such core into several virtual CPU cores directly during program execution.

This way can be applied with multi-tasking systems, including real-time systems.

Key words: parallel execution, dynamically reconfigurable core.

Вступ

Сьогодні актуальним завданням у сфері комп’ютерних технологій є підвищення продуктивності комп’ютерних систем. Одним з основних компонентів комп’ютерних систем, від якого істотно залежить продуктивність усієї комп’ютерної системи, є процесор.

Важливим способом підвищення продуктивності процесора є покращення показника ефективності використання обладнання, оскільки за однакової загальної кількості обладнання збільшується кількість апаратури, задіяної для виконання алгоритмів, і, як наслідок, кількість інструкцій, що виконуються за одиницю часу.

Крім того, підвищення ефективності використання обладнання сприяє відповідному зменшенню енергоспоживання системи, що є особливо актуально для портативних спеціалізованих комп’ютерних систем із автономним живленням, таких як кишенькові комп’ютери, мобільні телефони.

Постановка завдання

Перспективним напрямком підвищення ефективності обладнання процесора є реконфігурація комп’ютерних систем.

Використання процесорів з реконфігурованими ядрами дає змогу оптимізувати апаратну структуру процесора для виконання конкретної програми, а динамічна реконфігурація таких ядер, що здійснюється перед виконанням кожної машинної команди, дає можливість найкраще оптимізувати структуру процесорного ядра для цієї машинної команди.

Під час виконання багатопотокових програм постає завдання оптимізації паралельного виконання потоків на процесорі, яке для динамічно реконфігурованих процесорів може бути виконане шляхом розподілу обладнання процесора для кількох процесорів.

Тому поставлено актуальне завдання розроблення способу підвищення ефективності використання обладнання у динамічно реконфігурованому процесорному ядрі під час виконання.

Аналіз літературних джерел

З 2006 р. одним з основних способів підвищення продуктивності процесорів стало нарощування кількості ядер [1, 2]. Проте надмірне збільшення кількості процесорних ядер не приводить до істотного підвищення продуктивності у більшості задач [2].

Серед причин, які унеможливлюють безмежно нарощувати продуктивність, основними є складність програмування паралельних систем та проблеми синхронізації між обчислювачами у цих системах [3]. Проблеми, пов'язані із складністю паралельного програмування, часто призводять до ситуацій, коли у процесорах під час виконання критичних до продуктивності операцій задіяна лише частина ядер із наявних. Для вирішення проблем синхронізації процесорних ядер використовується додаткове обладнання, що також призводить до зниження показника внаслідок деяких особливостей.

Ще одним відомим способом підвищення ефективності використання обладнання є технологія Intel Hyper-Threading Technology. Вона забезпечує зберігання даних двох потоків у ядрі процесора з тим, щоб під час зупинки виконання одного потоку виконувати інструкції іншого потоку [4]. Прикладом такої затримки є кеш-промах, робота з апаратурою, очікування звільнення ресурсів та інші затримки синхронізації.

Збільшення розрядності машинного слова до 64-х біт або більшого переважно знижує ефективність використання обладнання [5]. Тому застосування 64-бітних процесорів виправдане переважно для розрахунків з великими цілими числами, а в інших випадках старші біти 64-розрядних регістрів простоюють, що є неефективним використанням обладнання. У сучасних персональних комп'ютерах перехід до 64-бітної розрядності машинного слова пов'язаний насамперед із збільшенням адресного простору процесів та відповідно необхідністю зберігати вказівники (адреси) у регістрах загального призначення та виконувати з ними арифметичні операції.

Опис запропонованого способу

Описане в [6, 7] динамічно реконфігуроване ядро під час виконання більшості задач часто матиме значну кількість невикористаних функціональних блоків – регістрів, арифметичних пристроїв, каналів доступу до пам'яті тощо, а отже, низьке значення ефективності використання обладнання.

Тому для такого процесорного ядра під час його використання в багатозадачних системах можливий спосіб підвищення ефективності використання обладнання, а отже, і продуктивності, який полягає у застосуванні вільних від виконання команд поточності функціональних блоків для виконання команд інших потоків. За наявності достатньої кількості вільних функціональних блоків вказаний спосіб, на відміну від Intel Hyper-Threading Technology, за певних умов дає можливість одночасно виконувати машинні команди різних потоків.

Оскільки під процесорним ядром часто розуміють частину процесора, що виконує один потік команд, можна стверджувати, що одне фізичне ядро процесора динамічно під час виконання розділятиметься на кілька віртуальних процесорних ядер.

