

**УКРАЇНА****(19) UA****(11) 104039****(13) C2****(51) МПК****G06F 9/44 (2006.01)****G06F 9/45 (2006.01)**

**ДЕРЖАВНА СЛУЖБА
ІНТЕЛЕКТУАЛЬНОЇ
ВЛАСНОСТІ
УКРАЇНИ**

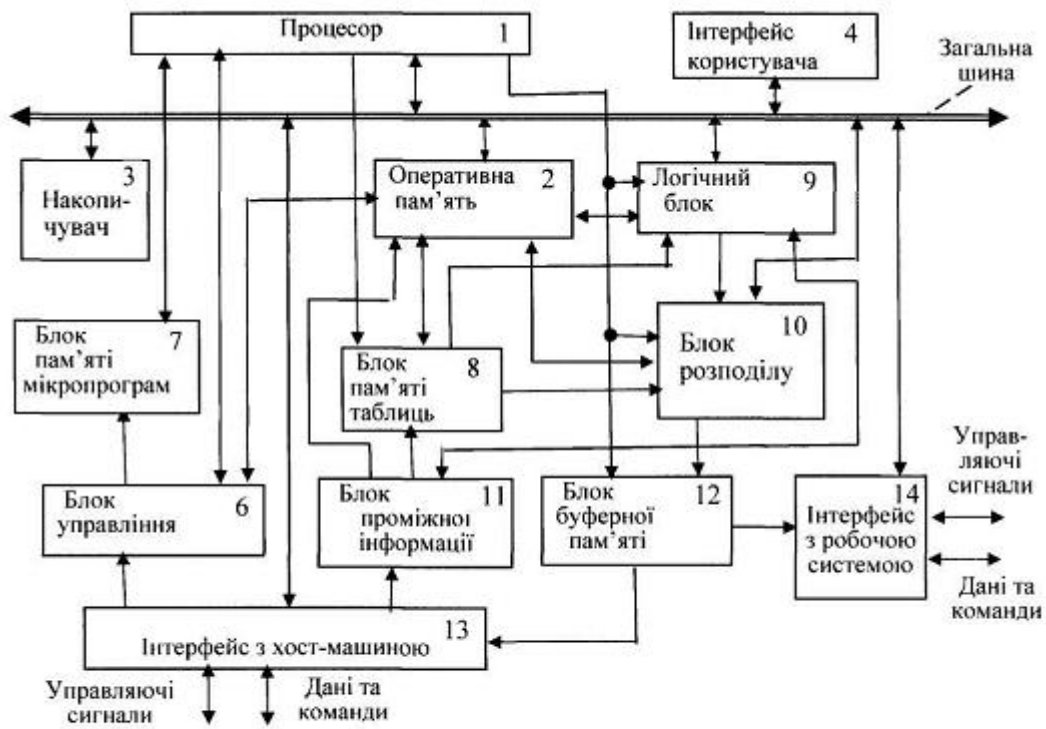
(12) ОПИС ДО ПАТЕНТУ НА ВІНАХІД

(21) Номер заявки: а 2012 02223	(72) Винахідник(и): Сергієнко Іван Васильович (UA), Палагін Олександр Васильович (UA), Боюн Віталій Петрович (UA), Яковлев Юрій Сергійович (UA), Слісєєва Олена Володимирівна (UA)
(22) Дата подання заявки: 27.02.2012	
(24) Дата, з якої є чинними права на винахід: 25.12.2013	
(41) Публікація відомостей про заяву: 27.08.2013, Бюл.№ 16	(73) Власник(и): ІНСТИТУТ КІБЕРНЕТИКИ ІМ. В.М. ГЛУШКОВА НАН УКРАЇНИ, пр. Академіка Глушкова, 40, м. Київ, 03680 (UA)
(46) Публікація відомостей про видачу патенту: 25.12.2013, Бюл.№ 24	(56) Перелік документів, взятих до уваги експертизою: US 5452461 A; 19.09.1995 US 2007169046 A1; 19.07.2007 US 7437723 B1; 14.10.2008 WO 2010138274 A1; 02.12.2010 JP 2005316577 A; 10.11.2005 US 7093258 B1; 15.08.2006 FR 2803060 A1; 29.06.2001 WO 0223328 A2; 21.03.2002 US 2004003204 A1; 01.01.2004

(54) СИСТЕМА ДЛЯ РОЗПОДІЛУ ПРОГРАМИ КОРИСТУВАЧА**(57) Реферат:**

Система для розподілу програм користувача належить до обчислювальної техніки, зокрема до розподілених гетерогенних комп'ютерних систем, наприклад систем типу "Процесор - в - пам'яті" ("Processor-in-memory") або до PIM-систем. Система містить процесор, оперативну пам'ять, накопичувач, інтерфейс користувача, загальну шину, блок розподілу, блок пам'яті мікропрограм, блок пам'яті таблиць, логічний блок, блок проміжної інформації, блок буферної пам'яті, блок управління, інтерфейс з хост-машиною, інтерфейс з робочою системою, які сполучені між собою відповідними зв'язками. Введення в систему вказаних блоків дозволяє виконувати поєднання за часом процесу розпаралелювання поточного фрагмента програми користувача і процесу підготовки інформації, необхідної для розпаралелювання наступного фрагмента, а також поєднання за часом роботи системи для розподілу програми користувача з виконанням робочою системою цієї програми за рахунок застосування блока буферної пам'яті. Технічним результатом є скорочення часу виконання процесу розподілу, розширення області застосування системи розподілу для робочих систем, виконаних на різних апаратно-програмних платформах, а також додатково позитивний ефект в частині підвищення продуктивності робочої системи, у тому числі PIM-системи, за рахунок апаратної підтримки запропонованої стратегії розпаралелювання, яка відображає особливості архітектури і структури робочої системи, що реалізовує програму користувача.

UA 104039 C2



Фіг. 2

Винахід належить до області обчислювальної техніки, зокрема до гетерогенних розподілених комп'ютерних систем, наприклад до кластерних систем, в яких вузловий процесор кластера відрізняється за параметрами продуктивності від інших процесорів цього кластера, а також до систем типу "Processor-in-memory" ("Процесор-в-пам'яті") або PIM-систем, в яких, так заний, ведучий процесор (ВП) істотно перевершує за параметрами продуктивності інші процесори, які розміщені на одному кристалі з ВП і приєднані до нього. Такі процесори ідентифіковані в літературі як процесорні ядра (ПЯ), оскільки вони мають спрощену структуру, скорочену систему команд і істотно менші в порівнянні з ВП функціональні можливості [Яковлев Ю.С. Однокристалые компьютерные системы высокой производительности. Особенности архитектурно-структурной организации и внутренних процессов: монография / Ю.С.Яковлев. - Винница: ВНТУ, 2009. - 294 с.]. Системи такого типу, що реалізують розподілений по процесорах алгоритм розв'язання задачі користувача, надалі називатимемо робочими системами на відміну від систем, призначених безпосередньо для розподілу задачі користувача.

Аналіз історії розвитку засобів обчислювальної техніки показав, що технологічні прийоми вдосконалення елементної бази, як основи побудови обчислювальних засобів, має потенціал підвищення їх продуктивності (за даними зарубіжних публікацій), приблизно, одного порядку, а архітектурно-структурні рішення і розпаралелювання програми користувача може забезпечити підвищення продуктивності на декілька порядків.

Відомі підходи до вирішення проблеми розподілу програми користувача на незалежні за даними фрагменти (блоки фрагментів) і розподілу їх по процесорах багатопроцесорної системи для паралельного виконання, як правило, засновані на трудомістких ітераційних методах розпаралелювання програми, на аналізі і виділенні для розпаралелювання циклів програми, на використанні методів апроксимації ділянок програми, які піддаються формульним описам, на застосуванні стандартних методів розпаралелювання при виконанні операцій над окремими математичними структурами, наприклад, операцій над матрицями, поліномами і так далі. Усе це реалізують програмними способами, у тому числі - використовуючи спеціально розроблені програми, наприклад програму SAGE [Slo-Li Chu. Exploiting Application Parallelism for Processor-in-Memory Architecture / Slo-Li Chu, Tsung-Chuan Huang // Proc. of National Computer Symposium, Taiwan, 2003, December 18-19. - 2003. - P. 293-303. - Режим доступу: http://dspace.lib.fcu.edu.tw/bitstream/2377/564/1/OT_1022003305.pdf. - Дата доступу: 17.08.11] та ін. При цьому для реалізації розпаралелювання, як правило, використовують ресурси двох і більше комп'ютерів розподіленої системи, тим самим знижуючи ефективність їх використання для вирішення безпосередньо задачі користувача.

Враховуючи, що розпаралелюванню підлягають задачі користувача, що відрізняються високою складністю і трудомісткістю та вимагають від комп'ютерної системи типу супер-ЕОМ високої продуктивності, такий процес розпаралелювання має високу енергоємність, оскільки часто при програмному способі реалізації розпаралелювання і великій кількості ітерацій час процесу розпаралелювання може істотно перевищувати час розв'язання задачі.

