



УКРАЇНА

(19) UA

(11) 71719

(13) U

(51) МПК

G06F 9/44 (2006.01)

G06F 9/45 (2006.01)

ДЕРЖАВНА СЛУЖБА  
ІНТЕЛЕКТУАЛЬНОЇ  
ВЛАСНОСТІ  
УКРАЇНИ

## (12) ОПИС ДО ПАТЕНТУ НА КОРИСНУ МОДЕЛЬ

(21) Номер заявки:	u 2012 00054	(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:	03.01.2012	(73) Власник(и):	ІНСТИТУТ КІБЕРНЕТИКИ ІМ. В.М. ГЛУШКОВА НАН УКРАЇНИ, вул. Академіка Глушкова, 40, м. Київ, 03680 (UA)
(24) Дата, з якої є чинними права на корисну модель:	25.07.2012		
(46) Публікація відомостей про видачу патенту:	25.07.2012, Бюл.№ 14		

## (54) СПОСІБ РОЗПОДІЛУ ПРОГРАМИ КОРИСТУВАЧА ДЛЯ КОМП'ЮТЕРНОЇ СИСТЕМИ

### (57) Реферат:

Спосіб розподілу програми користувача для комп'ютерної системи включає перетворення початкових кодів програми в проміжні коди, розділення проміжних кодів на множину кодів задач, генерацію інформації про відносини серед множини задач на основі даних в задачах і перетворення кожної задачі в об'єктну програму, яку передають до множини процесорів комп'ютерної системи. Формують багаторівневу модель розподілу кодів програми, блок послідовності дій. В межах цього блока визначають можливість розділення програми користувача на задачі і формують вхідний керуючий пакет розподілу. Аналізують та структурують інформацію. Формують базу поточних значень, послідовність сигналів кодів мікропрограм, виконують розділення кодів програми. За результатами розподілу на всіх рівнях, за даними кодів вихідних пакетів формують групи кодів задач програми користувача.

UA 71719 U



Корисна модель належить до області обчислювальної техніки, зокрема - до розподілених гетерогенних комп'ютерних систем с різноманітними процесорами, наприклад, кластерних систем з прискорювачами обчислень, систем типу "Процесор - в - пам'яті" ("Processor-in-memory" або PIM-систем) та ін. Особливості архітектурно-структурної організації PIM-систем забезпечують в порівнянні з КС з класичною архітектурою при однакових рівнях технології створення елементної бази істотно менший час реалізації одних і тих самих програм користувача (на порядок і більш) при менших значеннях параметрів споживаної потужності, габаритів і ваги [1]. Реалізація програми користувача PIM-систем особливо ефективна при вирішенні складних задач, які можуть бути розпаралелені на різних рівнях ієрархічної структури засобів обробки і зберігання інформації при масовому зверненні до пам'яті. Це задачі обробки зображень, управління літальними апаратами і космічними об'єктами, завдання обробки радарних сигналів і ін. Тому далі розглядатимемо запропонований спосіб розподілу програми користувача стосовно PIM-систем, беручи до уваги, що запропонований спосіб може бути використаний для гетерогенних розподілених систем іншого типу, оскільки PIM-система, що розглядається тут, містить множину PIM-пристроїв, кожен з яких виконано на одному кристалі (чипі), на якому ведучий процесор (ВП) можна приймати за вузловий процесор одного вузла кластера, а процесорні ядра (ПЯ), які підключені до ВП і розміщені на тому ж кристалі, можна приймати за процесорні елементи цього кластера [1]. Множина таких PIM-пристроїв (чипів) утворює розподілену кластерну систему, сполучену міжчиповою комутаційною схемою. Така кластерна система, виходячи з принципів побудови PIM-систем, є гетерогенною, оскільки ВП за своїми параметрами і системою команд істотно відрізняється від ПЯ, які орієнтовано на виконання простих, але масових операцій при масовому зверненні до пам'яті.

PIM-пристрої, які сполучено за допомогою комутаційної схеми, підключені до хост-машини (перший рівень ієрархії), а кристал (чип) кожного PIM-пристрою, на якому розміщено ведучий процесор разом з пам'яттю утворюють другий рівень ієрархії. Множина ПЯ, до кожного з яких підключено банк пам'яті (БП), утворює третій рівень ієрархії. Взаємодію компонентів в таких системах виконують за принципами передачі повідомлення, основною структурною одиницею якого є керуючий пакет.

Процес розпаралелювання кодів складних програм користувача для таких систем є вельми трудомістким. Його, як правило, виконують на супер-ЕОМ, витрачаючи при цьому час і ресурси, які можуть суттєво перевищувати час і ресурси, необхідні безпосередньо для реалізації програми користувача. Це, в основному, відбувається через те, що відомі способи розподілу програм користувача по процесорах складних комп'ютерних систем, і в першу чергу PIM-систем, не враховують особливості їх архітектурно-структурної організації, наприклад, гетерогенність, і тим самим не використовують повною мірою можливості систем такого класу. Тому створення нового способу розподілу програм користувача для гетерогенних розподілених систем, який в максимальному ступеню враховує особливості їх архітектурно - структурної організації і тим самим дозволяє істотно зменшити час розподілу програм користувача та істотно поліпшити технічні характеристики системи, є актуальним завданням.

Відомі система і спосіб 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). Винахід присвячений розподілу програм користувача серед процесорів, який засновано на застосуванні спеціальної програми розподілу Veil, що включає множину стратегій розподілу. Ця програма для кожної стратегії аналізує початковий текст програми, цикли початкового тексту, розміри даних і типи, щоб підготувати ряд дистрибутивних спроб, за допомогою чого виконують розподіл програм користувача згідно кожної стратегії та оцінку результатів, щоб визначити кращу стратегію і з її допомогою виконати оптимальний розподіл програм користувача по процесорах. Недоліками такого способу є:

Висока трудомісткість і великий час реалізації розпаралелювання цільової програми (програми користувача), оскільки вибір робочої (кращої) стратегії паралелізму виконується тільки після виконання всіх наявних стратегій розпаралелювання програми користувача і оцінки їх тимчасових параметрів за наслідками виконання, зокрема - часу обчислювального процесу і часу розриву між надходженням програм (або частини програми) і відповідних даних.

Необхідність використання для кожного процесора великої ємкості пам'яті унаслідок копіювання програми користувача (або її частин) в пам'ять кожного з n процесорів, а також розміщення в пам'яті кожного процесора відповідного підпотoku даних, отриманого в результаті демультимплексування всього потоку даних.

Відомий також спосіб розподілу програми користувача для PIM-системи, який описано в [2]. Спосіб засновано на застосуванні двох'ярусної моделі PIM-системи (хост - машина ↔ ведучий процесор) і програмної системи SAGE (Statement-Analysis-Grouping-Evaluation), яка містить

набір послідовно виконуваних програм, за допомогою якого виконують аналіз програми користувача, операторне розбиття програми користувача, конструювання вагового графа WPG (Weight Partition Dependence Graph), оцінку часу виконання фрагментів програм для кожного комп'ютера, генерування фронту реалізації паралельних фрагментів програми користувача (груп), визначення плану виконання програми користувача і генерування відповідних завдань (підпрограм або пакетів) ведучим процесорам і хост-машині.

До недоліків способу відносяться:

Даний спосіб є обмеженим за функціональними можливостями, оскільки розглядає обмежену модель і відповідно процес розподілу програми користувача тільки між хост-машиною і одного ВП. Не дивлячись на обмежену модель, процес розподілу програми користувача для РІМ-системи згідно способу є дуже трудомістким і займає велику кількість часу, оскільки включає більше 20 послідовно виконуваних великих програмних блоків, кожен з яких включає велику кількість переборів і підпрограм з великою кількістю операцій.