З іншого боку, кожен функціональний блок описаного ядра можна розглядати як окремий, самостійний процесор, що опрацьовує дані за певним, фіксованим алгоритмом, а все ядро – як мережу на кристалі. У такому випадку, згідно з класифікацією Хокні [8], описане процесорне ядро є MIMD-системою, побудованою на комутаторах, з розділеною пам'яттю, оскільки матриця зв'язків ядра дає змогу підключення довільних функціональних блоків між собою, а самі блоки не мають доступу до жодної спільної пам'яті.

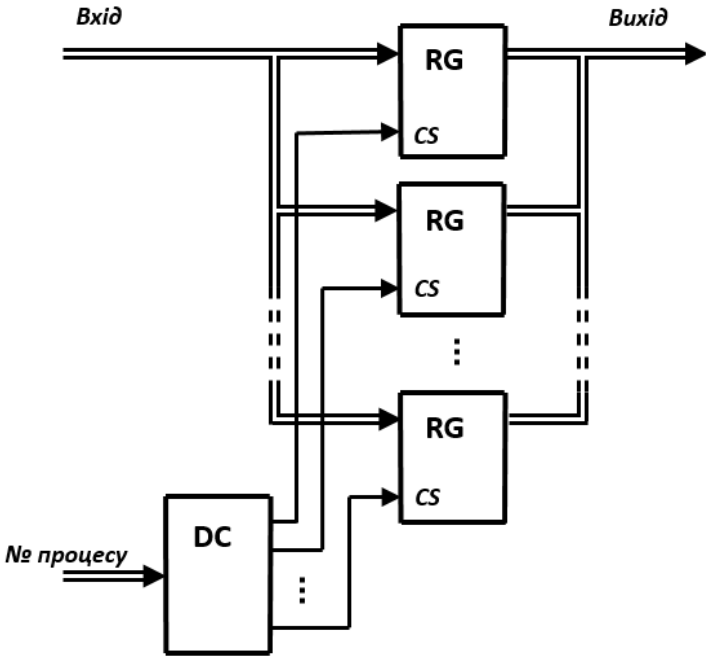
Проблеми розподілу функціональних блоків між потоками

Паралельне виконання кількох потоків на одному ядрі можливе лише, коли вільні усі необхідні функціональні блоки для усіх потоків. Тому одним із завдань апаратури описаного процесора є контроль зайнятості функціональних блоків, який повинен проводитись по черзі у порядку спаду пріоритетів виконуваних процесів.

Зайнятим функціональний блок вважається лише тоді, коли на його входи подані значення. Якщо хоча б один із необхідних для цього потоку функціональних блоків зайнятий, такий потік не може бути виконаний, а потрібно пробувати виконувати наступний потік – з меншим пріоритетом.

Дещо складнішою є проблема, пов'язана із використанням потоками функціональних блоків регістрів, оскільки зміна одним потоком раніше записаного іншим потоком значення не допускається, оскільки це може призвести до неправильної роботи програм.

Одним із способів уникнення таких ситуацій є використання кожним із одночасно виконуваних потоків свого набору регістрів. У зв'язку з цим функціональні блоки регістрів будуть завжди вільними для потоків. Схему функціонального блока із окремими регістрами для кожного потоку показано на рисунку.



Функціональний блок регістра

Іншим способом є введення спеціального регістра із бітами, що показують, чи використовується певний регістр, проте він має багато недоліків, а саме:

1) порушення структури описаного ядра та системи машинних команд, оскільки необхідні додаткові команди скиду бітів вказаного спеціального регістра;

2) можливе на програмному рівні надлишкове блокування функціональних блоків-регістрів, що може істотно збільшити час виконання задачі, або навіть унеможливити паралельне виконання потоків;

3) для компенсування заблокованих функціональних блоків регістрів потрібно збільшити їхню кількість, що приведе до збільшення розрядності машинних команд, і відповідно до розрядності шини даних.

Для спільного доступу до пам'яті, а також до інших пристроїв, відображених в адресному просторі пам’яті, залежно від характеру задач у фізичному ядрі процесора може бути передбачений як один інтерфейс доступу до пам'яті, який блокується під час роботи з пам’яттю, так і кілька, по одному інтерфейсу на кожен із виконуваних одночасно потоків.

Особливості структури дешифратора команд

Для декодування команд може бути використаний як один дешифратор на усі потоки, так і по одному дешифратору на кожен потік.

Використання одного дешифратора залежно від характеру задач може знижувати продуктивність ядра через затримки на зчитування операцій з пам’яті команд.