Застосування для розподілу апаратно-програмних засобів, наприклад комп'ютерів з класичною архітектурою, що не входять до складу системи, яка вирішує задачі користувача, також виявляється не ефективним, оскільки, по-перше, процес розподілу програми користувача на цих засобах реалізують також програмно, і, по-друге, спосіб розподілу і апаратно-програмні засоби для підтримки цього способу є, згідно з моделлю розподілу, проблемно-орієнтованими на конкретний тип архітектури (наприклад, на архітектуру типу PIM - системи) і на конкретний клас задач користувача, наприклад на задачі, що містять багато циклів і так далі [Slo-Li Chu. Exploiting Application Parallelism for Processor-in-Memory Architecture / Slo-Li Chu, Tsung-Chuan Huang // Proc. of National Computer Symposium, Taiwan, 2003, December 18-19. - 2003. - P. 293-303. - Режим доступу: http://dspace.lib.fcu.edu.tw/bitstream/2377/564/1/OT_1022003305.pdf. - Дата доступу: 17.08.11]

Відомі також і інші підходи використання окремих апаратно-програмних засобів, що також реалізують програмний спосіб розподілу задачі користувача, ефективність яких може бути досягнута при застосуванні спеціального комутатора розподіленої комп'ютерної системи і спеціального блока додатково для кожного процесора цієї системи.

Таким чином, відомі пристрої, за допомогою яких реалізують спосіб розподілу задач користувача по процесорах багатопроцесорної системи, є проблемно-орієнтованими і не можуть бути ефективно використані для розподілених комп'ютерних систем з архітектурою іншого типу, а також з системами, виконаними на основі інших апаратно-ограмних платформ, тобто мають обмежені функціональні можливості і вузьку сферу ефективного застосування. Крім того, процес розподілу програм користувача за допомогою таких пристроїв виконують

програмними способами, що згідно з моделями розподілу, що використовуються, призводить до великої кількості ітерацій і як наслідок - до тривалого процесу розподілу, великих енерговитрат і вартості експлуатації.

Тому створення системи для реалізації процесу розподілу програми користувача, яка має широкі функціональні можливості і може використовуватися для комп'ютерних систем з різною архітектурою і різною апаратно-програмною платформою, а також істотно скорочує час розподілу програми користувача, є актуальним технічним завданням.

Відома система для реалізації методу розподілу програм користувача [див. патент США "Program parallelizing apparatus capable of optimizing processing time". № 5452461 (A), 19.09.1995]. При цьому три перші втілення цього винаходу реалізують розпаралелювання програми користувача за допомогою програмних засобів, використовуючи комп'ютер з класичною архітектурою. Для четвертого втілення винаходу система розпаралелювання для паралельної обробки багатопроцесорною робочою системою, що має безліч процесорних блоків, включає: інформаційний процесор і модуль пам'яті, сполучені через інтерфейс з цільовою (робочою) системою. Програми розпаралелювання зберігають в пам'яті процесора системи розпаралелювання. Об'єктні коди і їх послідовність, як результат розпаралелювання, посилають цільовій (робочій) системі через спеціальні інтерфейс і зовнішній розширений канал, до якого підключені блоки пам'яті, що спільно використовуються усіма процесорними блоками робочої системи і процесором системи розпаралелювання. Кожен з процесорних блоків робочої системи містить процесор, локальну пам'ять і буфер, при цьому до кожного процесорного блока підключено спеціальний блок управління, який в сукупності з іншими такими ж блоками забезпечує механізм паралельної обробки.

Як апаратуру системи розпаралелювання програм, згідно з четвертим втіленням цього винаходу, може бути використано, наприклад, апаратуру персонального комп'ютеру, робочої станції, як окремо, так і в сукупності. Крім того, є можливим варіант побудови системи, коли один з багатьох процесорних блоків робочої (цільовий) системи може бути використаний для виконання функцій розпаралелювання програм.

Загальними ознаками пропонованого винаходу в порівнянні з аналогом є:

- Систему розпаралелювання реалізовано в "межах комп'ютера", який містить процесор, пам'ять і інтерфейс з реалізацією функцій розподілу програми користувача для паралельної обробки багатопроцесорною системою, що має множину процесорів.

Причинами, що заважають досягненню поставленого технічного завдання, є:

- усі засоби, включаючи засоби передкомпіляції програми користувача, розділення її на завдання, розподіл завдань по процесорах і інші реалізовано згідно з першими трьома втіленнями винаходу в "межах комп'ютера" програмними способами, що істотно збільшує час розподілу застосування;

- згідно з четвертим втіленням винаходу, де представлено структурну схему апаратної частини реалізації розпаралелювання, система розпаралелювання не може бути ефективно використана для робочих систем, виконаних на інших апаратно, - програмних платформах і інших принципах архітектурно-структурної організації, оскільки ця система розпаралелювання орієнтована на використання спільно з розподіленою обчислювальною системою, архітектура якої істотно відрізняється від поширеної класичної архітектури за рахунок застосування спеціального (розширеного) каналу зв'язку з блоками процесорів разом з додатковим блоком для кожного процесора, що реалізовує механізм паралельної обробки.

Відома також система і метод для розподілу програм серед взаємодіючих процесорів [див. пат. США "System and method for the distribution of a program among cooperating processors". № 7765536, 27.07.2010]. При цьому система включає множину мікропроцесорів, кожен з яких включає локальну пам'ять. В якості мікропроцесорів може бути будь-який мікропроцесор, наприклад Intel Pentium 5. Локальна пам'ять пов'язана з мікропроцесором і може бути будь-якого типу, наприклад, SRAM, DRAM, SDRAM, RDRAM, DDR-2 і так далі.

Мікропроцесори пов'язані один з одним системною шиною. Периферійні пристрої, типу жорсткого диска, CD-ROM і мережний інтерфейс також підключені до системної шини. Можливе застосування додаткового загальнодоступного блоку пам'яті, приєднаного до системної шини, через який мікропроцесори можуть спілкуватися один з одним, використовуючи семафори, щоб синхронізувати свої операції.

Архітектура такої багатопроцесорної системи добре відома, і описана тут система - тільки приклад архітектури такого типу.

Загальними ознаками пропонованого винаходу в порівнянні з цим аналогом є:

- як систему для розподілу програми користувача застосовують процесор з локальною пам'яттю, в якій зберігають програму користувача і спеціальну програму, що включає стратегії розподілу;

5 - як робочу систему, на якій реалізують розподілену програму користувача, застосовують систему, що містить множину процесорів, кожен з яких має власну локальну пам'ять і сполучений із загальною шиною, до якої також підключені накопичувачі на жорстких дисках, CD-ROM, адаптери і інші пристрої, необхідні для роботи з комп'ютером.

Причинами, що заважають досягненню поставленого технічного завдання, є:

10 - застосування для кожного процесора робочої системи великої місткості пам'яті внаслідок необхідності копіювання програми користувача або її частин в пам'ять кожного з n розташованих процесорів, і розміщення в пам'яті кожного процесора відповідного підпотoku даних, отриманого в результаті демультимплексування усього потоку даних; це збільшує вартість усієї розподіленої системи і ускладнює її реалізацію на одному кристалі згідно сучасної тенденції створення однокристальних розподілених комп'ютерних систем;

15 - висока трудомісткість і великий час реалізації розпаралелювання цільової програми (програми користувача), оскільки процес розподілу виконують програмним способом, при цьому вибір робочої (кращої) стратегії паралелізму здійснюють тільки після виконання усіх наявних стратегій розпаралелювання і оцінки їх часу виконання, у тому числі - часу обчислювального процесу і часу розриву між одержанням програм (чи частин програми) і відповідних даних.

20 Найбільш близьким до запропонованої системи по технічній суті і задачі, що розв'язується, є система для управління розподілом програм, яку викладено в патенті [див. патент США "System for managing distribution of programs". №7437723, 14.10.2008].

Система для розподілу програми користувача включає два ідентичні комп'ютери, які входять до складу робочої системи, що містить множину таких же комп'ютерів, пов'язаних мережею. При цьому кожен комп'ютер містить (фіг. 1): процесор 1, пам'ять 2, накопичувач 3, інтерфейс користувача 4 і мережевий інтерфейс 5, підключені через відповідні для кожного пристрою входи - виходи до загальної шини 6. Архітектура і структура кожного з перерахованих пристроїв ідентичні архітектурі і структурі цих пристроїв класичного комп'ютера. При цьому один з названих комп'ютерів системи для розподілу програми користувача виконує функції контролера (control apparatus), а інший - функції показника дії (target apparatus). Перший комп'ютер реєструє і виконує попередній аналіз програм користувача із запам'ятовуванням таблиць стану посилань між програмами, таблиць стану доставки застосування для обробки на другому комп'ютері системи, очікування доставки, а також формує блок інформаційного забезпечення і відповідний пакет для передачі інформації другому комп'ютеру.

35 Другий комп'ютер приймає від першого комп'ютера пакети інформації, аналізує їх, формує інформаційно-довідкові файли, інсталує і деінсталує програми, встановлює адресні посилання програм, управляє командами обробки і формує таблицю управління програмами.

Загальними ознаками пропонованого винаходу з прототипом є:

40 - процесор, оперативна пам'ять, накопичувач, інтерфейс користувача;
- для управління процесом розподілу інформації і її передачі між комп'ютерами робочої системи, що виконує програму користувача, використовують принцип передачі повідомлень, основою якого є інформаційні пакети.

Причинами, що заважають досягненню поставленого технічного завдання, є:

45 - тривалий час виконання процесу розподілу програми користувача, оскільки система розподілу виконана у вигляді комплексу програм, які розміщено на двох комп'ютерах, між якими в процесі розподілу формують інформаційні пакети і виконують численні пересилки цих пакетів з метою їх аналізу і обробки;

50 - відсутність апаратно-програмних засобів підтримки реалізації окремих найбільш відповідальних ділянок програми розподілу з метою виключення втручання на цих ділянках користувача (менеджера), що призводить до збільшення кількості ітерацій і необґрунтованого результату розподілу програми користувача.

