УДК 004.272.2

О.Б. Посвятовська
викладач,
Галицький коледж
імені В'ячеслава
Чорновола

e-mail:o.posv@gi.edu.ua

ОСОБЛИВОСТІ ГЕТЕРОГЕННОЇ АРХІТЕКТУРИ ДЛЯ АПАРАТНИХ РІШЕНЬ

О.Б. Посвятовська. **Особливості гетерогенної архітектури для апаратних рішень**. За останні десятиліття гетерогенні обчислювальні системи стають все більш привабливими. У порівнянні з традиційними, вони забезпечують високу пікову продуктивність нарівні з енергоефективністю та більш низькою вартістю. Зі збільшенням паралелізму в області високопродуктивних обчислень актуальною є необхідність у правильному розумінні особливостей та переваг гетерогенних архітектур.

O.B. Posvyatovskaya.. Features of a Heterogeneous Architecture for Hardware Solutions. In recent decades, heterogeneous computing systems have become increasingly attractive. Compared to traditional ones, they provide high peak performance along with energy efficiency and lower cost. With the increase in parallelism in the field of high-performance computing, the need for a proper understanding of the features and advantages of heterogeneous architectures is urgent.

Вступ. На сучасному етапі в галузі розробки апаратного забезпечення виділяється декілька принципів, яких намагаються дотримуватись.

Перший з них – зниження енергозалежності, спричинений, зокрема, тим, що популярності набувають мобільні пристрої, тому серед користувачів важливим параметром є час автономної роботи.

Другим принципом є збільшення потужності апаратної складової, оскільки постійно з'являються нові функціональні сервіси. Наприклад, керування пристроями за допомогою голосу та жестів, підтримка високих стандартів щодо якості аудіо та відезображень, а особливо використання хмарних сервісів.

Ці фактори сприяють більшому розвитку гетерогенних систем, які вже зараз знаходять застосування в мобільних, портативних, стаціонарних, серверних і суперкомп'ютерних пристроях, пе-

ревершуючи аналоги за рівнем паралелізму, енергоспоживанню і продуктивності.

Проте, варто зауважити, що застосування гетерогенної архітектури хоча підвищує ефективність відповідної апаратної складової, але часом одночасно зменшує її енергоефективність та збільшує габарити. Також, збільшення енерговитрат процесорів вимагає додаткових енерговитрат на їх охолодження, що має негативні наслідки при функціонуванні великих масивів процесорів, наприклад, дата-центрів.

Матеріал і результати дослідження. Гетерогенні архітектури мають різнорівневе застосування – від кластерних суперкомп'ютерів і високопродуктивних серверів до малопотужних пристроїв, що вбудовуються, включаючи мобільні телефони та планшети.

До типових прикладів гетерогенних архітектур можна віднести наступні:

– процесори з цифровими сигнальними процесорами (DSP);

– процесори з графічним процесором загального призначення (GPGPU). В цьому випадку графічний процесор може більш ефективно використовуватися для візуалізації предметів в машинній графіці, оскільки кінцевий користувач може запрограмувати його на виконання алгоритмічної обробки;

– центральні процесори (CPU), цифрові сигнальні процесори (DSP) і графічні процесори загального призначення (GPGPU);

– процесори з програмованими користувачем вентильними матрицями (FPGA);

– IP-блоки для конкретного додатка, реалізовані на базі будь-якого варіанту з перерахованих вище. .[1]

Всі ці приклади базової архітектури можуть існувати в дискретних компонентах або бути інтегровані в систему-на-кристалі (англ. System-on-chip, SoC). Залежно від пристрою, можлива реалізація додаткових IP-блоків, що призначені для використання в додатках, які відносяться до таких машинним збірок, як пристрої DSP з ШІМ-модулем (англ. Pulse Width Modulation module, PWM module).

Зокрема, останнім часом популярною гетерогенної архітектурою стали процесори з FPGA. Вони надають користувачеві від-

разу три елементи обробки, оскільки постачальники пристроїв FPGA інтегрували в них потужні блоки DSP. FPGA якраз призначені для забезпечення субмікросекундного рівня обробки даних, апаратного детермінізму і високої надійності. Крім того, вони характеризуються ще і високою гнучкістю, можливістю виконання повної настройки під конкретні індивідуальні вимоги споживача,

У деяких мікросхемах на одному кристалі об'єднуються графічний процесор і кілька ядер загального призначення. Аналогічно цьому багато однокристальних систем на додаток до одного або декількох процесорів спеціального призначення містять ядра загального призначення [2].