Як основну одиницю аналізу початкової програми користувача використовують оператори в циклі (застосовано операторний метод розбиття). Через складність операторного методу розглядають тільки оператори усередині циклів, і виявляють залежності по даним тільки для деяких конструкцій програми користувача (для ідеально вкладених циклів), при цьому інші конструкції не розглядають, що також указує на обмежені функціональні можливості способу.

Крім того, такий спосіб розподілу не може бути використано для виконання розподілу програм користувача, які розроблено для реалізації з використанням інших апаратно-програмних платформ.

Відомі також інші способи розподілу програм користувача (див. патент США "System for managing distribution of programs". № 7437723, 14.10.2008). Технічні рішення, які запропоновані в патенті, за своїм призначенням повинні полегшувати управління розподілом програм за наявності залежності посилань між програмами на різних комп'ютерах, а також управління для розподілу і видалення програми користувача. Систему розподілу виконано у вигляді комплексу програм, які розміщені на двох комп'ютерах, зв'язаних між собою мережевими засобами, між якими в процесі розподілу виконуються численні пересилки складних структур даних (пакетів) з метою їх аналізу і обробки. Це істотно ускладнює і збільшує трудомісткість процесу розподілу програм користувача. Крім того, втручання менеджера (оператора, користувача) в процес розподілу програми користувача на найбільш важливих ділянках алгоритму його реалізації, також збільшує кількість ітерацій і час реалізації алгоритму розподілу для досягнення поставлених цілей, оскільки рішення можуть ухвалюватися не достатньо обґрунтовано. Так менеджер перевіряє залежні програми користувача і апаратуру, на яких розміщені ці програми, визначає пакет передачі програми користувача як предмет обробки і блок завдання як адресат видачі, визначає зміст обробки і передачу пакету програми користувача, вибирає посилення адресів, які повинні бути збережені, а також комбінацію ідентифікатора програми користувача і ідентифікатора блока для програми користувача, які будуть видалені і визначає умови видалення програми користувача ("стерти", "не стирати", "змінити"), на які система реагує відповідним чином.

Найбільш близьким до пропонованого способу по технічній суті і вирішуваному завданню є спосіб розподілу програми користувача (див. патент США "Program parallelizing apparatus capable of optimizing processing time". № 5452461 (A), 19.09.1995), якій включає послідовність наступних кроків: перетворення початкової програми в проміжні коди; ділення проміжних кодів на множину задач; генерація інформації про відносини, що показують послідовні відносини для задач серед множини задач на основі даних операндів в задачах; розповсюдження множини задач до множини рівнів, отриманих діленням повної тривалості обробки часовими точками синхронізації, що вказують часовий послідовний порядок; призначення задач, які розподілені на кожному з рівнів, на процесори в багатопроцесорній системі; визначення тривалості обробки машинних команд, складових кожної з задач; перетворення задач в об'єктні програми, які будуть оброблені множиною процесорів. При цьому за допомогою програмних засобів розпаралелювання програм згідно винаходу можливо генерувати з високою швидкістю початкову програму в об'єктну програму, здатну оброблятися паралельно множиною процесорів, складових багатопроцесорної системи, яка включає механізм комунікації для міжпроцесорної комунікації і механізм синхронізації, щоб обробка виконувалася паралельно серед процесорів через координацію в багатопроцесорній системі, і призначення об'єктних програм на процесори.

Запропоновані засоби планування задачі наділені функцією генерувати інформацію зв'язків задачі для скріплення в одну задачу тих задач, які можуть зменшити після їх об'єднання кількість часу координації або очікування обробки, яка повинна бути виконана засобами

синхронізації і, отже, часу, потрібного для того, щоб виконати паралельну обробку в багатопроцесорній системі.

До недоліків найближчого способу відносяться:

Початкове планування задач для паралельної обробки вимагає розрахунку часу реалізації кожної задачі, а для кожної ітерації низхідному і висхідному переміщенням по рівнях задач при їх величезній кількості визначається загальний час реалізації всього алгоритму. Це призводить до великих витрат обчислювальних ресурсів і ресурсів пам'яті тим паче, що при цьому повинні формуватися і розміщуватися в пам'яті множина таблиць, наприклад, реєстраційна таблиця завдань, таблиця міжзадачних послідовних відносин і ін. Крім того, при переміщенні задач виникає складна проблема синхронізації процесорів як усередині кожної групи процесорів, так і між такими групами, що також ускладнює управління всім процесом розпаралелювання.

Апарат розподілення стосовно третьої реалізації винаходу має велику кількість компіляторів різного типу: керований компілятор проблемно-орієнтованої мови, паралельний компілятор, два компілятори операційно-орієнтованої мови, мовний компілятор, векторний компілятор, компілятор з розподіленими функціями. Тим самим спосіб і алгоритм розпаралелювання є проблемно-орієнтованими, і для розпаралелювання конкретних початкових програм потрібна розробка і використання конкретного типу компілятора, що збільшує вартість розробки і скорочує область застосування пропонованих методів розпаралелювання.

Спосіб розподілення програми користувача стосовно четвертої реалізації винаходу орієнтовано на використання спільно з розподіленою комп'ютерною системою, яка відрізняється від класичного варіанту архітектурно-структурної організації наявністю розширеного каналу зв'язку з процесорами разом зі схемами управління шинами і для кожного процесору - наявністю механізму паралельної обробки. Тим самим можна зробити висновок, що для інших комп'ютерних систем, зокрема для кластерних або PIM-систем цей метод є мало ефективним.

В основу корисної моделі, яка пропонується, покладено технічна задача створення способу розподілу кодів програми користувача для паралельного виконання на розподіленій гетерогенній комп'ютерній системі, у тому числі - PIM-системі, який дозволить зменшити загальний час розпаралелювання за рахунок використання запропонованої багаторівневої моделі розподілу, цілеспрямованого початкового розподілу кодів програми користувача з використанням критеріїв відповідності системи команд процесорів комп'ютерної системи (PIM-системи) наборам операцій програми користувача на кожному рівні розподілу, що істотно скорочує кількість ітерацій розподілу, а також за рахунок перевірки і при необхідності коректування на кожному рівні результатів розподілу. При цьому знижується трудомісткість, вартість розробки і використання програми розподілу в цілому за рахунок використання на всіх рівнях однотипних програм розділення програми користувача на складові її компоненти (коди задач, програмних частин, програмних модулів) і розподілу їх по процесорах КС (PIM-системи). При цьому розширюється область запропонованого способу розподілу кодів програми користувача на різні апаратно-програмні платформи за рахунок використання мікропрограмного принципу формування процесу розподілу для кожного рівня.

Впровадження запропонованого способу розподілу кодів програми користувача забезпечує також додатково позитивний ефект в частині підвищення продуктивності КС за рахунок розпаралелювання на всіх рівнях обробки інформації, перевірки і, при необхідності, коректування на кожному рівні розподілу рівномірності завантаження всіх процесорів корисною (обчислювальною) роботою.

Позитивний ефект запропонованого способу розподілу програми користувач досягається тим, що у способі максимально враховують наступні особливості архітектурно-структурної організації цього класу машин:

По-перше: багаторівневий спосіб розподілу кодів програми користувача відповідає багаторівневої організації засобів обробки і зберігання інформації в PIM-системі: процесор хост-машини ↔ ВП ↔ ПЯ; пам'ять хост-машини ↔ пам'ять ВП ↔ пам'ять ПЯ.

