



УКРАЇНА

(19) UA

(11) 55489

(13) C2

(51) 7 H04N7/173

МІНІСТЕРСТВО ОСВІТИ
І НАУКИ УКРАЇНИДЕРЖАВНИЙ ДЕПАРТАМЕНТ
ІНТЕЛЕКТУАЛЬНОЇ
ВЛАСНОСТІОПИС
ДО ПАТЕНТУ НА ВІНАХІД

(54) ПРИСТРІЙ ДЛЯ БАГАТОПОТОКОВОЇ ОБРОБКИ ДАНИХ (ВАРІАНТИ)

1

2

(21) 2000052616

(22) 07 10 1998

(24) 15 04 2003

(86) PCT/IB98/01627, 07 10 1998

(31) 97402361 6

(32) 07 10 1997

(33) EP

(31) 97402362 4

(32) 07 10 1997

(33) EP

(31) 97402430 9

(32) 07 10 1997

(33) EP

(46) 15 04 2003, Бюл. №4, 2003 р

(72) Ляо Хонгтао, FR, Янг Руі Лянг, CN

(73) КАНАЛЬ+ СОСЬЕТЕ АНОНІМ, FR

(56) EP, 0714684, 05 06 1996

EP, 0709769, 01 05 1996

(57) 1 Пристрій для обробки цифрових аудіовізуальних даних, що має щонайменше одну відповідну апаратну операційну систему, асоційовану з одним або декількома апаратними пристроями для передавання і приймання даних, і що також включає в себе систему обробки даних, яка має багатопотокову віртуальну машину, виконану з можливістю, серед іншого, приймання повідомлень про події, що посилаються апаратною операційною системою, і призначення відповідних об'єктів повідомлень одному або декільком потокам, причому потік може бути призупинений під час його виконання для уможливлення виконання іншого потоку

2 Пристрій за п. 1, в якому віртуальна машина має багатопотокову архітектуру з витісненням, де потік призупиняється під час його виконання при створенні потоку з більш високим пріоритетом

3 Пристрій за п. 1 або 2, в якому віртуальна машина має менеджер подій, виконаний з можливістю, у відповідь на повідомлення про подію, яке посилається апаратною операційною системою, збереження об'єкта події в одному або декількох потоках у впорядкований за пріоритетами черзі потоків

4 Пристрій за п. 3, в якому повідомлення про події, що посилаються апаратною операційною системою в менеджер подій, спочатку обробляються одним або декількома менеджерами пристроїв в системі обробки даних

5 Пристрій за п. 3 або 4, в якому менеджер подій виконаний з додатковою можливістю реагування на повідомлення про події, що генеруються всередині віртуальної машини, або повідомлення про події, що отримуються від високорівневих прикладних програм, які виконуються в системі обробки даних або поза нею

6 Пристрій за будь-яким з пп. 3-5, в якому менеджер подій ранжує об'єкти подій всередині потоку відповідно до пріоритету повідомлення про подію і/або часу надходження повідомлення про подію

7 Пристрій за будь-яким з пп. 3-6, в якому віртуальна машина додатково має таблицю маршрутизації, яка містить інформацію про можливі повідомлення про події, до якої звертається менеджер подій для уможливлення визначення менеджером подій відповідності отриманого повідомлення про подію тому або іншому потоку

8 Пристрій за п. 7, в якому таблиця маршрутизації містить інформацію, яка уможливорює визначення менеджером подій пріоритету об'єкта події в потоці

9 Пристрій за будь-яким з пп. 3-8, в якому віртуальна машина додатково містить планувальник, виконаний з можливістю аналізу потоків, що входять у впорядковану за пріоритетами чергу потоків, і ініціювання виконання потоку, що має найвищий пріоритет на даний момент часу

10 Пристрій за п. 9, в якому менеджер подій виконаний з можливістю повідомлення про надходження повідомлення про подію і ініціювання аналізу планувальником нового стану потоків, що входять у чергу потоків

11 Пристрій для обробки цифрових аудіовізуальних даних, який включає в себе один або декілька апаратних пристроїв для передавання і приймання даних, зовнішніх по відношенню до цього пристрою, причому цей пристрій включає в себе також систему обробки даних, яка включає в себе першу віртуальну машину, виконану з можливістю, серед іншого, приймання написаного на інтерпретаційній мові коду, що завантажується через один або декілька із зазначених апаратних пристроїв, причому зазначена віртуальна машина виконана з можливістю розрізнення коду, написаного на щонайменше двох інтерпретаційних мовах, виходячи зі структури прийнятого коду, і передавання такого коду

(13) C2

(11) 55489

(19) UA

на відповідний інтерпретувальний засіб для інтерпретації і виконання

12 Пристрій за п 11, в якому віртуальна машина розрізняє інтерпретаційний код на щонайменше двох інтерпретаційних мовах, виходячи з параметрів повідомлення-заголовка, що відноситься до модуля коду на одній з зазначених мов

13 Пристрій за п 11 або 12, в якому віртуальна машина розрізняє інтерпретаційний код, виходячи з наявності або відсутності повідомлення-заголовка, що відноситься до модуля коду на одній з зазначених мов

14 Пристрій за будь-яким з пп 11-13, в якому щонайменше одна з зазначених мов є об'єктно-орієнтованою мовою

15 Пристрій за будь-яким з пп 11-14, в якому віртуальна машина ідентифікує код, написаний на об'єктно-орієнтованій мові, за наявності повідомлення-заголовка, що відноситься до файлу класу цієї мови

16 Пристрій за будь-яким з пп 11-15, в якому кожний інтерпретувальний засіб виконує код із зверненням до однієї або декількох бібліотек функцій

17 Пристрій за п 16, в якому загальна бібліотека функцій спільно використовується декількома інтерпретувальними засобами

18 Пристрій за п 16 або 17, в якому один або декілька із зазначених інтерпретувальних засобів виконують код із зверненням до бібліотеки функцій виключно даного інтерпретувального засобу

19 Пристрій для обробки цифрових аудіовізуальних даних, який має систему обробки даних, яка включає в себе пам'ять і менеджер пам'яті для розміщення і зберігання об'єктів в зазначеній пам'яті, і в якому перша множина об'єктів розміщується менеджером пам'яті з використанням множини дескрипторів, при цьому кожний дескриптор включає в себе посилання на адресу в пам'яті відповідного об'єкта, і в якому друга множина об'єктів розміщується і зберігається в пам'яті безпосередньо, без використання дескрипторів

20 Пристрій за п 19, в якому об'єкти другої множини розміщуються і зберігаються в пам'яті менеджером пам'яті, але можуть зчитуватися безпосередньо іншими елементами системи обробки даних

21 Пристрій за п 19 або 20, в якому зазначена множина дескрипторів також зберігається в зазначеній пам'яті

22 Пристрій за будь-яким з пп 19-21, в якому менеджер пам'яті виконаний з можливістю переміщення об'єктів першої множини в зазначеній пам'яті і відповідного змінювання адресного посилання, що зберігається у відповідному дескрипторі

23 Пристрій за п 22, в якому об'єкти першої множини переміщуються в пам'яті менеджером пам'яті, коли подальше зберігання нових об'єктів в пам'яті неможливе без переміщення існуючих збережених об'єктів

24 Пристрій за п 22 або 23, в якому об'єкти першої множини переміщуються відповідно до певного алгоритму ущільнення, так щоб об'єднати максимальний об'єм вільної пам'яті як обчислено зазначеним алгоритмом ущільнення

25 Пристрій за будь-яким з пп 19-24, в якому об'єкти другої множини є такими, що не можуть переміщуватися в пам'яті

26 Пристрій за будь-яким з пп 19-25, в якому система обробки даних включає в себе віртуальну машину, причому менеджер пам'яті утворює частину цієї віртуальної машини

27 Пристрій за п 26, в якому віртуальна машина має багатопотокову архітектуру, де потік може тимчасово призупинятися під час його виконання для уможливлення виконання іншого потоку

28 Пристрій за п 27, в якому багатопотокова віртуальна машина має утворений всередині неї потік "збирач сміття", причому віртуальна машина при виконанні цього потоку видаляє з пам'яті об'єкти, на які немає посилань на даний момент часу