Так менеджер перевіряє наявність залежностей за даними між блоками програми користувача, визначає апаратуру, на якій розміщують ці програми, визначає пакет передачі програми користувача як предмет обробки і блок задання як адресат видачі, визначає зміст обробки і передачу пакета програми користувача, вибирає посилання адрес, які мають бути збережені, а також комбінацію ідентифікатора програми користувача і ідентифікатора блока для програми користувача, які будуть видалені, визначає умови видалення програми користувача ("стерти", "не стирати", "змінити"), на які система реагує відповідним чином.

60 В основу винаходу поставлено задачу створити систему розподілу програми користувача, в якій, завдяки введенню нових технічних засобів, що підтримують застосування нових моделей,

алгоритму і критеріїв розподілу, забезпечено скорочення часу виконання процесу розподілу програми користувача, розширення сфери застосування системи розподілу для робочих систем, виконаних на різних апаратно-програмних платформах, і тим самим підвищення серійно-спроможності і скорочення вартості експлуатації системи.

5 Розв'язання поставленої задачі досягається тим, що система для розподілу програми користувача містить процесор, оперативну пам'ять, накопичувач, інтерфейс користувача, перші входи - виходи яких підключені до загальної шини, блок управління, блок пам'яті мікропрограм, блок пам'яті таблиць, логічний блок, блок розподілу, блок проміжної інформації, блок буферної пам'яті, інтерфейс з хост-машиною, інтерфейс з робочою системою, при цьому входи - виходи
10 блока пам'яті мікропрограм сполучені з другими входами-виходами процесора, треті входи-виходи якого підключені до перших входів-виходів блока управління, виходи якого сполучені з відповідними входами блока пам'яті мікропрограм, а його входи підключені до перших виходів інтерфейсу з хост-машиною, другі виходи якого сполучені з першими входами блока проміжної інформації, виходи якого сполучені з першими входами блока пам'яті таблиць, другі входи якого
15 сполучені з першими виходами процесора, другі виходи якого сполучені з відповідними першими входами логічного блока, з першими входами блока розподілу і блока буферної пам'яті, другі входи якого сполучені з відповідними виходами блока розподілу, другі входи якого сполучені з першими виходами блока пам'яті таблиць, другі виходи якого сполучені з другими входами логічного блока, перші входи - виходи якого підключені до других входів-виходів
20 оперативної пам'яті, а його другі входи-виходи підключені до відповідних входів-виходів блока проміжної інформації, другі виходи якого сполучені з відповідними входами оперативної пам'яті, треті входи-виходи якої сполучені з другими входами - виходами блока управління, четверті входи- виходи оперативної пам'яті підключені до відповідних входів-виходів блока пам'яті таблиць, а її п'яті входи-виходи підключені до перших входів- виходів блока розподілу, треті
25 входи якого сполучені з відповідними виходами логічного блока, треті входи-виходи якого сполучені із загальною шиною, яка підключена до других входів-виходів блока розподілу і до перших входів - виходів інтерфейсу з робочою системою, входи якого сполучені з першими виходами блока буферної пам'яті, другі виходи якого сполучені з відповідними входами інтерфейсу з хост-машиною, перші входи-виходи якого підключені до загальної шини, а його
30 другі і треті входи-виходи є відповідно першими і другими входами-виходами системи, другі і треті входи - виходи інтерфейсу з робочою системою є відповідно третіми і четвертими входами- виходами системи для розподілу програми користувача.

Відмітними ознаками пропонованого пристрою є те, що до складу системи введені блок управління, блок пам'яті мікропрограм, блок пам'яті таблиць, логічний блок, блок розподілу,
35 блок проміжної інформації, блок буферної пам'яті, інтерфейс з хост-машиною, інтерфейс з робочою системою, при цьому входи-виходи блока мікропрограм сполучені з другими входами-виходами процесора, треті входи- виходи якого підключені до перших входів-виходів блока управління, виходи якого сполучені з відповідними входами блока пам'яті мікропрограм, а його входи підключені до перших виходів інтерфейсу з хост-машиною, другі виходи якого сполучені з
40 першими входами блока проміжної інформації, виходи якого сполучені з першими входами блока пам'яті таблиць, другі входи якого сполучені з першими виходами процесора, другі виходи якого сполучені з відповідними першими входами логічного блока, з першими входами блока розподілу і блока буферної пам'яті, другі входи якого сполучені з відповідними виходами блока розподілу, другі входи якого сполучені з першими виходами блока пам'яті таблиць, другі
45 виходи якого сполучені з другими входами логічного блока, перші входи-виходи якого підключені до других входів-виходів оперативної пам'яті, а його другі входи-виходи підключені до відповідних входів-виходів блока проміжної інформації, другі виходи якого сполучені з відповідними входами оперативної пам'яті, треті входи-виходи якої сполучені з другими входами-виходами блока управління, четверті входи-виходи оперативної пам'яті підключені до
50 відповідних входів-виходів блока пам'яті таблиць, а її п'яті входи- виходи підключені до перших входів-виходів блока розподілу, треті входи якого сполучені з відповідними виходами логічного блока, треті входи-виходи якого сполучені із загальною шиною, яка підключена до других входів - виходів блока розподілу і до перших входів - виходів інтерфейсу з робочою системою, входи якого сполучені з першими виходами блока буферної пам'яті, другі виходи якого сполучені з
55 відповідними входами інтерфейсу з хост - машиною, перші входи - виходи якого підключені до загальної шини, а його другі і треті входи-виходи є відповідно першими і другими входами-виходами системи, другі і треті входи-виходи інтерфейсу з робочою системою є відповідно третіми і четвертими входами-виходами системи для розподілу програми користувача.

Введення в запропоновану систему для розподілу програми користувача цих ознак в
60 порівнянні з системою-прототипом дозволяє:

- істотно скоротити час розподілу застосування за рахунок того, що процедуру підготовки даних усередині системи розподілу поєднують за часом з процедурою безпосереднього розподілу програми користувача, використовуючи наступні введені апаратні блоки, сполучені між собою відповідними зв'язками: блок управління, блок пам'яті мікропрограм, блок проміжної інформації, блок пам'яті таблиць, блок вихідних пакетів, логічний блок, блок розподілу, блок буферної пам'яті, блок інтерфейсу з робочою системою (PIM - системою), блок інтерфейсу з хост-машиною;

- розширити сферу застосування запропонованої системи розподілу програми користувача для робочих систем, виконаних на різних апаратно-програмних платформах за рахунок того, що процес розподілу виконують під управлінням мікропрограм, які записують у блок мікропрограм з урахуванням особливостей архітектурно-структурної організації робочої системи, яка повинна виконувати програми, і особливостей безпосередньо стратегії і алгоритму розподілу;

- зменшити вартість запропонованої системи розподілу за рахунок того, що така система має широку сферу застосування і отже - серійно-спроможність, а також додатково зменшити вартість її експлуатації і енергоспоживання за рахунок скорочення часу усього процесу розподілу.

Для ілюстрації запропонованої системи для розподілу програми користувача і пояснення суті її роботи наведені наступні креслення:

На фіг. 1 наведено блок-схему системи-прототипу для розподілу програми користувача.

На фіг. 2 наведено блок-схему запропонованої системи для розподілу програми користувача.

На фіг. 3 наведено укрупнену блок-схему стратегії розподілу застосування, алгоритм реалізації якої підтримує запропонована система розподілу.

На фіг. 4 наведено узагальнену структуру вхідного управляючого пакета.

На фіг. 5 наведено узагальнену структуру вихідного пакета, що відображає результати розподілу.

Блок-схему запропонованої системи для розподілу програми користувача наведено на фіг.2, яка містить: процесор 1, оперативну пам'ять 2, накопичувач 3, інтерфейс користувача 4, перші входи-виходи яких підключені до загальної шини 5, блок управління 6, блок пам'яті мікропрограм 7, блок пам'яті таблиць 8, логічний блок 9, блок розподілу 10, блок проміжної інформації 11, блок буферної пам'яті 12, інтерфейс з хост-машиною 13, інтерфейс з робочою системою 14.

При цьому входи-виходи блока мікропрограм 7 сполучені з другими входами-виходами процесора 1, треті входи-виходи якого підключені до перших входів-виходів блока управління 6, виходи якого сполучені з відповідними входами блока пам'яті мікропрограм 7, а його входи підключені до перших виходів інтерфейсу з хост-машиною 13, другі входи якого сполучені з першими входами блока проміжної інформації 11, виходи якого сполучені з першими входами блоку пам'яті таблиць 8, другі входи якого сполучені з першими виходами процесора 1, другі входи якого сполучені з відповідними першими входами логічного блоку 9, з першими входами блока розподілу 10 і блока буферної пам'яті 12, другі входи якого сполучені з відповідними виходами блока розподілу 10, другі входи якого сполучені з першими виходами блока пам'яті таблиць 8, другі входи якого сполучені з другими входами логічного блоку 9, перші входи-виходи якого підключені до других входів-виходів оперативної пам'яті 2, а його другі входи-виходи підключені до відповідних входів-виходів блока проміжної інформації 11, другі входи якого сполучені з відповідними входами оперативної пам'яті 2, треті входи-виходи якої сполучені з другими входами-виходами блока управління 6, четверті входи-виходи оперативної пам'яті 2 підключені до відповідних входів - виходів блоку пам'яті таблиць 8, а її п'яті входи - виходи підключені до перших входів виходів блока розподілу 10, треті входи якого сполучені з відповідними виходами логічного блоку 9, треті входи виходи якого сполучені із загальною шиною 5, яка підключена до других входів-виходів блока розподілу 10 і до перших входів-виходів інтерфейсу з робочою системою 14, входи якого сполучені з першими виходами блоку буферної пам'яті 12, другі входи якого сполучені з відповідними входами інтерфейсу з хост-машиною 13, перші входи-виходи якого підключені до загальної шини 5, його другі і треті входи-виходи є відповідно першими і другими входами-виходами системи, а другі і треті входи-виходи інтерфейсу з робочою системою 14 є відповідно третіми і четвертими входами-виходами системи.