По-друге: приймається до уваги гетерогенність системи, на який має бути виконано програму користувача, для цього при розподілі програми використовують нові критерії розподілу - відповідність на кожному рівні систем команд процесорів PIM-системи наборам операцій програми (стосовно рівня - задачі, модуля) користувача.

По-третє: приймається до уваги однотипність PIM-пристроїв (чипів), які складають PIM-систему, та однотипність множини ПЯ в межах кожного чипу. Це дозволяє спростити моделі розподілу на кожному рівні таким чином:

PIM-система містить однакові PIM-пристрої, кожен з яких виконано на одному кристалі (чипі) і сполучено з іншими чипами за допомогою міжчипової комутаційної схеми. Проте, разом з хост-машиною сукупність таких PIM-пристроїв, складових PIM-системи, утворює гетерогенну

систему, оскільки функціональні можливості і системи команд хост-машини і РІМ-системи істотно відрізняються. Тому формують модель першого рівня розподілу програми користувача у вигляді: процесор хост-машини  $\leftrightarrow$  еквівалентний ведучий процесор (ВП<sub>екв</sub>). При цьому як систему команд ВП<sub>екв</sub> приймають систему команд одного РІМ-пристрою (бо всі вони однакові),

5 ВП<sub>екв</sub> приймають рівній сумі ємкостей всіх РІМ-пристроїв, а за тривалість такту роботи ВП<sub>екв</sub> приймають тривалість такту одного ВП, поділену на кількість ВП у складі РІМ-системи, враховуючи можливість їх одночасної роботи. При цьому як критерій розділення кодів програми користувача на коди задач використовують критерій відповідності системи команд хост-машини та ВП<sub>екв</sub> наборам операцій відповідних задач для кожного РІМ-пристрою.

10 Кожен РІМ-пристрій, виконаний на одному кристалі (чипі), є гетерогенним, оскільки містить ВП разом з пам'яттю і множину ПЯ, кожен з яких підключений до свого банку пам'яті і сполучений з ВП. При цьому, згідно принципам організації РІМ-пристрою, як ВП використовують процесор з розширеними в порівнянні з ПЯ функціональними можливостями і розширеною системою команд, яку орієнтовано на виконання складних операцій. Як ПЯ використовують

15 спрощений процесор зі скороченою системою команд, яку орієнтовано на виконання простих, але масових операцій типу додавання, віднімання, зсуву і ін. при масовому зверненні до пам'яті за операндами. Тому формують модель другого рівня розподілу програми користувача у вигляді: ВП  $\leftrightarrow$  еквівалентне процесорне ядро (ПЯ<sub>екв</sub>). При цьому за систему команд ПЯ<sub>екв</sub> приймають систему команд одного ПЯ (оскільки всі ПЯ на кристалі - однакові), ємкість пам'яті

20 приймають рівній сумі ємкостей всіх банків пам'яті одного чипу, а за тривалість такту роботи ПЯ<sub>екв</sub> приймають тривалість такту одного ПЯ, поділену на кількість ПЯ у складі РІМ-пристрою (чипу), враховуючи можливість їх одночасної роботи усередині одного кристалу. При цьому враховуються обмеження на скорочений набір команд ПЯ і ємкість кожного банку пам'яті, підключеного до ПЯ на цьому ж кристалі. Таке положення пояснюється тенденцією розмістити

25 на одному кристалі з ВП і з пам'яттю максимально можливу кількість ПЯ для найбільшого розпаралелювання програмних частин програми користувача.

Модель розподілу програми користувача на третьому (кінцевому) рівні стосовно РІМ-системи формують у вигляді: ПЯ<sub>екв</sub>  $\leftrightarrow$  множина ПЯ зі своїми власними параметрами.

При розподілі програми користувача стосовно запропонованого способу спочатку формують

30 багаторівневу модель розподілу кодів програми користувача, потім формують блок послідовності дій (БПД.), який використовують для кожного і-го поточного рівня розподілу, в межах цього блока визначають можливість розділення програми користувача на незалежні за даними задачі (фіг. 1). Потім для і-го рівня розподілу формують вхідний керуючий пакет розподілу (фіг. 2), аналізують та структурують інформацію про параметри комп'ютерної системи

35 (табл. 1), та про параметри програми користувача та її частин (табл. 2), на основі цієї інформації формують бази поточних значень, які запам'ятовують в електронній пам'яті, потім формують послідовність сигналів кодів мікропрограм для розподілу на поточному рівні, виконують розділення кодів програми користувача або її задач на незалежні за даними коди частин і розподіляють їх між процесорами КС стосовно сформованої моделі розподілу для і-то поточного рівня, використовуючи критерій відповідності системи команд цих процесорів

40 наборам операцій програми користувача обраного рівня розподілу. Далі після визначення тривалості обробки машинних команд кожної із задач перевіряють баланс завантаження процесорів КС кодами програми користувача, які присутні в моделі розподілу поточного і-го рівня, і за наявності балансу генерують для поточного рівня коди керуючого вихідного пакету,

45 якій запам'ятовують в електронній пам'яті. При відсутності балансу коректують розподіл шляхом укрупнення або подрібнення отриманих задач або їх частин. Далі на кінцевому рівні розподілу кожен частину програми, яку призначено на попередньому рівні, розділяють на програмні модулі, які розподіляють поміж усіма процесорами, що входять в набір моделі розподілу кінцевого рівня, при цьому виконують послідовність дій стосовно БПД, за виключенням того, що використовують модель розподілу, а також формують послідовність кодів мікропрограм, коди полів керуючого вхідного та вихідного пакетів для кінцевого рівня розподілу. При цьому як критерій розділення кодів програмних блоків на коди модулів програми використовують час виконання на кожному процесорі, які присутні в моделі розподілу цього рівня, і зв'язки за даними. Далі за результатами розподілу на всіх рівнях, за даними кодів вихідних пакетів

50 формують групи кодів задач програми користувача, які не мають зв'язків між собою за даними, при цьому послідовність виконання кодів груп визначають на підставі зв'язків між задачами в різних групах згідно порядку зв'язків даних програми користувача, аналізують всі отримані задачі, щоб визначити наявність таких груп, після цього перетворюють ці програмні групи в об'єктні програми, які передають до відповідних процесорів комп'ютерної системи, а також до

60 хост-машини, формують та передають до комп'ютерної системи та хост-машини коди вихідних

керуючих пакетів, аналогічних вихідним пакетам для і-го рівня розподілу, з відповідними індексами і ознаками та їх значеннями для груп, що визначають розділені по процесорах вихідні команди, дані та керуючу інформацію, які необхідні для паралельного виконання кодів програми користувача на комп'ютерній системі та на хост-машині.

5 При цьому на кожному поточному рівні формують таблицю кодів параметрів КС, до якої включають тип і кількість процесорів, системи команд процесорів, тривалість виконання кожної команди відповідним процесором (табл. 1), а також формують таблицю кодів параметрів програми користувача, та її частин, до якої включають типи циклів та їх параметри, тип і кількість операцій кожного типу (табл. 2).

10 У вхідний керуючий пакет (фіг. 2), якій формують до кожного рівня розподілу, включають коди полів, що містять код і-го рівня розподілу програми користувача, а також стосовно рівня розподілу - код ідентифікатора параметрів програми користувача або її задачі; частини, модуля, код ідентифікатора параметрів комп'ютерної системи; код ідентифікатора адреси програми користувача або її задачі; частини, модуля; код запуску системи на обробку програми користувача, або її задачі, частини, модуля; код інсталяції (деінсталяції) програми користувача або її задачі, частини, модуля; код ознаки ініціалізації програми користувача або її задачі, частини, модуля; код ідентифікатора наявності посилань, якій включає: код ідентифікатора таблиці посилань; код ознаки наявності посилань на стандартні методи розпаралелювання; код ознаки наявності посилань на інші програмні блоки; код ідентифікатора джерела посилань програми користувача або її задачі, частини, модуля; код ідентифікатора приймача посилань програми користувача або її задачі, частини, модуля; код повідомлення про зміну посилань в таблиці посилань; код ідентифікатора початкового посилання на обробку.