29 Пристрій за п 28, в якому виконання потоку "збирання сміття" також ініціює виконання віртуальною машиною переміщення об'єктів першої множини відповідно до певного алгоритму ущільнення для об'єднання максимальної кількості вільної пам'яті

30 Пристрій за будь-яким з пп 19-29, в якому пам'ять визначається одним або декількома компонентами ОЗП

31 Пристрій за будь-яким з попередніх пунктів, який є декодером для цифрової системи трансляції, такої як система цифрового телебачення

32 Пристрій за будь-яким з пп 1-18, в якому один із зазначених апаратних пристроїв включає в себе MPEG-демультиплексор

33 Пристрій за будь-яким з пп 1-18, в якому зазначені апаратні пристрої можуть включати в себе щонайменше один засіб з таких: тюнер, послідовний інтерфейс, паралельний інтерфейс, модем і один або декілька пристроїв зчитування смарт-карт

34 Пристрій для обробки цифрових аудіовізуальних даних, який включає в себе засіб керуванням апаратним забезпеченням, асоційований з одним або декількома апаратними пристроями для передавання і приймання даних, і багатопотокову віртуальну машину, яка включає в себе засіб для приймання повідомлень про події, що посилюються засобом керуванням апаратним забезпеченням, засіб для призначення відповідних об'єктів повідомлень одному або декільком потокам і засіб для призупинення потоку під час його виконання для уможливлення виконання іншого потоку

35 Пристрій для обробки цифрових аудіовізуальних даних, який включає в себе один або декілька апаратних пристроїв для передавання і приймання даних, зовнішніх по відношенню до цього пристрою, і віртуальну машину, яка включає в себе засіб для приймання написаного на інтерпретаційній мові коду, що завантажується через один або декілька із зазначених апаратних пристроїв, засіб для розрізнення коду, написаного на щонайменше двох інтерпретаційних мовах, виходячи зі структури прийнятого коду, і засіб для передавання такого коду на відповідний інтерпретувальний засіб для інтерпретації і виконання

36 Пристрій для обробки цифрових аудіовізуальних даних, який включає в себе пам'ять і менеджер пам'яті, що має засіб для розміщення першої

множини об'єктів з використанням множини дескрипторів, при цьому кожний дескриптор включає в себе посилання на адресу відповідного об'єкта в

пам'яті, і засіб для розміщення і зберігання другої множини об'єктів в пам'яті безпосередньо, без використання дескрипторів

Даний винахід відноситься до пристрою для обробки цифрових аудіовізуальних даних, зокрема до декодера для системи цифрового телебачення, що містить багатопотокову систему обробки даних

У міжнародній заявці РСТ/EP97/02116 описується програмна система для управління декодером в системі цифрового телебачення, що використовує віртуальну машину і виконуючу підсистему для роботи з цифровими телевізійними даними і прикладними програмами, що завантажуються. Система має ряд переваг в порівнянні з відомими раніше системами для приймачів/декодерів, особливо відносно незалежності програмних рівнів системи від апаратних елементів декодерів, що виробляються, завдяки застосуванню віртуальної машини

Система, що описується в згаданій заявці, використовує для управління виникаючими в системі подіями і їх обробки принцип структури, заснованої на застосуванні однофайлової черги. Зі структурою, заснованою на принципі черги, пов'язаний ряд недоліків, включаючи відносно повільне реагування на події з високим пріоритетом і нездатність системи ефективно обробляти декілька паралельних вхідних потоків. Як викладено в заявці, система включає ряд модулів управління послідовністю процесів. Хоч система може управляти пріоритетами функціонування таких модулів, після того як деякий процес запущений, не можна перемкнутися на інший процес

Ці недоліки згаданої структури стають особливо відчутними у випадку, коли приймач/декодер містить інтерактивну прикладну програму. Наприклад, якщо система не може перемикає задачі у відповідь на пріоритетну команду і при цьому час, потрібний тривалий час для завантаження даних, то це може привести до блокування роботи системи, не зважаючи на команди користувача на перемикання в інший режим

Існує також необхідність в спрощенні структури драйвера пристрою для цієї відомої системи. Зв'язок між виконуючою підсистемою і пристроями апаратного рівня відомого декодера здійснюється за допомогою множини драйверів пристроїв, при цьому управління ними всіма здійснюється менеджером пристроїв, керуючим призначенням пріоритетів повідомленням про події і постановкою їх в структуру черги модулів управління послідовністю процесів. Як описано в згаданій заявці, хоч виконуюча підсистема надається постачальником системи, драйвери пристроїв і менеджер звичайно надаються виробником декодера у відповідності зі специфікаціями постачальника системи

У даному контексті термін "пристрій" звичайно використовується для позначення інтерфейсних пристроїв, що використовуються для обробки даних, які отримуються і передаються, декодером,

наприклад, даних, що отримуються через смарт-карту або через потік віщання, тощо

Відмінна (від тієї, що малася на увазі) інтерпретація специфікації виробником декодера може привести до проблем, коли, наприклад, менеджер не дотримується правильної класифікації подій з пріоритетом. У такому випадку функціонування системи, заснованої на черзі, буде порушено, оскільки події, що поступають в фільтр подій і модуль управління послідовністю процесів, будуть неправильно ідентифікуватися (в значенні їх пріоритетів) і неправильно оброблятися системою черги

Задачею одного з об'єктів даного винаходу є подолання даної проблеми

Відповідно до першого аспекту даного винаходу пропонується пристрій для обробки цифрових аудіовізуальних даних, причому пристрій має щонайменше одну відповідну апаратну операційну систему, асоційовану з одним або більш апаратними пристроями для передачі і прийому даних, і пристрій додатково містить систему обробки даних, що включає в себе багатопотокову віртуальну машину, виконану з можливістю, серед іншого, прийому повідомлень про події, що посилаються апаратною операційною системою, і призначення відповідних об'єктів повідомлень одному або більш потокам, і в якому потік, що містить об'єкт події, може бути припинений під час виконання, щоб зробити можливим виконання іншого потоку

Завдяки використанню багатопотокової архітектури даний винахід дозволяє системі ефективно реагувати на надходження подій, що отримуються через зовнішні інтерфейси пристрою, забезпечуючи швидку обробку подій з високим пріоритетом при тимчасовому припиненні нетермінових процесів

У одному з варіантів здійснення винаходу віртуальна машина має багатопотокову архітектуру з витісненням, при якій потік припиняється під час виконання при створенні потоку з більш високим пріоритетом. Хоч цей варіант здійснення винаходу є переважним через швидкість реагування на пріоритетні події, можуть розглядатися і інші варіанти здійснення, такі, як варіант здійснення з квантуванням часу, в якому віртуальна машина перериває виконання потоку через задані періодичні інтервали, щоб перевірити, чи існує інший потік для виконання

Переважно віртуальна машина містить менеджер подій, виконаний з можливістю, у відповідь на повідомлення, що посилається апаратною операційною системою про подію, збереження об'єкта події в одному або більш потоках у впорядкованій по пріоритетах черзі потоків

Таким чином, призначення пріоритетів подіям може проводитися безпосередньо віртуальною машиною, тим самим усуваючи проблеми, власти-

ві відомій системі, в якій спочатку події упорядковуються для постановки в чергу пристрою обробки низькорівневими драйверами пристроїв і менеджерами. Як описано вище, реалізації драйвера можуть розрізнятися у різних виробників. У протилежність цьому, в даному переважному варіанті здійснення винаходу повідомлення про події сортується і ранжується по пріоритетах менеджером подій у віртуальній машині, характеристики якої не змінюються в залежності від платформи.

Незважаючи на те, що обробка подій тепер виконується віртуальною машиною, в деяких варіантах здійснення винаходу система все ж може містити один або більш драйверів пристроїв, що служать інтерфейсом між операційною системою віртуальної машини і операційною системою апаратного рівня.

У доповнення до подій, що поступають від апаратної операційної системи, менеджер подій може також бути сконфігурований так, щоб забезпечити можливість реагування на повідомлення про події, що поступають з самої віртуальної машини або від високорівневих прикладних програм.