При цьому процесор 1, призначений для підготовки початкової інформації усередині системи, необхідної для процедур розподілу фрагментів програми користувача і розподілу отриманих її частин (блоків) між процесорами робочої системи (в даному випадку PIM-системи), для формування проміжної інформації у вигляді файлів і таблиць параметрів алгоритму

програми користувача, поточних таблиць параметрів робочої системи, для формування відповідних запитів і звернень до усіх блоків системи, формування інформаційних пакетів для усіх рівнів розподілу та ін.

Алгоритм функціонування процесора на різних рівнях розподілу програми користувача визначається відповідною до кожного рівня моделлю стратегії розподілу, яку представлено на вході процесора у вигляді послідовності відповідних кодів мікрокоманд, що поступає з входів - виходів блоку пам'яті мікропрограм 7. При цьому як процесор цієї системи може бути застосовано один з процесорів, що серійно випускаються, наприклад, Pentium IV ф. Intel, або Power PC 750 FX ф. IBM, або їх аналоги.

Оперативна пам'ять 2 призначена для тимчасового зберігання інформації, отриманої в результаті виконання процесором 1 арифметичних і логічних операцій, у тому числі - для тимчасового зберігання сформованих файлів, даних і програм різного призначення. Для побудови такої пам'яті можуть бути використані мікросхеми типу DDR, DR DRAM, SD RAM і інші.

Накопичувач 3 призначений для зберігання програми користувача, яка підлягає розпаралелюванню і реалізації на робочій системі (PIM-системі), її параметрів, моделей стратегії розподілу для кожного рівня розподілу і

безпосередньо самої програми розподілу, початкових параметрів робочої системи (PIM - системи), даних і команд, що поступають від хост-машини, а також іншої інформації, необхідної для роботи системи розподілу програми користувача. Цей накопичувач може бути виконаний у вигляді стандартних накопичувачів на жорстких дисках, що використовуються нині в персональних комп'ютерах і серверах, наприклад, фірм - виробників Hitachi, Samsung, Western Digital та ін.

Інтерфейс користувача 4 призначений для взаємодії оператора (користувача) з системою розподілу з метою контролю і при необхідності коригування початкових і проміжних даних при виконанні процедур розподілу програми користувача. Цей інтерфейс зазвичай представлений у вигляді клавіатури і монітора стандартних типів.

Загальна (системна) шина 5 призначена для зв'язку між блоками і пристроями системи через їх відповідні перші входи-виходи. Як загальна шина можуть бути використані стандартні шини типу PCI або PCI-Express.

Наступні блоки 6-14 (фіг.2) виконані із застосуванням логічних, процесорних, регістрових схем, а також схем пам'яті з використанням відповідних зв'язків. При цьому усі вказані блоки розміщені на одному кристалі програмованої логічної інтегральної схеми (ПЛИС) типу Virtex 5 або Virtex 6 фірм Xilinx [Сімейство ПЛИС фірми Xilinx типу FPGA. Створення принципових схем і програмування ПЛИС типу FPGA. Режим доступу: http://www.fpga-cpld.ru/sapr_7.html. - Дата доступу: 24.01.12, а також: "Офіційний сайт фірми Xilinx". Режим доступу: <http://www.Xilinx.com>. - Дата доступу: 24.01.12.].

Блок управління 6 призначений для прийому, тимчасового зберігання і аналізу вхідного управляючого пакета, що надходить від хост-машини, а також пакетів, сформованих за результатами розподілу програми користувача на кожному рівні. Цей блок розшифровує відповідні поля пакетів і видає управляючі сигнали, які поступають на блок пам'яті мікропрограм 7, процесор 1, оперативну пам'ять 2 для ініціалізації видачі процесору 1 послідовності кодів мікрокоманд, а також для запуску процедур, що виконуються блоками системи на кожному рівні розподілу у відповідності із запропонованою загальною стратегією і моделями розподілу програми користувача. Цей блок управління виконаний на процесорному елементі, регістрових і логічних схемах, сполучених між собою та з відповідними входами - виходами цього блока, який розміщений на кристалі ПЛИС.

Блок пам'яті мікропрограм 7 призначений для зберігання і видачі процесору 1 послідовності управляючих кодів мікрокоманд для кожного рівня розподілу у відповідність із запропонованою загальною стратегією і моделями розподілу програми користувача. Так само, як і блок управління 6, блок пам'яті мікропрограм 7 містить стандартні для пам'яті компоненти і виконаний на елементах пам'яті і логічних елементах ПЛИС типу Virtex 5 або Virtex 6, сполучених між собою і з входами-виходами відповідними зв'язками.

Блок пам'яті таблиць 8 призначений для зберігання і видачі сформованих кодів таблиць параметрів робочої системи, таблиць параметрів програми користувача, таблиць зв'язку програмних блоків за даними і кодів іншої інформації, необхідної для процедур розподілу і формування кодів полів інформаційних пакетів. Виконаний на елементах пам'яті і логічних елементах, розміщених на кристалі ПЛИС.

Логічний блок 9 призначений для порівняння кодів команд процесорів робочої системи з наборами операцій блоків фрагмента програми користувача на кожному рівні розподілу, формування таблиць їх відповідності і передачі у блок пам'яті таблиць 8, а також для

розрахунків вагових коефіцієнтів команд для процесорів ВП і ПЯ*. Виконаний на процесорному елементі, логічних елементах і елементах пам'яті, сполучених між собою і з входами-виходами блока 9 відповідними зв'язками, і розміщений на кристалі ПЛІС.

Блок розподілу 10 призначений для розподілу програмних блоків по ярусах ярусно-паралельного графа, за допомогою якого представлені програмні блоки і зв'язки між ними, а також для визначення порядку виконання програмних блоків і розподілу їх по процесорах PIM-системи. Виконаний на процесорному елементі, логічних елементах і елементах пам'яті, сполучених між собою і з входами-виходами блока відповідними зв'язками, і розміщений на кристалі ПЛІС.

Блок проміжної інформації 11 призначений для тимчасового зберігання кодів даних поточних параметрів таблиць робочої системи, параметрів алгоритму програми користувача, параметрів зв'язку блоків програми за даними, сформованих процесором 1 та іншими блоками впродовж виконання процедур розділення алгоритму на блоки і розподілу їх по процесорах. Також, як і інші блоки, виконаний на елементах ПЛІС.

Блок буферної пам'яті 12 призначений для зберігання черги видачі в робочу систему (PIM-систему) кодів вихідних пакетів кожного ярусу з можливістю перекриття в часі процесів розподілу кодів програми користувача або її частин і процесів виконання в робочій системі безпосередньо програми користувача або її частин, розподілених раніше по процесорах, що забезпечує скорочення загального часу розподілу програми користувача. Виконаний на логічних елементах і елементах пам'яті ПЛІС, що утворюють структуру буферної пам'яті типу FIFO (First in-First out).

Інтерфейс з хост - машиною 13 призначений для передачі кодів команд і даних між хост - машиною і системою для розподілу програми користувача. Виконаний на елементах ПЛІС.

Інтерфейс з робочою системою 14 призначений для передачі кодів команд і даних між робочою системою, що реалізовує програму користувача (PIM- системою) і системою для розподілу цієї програми. Виконаний на елементах ПЛІС.

Роботу системи для розподілу програми користувача розглянемо стосовно розподіленої гетерогенної робочої системи типу "Процесор - в -пам'яті" ("Processor-in-memory") або PIM - системи, яку виконано на одному кристалі, де розподілена по процесорах програма має бути реалізована. Модель стратегії розподілу програми користувача для її реалізації на PIM - системі представляють у вигляді трьох рівнів, що відображає трирівневу архітектурно-структурну організацію PIM- системи [Яковлев Ю.С. Однокристалые компьютерные системы высокой производительности. Особенности архитектурно-структурной организации и внутренних процессов: монография / Ю.С. Яковлев. - Винница: ВНТУ, 2009. - 294 с.]

На першому рівні виконують розділення програми користувача на фрагменти програми, що містять блоки програм, у відповідності з моделлю першого рівня і розподіляють отримані фрагменти програм між хост-машиною і чипами (кристалами) робочої системи, кожен з яких містить розподілену багатопроцесорну PIM-систему. При цьому перевага віддається розділенню на великі фрагменти, де кожен фрагмент призначений для самостійної реалізації на окремому чипі (кристалі) PIM-системи. Розділення виконують так, щоб взаємозв'язок за даними між цими фрагментами здійснювався в основному підсумковими значеннями результатів їх реалізації. При цьому взаємодію між компонентами розподіленої системи здійснюють за принципами передачі повідомлень, тому основними структурними одиницями вхідної і вихідної інформації системи для розподілу програми користувача є інформаційні пакети.