20 У вихідні пакети (Фіг. 3), які формують після кожного рівня розподілу на і-ому рівні програми користувача, включають коди полів, що містять код рівня розподілу програми користувача, а також стосовно рівня розподілу - код ідентифікатора адресів задач програми користувача, або їх частин та модулів після розділення; код ідентифікатора апаратних блоків, до яких направлено задачі програми користувача або їх частини та модулі; код інсталяції (деінсталяції) задач програми користувача або їх частин та модулів; код типа обробки розподіленої програми користувача; код ідентифікатора адресів груп з паралельними задачами або частинами, або модулями програми користувача; код ідентифікатора посилань між групами програми користувача; код ідентифікатора адресів груп джерел посилань; код ідентифікатора адресів груп, які приймають посилання; код ознаки повідомлення від комп'ютерної системи про виконання групи; код ознаки повідомлення про зміну зв'язків між групами; код ознаки відповіді комп'ютерної системи про виконання обробки; код ідентифікатора адреси наступного вихідного пакету; вільне поле.

35 На фіг. 1 наведено узагальнену блок-схему багаторівневої стратегії розподілу кодів програми користувача.

На фіг. 2 наведено узагальнену структуру вхідного керуючого пакету

40 На фіг. 3 наведено узагальнену структуру вихідного пакету, що відображає результати багаторівневої стратегії розподілу.

На фіг. 4 наведено блок-схему пристрою для здійснення запропонованого способу розподілу кодів програми користувача.

Блок-схема пристрою розподілу 1 для здійснення запропонованого способу розподілу кодів програми користувача містить (фіг. 4):

45 2 - процесор, призначений для обробки інформації, необхідної для процедур розділення кодів програм користувача і розподілу кодів отриманих задач і їх частин між процесорами PIM-системи, а також для формування проміжних і кінцевих таблиць, відповідних запитів до блоків постійної пам'яті 3, оперативної пам'яті 4 та дискового накопичувача 5. Функціонування процесора на різних рівнях розподілу програми користувача визначається відповідною кожному рівню моделлю стратегії, що представлено на вході процесора у вигляді сформованої послідовності кодів відповідних мікрокоманд, які поступають з виходів блока 3. Основні функції і процедури процесора представлені в табл. 3.

3 - блок постійної пам'яті (БПП) для зберігання кодів мікропрограм для розподілу кодів програм користувача на різних рівнях.

55 4 - блок оперативної пам'яті (БОП), призначений для тимчасового зберігання інформації, отриманої в результаті виконання процесором арифметичних і логічних операцій, зокрема - для тимчасового зберігання сформованих таблиць різного призначення.

60 5 - дисковий накопичувач (ДН) для зберігання програми, за допомогою якої формують моделі розподілу кодів програми користувача кожного рівня, бази даних, призначеної для зберігання кодів параметрів PIM-системи згідно табл.1, кодів параметрів програми користувача

згідно табл. 2, кодів даних і команд, що поступають від хост-машини, інформації про зв'язність компонентів програми користувача за даними, кодів вхідних керуючих пакетів та вихідних пакетів, що відображають результати розподілення програми користувача на різних рівнях, а також іншої інформації, необхідної для роботи системи розподілення кодів програм користувача.

6 - загальна шина типу PCI або PCI-Express, за допомогою якої зв'язані між собою основні вузли і блоки пристрою.

7 - блок інтерфейсу для зв'язку пристрою розподілу з PIM-системою видає в PIM-систему інформацію, що відображає результати розподілу кодів між процесорами PIM-системи, а також приймає від PIM-системи службову інформацію, зокрема - про закінчення процесів обробки інформації кодів розподілених задач, частин задач і модулів програми користувача процесорами.

8 - блок інтерфейсу пристрою розподілу для зв'язку з хост-машиною реалізує зв'язок пристрою розподілу кодів програми користувача з хост-машиною, приймаючи коди даних, команд і керуючого пакету від хост-машини і направляючи в хост-машину сформовані для хост-машини коди пакетів інформації, що відображають результати розподілу програми користувача між хост-машиною і PIM - пристроями.

9 і 10 - монітор та клавіатура для відображення та введення-виводу різноманітної інформації при взаємодії користувача з пристроєм розподілу 1.

При цьому виходи постійного блока пам'яті 3 кодів мікропрограм сполучені з входами процесора 2, входи-виходи якого підключені до загальної шини 6. Входи-виходи блока оперативної пам'яті 4, дискового накопичувача 5, перші входи-виходи інтерфейсу з PIM-системою 7, перші входи-виходи інтерфейсу з хост-машиною 8, входи-виходи монітора 9 і входи-виходи клавіатури 10 також підключені до загальної шини 6. При цьому другі входи-виходи блока інтерфейсу з PIM-системою 7 підключені до відповідних перших входів-виходів пристрою розподілу програми користувача 1, другі входи-виходи якого сполучені з другими входами-виходами блока інтерфейсу з хост-машиною.

Спосіб розподілу програми користувача для паралельного виконання на PIM-системі здійснюють за допомогою описаного пристрою 1 таким чином.

Перед запуском процесу розподілу кодів програми користувача за допомогою засобів обчислювальної техніки, наприклад, за допомогою хост-машини, готують початкову інформацію. Для цього будують модель стратегії розподілу кодів програми користувача, яку представляють у вигляді послідовності кодів для кожного рівня розподілу та запам'ятовують в блоці 5 пристрою 1. При цьому на першому рівні модель стратегії розподілу програми користувача виглядає так: процесор хост-машини  $\leftrightarrow$  еквівалентний ведучий процесор (ВП\*) PIM-системи". Параметри ВП\* формують, як наведено вище, на основі параметрів одного ВП, розміщеного на кристалі PIM-пристрою, і кількості PIM-пристроїв N, які входять до складу PIM-системи. Таке представлення моделі є допустимим, оскільки всі PIM-пристрої PIM - системи є однаковими, а хост-машина безпосередньо зв'язана з ВП кожного PIM-пристрою. Блок-схему стратегії розподілу наведено на фіг.1.

Також за допомогою процесору 2 аналізують та структурують параметри PIM - системи, параметри програми користувача, які для аналізу передають з блока 5 через загальну шину 6 в блок 4 та формують з цієї інформації відповідні набори параметрів, які запам'ятовують в дисковому накопичувачу 5. Перелік основних параметрів цих наборів наведено в табл. 1 і табл. 2. Крім того, за допомогою процесору 2, якій використовує специфічну програму з блока 5, формують керуючий пакет спочатку для першого рівня розподілу. Узагальнену структуру такого пакету для трьох рівнів розподілу стосовно PIM-системи наведено на фіг. 2. Керуючий пакет розміщують в БОП 4, а решту всієї початкової інформації розміщують в дисковому накопичувачу 5 (фіг. 4) на додаток до іншої, розміщеної там інформації, серед якої доцільно виділити: безпосередньо програму користувача, яку необхідно розпаралелювати, інформацію про зв'язність компонентів програми користувача за даними, програми, що відображають моделі стратегії розподілу програми користувача на кожному з трьох рівнів розподілу, програму для лексичного аналізу програми користувача і її фрагментів та ін.