У переважних варіантах здійснення винаходу об'єкти подій в потоку можуть також ранжуватися по пріоритетах подій і/або часу надходження події. Це може виконуватися додатково до згаданого призначення пріоритетів, що виконується при розподілі подій по потоках в черзі потоків.

У одній з реалізацій винаходу віртуальна машина може також містити таблицю маршрутизації, яка містить інформацію про можливі повідомлення про події, до якої звертається менеджер подій, з тим, щоб менеджер подій міг визначати відповідність тому або іншому потоку отриманого повідомлення про подію. Ця таблиця маршрутизації може також використовуватися для визначення пріоритету об'єкта події в потоку. Як буде зрозуміло фахівцям в даній галузі, можливо застосування і альтернативних засобів.

У доповнення до менеджера подій і таблиці маршрутизації, віртуальна машина також переважно містить планувальник, виконаний з можливістю аналізу потоків, що входять у впорядковану по пріоритетах чергу потоків, і ініціювання виконання потоку, що має найвищий пріоритет в даний час. Для реалізації управління потоками з витісненням менеджер подій може бути виконаний з можливістю сигналізування про надходження повідомлення про подію і виклику планувальника для перевірки нового стану потоків, що містяться в черзі потоків.

Ще один недолік описаної в згаданій міжнародній заявці РСТ/EP97/02116 системи пов'язаний з обробкою коду, що отримується. Хоч використання віртуальної машини і виконуючої підсистеми дозволяє системі, описаній в згаданій заявці, бути в істотній мірі незалежною від апаратного рівня системи, розширюваність відомої системи все ж обмежується кодом, що застосовується для написання прикладних програм, які представляють більш високий рівень в порівнянні з рівнем віртуальної машини. Як описано в згаданій заявці, код пишеться на мові, що інтерпретується, яка завантажується в приймач з центра віщання і інтерпретується інтерпретатором у віртуальній машині.

Хоч як код може бути вибрана мова, що є по-

ширеною на ринку і стандартизованою, можливе виникнення проблем у випадку, коли, наприклад, приймач повинен виконувати прикладні програми, написані в двох або більш різних кодах. Ця проблема може виникати, наприклад, коли декодер встановлюється в системі віщання, в якій наявні декодери пристосовані для прийому прикладних програм, написаних в код, відмінний від того, який використовується в даному декодері. У такому випадку оператор може виявитися вимушеним завантажувати деяку прикладну програму двічі: один раз написану на оригінальній мові для наявних декодерів і один раз написану в новому код для нових декодерів. Зрозуміло, що такий режим роботи є відносно неефективним в тому, що стосується використання смуги пропускання.

Метою ще одного об'єкта даного винаходу є подолання цього недоліку.

У відповідності з другим аспектом даного винаходу пропонується пристрій для обробки цифрових аудіовізуальних даних, який містить один або більш апаратних пристроїв для передачі і прийому даних, зовнішніх по відношенню до названого пристрою, причому названий пристрій додатково містить систему обробки даних, яка містить першу віртуальну машину, виконану з можливістю, серед іншого, прийому написаного на мові, що інтерпретується, коду, що завантажується через один або більш із згаданих апаратних пристроїв, причому згадана віртуальна машина виконана з можливістю розрізнення коду, написаного на принаймні двох мовах, що інтерпретуються, в залежності від структури коду, що приймається, і передачі такого коду на відповідний інтерпретуючий засіб для інтерпретації і виконання.

Застосовуючи віртуальну машину, виконану з можливістю розрізнення коду, що приймається, і декілька інтерпретуючих засобів для інтерпретації такого коду, даний винахід усуває проблеми, властиві існуючим системам, і дозволяє пристрою обробляти інструкції на різних мовах, що інтерпретуються. Таким чином, може бути надана система, що повністю розширюється, як відносно високорівневих прикладних, так і низькорівневих апаратних інтерфейсів.

У одному з варіантів здійснення віртуальна машина розрізняє код, що інтерпретується на щонайменше двох мовах, що інтерпретуються, виходячи з параметрів повідомлення-заголовка, що відноситься до модуля коду на одній з таких мов. Зокрема, віртуальна машина може розрізняти код, що інтерпретується, виходячи з наявності або відсутності повідомлення-заголовка, що відноситься до модуля коду, для однієї з мов. Можна представити і інші варіанти здійснення, в яких віртуальна машина розрізняє код на основі "прапора" або іншого елемента коду всередині або в кінці потоку коду, що передається, або по імені файла модуля коду.

Даний винахід особливо застосовний в ситуації, коли одна або більш із згаданих мов, що інтерпретуються, є об'єктно-орієнтованою мовою. У такому випадку пристрій може перевіряти наявність повідомлення-заголовка, що відноситься до файла класу цієї мови.

Для виконання коду кожний інтерпретуючий

засіб може виконувати код із зверненням до однієї або більш бібліотек функцій. Переважно загальна бібліотека функцій спільно використовується декількома інтерпретуючими засобами для зменшення об'єму пам'яті, що займається бібліотеками функцій.

Незважаючи на наявність загальної бібліотеки функцій, один або більш із згаданих інтерпретуючих засобів можуть виконувати код із зверненням до бібліотеки функцій виключно даного інтерпретатора. Це може бути бажане, наприклад, коли певні спеціалізовані функції простіше виконуються при зверненні до спеціальної бібліотеки функцій. Зрозуміло, що розмір системи може бути зменшений при використанні бібліотек функцій, спільних для обох інтерпретуючих засобів, де це можливе і/або зручне.

Ще один недолік запропонованої в згаданій міжнародній заявці RCT/EP97/02116 системи пов'язаний з роботою з пам'яттю, що використовується системою при обробці команд. Система, описана в згаданій заявці, використовує менеджер пристроїв для управління пам'яттю. У цій системі всі звернення до пам'яті з віртуальної машини обробляються однаково. Крім того, не розглядається, як може бути оптимізоване використання пам'яті менеджером пристроїв.

Менеджер пристроїв є частиною рівня більш низького, ніж рівень віртуальної машини, і реалізовується виробником приймача/декодера, а не постачальником системи. Тому існує небезпека того, що реалізація, вибрана виробником приймача/декодера, не буде оптимальною з урахуванням потреб елементів системи більш високих рівнів, таких, як віртуальна машина, що розробляється постачальником системи.

Метою даного винаходу у відповідності з наступним аспектом є подолання деяких або всіх цих проблем і надання поліпшеної системи для управління пам'яттю в аудіовізуальному пристрої.

Відповідно до третього аспекту даного винаходу пропонується пристрій для обробки цифрових аудіовізуальних даних, що містить систему обробки даних, включаючи в себе пам'ять і менеджер пам'яті, для розміщення і зберігання об'єктів в пам'яті, і в якому перша множина об'єктів розміщується менеджером пам'яті з використанням множини дескрипторів, при цьому кожний дескриптор містить показник на адресу в пам'яті відповідного об'єкта, і в якому друга множина об'єктів розміщується і зберігається в пам'яті безпосередньо, без використання дескрипторів.

Розподіленням пам'яті між об'єктами, доступними через дескриптор, і безпосередньо доступними об'єктами в даному винаході встановлюється відмінність між першою множиною об'єктів, які можуть оброблятися менеджером пам'яті різними способами для оптимізації використання пам'яті, як буде описано нижче, і тими об'єктами, доступ до яких здійснюється частіше, до яких можна адресуватися прямо, без необхідності в посиланні на дескриптор.

Зокрема, в одному з варіантів здійснення винаходу, об'єкт з другої множини може бути доступний іншим елементам в системі обробки даних безпосередньо, без необхідності передачі через

менеджер пам'яті. Менеджер пам'яті, проте, може бути необхідним для розміщення і збереження об'єктів другої множини в пам'яті, для того щоб зберігати контроль за вмістом пам'яті.

У одному з варіантів здійснення дескриптори можуть зберігатися в згаданій пам'яті. Однак можливі інші варіанти реалізації, в яких дескриптори зберігаються в іншій області пам'яті. Дескриптори можуть зберігатися динамічно або в статичному масиві.