Для досягнення високої продуктивності доцільно використовувати окремий дешифратор команд для кожного з потоків, що можуть бути одночасно виконані на фізичному ядрі.

Для забезпечення одночасного виконання кількох потоків для кожного з них використовується окремий дешифратор аналогічної структури, на вхід кожного з яких подаються машинні інструкції відповідного потоку.

Виходи цих дешифраторів подаються на спеціальний пристрій, який резервує функціональні блоки. Якщо функціональні блоки, необхідні для виконання дешифрованої інструкції, зарезервовані одним із попередніх дешифраторів, виконання поточного потоку призупиняється до звільнення відповідних необхідних блоків.

Оцінка ефективності використання обладнання

Для оцінки ефективності використання обладнання визначатимемо частку задіяних функціональних блоків, незалежно від їх типів.

Відповідно ефективність використання обладнання під час виконання описаним ядром певної машинної команди, тобто під час виконання лише одного потоку, можна визначити за такою формулою:

$$E_к = \frac{N}{N_{заг.}} \cdot 100\ \%, \quad (1)$$

де $E_к$ – ефективність використання обладнання під час виконання певної машинної команди; N – кількість одночасно задіяних функціональних блоків; $N_{заг.}$ – загальна кількість функціональних блоків.

За одночасного виконання на одному процесорному ядрі кількох машинних команд, тобто кількох потоків одночасно, вираз ефективності використання обладнання набуде такого вигляду:

$$E_к = \frac{\sum_{i=1}^{n} N_i}{N_{заг.}} \cdot 100\ \%, \quad (2)$$

де $E_к$ – ефективність використання обладнання; $N_{i.}$ – кількість одночасно задіяних функціональних блоків i-м потоком; $N_{заг.}$ – загальна кількість функціональних блоків; n – кількість потоків, що одночасно виконуються на процесорному ядрі.

Враховуючи співвідношення (2) за збільшення кількості потоків n, якщо при цьому кожен з них виконує різні машинні команди, тобто використовує різні набори функціональних блоків, у такому випадку підвищується ефективність використання обладнання.

Висновки

Описаний спосіб підвищення продуктивності мікропроцесора з динамічно реконфігурованим ядром за допомогою розподілу ресурсів на кілька процесорних ядер безпосередньо під час виконання програм дає можливість значно підвищити продуктивність виконання задач реального часу, ресурсомістких обчислень, а також програм із кількома потоками з диференційованими пріоритетами.

Подальшим розвитком цієї технології може бути удосконалення способів розподілу функціональних блоків між потоками, яке забезпечувало б максимальний коефіцієнт використання апаратури. Також важливим завданням щодо розвитку описаного способу є визначення оптимального набору функціональних блоків для цього процесорного ядра залежно від кількості та характеру виконуваних потоків.

1. Ланина Э.П. Многоядерность, как способ увеличения производительности вычислительной системы: Всероссийский конкурсный отбор обзорно-аналитических статей по приоритетному направлению" // Информационно-телекоммуникационные системы. – 2008. 2. Путря Ф.М. Архитектурные особенности процессоров с большим числом вычислительных ядер // Информационные технологии. – 2009. – № 4. – С. 2–7. 3. Shaoshan L., Gaudiot J.-L. Synchronization Mechanisms on Modern Multi-core Architectures // Advances in Computer Systems Architecture / Lynn C., Yunheung P., Sangyeun C. – 2007. 4. Deborah T. Marr та ін. Hyper-Threading Technology Architecture and Microarchitecture // Intel Technology Journal Q1. – 2002. 5. Щукин А. Анализ производительности 64- и 32-разрядных многопроцессорных вычислительных систем в программном комплексе вычислительной гидрогазодинамики STAR-CD // iXBT.com, 2004. – Режим доступу: http://www.ixbt.com/cpu/star-cd-test.shtml. 6. Березовський М., Дунець Р. Динамічно реконфігуроване процесорне ядро // "Сучасні комп'ютерні системи та мережі: розробка та використання": Матер. 5-ї Міжнар. наук.-техн. конф. ACSN-2011. – С.204–205. 7. Березовський М., Дунець Р. Оптимізація формату машинних команд динамічно реконфігурованого процесорного ядра // Радіоелектронні і комп'ютерні системи. – 2012. – № 5 (57). – С. 120–124. 8. Hockney R. Classification and Evaluation of ParallelComputer Systems // Lecture Notes in Computer Science. – 1987. – № 295.