Програму моделі розподілу першого рівня представляють у вигляді кодів програмного блока стратегії розподілу програми користувача, який розміщують в пам'яті дискового накопичувача 5, при цьому набори кодів всіх параметрів, включаючи параметри зв'язності по даним, представляють в базі даних, яку також розміщують в пам'яті дискового накопичувача 5 пристрою розподілу 1.

При розділенні кодів програми користувача у відповідності з моделлю першого рівня перевага віддається розділенню на крупні фрагменти - задачі, де кожна задача призначена для



самостійної реалізації на окремому PIM-пристрої. Розділення роблять так, щоб взаємозв'язок по даним між цими задачами здійснювався, в основному, підсумковими значеннями результатів їх реалізації.

Процес розподілу кодів програми користувача на першому рівні на незалежні за даними задачі починають з того, що формують блок послідовності дій (БПДі), який використовують для всіх рівнів розподілу. У межах цього блока спочатку перевіряють можливості розподілу, що здійснюють за допомогою процесору 2 і відповідної програми, яка перед початком розподілу розміщена в дисковому накопичувачі 5. Для цього виконують аналіз програми користувача, в результаті якого може бути виявлено два варіанти реалізації процедур розпаралелювання (фіг. 1):

1. Коди програми користувача не підлягають подальшому розділенню на коди задач або їх частини, тобто відсутні цикли, стандартні функції, які розпаралелюють стандартними методами, відсутні послідовні ділянки програми, що реалізують набір послідовних операцій, які можна представити у вигляді функцій, що підлягають при обчисленні розпаралелюванню і інші відомі ознаки.

2. Коди програми користувача може бути розділено на коди програмних задач згідно вказаним в п.1 і іншим відомим ознакам і може бути розпаралелено.

У другому випадку процес розподілу кодів програми користувача починають з розшифровки за допомогою процесору 2 коду першого поля керуючого пакету, розміщеного в блоці 4, який вказує, що коди параметрів і коди ідентифікаторів параметрів всіх полів пакету відносяться до першого рівня розподілу програми користувача. Керуючі сигнали, які отримано після розшифровки цього коду, через загальну шину 6 і відповідні входи-виходи блока 3 безпосередньо направляють в блок 3, який видає коди мікрокоманд відповідно для першого рівню розподілу. Одночасно з отриманням послідовності кодів мікропрограм, процесор 2 через його входи-виходи і загальну шину 6 виставляє запити до бази даних, яку розміщено на дисковому накопичувачу 5, з метою отримання кодів параметрів PIM-системи, параметрів програми користувача, що відносяться тільки до першого рівня розподілу, а також інформації про зв'язність за даними. Ці параметри зчитують з бази даних на основі значень кодів ідентифікаторів другого і третього полів керуючого пакету (фіг.2), які розшифровані за допомогою процесору 2. Отримані в результаті сигнали використовують для структуризації, формування з них відповідних таблиць, які запам'ятовують в блоці 4, вхід-вихід пристрою 4 передають для тимчасового зберігання в оперативний запам'ятовувачий пристрій 4. Далі за допомогою процесору 2 розшифровують код шостого поля керуючого пакету і за наявності коду інсталяції виконують інсталяцію програм шляхом розміщення всіх необхідних програмних файлів у відповідних місцях пам'яті програм користувача пристрою розподілу 1.

Крім того, блок процесор 2 аналізує код сьомого поля керуючого пакету і за наявності коду ознаки ініціалізації привласнюють початкові значення змінним. При цьому процесор 2 формує запит в пристрій 5 для читання з відповідних таблиць і формування списку операцій програми користувача і списку команд PIM-системи для першого рівня розподілу. Цю інформацію представляють у вигляді файлів: код ідентифікатора програми користувача, список операцій програми користувача; код ідентифікатора процесора хост-машини, список кодів команд хост-машини, код ідентифікатора процесора ВП\* PIM-пристрою, список команд ВП\*, який рівний списку команд одного ВП. Ці файли записують в блок 4 по ланцюгу: входи-виходи блока 5, загальна шина 6, відповідні входи-виходи блока 4. Формують та по цьому ж ланцюгу записують в блок 4 файл, який містить інформацію про кількість ітерацій для кожного циклу, що входить в програму користувача, а також файл з описом залежності за даними. При формуванні файлу залежності за даними використовують значення кодів полів посилань 8-14 керуючого пакету (фіг. 2), які аналізують за допомогою блока 2, і отриману при цьому інформацію передають з перших входів-виходів блока 2 через загальну шину 6 на відповідні входи-виходи блока 4, для зберігання на час виконання процесу розподілу кодів програми користувача.

Далі за допомогою процесора 2 у відповідності з послідовністю кодів мікрокоманд для першого рівня розподілу, яка поступає з виходів блока 3 на відповідні входи процесора 2, виконують розподіл задач програми користувача між хост-машиною і еквівалентним ВП\* PIM-системи. Для цього за допомогою процесору 2 та програми, яка розміщена в блоці 5, виконують лексичний аналіз тексту програми користувача, який використовують для подальшого визначення можливості виконання задач програми користувача за допомогою хост-машини і за допомогою PIM-системи, а також виконують для них по відповідним формулам розрахунки ваги (часу реалізації) задач програми користувача. В процесі аналізу виділяють операції задач програми користувача і записують в пристрій 4 в таблицю операцій.

На підставі таблиці операцій задачі програми користувача обчислюють вагу задачі для процесора хост-машини і ВП\* за формулою:

$$W_{\text{хост}} = \sum_{i=1}^n p_i \times q_i, \quad (1)$$

де  $n$  - кількість записів в таблиці операцій задачі програми користувача,  $p_i$  - час виконання процесором хост-машини  $i$ -ї операції,  $q_i$  - скільки разів  $i$ -а операція може бути виконана в даній задачі програми користувача.

Вагу задачі для еквівалентного процесора ВП\* РІМ-системи обчислюють за аналогічною формулою:

$$W_{\text{ВП}^*} = \sum_{j=1}^m \delta_j \times \lambda_j, \quad (2)$$

де  $m$  - кількість записів в таблиці операцій задачі програми користувача,  $\delta_j$  - час виконання команди за допомогою ВП\* РІМ-системи, яка відповідає  $j$ -ї операції,  $\lambda_j$  - скільки разів  $j$ -а операція може бути виконана в даній задачі програми користувача. Результати обчислень запам'ятовують в блоці пам'яті 4. Таку процедуру визначення часу виконання задачі виконують аналітично для всіх задач програми користувача, які були виділені для хост-машини і окремо для ВП\* РІМ - системи.

Далі розподіляють набір кодів задач між хост-машиною і ВП\*. При цьому набір кодів задач, отриманий для ВП\*, розподіляють між однаковими ВП усіх РІМ-пристроїв одним з відомих методів, наприклад, методом пошуку незалежних задач, і перевіряють баланс завантаження (час виконання кожної задачі) для процесора хост-машини і ВП\* РІМ-системи. Якщо є баланс завантаження, то переходять на другий рівень розподілу програми користувача, при цьому записують в блок 5 результати розподілу на першому рівні, при цьому формують набір кодів задач програми користувача для хост-машини і відповідний набір кодів задач для ВП кожного РІМ-пристрою РІМ-системи з відповідними кодами атрибутів (ідентифікаторів), які вказують на їх приналежність до апаратних засобів. Всю цю інформацію також представляють у вигляді кодів вихідного пакету за наслідками розподілу на першому рівні. Узагальнену (для трьох рівнів) структуру вихідного пакету для РІМ-системи наведено на фіг. 3.

Якщо баланс завантаження процесора хост-машини і ВП\* РІМ-системи відсутній, то виконують коректування розподілу початкових задач методом їх укрупнення або розділення на складові частини, і процес розподілу задач програми користувача повторюють за тим же сценарієм.