Системи, що об'єднують кілька різнорідних процесорів на одному кристалі, мають загальну назву гетерогенних мультиядерних процесорів. Як приклад таких процесорів може послужити лінійка мережевих процесорів IXP, спочатку представлена компанією Intel в 2000 році і регулярно оновлювана з використанням самих передових технологій. Мережеві процесори зазвичай містять одне керуюче ядро загального призначення (наприклад, ARM-процесор, на якому запущена Linux) і багато десятків вузькоспеціалізованих потокових процесорів, добре проявляють себе в обробці мережевих пакетів і більше ні в чому іншому. Вони широко використовуються в мережевому обладнанні, такому як маршрутизатори і брандмауери. Для маршрутизації мережевих пакетів, ймовірно, не знадобляться великі обсяги обчислень з плаваючою точкою, тому в більшості моделей потокових процесорів взагалі відсутній блок таких обчислень. У той же час високошвидкісної мережевої обмін даними сильно залежить від швидкого доступу до пам'яті (для читання даних пакета), і у потокових процесорів є спеціальне обладнання для здійснення такого доступу.

Зрозуміло, що в попередніх прикладах йшлося про гетерогенні системи. Потокові процесори і керуючі процесори в IXP абсолютно різні за будовою, з різними наборами інструкцій. Те ж саме можна сказати про ядрах графічних процесорів і ядрах загального призначення. Але гетерогенність можна впроваджувати і за підтримки однакового набору інструкцій. Наприклад, у центрального процесора може бути невелика кількість «великих» ядер з великими конвеєрами і, можливо, високими тактовими частотами і велика кількість «малих» ядер, які просто мають меншу потуж-

ність і, можливо, працюють на більш низьких тактових частотах. Потужні ядра потрібні для запуску коду, що вимагає швидкої послідовної обробки, а малі ядра знадобляться для задач, які можуть бути ефективно виконані в паралельному режимі. Прикладом гетерогенної архітектури, яка відповідає цьому напрямку, може послужити сімейство ARM-процесорів big.LITTLE [3].

В загальному перспективність гетерогенних архітектур пов'язують з успішною розробкою спеціалізованих процесорів, орієнтованих тільки на певний клас задач, основним з яких стали графічні додатки. Проте, у наш час зростання потреб в більш якісній обробці зображень і розширення технологічних можливостей виробництва напівпровідникових кристалів привели до того, що графічні процесори стали швидко і ґрунтовно вдосконалюватися, чим дали гетерогенним системам новий напрямок розвитку у спільному використанні універсальних (CPU) і графічних (GPU) процесорів в складі однієї обчислювальної системи при вирішенні широкого класу задач.

На даний час гетерогенні платформи мають наступний ряд архітектурних особливостей.

По-перше, орієнтовані на високу продуктивність ядра. Наприклад, GPU можуть бути вбудовані в систему як в інтегральному, так і в дискретному вигляді. Система може також мати гібридну конфігурацію, в якій порівняно малопродуктивний GPU з низьким енергоспоживанням інтегрований в чіпсет, а високопродуктивний GPU виконаний як дискретне пристрій. Нарешті, платформа може містити кілька дискретних карт з вбудованим GPU. Конфігурація платформи визначає багато параметрів, наприклад пропускну спроможність і затримки між різними типами процесорів, а також можливість підтримки когерентності кеша і т.д.

По-друге, скалярні і високопродуктивні ядра можуть мати різні операційні системи (ОС). Наприклад, процесор Larrabee компанії Intel, призначений для виконання складних графічних завдань, може містити власне ядро ОС. Це означає, що схеми переміщення віртуальної пам'яті (перетворення віртуальних адрес в фізичні) можуть бути різними у різних видів ядер. Одні і ті ж віртуальні адреси можуть бути одночасно розподілені за двома різними фізичним адресами, один з яких розташований в пам'яті CPU, а інший – в пам'яті процесора Larrabee. Це також означає, що

системне середовище (завантажувачі, компоновщики і ін.) може бути відмінним в різних ядрах.

По-третє, скалярні і високопродуктивні ядра можуть мати різну структуру системи команд (ISA) і, отже, один і той же код не можна запускати на ядрах різних типів [4].

Сучасні GPU містять сотні арифметичних пристроїв і мають високу швидкість доступу до внутрішніх і зовнішніх модулів пам'яті, що дозволяє їм паралельно обробляти величезні масиви даних і досягати продуктивності в кілька TFlops. Всі ці показники, що перевершують майже на порядок аналогічні характеристики CPU, а також поширення паралельних обчислень і складність масштабування багатоядерних і багатопроцесорних систем дають підстави для використання графічного процесора при вирішеніі обчислювальних задач, що володіють високим рівнем паралелізму. Зокрема, це зумовило виникнення напрямку – обробки завдань загального призначення (general purpose, GP) на графічних процесорах (GPU), що позначається абревіатурою GPGPU(General-Purpose computing on Graphic Processing Units)[1]. Реалізовувати GPGPU, обчислення загального призначення на графічному прискорювачі, можна було з використанням відкритого Стадарт і OpenCL, або спрощеного діалекту мови С.