У одній з реалізацій менеджер пам'яті виконаний з можливістю переміщення в пам'яті об'єктів першої множини, з відповідною зміною адресного показника, який зберігається у відповідному дескрипторі. Об'єкти можуть переміщатися, наприклад, коли подальше збереження нових об'єктів в пам'яті неможливе. Переміщення може здійснюватися, наприклад, відповідно до відповідного алгоритму ущільнення. Таким чином можна оптимізувати вільну пам'ять, при збереженні можливості простого і ефективного контролю за об'єктами, що зберігаються.

Для того, щоб доступ до об'єктів другого типу був завжди можливий, ці об'єкти переважно є непереміщуваними в пам'яті. Можливі і інші реалізації, в яких, наприклад, переміщення об'єкта другого типу буде пов'язане з процедурою зміни адреси об'єкта, по якій він знаходиться в системі.

Даний винахід особливо застосовний до варіанту здійснення, в якому віртуальна машина має багатопотокову архітектуру того типу, який описаний в зв'язку з першим аспектом винаходу, при якому потік може бути тимчасово припинений під час виконання, щоб дозволити виконання іншого потоку.

У цьому випадку багатопотокова віртуальна машина може переважно включати потік "складальника сміття, що внутрішньо формується", віртуальна машина при виконанні цього потоку звільняє пам'ять від об'єктів, на які немає посилань в даний час. Як альтернатива або додаткова можливість при виконанні потоку "складальника сміття" віртуальною машиною може також виконуватися переміщення об'єктів першої множини відповідно до алгоритму ущільнення, для того щоб об'єднати максимальний об'єм вільної пам'яті.

Область пам'яті, що розглядається, може відповідати ОЗП системи, хоч даний винахід в рівній мірі застосовний до інших компонентів пам'яті, таких як пристрої флеш-пам'яті або ЕСППЗП.

Хоч даний винахід особливо застосовний в декодері для прийому і обробки сигналів цифрового телебачення, очевидно, що принципи роботи системи обробки даних, викладені в даній заявці, можуть також застосовуватися в інших пристроях для обробки цифрових аудіовізуальних даних, таких як цифрові відеомагнітофони тощо.

У випадку декодера для цифрового телевізійного віщання, апаратні пристрої декодера можуть включати один або всі з наступних: MPEG демультіплексор разом з тюнером, послідовний інтерфейс, паралельний інтерфейс, модем і один або більш пристроїв зчитування смарт-карт.

Термін "приймач/декодер" або "декодер", що використовується тут, може означати приймач для прийому або закодованих, або незакодованих сиг-

налів, наприклад, телевізійних і/або радіосигналів, які можуть передаватися або трансплюватися деякими іншими засобами. Цей термін може також означати декодер для декодування сигналів, що приймаються. Варіанти здійснення таких приймачів/декодерів можуть включати декодер, суміщений з приймачем, наприклад, в телеприставці, для декодування сигналів, що приймаються, декодер, що функціонує в поєднанні з фізично окремим приймачем, декодер, що має додаткові функції, такі, як Web-браузер, або декодер, інтегрований з іншими пристроями, такими, як відеомагнітофон або телевізор.

Термін "цифрова система трансляції", що використовується тут, включає будь-яку систему трансляції для передачі або віщання, наприклад, головним чином аудіовізуальних або мультимедійних цифрових даних. Хоч даний винахід особливо застосовний у віщальній системі цифрового телебачення, він може також застосовуватися в фіксованій телекомунікаційній мережі для мультимедійних прикладних Internet-програм, в кабельному телебаченні, і т.д.

Термін "система цифрового телебачення", що використовується тут, включає, наприклад, будь-які супутникові, наземні, кабельні і інші системи.

Термін MPEG, що використовується нижче в описі, відноситься до стандартів передачі даних, розроблених робочою групою Міжнародної організації по стандартизації "Motion Pictures Expert Group" ("Експертна група по рухомих зображеннях") і, зокрема, але не виключно, до стандарту MPEG-2, розробленого для прикладних програм цифрового телебачення і викладеному в документах ISO 13818-1, ISO 13818-2, ISO 13818-3 і ISO 13818-4. У контексті даної патентної заявки цей термін включає всі варіанти, модифікації і розвинуті формати MPEG, застосовні в галузі цифрової передачі даних.

Нижче буде описаний варіант здійснення даного винаходу, виключно як ілюстративний приклад, з посиланнями на прикладні малюнки, де:

На фіг 1 показана загальна архітектура системи цифрового телебачення,

На фіг 2 показані елементи інтерактивної системи в системі цифрового телебачення, показаній на фіг 1,

На фіг 3 показана архітектура програмної системи відповідно до даного винаходу, реалізованої в приймачі/декодері,

На фіг 4 показана архітектура віртуальної машини, що входить в показану на фіг 3 систему, що включає, зокрема, модуль менеджера подій, модуль інтерпретації і модуль управління пам'яттю,

На фіг 5 показана структура інтерпретатора, що використовується у віртуальній машині,

На фіг 6 показана організація підтримки потоків у віртуальній машині,

На фіг 7 показано функціонування менеджера подій і планувальника віртуальної машини,

На фіг 8 показана організація пулу пам'яті віртуальної машини.

Мережа цифрового телебачення

Загальна структура системи цифрового телебачення 1000 відповідно до даного винаходу показана на фіг 1. Винахід включає практично звичай-

ну систему цифрового телебачення 2000, яка використовує відому систему компресії MPEG-2 для передачі ущільнених цифрових сигналів. Більш детально, пристрій компресії MPEG-2 2002 в центрі віщання приймає потік цифрових сигналів (звичайно потік відеосигналів). Пристрій компресії 2002 підключається до мультиплексора і скремблера 2004 за допомогою каналу 2006.

Мультиплексор 2004 приймає множинну вхідних сигналів, формує один або декілька несучих потоків і передає ущільнені цифрові сигнали в передавач 2008 центра віщання через канал 2010, тип якого, природно, може бути різним, включаючи канали телекомунікацій. Передавач 2008 передає електромагнітні сигнали через канал "Земля-супутник" 2012 на супутниковий ретранслятор 2014, де виконується їх обробка електронними засобами і віщання через віртуальний канал "супутник-Земля" 2016 на наземний приймач 2018, що звичайно має форму тарілки, що належить кінцевому користувачеві або орендується ним. Сигнали, що приймаються приймачем 2018, передаються в суміщений приймач/декодер 2020, що належить кінцевому користувачеві або орендується ним, і підключений до телевізора 2022 кінцевого користувача. Приймач/декодер 2020 декодує ущільнений MPEG-2 сигнал в телевізійний сигнал для телевізора 2022.

Система умовного доступу 3000 підключається до мультиплексора 2004 і приймача/декодера 2020 і розташовується частково в центрі віщання і частково в декодері. Вона дозволяє кінцевому користувачеві здійснювати доступ до цифрових телепередач від одного або декількох операторів віщання. У приймачі/декодері 2020 може встановлюватися смарт-карта, яка може дешифрувати повідомлення, що відносяться до комерційних пропозицій (однієї або декількох телепередач, що продаються оператором віщання). З використанням декодера 2020 і смарт-карти кінцевий користувач може купувати комерційні пропозиції в режимі підписки або оплати за перегляд. Інтерактивна система мережі цифрового телебачення.

Інтерактивна система 4000, також підключена до мультиплексора 2004 і приймача/декодера 2020 і також розташована частково в центрі віщання і частково в декодері, дозволяє кінцевому користувачеві взаємодіяти з різними прикладними програмами через модемний зворотний канал 4002.

На фіг 2 показані елементи загальної архітектури інтерактивної телевізійної системи 4000, що включає, взагалі кажучи, чотири основних елементи.

1 Засіб розробки 4004 в центрі віщання або в іншому місці, що дозволяє оператору віщання створювати, розробляти, налагоджувати і тестувати прикладні програми.

