1

УДК 004.023

ЕВРИСТИЧНА ОПТИМІЗАЦІЯ РОБОТИ КЕШ-ПАМ'ЯТІ У КОМПОЗИЦІЙНОМУ МІКРОПРОГРАМНОМУ ПРИСТРОЇ КЕРУВАННЯ

Мокроусов О.С., Ковальов С.О., Ніколаєнко Д.В., Бабаков Р.М.
ДонНТУ, Донецьк, Україна

The heuristic algorithm of increasing of efficiency of using of cache-memory module in compositional microprogram control unit, based on special addressing of operator linear chains, is developed. The different strategies for combination of some linear chains operators in one memory block are proposed. They allow in common case to increase value of probability of cache hits for given flow-chart.

Key Words: *cache-memory, heuristic algorithm, probability of cache hit.*

Введення

Одним із структурних елементів сучасних обчислювальних систем є пристрій керування (ПК), який може бути реалізований у вигляді композиційного мікропрограмного пристрою керування (КМПК) з поділом кодів, в якому досягається мінімальне число виходів схеми адресації [1]. Використання елементного базису ПЗП при реалізації керуючої пам'яті (КП) здешевлює схему пристрою, однак у той же час приводить до значного зниження швидкодії схеми [2]. Таким чином, збільшення швидкодії схеми КМПК при проектуванні засобів обчислювальної техніки є актуальною науково-технічною задачею. Рішення даної задачі спричинить за собою збільшення швидкодії обчислювальних систем, реалізованих на базі КМПК, і, як наслідок, розширить сферу їх застосування.

Для збільшення швидкодії схеми КМПК з розділенням кодів в роботі [3] запропоновано метод, що полягає у використанні додаткового модуля кеш-пам'яті мікрокоманд для зберігання найбільш часто використовуваних послідовностей мікрокоманд керуючої пам'яті (рис.1).

Один із способів,що дозволяє вбудувати кеш-пам'ять (КП) у структуру КМПК зображений на рисунку 2.

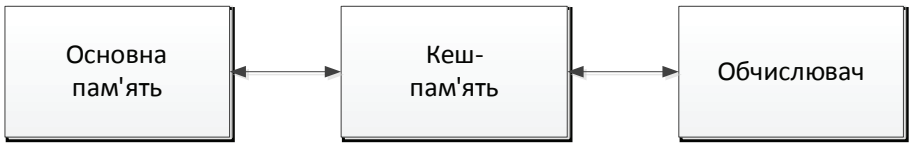

Рисунок 1. Структурне розташування кеш-пам'яті в обчислювальному пристрої

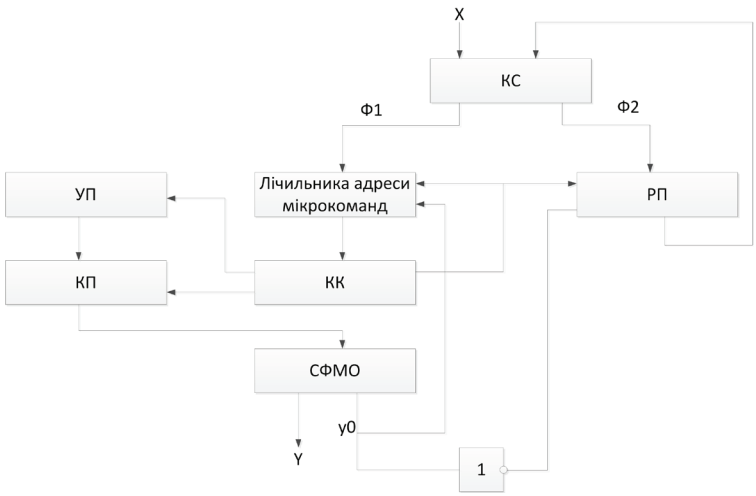

Рисунок 2. Структурна схема КМПК з кеш-пам'яттю

Тут КК – кеш-контролер, призначений для управління процесом вибору мікрокоманди з пам'яті. Кеш-контролер, крім управління УП(управляючий пристрій) та КП, також управляє подачею синхронізації на входи лічильника адреси мікрокоманд і регістра пам'яті (РП).