У рамках цього винаходу перший рівень розподілу усієї програми користувача не розглядається. Справжній об'єкт винаходу поширюється на однокристалний варіант організації робочої системи (в даному випадку PIM-системи), тобто на другий і третій рівні розподілу фрагмента програми (задачі) користувача, що на даний момент часу є найбільш перспективним у зв'язку з досягнутими успіхами інтегральної технології, що забезпечують на одному кристалі розміщення множини процесорів з пам'яттю і комутаційним середовищем.

У відповідність з моделлю розподілу програми користувача на другому рівні процес розподілу виконують між ВП і еквівалентним ПЯ*. При цьому формують основні параметри еквівалентного ПЯ* для цього чипа. Оскільки усі ПЯ в наборі ПЯ* на одному чипі - однакові, то за систему команд ПЯ* приймають систему команд одного ПЯ, місткість пам'яті ПЯ* приймають рівній сумарній місткості пам'яті, підключеної до усіх ПЯ на кристалі, а в якості часу виконання будь-якої команди ПЯ* приймають час виконання команди одного ПЯ, поділену на кількість ПЯ в наборі ПЯ*, враховуючи одночасну їх роботу.

На третьому рівні виконують розподіл програмних блоків по ПЯ, які входять в набір ПЯ*. Розміщення цих програмних блоків між однаковими ПЯ може бути виконано з використанням параметричної ваги кожного програмного блока і зв'язків за даними між блоками. З програмних

блоків, отриманих в результаті розподілу між ВП і усіма ПЯ одного чипа, формують групи (яруси) блоків, які можуть виконуватися паралельно, і визначають послідовність виконання таких груп (ярусів). На основі цієї інформації формують для кожного ярусу вихідні інформаційні пакети, які містять атрибути, що визначають їх приналежність до відповідних ВП і ПЯ усередині чипа. Ці пакети передають у PIM-систему для реалізації програми користувача, яку розпаралелено.

До початку безпосереднього розподілу програми користувача формують за допомогою хост-машини початкову інформацію, яку разом з програмою користувача, представленою у вигляді її фрагментів (задач), що містять програмні блоки і написані на мові програмування, записують в накопичувач 3 через його відповідні входи-виходи, загальну шину 5, перші, другі і треті входи-виходи інтерфейсу з хост-машиною 13 (фіг. 2).

До основної початкової інформації, яка потрібна для роботи системи розподілу програми користувача, належить:

1). Файл з описом багаторівневої стратегії і моделей розподілу програми користувача для кожного рівня розподілу з наступною структурою: ідентифікатор рівня розподілу програми користувача, ідентифікатор коду адреси мікропрограми, розміщеної у блоці мікропрограм 7 і відповідної до номеру рівня стратегії розподілу, ідентифікатор коду адреси першого критерію для цього номера рівня розподілу (по відповідності систем команд процесорів PIM - системи наборам операцій фрагмента програми користувача), ідентифікатор коду адреси другого критерію розподілу (по значеннях параметричної ваги фрагментів програми), ознака наявності зв'язку програмних блоків усередині фрагмента (задачі).

2). Фрагмент програми, розділений на блоки, у вигляді: ідентифікатор першого програмного блока, список інструкцій мови програмування, що входять до складу цього блоку, ідентифікатор другого програмного блока, список інструкцій цього блока і т. д.

3). Файл з описом графа залежності між блоками фрагмента (задачі) початкової програми. Структура файлу наступна: ідентифікатор вершини, список дочірніх вершин, ідентифікатор наступної вершини, список дочірніх вершин і так далі доки не будуть перераховані усі вершини графа. Під вершинами графа тут розуміються програмні блоки, на які розділений фрагмент початкової програми. Множина ребер графа - це залежності між блоками.

4). Файл з інформацією про кількість ітерацій для кожного циклу, що входить в початковий фрагмент програми.

5). Описи системи команд ВП і ПЯ, представлені у вигляді таблиць, що складаються з наступних полів: код команди, ідентифікатор приналежності команди конкретному типу процесора (ВП або ПЯ), параметрична вага команди для відповідного процесора. Під параметричною вагою команди для цього процесора розуміється час виконання цієї команди на цьому процесорі. Причому, передбачається, що система команд ПЯ є підмножиною системи команд ВП.

6). Описи першого і другого критеріїв розподілу, представлені у вигляді таблиць, що складаються з наступних полів: ідентифікатор номера рівня розподілу, ідентифікатор коду адреси розміщення першого критерію, що відповідає номеру рівня розподілу, ідентифікатор коду адреси розміщення другого критерію для цього рівня.

7). Додаткова інформація про параметри PIM-системи, яка може бути представлена у вигляді відповідної початкової таблиці або набору файлів: розрядність і місткість пам'яті ВП, тривалість одного такту роботи ВП, набір і значення параметрів еквівалентного ВП*, кількість ПЯ на одному чипі,

розрядність ПЯ і місткість його банку пам'яті, тривалість одного такту роботи ПЯ, набір і значення параметрів еквівалентного ПЯ*, швидкість передачі інформації по каналу процесор - пам'ять (ширина смуги пропускання інформації по цьому каналу) та ін.

8). Додаткова інформація про параметри програми (задачі) користувача, яка також може бути представлена у вигляді таблиці або набору файлів: типи і кількість операцій програми (задачі) користувача, типи і кількість незалежних за даними задач і блоків програм, типи і кількість послідовних ділянок програм, частота повторення кожної j-ої операції програми користувача, розрядність обробки даних, набір і параметри програм реалізації відомих методів розпаралелювання при розв'язанні типових завдань (добуток матриць та інші).

Для роботи системи розподілу програми користувача спочатку готують первинну початкову інформацію, яку запам'ятовують в оперативній пам'яті 2. Для цього від хост-машини передають вхідний управляючий пакет, який через перші входи-виходи цієї системи і через інтерфейс з хост-машиною 13 надходить на відповідні входи блока управління 6, де цей пакет запам'ятовують на регістрах на час його аналізу і формування відповідних сигналів, що управляють. Укрупнену блок-схему стратегії розподілу програми користувача, алгоритм

реалізації якої підтримує запропонована система для розподілу програми користувача, відображено на фіг.3, а варіант структури вхідного управляючого пакету - на фіг. 4.

Спочатку блок 6 розшифровує код першого поля вхідного управляючого пакета (фіг.4), який вказує, що параметри і ідентифікатори параметрів усіх полів цього пакета належать до другого рівня розподілу. Управляючі сигнали отримані після розшифровки цього коду, з перших входів-виходів блока 6 надходять на треті входи-виходи процесора 1. Одночасно з других входів-виходів блока управління 6 на треті входи-виходи оперативної пам'яті 2 поступають коди ідентифікаторів другого, третього, четвертого і п'ятого полів вхідного пакету, використовуючи які процесор 1 формує коди адрес для зчитування з накопичувача 3 і передачі в оперативну пам'ять 2 початкової інформації для цього рівня розподілу, у тому числі: відповідні файли з параметрами фрагмента алгоритму програми користувача, параметрами робочої системи (PIM-системи), безпосередньо сам фрагмент програми, блоки якого підлягають розподілу по процесорах робочої системи, інформація про критерії розподілу, файл з описом таблиці зв'язності за даними і ін... Процесор 1 сприймає вказані коди полів пакета по ланцюгу: перші входи-виходи оперативної пам'яті 2, загальна шина 5, перші входи-виходи процесора 1, а передача прочитаних з блока 3 у блок 2 кодів параметрів фрагмента алгоритму, кодів параметрів робочої системи, безпосередньо кодів самого фрагмента алгоритму і іншої початкової інформації відбувається по ланцюгу: перші входи - виходи накопичувача 3, загальна шина 5 і перші входи-виходи оперативної пам'яті 2. При цьому процесор 1 структурує отримані коди параметрів у вигляді таблиць і записує їх у блок пам'яті таблиць 8 по ланцюгу: перші входи-виходи процесора 1, другі входи-виходи блока пам'яті таблиць 8.

Далі за допомогою блока 6 розшифровують код шостого поля управляючого пакета і за наявності коду інсталяції виконують інсталяцію програми розподілу шляхом розміщення усіх необхідних програмних файлів у відповідних місцях пам'яті системи для розподілу програм користувача. Крім того, блок 6 аналізує інформацію сьомого поля управляючого пакету і за наявності ознаки ініціалізації процесор 1 присвоює початкові значення змінним.

Таким чином, підготовлена необхідна інформація для розподілу програми користувача між ВП і ПЯ*, яку розміщено в оперативній пам'яті 2, а систему для розподілу фрагмента програми користувача між ВП і ПЯ* приведено в початковий стартовий стан. При цьому користувач може ознайомитися з цією інформацією і при необхідності уточнити її, використовуючи для цього інтерфейс користувача 4, який відповідними входами-виходами підключений до загальної шини 5.