2 Сервер прикладних програм і даних 4006 в центрі віщання, з'єднаний зі засобом розробки 4004, для надання оператору віщання можливості готувати, засвідчувати автентичність і формувати прикладні програми і дані для відправлення в мультиплексор і скремблер 2004 для їх вставки в несучий потік MPEG-2 (звичайно в його приватну

секцію), що підлягає віщанню для кінцевого користувача

3 Систему обробки даних 4008 в приймачі/декодері для прийому і обробки прикладних програм, що завантажуються, і даних для управління обміном інформацією з іншими елементами інтерактивної системи і апаратними елементами приймача/декодера, причому система 4008 містить віртуальну машину з виконуючою підсистемою (RTE), реалізованою у вигляді здійснимого коду, інсталюваного в приймачі/декодері

4 Зворотний модемний канал 4002 між приймачем/декодером 2020 і сервером прикладних програм і даних 4006 для подачі сигналів, вказуючих серверу 4006 вставляти дані і прикладні програми в несучий потік MPEG-2 на вимогу кінцевого користувача. Інформація може також передаватися в зворотному напрямі

Приймачі/декодер 2020 містить ряд пристроїв для обміну даними із зовнішніми пристроями в інтерактивній системі, такі, як тюнер для настройки приймача, MPEG демультимплексор для демультимплексування MPEG сигналу, послідовний інтерфейс, паралельний інтерфейс, модем і один або два пристрої зчитування смарт-карт, призначених для зчитування, наприклад, кредитних карт або смарт-карт підписки, що випускаються разом з системою. Характеристики таких пристроїв добре відомі в галузі систем цифрового телебачення і не будуть далі детально описуватися

Аналогічно, різновиди інтерактивних прикладних програм, які можуть бути надані (віддалений доступ до банківських рахунків з будинку, дистанційні покупки, завантаження програмного забезпечення), є відомими для фахівців в даній галузі, і не будуть далі детально описуватися. Хоч архітектура системи декодера, описана нижче, особливо підходить для інтерактивних прикладних програм, зрозуміло, що описана архітектура може бути використана в більш простих неінтерактивних цифрових системах телебачення, таких, як звичайні платні системи телебачення

Архітектура системи декодера

Переходячи тепер до архітектури системи приймача/декодера, показаної на фіг 3, можна бачити, що використовується багаторівнева архітектура. Перший рівень 4100 представляє операційну систему апаратури приймача/декодера. Це операційна система реального часу, вибрана виробником для управління апаратними елементами приймача/декодера. Операційна система реального часу має відносно малий час реакції, що дозволяє правильно синхронізувати апаратні операції. Повідомлення про події передаються між цим рівнем і рівнем проміжного програмного забезпечення 4200, розташованим безпосередньо над ним

Система обробки даних 4008 знаходиться над рівнем апаратної операційної системи і включає рівень проміжного програмного забезпечення і рівень прикладних програм (або, більш правильно, рівень інтерфейсу прикладних програм)

Рівень проміжного програмного забезпечення пишеться на такій мові, як ANSI C, і включає елементи віртуальної машини 4250 і ряд інтерфейсів 4260, в тому числі графічний інтерфейс 4261, інтерфейс флеш-пам'яті і ППЗП 4262, інтерфейс

протоколів 4263 і інтерфейс пристроїв 4264

Як і система, запропонована в заявці РСТ/EP97/02116 і більш детально описана вище, даний винахід використовує віртуальну машину для забезпечення незалежності прикладних програм верхнього рівня від низькорівневої операційної системи, що реалізовується виробником

Інтерфейси 4260 забезпечують зв'язок операцій віртуальної машини і низькорівневої операційної системи 4100, а також включають ряд прикладних програм проміжного рівня, що більш легко виконуються на цьому рівні

Рівень інтерфейсу прикладних програм (API) 4300 включає ряд модулів високого рівня 4310 - 4314, написаних на об'єктно-орієнтованій мові, що інтерпретується, такий, як Java. Ці модулі забезпечують інтерфейс між прикладними програмами, що створюються постачальником послуг (інтерактивне керівництво по програмах, дистанційні покупки, Internet-браузер, і т.ін.), і віртуальною машиною системи. Приклади таких прикладних програм наведені нижче

Операційна система (ОС) нижнього рівня звичайно вбудовується в апаратні компоненти приймача/декодера, хоч в деяких реалізаціях ОС нижнього рівня може бути такою, що завантажується. Проміжне програмне забезпечення і модулі рівня інтерфейсу прикладних програм можуть завантажуватися в ОЗП або флеш-пам'ять декодера з віщальної трансляції. У альтернативному варіанті, деяке або все проміжне програмне забезпечення або елементи рівня інтерфейсу прикладних програм можуть зберігатися в ПЗП або у флеш-пам'яті (якщо така є) декодера. Зрозуміло, що фізична організація пам'яті декодера відрізняється від логічної організації пам'яті

Рівень інтерфейсу прикладних програм

Як показано на фіг 3, де зображений рівень інтерфейсу прикладних програм 4300, згаданий вище, модулі цього рівня написані на об'єктно-орієнтованій мові, такий, як Java. Кожний модуль визначає набір бібліотек класів, що викликаються під час роботи системи. У даній системі встановлені такі модулі

Модуль Lang/Util (Мова/Утиліта) 4310. Ці модулі визначають класи, необхідні для роботи віртуальної машини з об'єктами. Такі бібліотеки класів звичайно утворюють частину стандартної бібліотеки, асоційованої з вибраною об'єктно-орієнтованою мовою

Модуль MHEG-5 4311. Цей модуль визначає класи, пов'язані з маніпулюванням графічними об'єктами на телевізійному екрані. Такі об'єкти відрізняються від аудіовізуальних даних і можуть утворювати, наприклад, ідентифікатори каналів або текст, що накладається понад зображення, що показуються. Визначення класів в цьому модулі повинно відповідати вимогам MHEG-5, що визначаються стандартами ETS 300777-3 і ISO/IEC 13522-5 (і стандартом ISO/IEC 13522-6 у разі системи, реалізованої на Java)

Інструментальний модуль 4312. Цей модуль містить класи, що використовуються для завантаження і декомпресії інформації, класи, пов'язані з управлінням файловою системою і пам'яттю в приймачі/декодері, а також класи, пов'язані із спо-

лученням з Internet, і т ін

Модуль пристроїв 4313 Цей модуль визначає класи, необхідні для управління периферійними пристроями, підключеними до приймача/декодера, згаданими вище, в тому числі модемом, пристроями зчитування смарт-карт, тюнером потоку MPEG, і т ін

Сервісний модуль 4314 Цей модуль визначає класи, необхідні для виконання розробки високорівневих інтерактивних прикладних програм, таких, як управління даними кредитних карт і т ін

Модуль DSMCC-UU 4315 Цей модуль реалізовує протоколи, необхідні для обміну інформацією між клієнтом і сервером для пошуку файлів даних і їх зчитування. Реалізація цього модуля повинна відповідати вимогам ISO/IEC 13818-6 і директивам, визначеним в частині 9 DAVIC

Наступний рівень інтерактивних прикладних програм, який пишеться постачальником послуг і завантажується під час віщання, як і в звичайній системі, буде базуватися на вищеописаних інтерфейсних модулях. У залежності від прикладних програм, що використовуються, деякі з названих модулів можуть бути відсутніми. Наприклад, якщо постачальник послуг не має намір надавати загальний канал для зчитування даних, модуль DSMCC-UU в остаточній системі може бути відсутнім

Модулі 4300 надають бібліотеки класів для об'єктно-орієнтованого середовища програмування. Поведінка їх класів буде залежати від вибраної мови. У разі прикладної Java-програми, наприклад, буде дотримуватися структура класів з одним класом успадкуванням

Інтерфейсний рівень

Як показано, Інтерфейсний рівень складається з чотирьох модулів - графічного модуля 4261, модуля управління файлом пам'яті, модуля протоколів 4263 і менеджера пристроїв 4264. Хоч модулі на цьому рівні описуються як інтерфейсні модулі, їх функцією в основному є надання "зв'язуючого" рівня для реалізації модулів інтерфейсу прикладних програм і для функціонування віртуальної машини