Використання кеш-пам'яті дозволяє знизити середній час доступу до керуючої пам'яті, що призводить до зменшення середньої тривалості такту роботи пристрою і до збільшення його середньої швидкодії. При цьому основним параметром, що впливає на ефективність використання кеш-пам'яті мікрокоманд, є ймовірність кеш-влучень $p_{h,}$ що характеризує відношення кількості тактів, в яких відбулося кеш-попадання, до загальної кількості тактів роботи пристрою.

Характерною особливістю всіх структур КМПК є їх апаратна прив'язка до реалізованого алгоритму керування. Це обумовлено наявністю в структурі автомата модуля керування з жорсткою логікою. Дана особливість дозволяє провести оптимізацію схеми КМПК для кожного конкретного випадку реалізації. У структурі КМПК з розділенням кодів і кеш-пам'яттю, запропонованої в [3], одним з факторів, що впливає на величину ймовірності кеш-попадань, є реалізований алгоритм керування, традиційно представлений у вигляді граф-схеми алгоритму (ГСА). Кількість мікрокоманд (МК), операторних лінійних ланцюгів (ОЛЛ) і переходів між ОЛЛ, а також сама структура ГСА - все це впливає на величину $p_{h.}$ При цьому на сьогоднішній день недослідженими залишаються фактори, що впливають на збільшення значення p_h і, як наслідок, на збільшення середньої швидкодії пристрою керування.

Евристичний підхід до оптимізації розміщення мікрокоманд в керуючої пам'яті

У структурах КМПК з розділенням кодів мають місце природна адресація мікрокоманд всередині кожного ОЛЛ, а також той факт, що в першій мікрокоманді ОЛЛ адресні розряди, наступні після коду ОЛЛ, рівні нулям [2]. З цих причин процес адресації мікрокоманд, що є одним з етапів синтезу КМПК, в структурах з поділом кодів зводиться по суті до адресації ОЛЛ.

Нехай в заданій ГСА існує перехід з ОЛЛα_i в ОЛЛ $\alpha_{j.}$ При цьому, якщо обидва ОЛЛ в даний момент знаходяться в кеш-пам’яті, то виникне ситуація кеш-попадання, в іншому випадку - ситуація кеш-промаху. У тому випадку, якщо обидва ОЛЛ розташовані в керуючій пам’яті послідовно і належать одному блоку пам’яті, вони опиняться разом в одному рядку кеш-пам’яті, і згаданий перехід з $\alpha_i \rightarrow \alpha_j$ завжди буде призводити до кеш-попадання. Якщо ж дані ОЛЛ знаходяться в різних блоках пам’яті, то при виконанні переходу $\alpha_i \rightarrow \alpha_j$ можлива ситуація кеш-промаху, і значення p_h для ГСА буде трохи нижче. Якщо ж одночасне перебування даних ОЛЛ в кеш-пам’яті неможливо (наприклад, при використанні кеш-пам’яті з прямим відображенням даних), то цей перехід завжди буде призводити до кеш-промаху, що ще більше знизить величину ймовірності кеш-попадань для заданої ГСА.

Згідно з принципом просторової локальності даних, у випадку кеш-промаху в кеш з ПК поміщається не тільки запитувана мікрокоманда, а й кілька мікрокоманд з найближчих до неї адрес пам’яті. При цьому кількість завантажених команд визначається розміром рядка кеш-пам’яті, а для кодування мікрокоманд усередині рядка використовується частина молодших розрядів адреси мікрокоманди. З урахуванням цього вміст керуючої пам’яті може бути представлено у вигляді *блоків мікрокоманд,* що мають розмір, що дорівнює розміру рядка кеш-пам’яті і розташовуються послідовно починаючи з нульової адреси. Розподіл ПК на блоки є постійним і не залежить від розміщення мікрокоманд в адресному просторі ПК.