Безпосередньо процес розподілу блоків фрагменту програми користувача по процесорах робочої системи (PIM-системи) починають з розшифровки восьмого поля вхідного управляючого пакета (фіг.4) за допомогою блока аналізу вхідних пакетів 6. При цьому відповідні управляючі сигнали, з перших входів-виходів цього блока надходять на треті входи-виходи процесора 1. Процесор по цих сигналах формує перший запит, який через перші свої входи-виходи і через загальну шину 5 поступає до накопичувача 3. Накопичувач 3 по цьому запиту видає в оперативну пам'ять 2 через відповідні входи - виходи і через загальну шину 5 файл з описом стратегії і моделей розподілу (див. початкову інформацію, позначену вище). Процесор 1, використовуючи цей файл, формує другий запит, який посилає через другі свої входи - виходи у блок пам'яті мікропрограм 7 з метою ініціалізації цього блоку, який через відповідні другі входи - виходи на відповідні реєстри процесора 1 видає послідовність кодів мікрокоманд, ініціюючи його на реалізацію функцій для другого рівня розподілу програми користувача.

У відповідність з цією послідовністю процесор 1 формує коди запитів до оперативної пам'яті 2, який передає по ланцюгу: перші входи - виходи процесора 1, загальна шина 5, перші входи - виходи оперативної пам'яті 2, яка по цьому запиту видає в логічний блок 9 початкову інформацію, необхідну для визначення цим блоком відповідності системи команд ВП і ПЯ* наборам операцій блоків програми по ланцюгу: другі входи - виходи оперативної пам'яті 2, перші входи - виходи логічного блоку 9. Одночасно процесор 1 по ланцюгу: перші виходи процесора 1, другі входи блока пам'яті таблиць 8 формує запит до блока пам'яті таблиць 8 для видачі в логічний блок 9 кодів таблиць параметрів PIM-системи, кодів таблиць параметрів блоків алгоритму завдання користувача по ланцюгу: другі виходи блока пам'яті таблиць 8, другі входи логічного блоку 9. Логічний блок 9 за наявності на його перших входах коду сигналу запуску, який надходить від других виходів процесора 1, виконує наступні функції:

1). Вибір чергового блока фрагмента програми по ланцюгу: треті входи-виходи блока 9, загальна шина 5, входи-виходи оперативної пам'яті 2.

2). Формування структури таблиці команд для програмного блока, що аналізується: код операції, вага операції для ВП, вага операції для ПЯ; L (число, що показує, скільки раз цю

операцію буде виконано у блоці). На цьому етапі таблиця порожня і заповнюватиметься по мірі знаходження операцій у блоці.

3). Лексичний аналіз блока і отримання списку tokenів цього блока з використанням програми аналізу, яку розміщено в накопичувачі 3 і передано в логічний блок 9 по ланцюгу: входи-виходи накопичувача 3, загальна шина 5, треті входи- виходи логічного блока 9. За результатами лексичного аналізу програмний код блока розділяють на відповідні токени. Кожен токен складається з двох частин: код токenu і лексема. Лексема - послідовність символів мови високого рівня (МВР), на якій написано програмний код блоку, що має сенс для компілятора цієї МВР. Код токenu вказує на тип, до якого належить лексема, наприклад ідентифікатор, арифметична операція, знак пунктуації і так далі.

4). Аналіз списку tokenів: виділення у блоці чергової операції і встановлення усередині яких циклів знаходиться ця операція і, відповідно, скільки раз її буде виконано на цьому етапі (позначимо цю кількість через ℓ).

5). Пошук виділеної операції в таблиці команд цього блока. Якщо такої операції в таблиці команд блока немає, то перейти до п. 6, інакше - до п. 7.

6). Додавання нової операції в таблицю команд блока: в таблицю команд додається новий запис, в поле коду операції вноситься код операції, виділеної у блоці, в полі L записується значення ℓ . Інші поля поки що не заповнюються. Перехід до п. 9.

7). Зміна запису про цю операцію в таблиці команд: збільшення числа L на значення ℓ . Інші поля (з даними про вагу) не змінюються, оскільки ця операція вже зустрічалася раніше у блоці, вага і можливість виконання її на кожному з процесорів вже встановлені. Якщо перегляд блоку завершений, то перехід до п. 10, інакше повернення до п. 4.

8). Пошук в системі команд ВП. Пошук в таблиці, що містить систему команд ВП команди з кодом, що дорівнює коду операції, виділеної у блоці.

Якщо код відповідної операції знайдено, то з таблиці вибирається вага цієї команди і записується в таблицю команд цього блоку в полі "вага операції для ВП", інакше в таблицю команд в полі "вага операції для ВП" заноситься -1, що показує, що виконання цієї операції, а значить і усього блока в цілому на ВП неможливе.

9). Пошук в системі команд ПЯ. Пошук в таблиці, що містить систему команд ПЯ команди з кодом, що дорівнює коду операції, виділеної у блоці. Якщо код відповідної операції знайдено, то з таблиці вибирається вага цієї команди і записується в таблицю команд цього блока в полі "вага операції для ПЯ", інакше в таблицю команд в полі "вага операції для ПЯ" заноситься - 1, що показуватиме надалі, що виконання цієї операції, а значить і усього блока в цілому, на ПЯ неможливо. Якщо перегляд блока ще не завершений (не усі токени проглянуто), то перехід до п. 4.

10). Обчислення ваги блока для ВП. Якщо раніше не було встановлено, що виконання блока на ВП неможливе, то обчислюють вагу блока для ВП за формулою:

$$W_{ВП} = \sum_{i=1}^n p_{ВП_i} \times L_i$$

де n - кількість записів в таблиці команд, $p_{ВП_i}$ - вага i -ої команди для ВП, L_i - скільки раз i-ту команду може бути виконано у цьому блоці. Інакше вазі цього блока для ВП присвоюють значення - 1.

11). Обчислення ваги блока для ПЯ. Якщо раніше не було встановлено, що виконання блоку на ПЯ неможливе, то обчислюють вагу блока для ПЯ за формулою:

$$W_{ПЯ} = \sum_{i=1}^n p_{ПЯ_i} \times L_i$$

де n - кількість записів в таблиці команд, $p_{ПЯ_i}$ - вага i -ої команди для ПЯ, L_i - скільки раз i-ту команду може бути виконано в цьому блоці. Інакше вазі цього блоку для ПЯ присвоюють значення - 1.

12). Якщо не усі блоки фрагменту програми розглянуті, то виконують п.1.

13). Результати роботи логічного блока 9 у вигляді таблиці ваги програмних блоків передають у блок проміжної інформації 11 для їх тимчасового збереження по ланцюгу: другі входи - виходи логічного блоку 9, відповідні входи - виходи блока проміжної інформації 11. У цій таблиці записані значення ваги для ВП і ПЯ для кожного програмного блок-фрагменту. Таблиця ваги блоків містить наступні поля: ідентифікатор блоку 1, вага блоку 1 для ВП, вага блоку 1 для ПЯ, ідентифікатор блоку 2, вага блоку 2 для ВП, вага блоку 2 для ПЯ, і так далі, доки не будуть

перераховані усі блоки, що входять у фрагмент. Якщо в якості значення ваги для якого-небудь процесора стоїть -1, це означає що цей блок на цьому процесорі виконуватися не може.

Після заповнення таблиці ваги для усіх програмних блоків фрагменту програми цю таблицю передають у блок пам'яті таблиць 8 по ланцюгу: виходи блоку проміжної інформації 11, перші входи блоку пам'яті таблиць 8.

Далі процесор 1 відповідно до послідовності прийнятих ним мікрокоманд видає код запуску блоку розподілу 10 по ланцюгу: другі виходи процесора 1, перші входи блоку розподілу 10. При цьому на другі входи цього блоку поступають коди таблиці ваги програмних блоків по ланцюгу: перші входи блоку пам'яті таблиць 8, другі входи блоку розподілу 10. Одночасно на другі входи-виходи блока розподілу 10 з п'ятих входів - виходів оперативної пам'яті 2 поступають коди програми для управління розподілом і інформація про зв'язок між блоками за даними, яку представлено у вигляді ациклічного графа з урахуванням розшифровки п'ятого, а також з дев'ятого по п'ятнадцяте поле вхідного пакету, які визначають джерела і приймачі посилань різного призначення між програмними блоками (фіг. 4). Вершинами цього графа є самі блоки, а ребрами - зв'язки між ними. Цей граф подається в ярусно-паралельній формі. До одного ярусу включають тільки ті блоки, для виконання яких потрібні дані, отримані на попередніх ярусах. В межах одного ярусу блоки не пов'язані і можуть виконуватися паралельно. Блок b_i , вважається безпосередньо передуючим (чи батьком) блоку b_j , якщо для виконання блоку b_j потрібні дані, отримані у блоці b_i (тобто існує ребро спрямоване від вершини b_i до вершини b_j). Блок b_i при цьому вважається безпосередньо наступним за блоком b_j (чи спадкоємцем блоку b_j).

Для розподілу блоків по ярусах для кожного блока обчислюється значення, яке називається порядком виконання блока, та показує до якого ярусу слід віднести блок. Порядок виконання визначається за наступними правилами. Для усіх блоків, що не мають попередників (батьків) призначається порядок рівний 1. Відповідно ці блоки будуть віднесені до ярусу з номером 1. Для усіх інших блоків порядок виконання визначається, як максимальне значення порядків виконання усіх батьківських блоків плюс 1. Значення порядку виконання відповідатиме номеру ярусу, до якого належить блок. Блоки з однаковим порядком можуть виконуватися паралельно.