Графічний модуль 4261, наприклад, забезпечує створення і управління графічними об'єктами. Він запитує у низькорівневої ОС відображення основних графічних примитивів, таких, як одиничні пікселі, лінії, прямокутники, і т ін. Реалізація цього модуля залежить від графічних можливостей низькорівневої ОС виробника. Будучи в деякій мірі доповнюючим пакет MHEG-5 4311, ці функції можуть більш ефективно виконуватися на цьому рівні коду в порівнянні з високорівневим кодом, вибраним для рівня інтерфейсу прикладних програм над ним

Аналогічним образом модуль управління файлом пам'яті 4262 включає низькорівневі команди читання/запису файлів, пов'язані з компонентами пам'яті системи. Звичайно апаратна операційна система включає тільки команди, необхідні для запису/читання сектору або сторінки в компоненти пам'яті. Як і для випадку графічного модуля 4261, цей модуль дозволяє ефективно використати в системі набір більш простих низькорівневих задач

Модуль управління протоколами 4263 визна-

чає бібліотеку протоколів обміну інформацією, яка може викликатися при обміні інформацією через, наприклад, TCP/IP-рівень декодера

Менеджер пристроїв 4264 дещо відрізняється від інших модулів цього рівня тим, що він надає канал або інтерфейс між апаратною операційною системою і рівнями над нею, в тому числі і іншими модулями інтерфейсного рівня і віртуальною машиною. Наприклад, команди або повідомлення про події, що отримуються/посилаються апаратною ОС від віртуальної машини, обов'язково передаються менеджером пристроїв для перетворення відповідно до специфікацій інтерфейсу між цими двома рівнями

Опис віртуальної машини

Далі з посиланням на фіг 4 буде описана структура віртуальної машини 4250, яка використовується в системі відповідно до даного винаходу. Віртуальна машина, що використовується в даному винаході, є багатопотоковою машиною витіснюючого типу. Загальні характеристики такої машини відомі з інших галузей, відмінних від аудіовізуальної і цифрової телевізійної галузей, і подальший опис буде зосереджений на тих особливостях, які стосуються даної заявки

Віртуальна машина складається з ряду елементів, які взаємодіють, загалом, як показано на фіг 4

Планувальник 4270, що складається з сервісу менеджера потоків 4271 і сервісу менеджера монітора 4272, є центральним компонентом багатопотокової машини. Планувальник 4270 задає порядок виконання потоків, що створюються прикладною програмою поза віртуальною машиною і що створюються самою віртуальною машиною (наприклад, потік "складальника сміття", описаний нижче)

Менеджер подій 4273 підтримує таблицю маршрутизації подій і списки подій, що отримуються потоками, і централізує диспетчеризацію обробки подій

Менеджер пам'яті 4274 обробляє виділення і звільнення областей пам'яті в пам'яті системи, а також виконує видалення з пам'яті об'єктів, на які немає посилань ("збирання сміття")

Менеджер класів 4275 завантажує класи коду прикладної програми, що завантажується з віщального сигналу, взаємодіючи з менеджером безпеки 4280 для перевірки цілісності коду, що завантажується, і з менеджером файлів 4276, який формує прикладні програми

Менеджер файлів 4276 виконує формування файлів системи і реалізує механізм завантаження інтерактивних прикладних програм і даних

Менеджер безпеки 4280 відповідає за рівень доступу, що надається завантаженим прикладним програмам, деякі прикладні програми можуть виконувати більше число операцій, ніж інші, відносно файлової системи

Інтерпретатор 4277, що включає сервіс інтерпретації байт-коду 4278 і сервіс інтерпретації "т-коду" 4279, виконує інтерпретацію прикладних програм, написаних в одному з цих двох кодів, причому байт-код відповідає прикладним програмам Java, а т-код — це назва, дана власному коду, розробленому заявником. Як буде описано,

при необхідності можуть бути додані додаткові сервіси інтерпретації

Функціонування і реалізація менеджера класів, менеджера файлів і менеджера безпеки можуть бути звичайними. Опис далі буде зосереджений на інтерпретаторові, планувальникові, менеджерів подій і менеджерів пам'яті

Інтерпретатор

Далі з посиланням на фіг 5 буде описано функціонування інтерпретатора 4277, який використовується у варіанті здійснення даного винаходу. Як описано вище, недоліком звичайних операційних систем, що використовуються у відомих декодерах, є їх прив'язка до одного типу коду для високорівневих прикладних програм. Хоч може бути вибраний код, що є широко відомим і що застосовується, проте можуть виникати проблеми у випадку, коли необхідно підтримувати множину декодерів, що використовують різні прикладні програми, написані у множині різних кодів. Інтерпретатор даної системи дозволяє інтерпретувати декілька типів коду.

Як показано, файли, що поступають в систему, оцінюються менеджером класів 4500 у відповідності зі структурою файла на предмет того, чи є вони класами байт-коду або модулями m-коду, так що код прикладної програми, що поступає в інтерпретатор 4510, має показник формату. У разі байт-кодової прикладної програми, наприклад, завантажений файл класу буде мати відмітний 4-байтний заголовок-ідентифікатор, за яким буде слідувати номер версії, також 4-байтний. Інтерпретатор може розрізняти коди в залежності від присутності або відсутності цього заголовка байт-коду.

У інших варіантах здійснення для розрізнення будь-якої кількості мов прикладних програм можуть використовуватися інші характеристики типів коду, такі, як, наприклад, ім'я файла.

У залежності від показника формату, інструкції байт-коду прямують на інтерпретатор байт-коду 4278, де вони виконуються із зверненням до бібліотеки функцій 4520, асоційованої з інструкціями байт-коду, як це і відбувається звичайно для інструкцій коду, що інтерпретується. Бібліотека функцій з інструкціями "рідного" коду визначається у віртуальній машині.

У разі інструкцій m-коду, вони передаються на інтерпретатор t-коду 4278. Більшість інструкцій m-коду можуть формуватися і виконуватися із зверненням до бібліотеки функцій 4520, асоційованої з інструкціями байт-коду, інтерпретатор 4278 здійснює виклики бібліотеки 4520 для виконання таких m-кодових функцій завжди, коли це можливе.

Однак при певних обставинах деякі інструкції m-коду можуть вимагати спеціальних функцій виконання, які не можуть легко виконуватися із зверненням до загальної бібліотеки функцій. Для таких випадків може передбачатися виконання інструкцій із зверненням до окремої бібліотеки функцій 4530.

Планувальник і менеджер подій

Далі буде описано функціонування планувальника 4270 і менеджера подій 4273, з посиланням на фіг 6, на якій показаний життєвий цикл потоку в системі, і на фіг 7, на якій показане розміщення

системою об'єкта події в потік, у відповідь на подію, про яку повідомила низькорівнева операційна система.

Будуть детально описані дії по створенню потоку, який представляє середовище виконання, що виникає, зокрема, при повідомленні про подію. Зрозуміло, що створення потоку може ініціюватися командою, що генерується високорівневою прикладною програмою і що посилюється апаратною ОС, і при поверненні цієї команди. Потік також може створюватися в рамках самої віртуальної машини, наприклад, потік "складальника сміття".

Як було згадано вище, варіант здійснення винаходу, що розглядається, засновується на віртуальній машині з витіснюючим виконанням потоків, як такої, що використовуються в Java-системах. У такій машині формується множина потоків, які зберігаються в черзі потоків. Планувальник стежить за чергою потоків і вибирає для виконання потік з найвищим пріоритетом. Звичайно потік, який виконується, має найвищий пріоритет, але такий потік може уриватися потоком з ще більш високим пріоритетом, що є звичайною практикою для багатопотокових систем з витісненням. У такому випадку стан перерваного потоку зберігається, і потік повторно активується, коли він знов вибирається для виконання.

У деяких випадках сам потік може містити так звану інструкцію "yield" ("повернути управління"), яка ініціює припинення планувальником виконання цього потоку і пошук планувальником в черзі потоків будь-яких інших потоків для виконання. Інструкція "yield" може застосовуватися в низькопріоритетних внутрішніх задачах, таких, як функція "складальник сміття", що виконується системою для видалення з пам'яті системи неживих об'єктів.