На другому рівні виконують розподіл виділених на першому рівні кодів задач для кожного РІМ-пристрою РІМ-системи. Оскільки всі РІМ-пристрої РІМ-системи однакові, то можна вважати, що процеси розподілу для всіх РІМ-пристроїв виконують за одним і тим самим сценарієм.

Для реалізації розподілу на другому рівні, на якому кожну задачу програми, призначену на другому рівні для кожного РІМ-пристрою, розділяють на програмні частини, виконують ту ж саму послідовність дій, як і для першого рівня, використовуючи БПД і за наступним виключенням.

Процес розподілу також починають з перевірки можливості розподілу, але кодів задач програми користувача на незалежні за даними програмні частини (фіг. 1). І якщо така можливість є, використовують послідовність кодів мікропрограм для другого рівня розподілу, коди полів керуючого вхідного та вихідного пакетів заміняють на відповідні коди полів для другого рівня згідно фіг. 2, фіг. 3. Формують основні параметри еквівалентного процесорного ядра (ПЯ\*) для одного РІМ-пристрою та модель "один ВП - еквівалентний ПЯ\*". При цьому як критерій розділення кодів задач на коди програмних частин використовують критерій відповідності системи команд ВП і еквівалентного процесорного ядра ПЯ\* набором операцій відповідних програмних частин для кожного РІМ-пристрою. Оскільки всі ПЯ в наборі ПЯ\* РІМ-пристрою однакові, то за систему команд ПЯ\* приймають систему команд одного ПЯ, ємкість пам'яті ПЯ\* приймають рівною сумарній ємкості пам'яті, підключеної до всіх ПЯ РІМ-пристрою, а як час виконання будь-якої команди ПЯ\* приймають час виконання цієї команди одним ПЯ, поділений на кількість ПЯ в наборі ПЯ\*, враховуючи одночасну їх роботу.

Керуючий пакет розміщують в пам'яті 4, а решта всієї початкової інформації вже знаходиться в блоці оперативної пам'яті 4 і дисковій пам'яті 5 (фіг.4.) пристрою розподілу 1, зокрема: у пристрої 4 знаходяться коди задач програми користувача і коди їх параметрів, розподілені на першому рівні; на дисковому накопичувачі 5 знаходиться програма, що відображає модель стратегії розподілу програми користувача на другому рівні розподілу, у базі даних дискового накопичувача 5 розміщені коди наборів параметрів програми користувача і РІМ-пристроїв РІМ-системи, параметри зв'язності задач за даними та інша інформація, яка необхідна для розподілу на цьому рівні.

Далі послідовність процесу розподілу кодів на третьому (для PIM-системи на кінцевому) рівні аналогічна послідовності процесу розподілу кодів програми користувача на першому рівні, однак при цьому кожен частину програми, яка призначена на попередньому рівні, розділяють на програмні модулі, які розподіляють між всіма процесорами, що входять в набір моделі розподілу кінцевого рівня. При цьому також використовують блок послідовності дій БПДі, в межах якого виконують послідовність дій стосовно послідовності дій цього блока за виключенням того, що використовують модель розподілу, а також формують послідовність кодів мікропрограм, коди полів керуючого вхідного та вихідного пакетів для кінцевого рівня розподілу. Як критерій розділення кодів програмних частин на коди модулів програми використовують час виконання модулів на кожному процесорі, які присутні в моделі розподілу цього рівня, і зв'язки за даними. Час виконання обчислюють за допомогою процесору 2 за виразом (2) з відповідними індексами та їх значенням для кінцевого рівня розподілу. Результат записують в блок пам'яті 4 за ланцюгом: вхід-вихід блока 2, загальна шина 6 - вхід-вихід блока 4.

За результатами розподілу на всіх рівнях, за даними вихідних керуючих пакетів, які на даний час розміщені в блоці 4, за допомогою процесора 2 та з використанням програми в дисковому накопичувачу 5, формують групи задач, програми користувача, які не мають зв'язків між собою за даними стосовно інформації про зв'язки, яка розміщена в блоці 5. Перебирають всі отримані задачі, щоб визначити наявність таких груп. При цьому послідовність виконання груп визначають на підставі зв'язків між задачами в різних групах згідно порядку зв'язків даних програми користувача. Після цього, за допомогою процесору 2 з використанням спеціальної програми, яка розміщена в блоці 5, перетворюють ці програмні групи в об'єктні програми, які через інтерфейс 7 передають через перший вхід-вихід пристрою розподілу 1 до стосовних процесорів PIM-системи, а також через інтерфейс 8 і другий вхід-вихід пристрою розподілу 1 до хост-машини. Одночасно формують вихідні керуючі пакети з відповідними ознаками для груп, аналогічно пакету на фіг. 3, які запам'ятовують в блоці 5 та через треті та четверті входи-виходи пристрою розподілу 1 з третіх входів-виходів інтерфейсів 7 і 8 передають відповідно до PIM-системи та хост-машини, що визначає розділені по процесорах вихідні команди, дані та керуючу інформацію, які необхідні для паралельного виконання кодів програми користувача на комп'ютерній системі та на хост-машині.

Таким чином, запропонований спосіб розподілу програми користувача для паралельного виконання на комп'ютерній системі забезпечує широкі функціональні можливості розподілу кодів програм користувача і зменшує загальний час її розпаралелювання за рахунок використання запропонованої багаторівневої моделі розподілу, яка відображає багаторівневу архітектурно-структурну організацію комп'ютерної системи, цілеспрямованого початкового розподілу кодів програми користувача з використанням критеріїв відповідності системи команд процесорів PIM-системи набором операцій програми користувача на кожному рівні розподілу, що істотно скорочує кількість ітерацій розподілу, а також за рахунок перевірки і при необхідності коректування на кожному рівні результатів розподілу.

Крім того, запропонований спосіб розподілу кодів програм користувача знижує трудомісткість, вартість розробки програми розподілу в цілому за рахунок використання на всіх рівнях розподілу однотипного блока послідовності дій (БПДі) при розділенні програми користувача на складові компоненти (задачі, частини задач, програмні модулі) і розподілі їх по процесорах (див. фіг. 1) а також розширює область застосування запропонованого способу і пристрою розподілу на різні апаратно-програмні платформи за рахунок використання мікропрограмного принципу формування процесу розподілу для кожного рівня.

Впровадження запропонованого способу і системи розподілу програми користувача забезпечує також додатково позитивний ефект в частині підвищення продуктивності комп'ютерної системи за рахунок глибокого розпаралелювання на всіх рівнях ієрархічної організації засобів обробки і зберігання інформації, перевірки і при необхідності коректування на кожному рівні розподілу рівномірності завантаження всіх процесорів корисною (обчислювальною) роботою.

Джерела інформації:

1. Яковлев Ю.С. Однокристалльные компьютерные системы высокой производительности. Особенности архитектурно-структурной организации и внутренних процессов: монография / Ю.С. Яковлев. - Винница: ВНТУ, 2009.-294 с.