Далі розглядають кожен ярус окремо. У середині одного ярусу блоки розподіляються по процесорах таким чином. Вибираються тільки ті блоки, що належать до ярусу, що розглядається. Блоки, які можуть виконуватися тільки на ВП і не можуть виконуватися на ПЯ, об'єднуються в один складений блок B_c і призначаються ВП. Усі інші блоки призначаються ПЯ*. Виконується баланс завантаження між ВП і ПЯ*. Якщо вага складеного блоку B_c для ВП менша, ніж найбільше значення ваги для програмних блоків ПЯ, призначених на ПЯ*, то додають в складений блок B_c блок (чи блоки) з призначених для ПЯ*, але так, щоб вага блоку B_c для ВП не перевищувала найбільше значення ваги для ПЯ блоків, що входять в ПЯ*.

Далі розподіляють блоки, призначені на ПЯ* між ПЯ, що входять в ПЯ* (третій рівень стратегії розподілу). Якщо кількість блоків не перевищує кількість процесорів ПЯ, то кожен блок призначається черговому вільному ПЯ. Якщо деякі блоки містять цикли з незалежними ітераціями, то можна їх розділити ще на декілька блоків з різними значеннями лічильників циклу, і заповнити ці незайняті ПЯ. Але при цьому за часом буде виграв тільки у тому випадку, якщо блоки з циклами виявляться найбільшими по вазі для ПЯ. Якщо кількість блоків перевищує кількість ПЯ, то обчислюють середнє значення ваги цих блоків для ПЯ (позначимо його M_{cp}). Далі виконуємо наступні дії:

1). Усі блоки, для яких вага ПЯ більша, ніж M_{cp} , призначають по одному на кожного ПЯ.

2). Блоки, що залишилися, об'єднують в складені блоки, так щоб їх вага не перевищувала M_{cp} .

3). Якщо кількість отриманих складених блоків перевищує кількість незайнятих процесорів, то анулюють результати розподілу, збільшують значення M_{cp} , наприклад, на одиницю, і повертаються знову до п. 1. Інакше кожен складений блок, отриманий на кроці 2 призначають на окремий ПЯ.

4). Якщо залишаться незайняті ПЯ, то можна деякі складені блоки розділити (якщо це можливо) і призначити незайнятим ПЯ.

5). Переходять до наступного ярусу і повторюють розподіл для блоків на наступному ярусі, і так далі доки не будуть розглянуті усі яруси.

За результатами розподілу програмних блоків по ярусах формують вихідну інформацію у вигляді наступної структури:

- Ідентифікатор ярусу 1, ідентифікатор ВП, ідентифікатори адрес кодів програмних блоків, призначених на ВП в межах ярусу 1; ідентифікатор ПЯ1, ідентифікатори адрес кодів програмних блоків, призначених на ПЯ1 в межах ярусу 1; ідентифікатор ПЯ2, ідентифікатори адрес кодів

програмних блоків, призначених на ПЯ2 в межах ярусу 1, і так далі, доки не будуть перераховані усі процесори чипу

- Ідентифікатор ярусу 2; ідентифікатор ВП, ідентифікатори адрес кодів програмних блоків, призначених на ВП в межах ярусу 2; ідентифікатор ПЯ1, ідентифікатори адрес кодів програмних блоків, призначених на ПЯ1 в межах ярусу 2; ідентифікатор ПЯ2, ідентифікатори адрес кодів програмних блоків, призначених на ПЯ2 в межах ярусу 2, і так далі, доки не будуть перераховані усі процесори чипу, і усі яруси на які розподілені програмні блоки.

Блок розподілу 10 на основі отриманої інформації формує для кожного ярусу вихідний пакет, узагальнена структура якого має вигляд, показаний на фіг.5, де позначені поля: ідентифікатор k-го ярусу; ідентифікатор ВП; ідентифікатори адрес кодів програмних блоків, призначених на ВП в межах k-го ярусу, ідентифікатор j-го ПЯ; ідентифікатори адрес кодів програмних блоків, призначених на j-й ПЯ в межах k-го ярусу; ідентифікатор наступного пакету (ярусу); ідентифікатор наступного фрагмента алгоритму.

Ці пакети в послідовності зростання номерів ярусів з виходів блоку розподілу 10 поступають на другі входи блоку буферної пам'яті 12, де їх запам'ятовують на регістрах блоку буферної пам'яті 12 типу FIFO (First in-First out) за наявності коду дозволу, який поступає з других виходів процесора 1 на перші входи буферної пам'яті 12. Далі ці пакети у відповідність з номерами ярусів через відповідні входи-виходи інтерфейсу з робочою системою 14 і через треті і четверті входи-виходи системи розподілу надходять на відповідні процесори і пам'ять PIM-системи, де вони виконуються. Після виконання блоків програми, які належать першому ярусу, PIM-система видає відповідний сигнал, який через треті входи-виходи системи розподілу, інтерфейс з робочою системою 14 і загальну шину 5 надходить через відповідні входи-виходи в процесор 1, який на других входах-виходах виробляє сигнал дозволу, який надходить на перші входи блока буферної пам'яті 12 для передачі через інтерфейс з робочою системою вихідного пакета для другого ярусу програмних блоків, і т. д., поки не будуть виконані в PIM- системі програмні блоки усіх ярусів.

Таким чином, запропонована система для розподілу програми користувача істотно зменшує загальний час її розподілу за рахунок:

- введення апаратно-програмних засобів, що забезпечують перекриття в часі процесу виконання безпосередньо розподілу програмних блоків поточного фрагменту і процесу підготовки інформації, необхідної для розподілу програмних блоків наступного програмного фрагмента;

- застосування у складі введених апаратно-програмних засобів блоку буферної пам'яті типу FIFO, що забезпечує можливість застосування конвеєрного поєднання процесу розподілу програмних блоків усередині

системи розподілу і процесів паралельної обробки програмних блоків, вже розподілених раніше, в робочій PIM - системі;

- за рахунок підтримки запропонованою системою нової стратегії, застосування трирівневої моделі розподілу програми користувача, цілеспрямованого початкового розподілу програми користувача з використанням критеріїв відповідності системи команд процесорів робочої системи (PIM - системи) наборам операцій фрагментів програми користувача на кожному рівні розподілу, що істотно скорочує кількість ітерацій розподілу, а також за рахунок перевірки і при необхідності коригування на кожному рівні результатів розподілу.

Запропонована система для розподілу програми користувача також розширює сферу її застосування для робочих систем, виконаних на різних апаратно-програмних платформах за рахунок використання мікропрограмного принципу формування процесу розподілу для кожного рівня, що, окрім зменшення часу розподілу, у свою чергу, знижує трудомісткість, вартість розробки безпосередньо програми розподілу в цілому за рахунок використання на усіх рівнях розподілу однотипних ядер програм.

Впровадження запропонованої системи розподілу програми користувача забезпечує також додатково позитивний ефект в частині підвищення продуктивності PIM-системи за рахунок глибокого розпаралелювання на усіх рівнях ієрархічної організації засобів обробки і зберігання інформації, перевірки і при необхідності коригування на кожному рівні розподілу рівномірності завантаження усіх процесорів корисною (обчислювальною) роботою.

ФОРМУЛА ВИНАХОДУ

Система для розподілу програми користувача, що містить процесор, оперативну пам'ять, накопичувач, інтерфейс користувача, перші входи-виходи яких підключені до загальної шини, яка відрізняється тим, що до складу системи додатково введені блок керування, блок пам'яті

мікропрограми, блок пам'яті таблиць, логічний блок, блок розподілу, блок проміжної інформації, блок буферної пам'яті, інтерфейс з хост-машиною, інтерфейс з робочою системою, при цьому входи-виходи блока мікропрограми сполучені з другими входами-виходами процесора, треті входи-виходи процесора підключені до перших входів-виходів блока керування, виходи якого

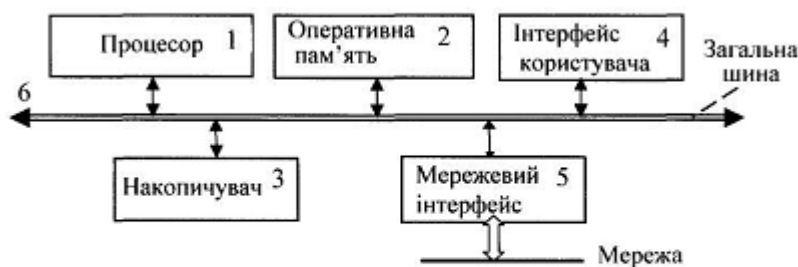
5 сполучені з відповідними входами блока пам'яті мікропрограми, а входи блока керування підключені до перших виходів інтерфейсу з хост-машиною, другі виходи якого сполучені з першими входами блока проміжної інформації, виходи блока проміжної інформації сполучені з першими входами блока пам'яті таблиць, другі входи якого сполучені з першими виходами процесора, другі виходи процесора сполучені з відповідними першими входами логічного блока,

10 а також із першими входами блока розподілу і блока буферної пам'яті, другі входи блока буферної пам'яті сполучені з відповідними виходами блока розподілу, другі входи якого сполучені з першими виходами пам'яті таблиць, другі виходи блока пам'яті таблиць сполучені з другими входами логічного блока, перші входи-виходи якого підключені до других входів-виходів оперативної пам'яті, а його другі входи-виходи підключені до відповідних входів-виходів

15 блока проміжної інформації, другі виходи блока проміжної інформації сполучені з відповідними входами оперативної пам'яті, треті входи-виходи оперативної пам'яті сполучені з другими входами-виходами блока керування, четверті входи-виходи оперативної пам'яті підключені до відповідних входів-виходів блока пам'яті таблиць, а її п'яті входи-виходи підключені до перших входів-виходів блока розподілу, треті входи блока розподілу сполучені з відповідними виходами логічного блока, треті входи-виходи якого сполучені із загальною шиною, яка підключена до

20 других входів-виходів блока розподілу і до перших входів-виходів інтерфейсу з робочою системою, входи інтерфейсу з робочою системою сполучені з першими виходами блока буферної пам'яті, другі виходи блока буферної пам'яті сполучені з відповідними входами інтерфейсу з хост-машиною, перші входи-виходи якого підключені до загальної шини, а його

25 другі і треті входи-виходи є відповідно першими і другими входами-виходами системи, а другі і треті входи-виходи інтерфейсу з робочою системою є відповідно третіми і четвертими входами-виходами системи для розподілу програми користувача.