Ці аспекти системи показані на фіг 6. Створення потоку (4550) приводить до збереження потоку в черзі потоків 4551. Знову створений потік має стан "init" ("ініціалізація") (4552). Якщо немає потоків, що мають більш високий пріоритет, потік виконується інструкцією start() і буде мати стан "executable" ("виконання") (4553). Якщо в потоку виконується інструкція stop(), потік переходить в стан "dead" ("завершений") (4554). Потік може також перейти в цей стан, якщо завершується інструкцією run()/fini (завершити виконання). Якщо в потоку зустрічається інструкція yield(), або поза потоком виконується інструкція suspend() (припинити), потік припиняється і переходить в стан "non-executable" ("невиконання") (4556).

Далі з посиланням на фіг 7 буде пояснена взаємодія низькорівневої операційної системи 4100, менеджера подій 4273 і планувальника 4270. Необроблені події, про які сигналізує низькорівнева операційна система 4100, передаються через менеджер пристроїв 4264 в менеджер подій 4273. У переважній реалізації менеджером пристроїв 4264 і/або багатозадачною системою, що використовується в операційній системі 4100, може використовуватися деяке впорядкування отриманих подій по пріоритетах. Однак, як стане ясно, однією з переваг даної системи є той факт, що, на відміну від системи, описаної в PCT/EP97/02116, обробка подій організується всередині віртуальної машини.

4250, таким чином надаючи постачальнику програмного забезпечення проміжного рівня можливість отримання повного контролю над процесом обробки подій

У варіанті здійснення винаходу, що розглядається, події, що посилаються через менеджер пристроїв 4264, класифікуються по їх коду і по їх типу. Код ідентифікує параметри події, наприклад, у разі події, що генерується через пристрій дистанційного управління, асоційований з даним декодером, код може ідентифікувати натиснену кнопку. Тип ідентифікує джерело події, наприклад, пристрій дистанційного управління

При отриманні сигналу про подію менеджер подій 4273 використовує таблицю маршрутизації 4560 для визначення пріоритету події і визначення необхідного потоку, і вставляє відповідний об'єкт події 4564 в один або більш потоки 4561, розміщені в черзі потоків 4562, організований за принципом пріоритетності. У деякому потоку можуть зберігатися один або більш об'єкти подій 4564, як показано позицією 4563. Об'єкти подій 4564 зберігаються в потоку відповідно до ранжування їх пріоритетів. Об'єкти подій однакового пріоритету в потоку ранжуються за часом їх надходження (черга типу FIFO - "першим прибув - першим обслужений")

При отриманні події менеджер подій 4273 аналізує про її надходження планувальнику 4270, який потім переглядає чергу потоків для визначення того, чи має новий потік більш високий пріоритет, ніж потік, що виконується в даний момент. Якщо це так, то потік, що виконується в даний момент, припиняється, як описано вище, і виконується новий потік. Так реалізовується схема обробки потоків з витісненням

Таким чином, переважний варіант здійснення винаходу забезпечує ефективну обробку потоків в декодері, дозволяючи системі швидко реагувати на події, навіть у випадку, коли система обробляє подію, що поступила раніше. Тим самим долаються недоліки відомої системи масового обслуговування з одним пристроєм обробки

Хоч в описаному переважному варіанті здійснення винаходу застосована система з витісненням, в якій надходження події примушує менеджер подій сигналізувати планувальнику, з тим, щоб потік був перерваний, можливі і інші реалізації. Наприклад, в системі з квантуванням часу планувальник може періодично переривати виконання потоку для перевірки стану черги потоків. Як альтернатива планувальник може бути виконаний з можливістю переривання виконання потоку з метою перевірки стану черги потоків після обробки кожної інструкції в цьому потоку

Менеджер пам'яті

Зрозуміло, що у разі приймача/декодера управління пулом пам'яті в системі особливо важливе, оскільки об'єм пам'яті відносно обмежений в порівнянні, наприклад, з ПК або іншою системою, що використовує жорсткий диск. У наведеному нижче описі пул пам'яті відповідає області пам'яті в ОЗП приймача/декодера. Однак, як було згадано вище, відповідність між фізичною і логічною організацією пам'яті не є точною, і пул пам'яті, що описується нижче, може розташовуватися в інших пристроях фізичної пам'яті в приймачі/декодері,

таких, як флеш-пам'ять, ЕСПЗП тощо, або розділятися ними

На фіг. 8 показана організація наявної в системі пам'яті. Як видно, область пам'яті розподіляється між пулом дескрипторів 4600, пулом переміщуваних об'єктів 4610 і пулом непереміщуваних об'єктів 4620

Кожний об'єкт в пулі 4610 ідентифікується дескриптором, що зберігається в пулі 4600, і відповідає йому. Відповідність між дескриптором і відповідним об'єктом підтримується модулем управління пам'яттю 4274 (див. фіг. 4), який також управляє доступом до пулу. Всі виклики об'єктів цього пулу здійснюються через їх дескриптори. Межа між пулами 4600 і 4610 рухлива. Коли в пам'яті повинен зберігатися новий об'єкт, в пулі 4600 створюється дескриптор, що містить покажчик на адресу об'єкта в пулі 4610. У такому випадку список дескрипторів збільшується на один. Дескриптори в пулі організуються у вигляді списку, щоб зробити можливим виконання ущільнення менеджера пам'яті

Об'єкти розміщуються в пулі по мірі необхідності і з урахуванням пам'яті, що є в наявності. У випадку, коли запитується розміщення деякого об'єкта, що вимагає більше пам'яті в одному блоці, ніж доступно, необхідно ущільнити об'єкти, вже розміщені в пам'яті. Ущільнення об'єктів в пам'яті може виконуватися відповідно до будь-якого відомого алгоритму ущільнення, наприклад, алгоритмом *sort-compact*. У варіанті здійснення винаходу, що розглядається, застосовується алгоритм ущільнення *Mark Sweep*. Для ущільнення пам'яті об'єкти переміщуються всередині зони 4610, для більш щільного групування об'єктів в пам'яті і усунення наявності вільного простору між сусідніми об'єктами. Таким чином, вся вільна пам'ять об'єднується в один блок для розміщення нового об'єкта в пулі 4610

Як було згадано вище, модуль управління пам'яттю підтримує відповідність між дескрипторами в пулі 4600 і об'єктами в пулі 4610, і нові адреси об'єктів в пулі будуть оновлюватися у відповідних дескрипторах, для забезпечення можливості подальшого доступу

Хоч використання дескрипторів для доступу до деяких об'єктів дозволяє системі оптимізувати розподіл пам'яті в пулі, цей процес збільшує час, необхідний для доступу до таких об'єктів, оскільки завжди необхідно спочатку витягнути дескриптор для того, щоб визначити адресу об'єкта. У деяких ситуаціях, а також для об'єктів, відповідних деяким певним подіям, може вимагатися більш швидкий доступ

У такому випадку об'єкти можуть розміщуватися в пулі непереміщуваних об'єктів 4620. Адреси таких об'єктів фіксовані всередині пулу. Таким чином, немає необхідності в створенні дескриптора, об'єкти використовуються системою напряму, і тим самим спрощується процедура доступу до цих спеціальних об'єктів. Знову ж, як і межа між пулами 4600 і 4610, межа між пулами 4620 і 4610 буде зміщатися в залежності від інформації, що зберігається в пулі 4620

У випадку, якщо, наприклад, непереміщуваний об'єкт повинен бути розміщений в пулі 4620, і є в

наявності нестача пам'яті через конфігурацію розміщення переміщуваних об'єктів в пулі 4610, може виконуватися ущільнення переміщуваних об'єктів, як описано вище. Після того, як об'єкти в цьому пулі будуть реорганізовані для звільнення максимальної кількості пам'яті, розміщення згаданого непереміщуваного блоку в пулі 4620 може виявитися можливим.