2. 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](http://dSPACE.lib.fcu.edu.tw/bitstream/2377/564/1/OT_1022003305.pdf). -Дата доступу: 17.08.11

Таблиця 1

	Найменування параметра
1	Кількість кристалів РІМ-системи
2	Склад і типи процесорів РІМ-системи на одному і-му кристалі
3	Набір системи команд ведучого процесора (ВП)
4	Розрядність ВП
5	Ємкість пам'яті ВП
6	Тривалість одного такту роботи ВП
7	Набір параметрів еквівалентного ВП*
8	Кількість ПЯ на одному кристалі
9	Набір системи команд ПЯ
10	Розрядність ПЯ
11	Ємкість банку пам'яті, підключеного до кожного ПЯ у складі кристала
12	Тривалість одного такту роботи ПЯ
13	Набір параметрів еквівалентного ПЯ*
--	-----
Х	Швидкість передачі інформації по каналу процесор - пам'ять (ширина смуги пропускання інформації по даному каналу)

Таблиця 2

	Найменування параметра
	Типи і кількість циклів програми користувача і їх параметри
	Типи і кількість незалежних за даними завдань і програм
	Типи і кількість послідовних ділянок програм
	Інформація про зв'язність усередині програми користувача за даними
	Типи і кількість операцій програми користувача
	Частота зустрічі кожної j-й операції програми користувача
	Розрядність обробки даних
----	-----
g	Програми реалізації відомих методів розпаралелювання при вирішенні типових завдань (множення матриць і тому подібне)

Таблиця 3

№ п/п	Найменування функцій і процедур
1	Формування параметрів ВП* і ПЯ*
2	Розділення кодів програми користувача на паралельні фрагменти і розподіл їх між хост-машиною і ВП*
3	Розділення і розподіл кодів фрагментів програми користувача між ВП* і набором кристалів РІМ-системи (ВП).
4	Розділення кодів фрагмента програми користувача кожного кристала на програмні блоки і розподіл їх між ВП і ПЯ*
5	Розділення кодів кожного програмного блока на програмні модулі і розподіл їх між ПЯ* і набором ПЯ кожного кристала
6	Розрахунок параметричної ваги (часу виконання) фрагментів, програмних блоків і модулів програми користувача для відповідних процесорів РІМ - системи, хост-машини і рівнів розподілу.
7	Перевірка балансу завантаження між хост-машиною і РІМ - системою (ВП*), а також між ВП і ПЯ*, і між кожним ПЯ.
8	Перерозподіл кодів програми користувача за відсутності балансу завантаження процесорів

9	Формування кодів таблиць при розподілі програми користувача на даний момент часу: таблиць параметричної ваги фрагментів, програмних блоків і модулів програми користувача, таблиць балансу завантаження ВП і ПЯ, таблиць відповідності системи команд ВП, системи команд ПЯ відповідним наборам операцій для кожного рівня розподілу програми користувача, таблиці зв'язності за даними і іншої поточної інформації.
10	Формування пакетів, що відображають результати розподілу програми користувача на кожному рівні і передача їх в хост-машину і РІМ-систему
11	Перетворення послідовних ділянок програм в паралельні (за наявності такої можливості)
12	Управління умовами переходу при реалізації алгоритмів розділення і розподілу програм користувача
---	-----
q	Формування запитів до БД і пам'яті для запису і читання інформації

### ФОРМУЛА КОРИСНОЇ МОДЕЛІ

5

1. Спосіб розподілу програми користувача для комп'ютерної системи, що включає перетворення початкових кодів програми в проміжні коди, розділення проміжних кодів на множину кодів задач, генерацію інформації про відносини серед множини задач на основі даних в задачах і перетворення кожної задачі в об'єктну програму, яку передають до множини процесорів комп'ютерної системи, який **відрізняється** тим, що спочатку формують багаторівневу модель розподілу кодів програми користувача, потім формують блок послідовності дій, який використовують для кожного i-го поточного рівня розподілу, в межах цього блока визначають можливість розділення програми користувача на незалежні за даними задачі, або частини задач, потім для i-го рівня розподілу формують вхідний керуючий пакет розподілу, аналізують та структурують інформацію про параметри комп'ютерної системи, та про параметри програми користувача та її частин, на основі цієї інформації формують базу поточних значень, які запам'ятовують в електронній пам'яті, потім формують послідовність сигналів кодів мікропрограм для розподілу на поточному рівні, виконують розділення кодів програми користувача або її задач на незалежні за даними коди частин і розподіляють їх між процесорами комп'ютерної системи стосовно сформованої моделі розподілу для i-го поточного рівня, використовуючи критерій відповідності системи команд цих процесорів наборам операцій програми користувача вибраного рівня розподілу, далі після визначення тривалості обробки машинних команд кожної із задач перевіряють баланс завантаження процесорів комп'ютерної системи кодами програми користувача, які присутні в моделі розподілу поточного i-го рівня, і за наявності балансу генерують для поточного рівня коди вихідного пакету, інакше коректують розподіл шляхом укрупнення або подрібнення отриманих задач програми користувача або їх частин, далі на кінцевому рівні розподілу кожну частину програми, яка призначена на попередньому рівні, розділяють на програмні модулі, які розподіляють між усіма процесорами, що входять в набір моделі розподілу кінцевого рівня, при цьому виконують послідовність дій стосовно блока послідовності дій за виключенням того, що використовують модель розподілу, а також формують послідовність кодів мікропрограм, коди полів керуючого вхідного та вихідного пакетів для кінцевого рівня розподілу, при цьому як критерій розділення кодів програмних частин на коди модулів програми використовують час виконання модулів на кожному процесорі, які присутні в моделі розподілу цього рівня, і зв'язки за даними, далі за результатами розподілу на всіх рівнях, за даними кодів вихідних пакетів формують групи кодів задач програми користувача, які не мають зв'язків між собою за даними, при цьому послідовність виконання кодів груп визначають на підставі зв'язків між задачами в різних групах згідно порядку зв'язків даних програми користувача, аналізують всі отримані задачі, щоб визначити наявність таких груп, після цього перетворюють ці програмні групи в об'єктні програми, які передають до відповідних процесорів комп'ютерної системи, а також до хост-машини, формують та передають до комп'ютерної системи та хост-машини коди вихідних керуючих пакетів, аналогічних вихідним пакетам для i-го рівня розподілу, з відповідними індексами і ознаками та їх значеннями для груп, що визначає розділені по процесорах вихідні команди, дані та керуючу інформацію, які

необхідні для паралельного виконання кодів програми користувача на комп'ютерній системі та на хост-машині.

2. Спосіб за п. 1, який **відрізняється** тим, що у вхідний керуючий пакет, який формують до кожного рівня розподілу, включають коди полів, що містять код i-го рівня розподілу програми користувача; а також стосовно рівня розподілу - код ідентифікатора параметрів програми користувача або її задачі; частини, модуля, код ідентифікатора параметрів комп'ютерної системи; код ідентифікатора адреси програми користувача або її задачі; частини, модуля; код запуску системи на обробку програми користувача, або її задачі; частини, модуля; код інсталяції (деінсталяції) програми користувача або її задачі, частини, модуля; код ознаки ініціалізації програми користувача або її задачі, частини, модуля; код ідентифікатора наявності посилань, якій включає: код ідентифікатора таблиці посилань; код ознаки наявності посилань на стандартні методи розпаралелювання; код ознаки наявності посилань на інші програмні блоки; код ідентифікатора джерела посилань програми користувача або її задачі, частини, модуля; код ідентифікатора приймача посилань програми користувача або її задачі, частини, модуля; код повідомлення про зміну посилань в таблиці посилань; код ідентифікатора початкового посилання на обробку.

3. Спосіб за п. 1, який **відрізняється** тим, що у вихідні пакети, які формують після кожного рівня розподілу на i-ому рівні програми користувача, включають коди полів, що містять код рівня розподілу програми користувача; а також стосовно рівня розподілу - код ідентифікатора адресів задач програми користувача, або їх частин та модулів після розділення; код ідентифікатора апаратних блоків, до яких направлено задачі програми користувача або їх частини та модулів; код інсталяції (деінсталяції) задач програми користувача або їх частин та модулів; код типу обробки розподіленої програми користувача; код ідентифікатора адресів груп з паралельними задачами або частинами, або модулями програми користувача; код ідентифікатора посилань між групами програми користувача; код ідентифікатора адресів груп джерел посилань; код ідентифікатора адресів груп, які приймають посилання; код ознаки повідомлення від комп'ютерної системи про виконання групи; код ознаки повідомлення про зміну зв'язків між групами; код ознаки відповіді комп'ютерної системи про виконання обробки; код ідентифікатора адреси наступного вихідного пакету, вільне поле.