Фіг. 1

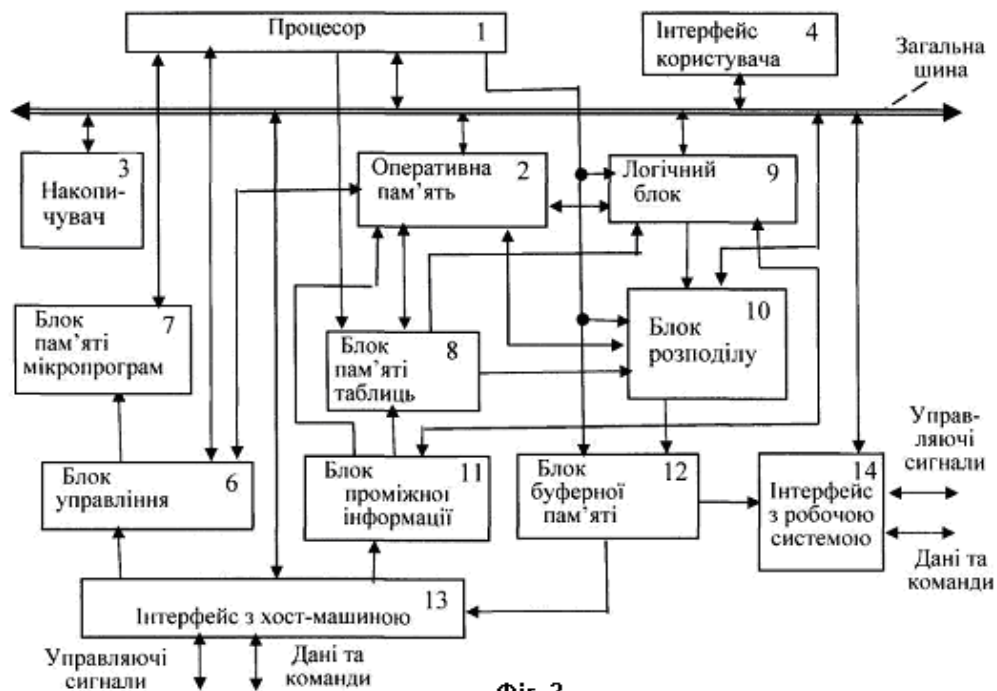


Fig. 2

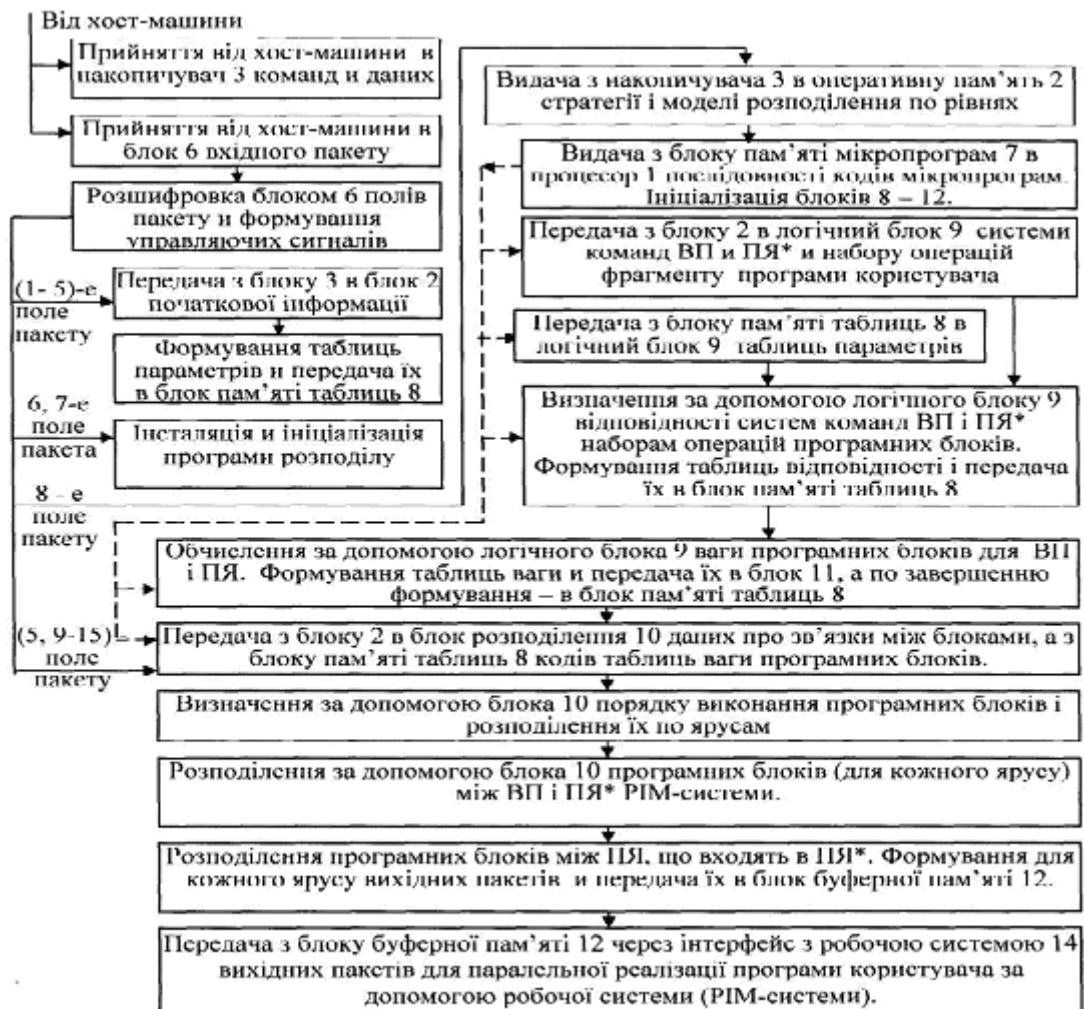


Fig. 3

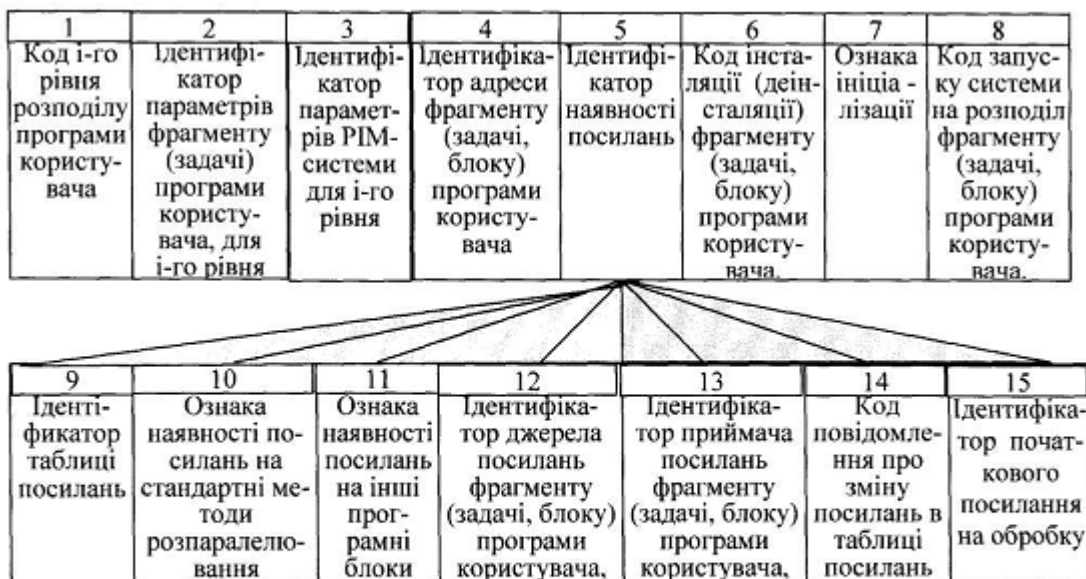


Fig. 4

Ідентифікатор k - го ярусу s - го фрагменту алгоритму	Ідентифікатор ВП	Ідентифікатори адрес програмних блоків, призначених на ВП для k - го ярусу	Ідентифікатор j - го ПЯ	Ідентифікатори адрес програмних блоків, призначених на j - й ПЯ для k - го ярусу	Ідентифікатор наступного пакету (ярусу)	Ідентифікатор наступного фрагменту алгоритму
-------------------------------------------------------	------------------	----------------------------------------------------------------------------	-------------------------	----------------------------------------------------------------------------------	-----------------------------------------	----------------------------------------------

Fig. 5