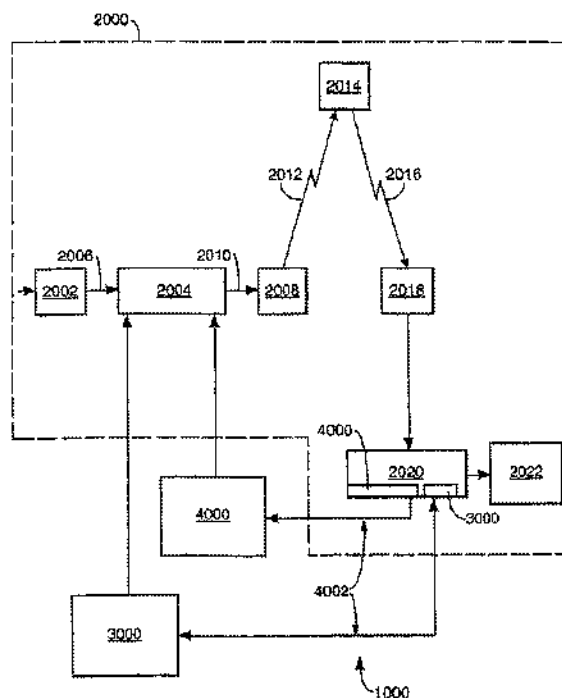
Вибір того, які об'єкти переміщувати, а які непереміщувати, відбувається по розсуду розробника. Наприклад, об'єкти, необхідні системі, можуть вибиратися як непереміщувані в зв'язку з важливістю цих об'єктів, тоді як об'єкти високорівневих прикладних програм можуть бути переміщуваними. У деяких випадках переміщувані об'єкти тимчасово фіксуються на місці і вважаються непереміщуваними.

Для видалення непотрібних об'єктів з пулу пам'яті система може також включати в себе так

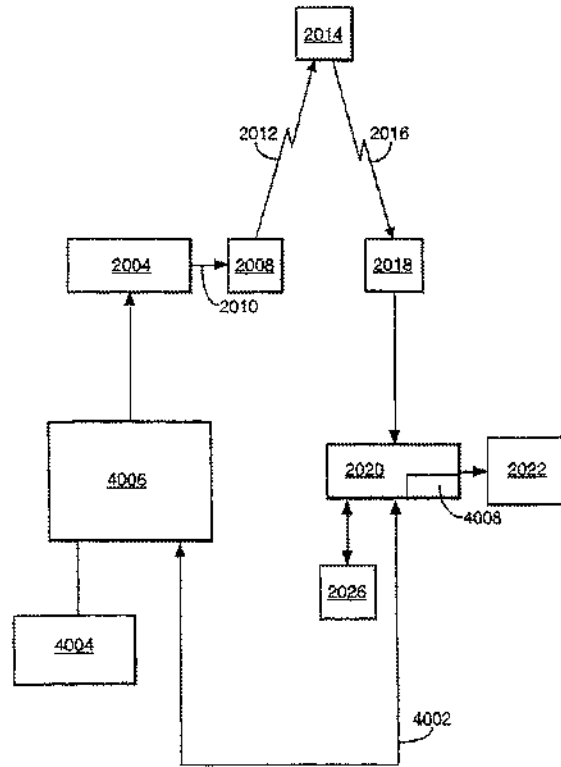
зване "збирання сміття". Під цим розуміють створення спеціального потоку "збирання сміття" з найменшим пріоритетом, до якого звертається планувальник у випадку, якщо в черзі потоків не міститься інших потоків. При виконанні цього потоку всі переміщувані об'єкти, які розміщені в пулі і на які немає посилань в даний час, будуть видалені. Потік "збирання сміття" може також виконувати ущільнення всіх інших переміщуваних об'єктів, як описано вище.

Створення потоку "збирання сміття" відоме з інших багатопотокових систем, що використовуються в інших прикладних програмах поза галуззю систем цифрового телебачення, і не буде розглядатися далі більш детально.

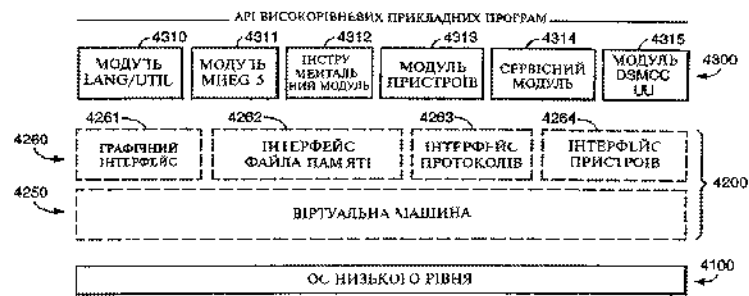
Однак зрозуміло, що використання процедури "збирання сміття" в поєднанні з іншими методами управління пам'яттю, описаними вище, надає особливі переваги в контексті, що розглядається.



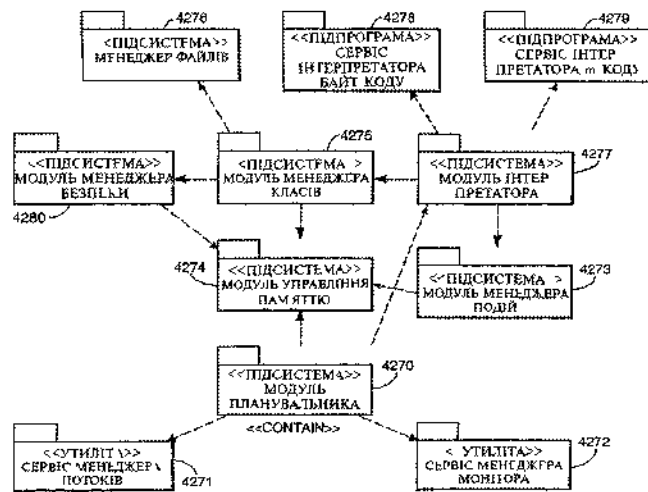
Фиг. 1



Фиг. 2



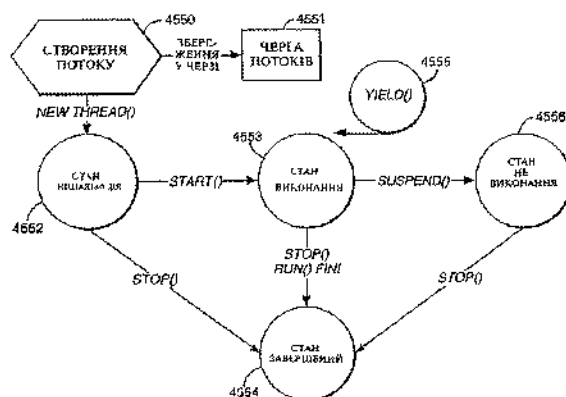
Фиг. 3



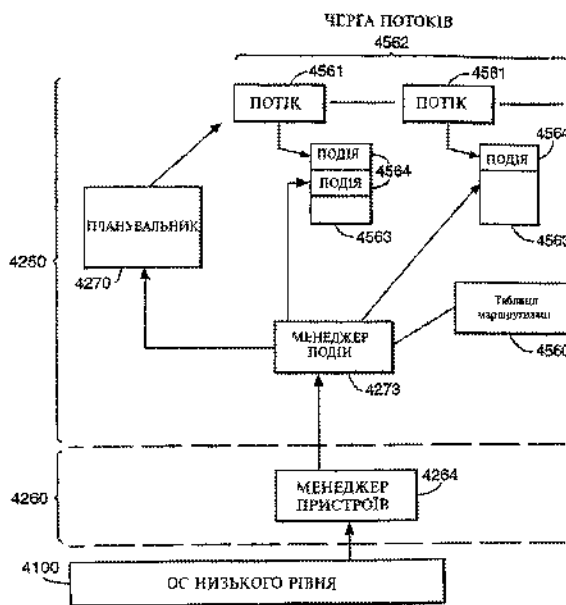
Фиг. 4



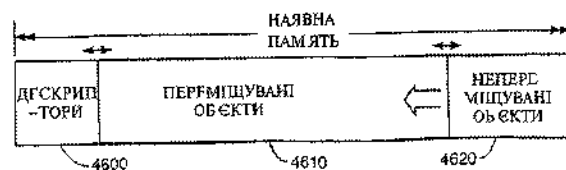
Фіг. 5



Фіг. 6



Фіг. 7



Фіг. 8