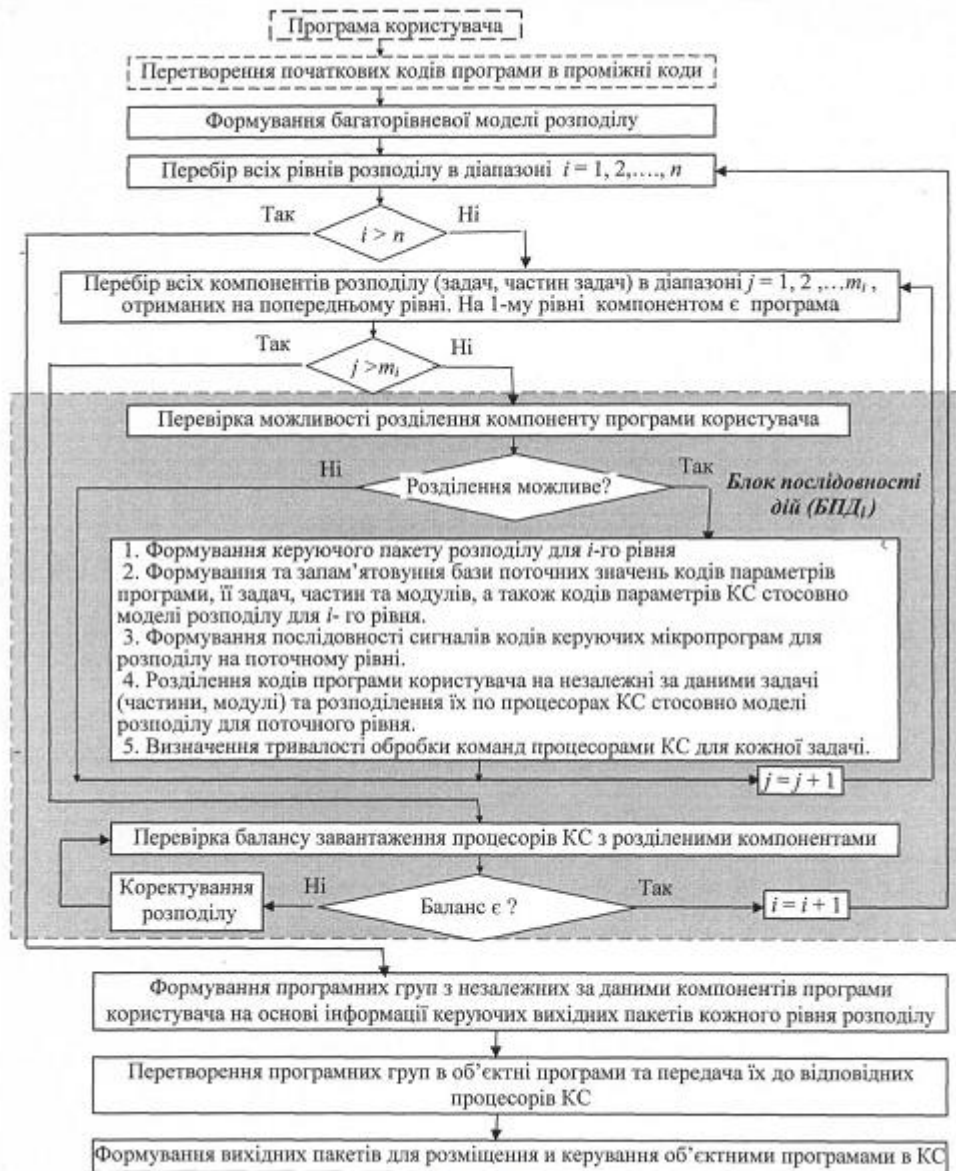


Fig. 1



1	2	3	4	5	6	7	8
Код і-го рівня розподілу програми користувача	Ідентифікатор параметрів програми користувача для і-го рівня	Ідентифікатор параметрів РІМ-системи для і-го рівня	Ідентифікатор адреси програми користувача (задачі, частини, модуля)	Код запуску системи на обробку програми користувача (задачі, частини, модуля)	Код інсталяції (деінсталяції) програми користувача (задачі, частини, модуля)	Ознака ініціалізації	Ідентифікатор наявності посилань

9	10	11	12	13	14	15
Ідентифікатор таблиці посилань	Ознака наявності посилань на стандартні методи розпаралелювання	Ознака наявності посилань на інші програмні блоки	Ідентифікатор джерела посилань програми користувача (задачі, частини, модуля)	Ідентифікатор приймача посилань програми користувача (задачі, частини, модуля)	Код повідомлення про зміну посилань в таблиці посилань	Ідентифікатор початкового посилання на обробку

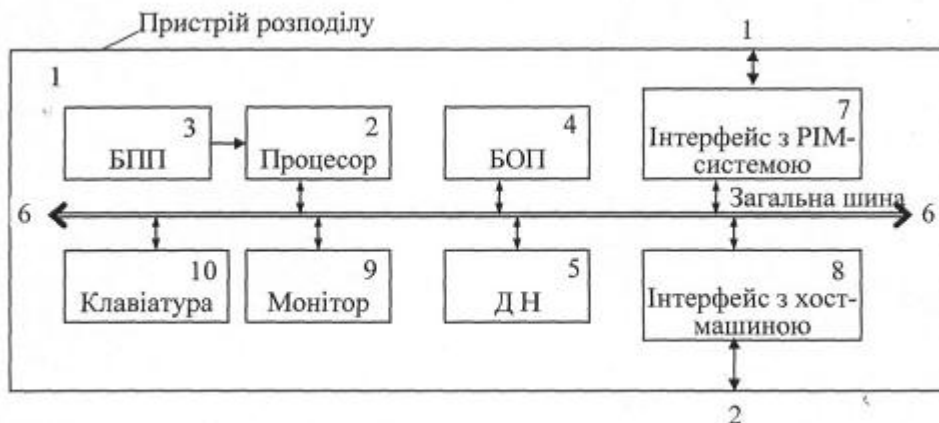
Фіг. 2

Код рівня моделі розподілу програми користувача	Ідентифікатор адресів задач, або їх частин і модулів програми користувача	Ідентифікатор апаратних блоків, до яких направлені задачі (їх частини і модулі) програми користувача	Код інсталяції (деінсталяції) задач (їх частин і модулів) програми користувача	Тип обробки розподіленої програми користувача (її частин і модулів)	Ідентифікатор адресів груп з паралельними задачами, (частинами і модулями) програми користувача
---	---	--	--	---	---

Ідентифікатор посилань між групами програми користувача	Ідентифікатор адресів груп джерел посилань	Ідентифікатор адресів груп, які приймають посилання	Ознака повідомлення про виконання групи	Ознака повідомлення про зміну зв'язків між групами	Ознака відповіді РІМ-системи про виконання обробки	Ідентифікатор адреси наступного пакету	Вільне поле
---	--	---	---	--	--	--	-------------

Фіг. 3



Фіг. 4



---

Комп'ютерна верстка М. Мацело

---

Державна служба інтелектуальної власності України, вул. Урицького, 45, м. Київ, МСП, 03680, Україна

---

ДП "Український інститут промислової власності", вул. Глазунова, 1, м. Київ – 42, 01601