Експериментальним шляхом авторами вироблені чотири основних евристичних правила (евристики), що характеризують шляхи підвищення ефективності розміщення ОЛЛ в блоках керуючої пам’яті:

Евристика 1. Ймовірність кеш-попадання не залежить від того, якому по порядку блоку пам’яті належить ОЛЛ. Іншими словами, конкретні значення адрес пам’яті, займані ОЛЛ, не роблять впливу на ймовірність кеш-попадань. З розглянутого правила випливає наступний висновок: якщо вміст блоку пам’яті змістити в адресному просторі ПК вперед або назад на кількість комірок, кратне розміром блоку, то значення ймовірності кеш-попадань не зміниться.

Евристика 2. Ймовірність кеш-попадань не залежить від порядку проходження ОЛЛ в

блоці. Евристика діє і в тому випадку, якщо блок пам'яті містить більше двох ОЛЛ. Висновок, який випливає з даного правила, може бути сформульовано таким чином: якщо кілька ОЛЛ можуть бути розміщені в одному блоці ПК, то порядок розміщення ОЛЛ в блоці не впливає на результуюче значення ймовірності кеш-попадань.

Для пояснення третього евристичного правила введемо наступні визначення:

1. «Небезпечною» МК вважається така МК, при виконанні якої кеш-промах можливий (але не обов'язковий). МК, при виконанні якої кеш-промах неможливий ні за яких обставин, вважається «безпечною». До «небезпечних» МК відносяться мікрокоманди, що є входами ОЛЛ.

2. Ймовірністю виконання (або вагою) $p(a_i)$ мікрокоманд a_i будемо називати відношення середньої кількості виконань $K(a_i)$ МК a_i за один прохід алгоритму до середнього кількості мікрокоманд K, що виконуються за один прохід алгоритму:

$$p(a_i) = K(a_i) / K. \quad (1)$$

3. «Небезпечна» вага $v(a_i)$ мікрокоманд a_i дорівнює сумі ваг мікрокоманд інших ОЛЛ, з яких є безпосередній перехід в мікрокоманду a_i, помножених на імовірність даного переходу.

4. «Небезпечна» вага блоку пам'яті дорівнює сумі «небезпечних» ваг ОЛЛ, що входять в блок.

Евристика 3. Якщо ОЛЛ O_i має переходи в ОЛЛ O_j, то розміщення цих ОЛЛ в одному блоці пам'яті знижує «небезпечну» вагу ОЛЛ O_j. З евристики 3 може бути зроблено наступний висновок: найбільш доцільно додати в поточний блок тій ОЛЛ, при якому «небезпечна» вага поточного блоку буде мінімальною.

Евристика 4. Збільшення кількості ОЛЛ, на які розбита безліч мікрокоманд розглянутої ГСА, не впливає на ймовірності виконання мікрокоманд, але збільшує кількість варіантів розміщення ОЛЛ в керуючої пам'яті.

Розробка евристичного алгоритму оптимізованого розміщення ОЛЛ в керуючої пам'яті

Ґрунтуючись на отриманих евристиках, побудуємо евристичний алгоритм оптимізації розміщення ОЛЛ в адресному просторі керуючої пам'яті з метою підвищення значення ймовірності кеш-попадань.

Вихідними даними при цьому виступають:

- вихідна ГСА;
- ймовірності виконання логічних умов;
- розмір рядка даних у модулі кеш-пам'яті;
- кількість і вміст ОЛЛ.

Алгоритм має наступні основні етапи:

1. Визначення ймовірностей виконання мікрокоманд.
2. Віднесення всіх ОЛЛ до безлічі нерозподілених ОЛЛ.
3. Вважати поточним початковий блок ПК.
4. Якщо поточний блок повністю заповнений, вважати поточним наступний порожній блок.
5. Якщо поточний блок порожній, помістити в нього один з нерозподілених ОЛЛ.
6. Якщо поточний блок заповнений частково, додати в нього той нерозподілений ОЛЛ, разом з яким сумарна «небезпечна» вага блоку буде мінімальною.
7. Якщо залишилися нерозподілені ОЛЛ, перейти до п. 4.
8. Кінець.

У пункті 5 алгоритму в порожній блок додається один з нерозподілених ОЛЛ. Вибір нерозподіленого ОЛЛ в якомусь сенсі може бути довільним: який б ОЛЛ не був обраний першим в блоці, алгоритм під дією евристики 3 повинен заповнити блок оптимальним способом. При цьому ті ОЛЛ, які будуть додані до блок, приречені на «сусідство» з першим вибраним ОЛЛ, і варіанти їх «сусідства» з іншими ОЛЛ не розглядаються. Таким чином, спосіб вибору першого ОЛЛ в блоці впливає на кінцевий вміст блоку, тобто на «небезпечну» вагу блоку і, зрештою, на ймовірність кеш-

попадань ГСА.

Авторами пропонуються шість варіантів способу вибору першого ОЛЛ для порожнього блоку, які домовимося називати стратегіями вибору нерозподілених ОЛЛ:

Стратегія 1: Вибирається ОЛЛ з максимальною вагою.

Стратегія 2. Вибирається ОЛЛ з мінімальною вагою.

Стратегія 3. Вибирається ОЛЛ з максимальною «небезпечною» вагою.

Стратегія 4. Вибирається ОЛЛ з мінімальною «небезпечною» вагою.

Стратегія 5. Вибирається ОЛЛ з максимальним відношенням «небезпечної» ваги до звичайного вазі.

Стратегія 6. Вибирається ОЛЛ з мінімальним відношенням «небезпечної» ваги до звичайної ваги.

При використанні кожної з шести стратегій в загальному випадку може бути отриманий різний результат (варіант розміщення мікрокоманд і значення $p_{h)}$. Безумовно, крім запропонованих, може бути знайдено безліч інших стратегій, що роблять різний вплив на результат алгоритму: наприклад, стратегія, при якій перший завжди вибирається ОЛЛ, що містить стартову МК $a_{1,}$ або стратегія випадкового вибору. Пошук найбільш оптимальної стратегії вибору ОЛЛ є окремою гілкою досліджень і в цій роботі не розглядається.

Евристичний характер запропонованого алгоритму полягає в наступному:

1. Оптимізація розміщення проводиться для кожного блоку окремо без урахування зв'язків між блоками. Велика кількість переходів між ОЛЛ, що знаходяться в різних блоках, може призводити до збільшення кількості кеш-промахів і зниження величини $p_{h.}$
2. Не враховується кількість рядків у модулі кеш-пам'яті. Якщо кеш-пам'ять містить лише один рядок фіксованої довжини, алгоритм заміщення даних, призначений для вибору одного з кількох рядків кеш-пам'яті, не використовується. У цьому випадку значення ймовірності кеш-попадань буде однаковим для будь-якої архітектури кеш-пам'яті. Якщо ж кеш-пам'ять містить кілька рядків, то вплив архітектури кеш-пам'яті і алгоритму заміщення даних на значення ймовірності кеш-влучень може бути значним.
3. Не використовується оптимальна стратегія вибору нерозподіленого ОЛЛ. Хоча експериментальні дослідження показують, що запропоновані стратегії виявляються досить ефективними, не можна говорити про абсолютну оптимальності будь-якої з них.

Висновок

Розроблений евристичний алгоритм оптимізації розміщення мікрокоманд в керуючої пам'яті дозволяє підвищити ефективність використання модуля кеш-пам'яті в структурі КМПК з розділенням кодів, що позитивно відображається на швидкодії логічної схеми пристрою. Практична реалізація даного алгоритму можлива в спеціалізованих САПР цифрових пристроїв керування.

Література

[1] Баркалов О.О., Палагін О.В. Синтез мікропрограмних пристроїв керування. – Київ: Інститут кібернетики НАН України, 1997. – 135 с.

[2] Баркалов О.О. Синтез пристроїв керування на програмованих логічних пристроях. – Донецьк: ДонНТУ, 2002. – 262 с.

[3] Баркалов О.О., Ковальов С.О., Бабаков Р.М., Ніколаєнко Д.В. Організація композиційних мікропрограмних пристроїв керування з розділенням кодів і кеш-пам'яттю // Штучний інтелект, 2007. – № 3. – С. 135-138.