Сьогодні найбільш поширеними багатоядерними процесорами є графічні процесори (GPU), які можна знайти практично в будь-якій комп’ютерній системі, яка не є вбудованою і має монітор. Графічний процесор має виділену пам'ять і тисячі крихітних ядер. У порівнянні з процесорами загального призначення транзисторний бюджет графічних процесорів витрачається в основному на схеми, що виробляють обчислення, і в меншій мірі на кеші і логіку управління. Вони оптимальні для множини невеликих паралельних обчислень, подібних до побудови багатокутників в графічних додатках. Для звичайних задач вони мало підходять. Їх також важко програмувати. Суттєвою різницею між програмуванням графічних процесорів і процесорів загального призначення є те, що графічні процесори по суті є машинами, що працюють за принципом «одна інструкція, множина даних», а це означає, що велика кількість ядер виконує одну і ту ж інструкцію над різними ділянками даних (архітектура SIMD). Ця модель програмування ефективна для паралелізму даних, але не завжди зручна для інших

стилів програмування (таких, як паралелізм завдань). Хоча графічні процесори можуть стати в нагоді для операційних систем (наприклад, при шифруванні або обробці мережевого трафіку), навряд чи на них буде працювати основна частина самої операційної системи.

Графічними процесорами (GPU) все частіше обробляються і інші обчислювальні завдання, особливо ті, що потребують великих обчислювальних потужностей, наприклад у наукових обчисленнях. На жаль, ефективне програмування графічних процесорів є досить складним завданням і вимагає застосування спеціальних мов програмування, таких як OpenGL або CUDA, право власності на які належить компанії NVIDIA. На даний час продукти цієї компанії найбільш популярні для вирішення завдань машинного навчання.

Машинне навчання – один із напрямів штучного інтелекту, для досягнення кінцевої мети в якому застосовується «навчання» комп'ютерних систем на основі рішення множини подібних завдань. Алгоритми машинного навчання засновані на засобах математичної статистики, теорії ймовірностей, теорії графів, чисельних методах. Їх продуктивність підвищується в міру обробки наростаючих обсягів даних. Для складних завдань, що вимагають аналітичних обчислень, які ґрунтуються на сучасних уявленнях про можливості мозку людини, використовуються нейронні мережі, що надають, в тому числі, можливість вирішення в IT-системах проблеми ефективного паралелізму.

Найважливіші компоненти нейронних мереж – графічні процесори, що забезпечують високу продуктивність при виконанні таких операцій, як множення великих матриць, згортка функцій і ряд інших, де дуже важливий масовий паралелізм.

Варто звернути увагу на те, що сучасні розробки в галузі розвитку штучного інтелекту вимагають адекватного розпізнавання людської мови, міміки, жестів, що, в свою чергу, вимагає збільшення як прямої продуктивності, так і оптимізації виконуваних операцій по декодуванню аудіо- та відео- потоку.

У сегменті дискретних графічних процесорів єдиним великим конкурентом NVIDIA залишається компанія AMD

Організація HSA Foundation, що заснована компаніями AMD, ARM, Samsung, Qualcomm, Texas Instruments, Imagination та

MediaTek, представила специфікацію HSA 1.0 (Heterogenous System Architecture), визначальну архітектуру, набір runtime-компонентів та програмні інтерфейси гетерогенних обчислювальних систем. Архітектура HSA визначає роботу обладнання. Програмні інтерфейси призначені для розробників програмного забезпечення, інструментаріїв та компіляторів. Специфікація на runtime визначає, як додатки повинні взаємодіяти з платформами HSA.

Компанія AMD займає особливе становище на ринку, так як випускає і процесори на базі на базі архітектури x86, і власні графічні процесори. Саме тому дана компанія активно впроваджує концепцію гетерогенної системної архітектури HSA, специфікації якої дозволяють організувати взаємодію центральних і графічних процесорів, що працюють на одній шині зі спільними завданнями.

Можливості архітектури HSA використовуються для налагодження спільної роботи CPU, GPU і різних DSP-процесорів, та організації гібридних обчисленнь, в яких відповідний обчислювальний пристрій обирається в прозорому режимі в залежності від задачі. HSA позиціонується як єдина оптимізована платформа, на базі якої може функціонувати OpenCL та OpenMP. Особливістю HSA є те, що CPU та GPU мають доступ до спільних областей пам'яті, що спрощує організацію роботи гібридних додатків та мінімізує кількість операцій з копіювання пам’яті. Наприклад, GPU та CPU можуть напряму працювати зі спільним блоком пам'яті додатку за аналогією з роботою з пам'яттю в багатопоточних програмах. Специфікації охоплюють також методи відображення віртуальної пам'яті, забезпечення когерентності пам'яті та передачі повідомлень.

В основі HSA лежить спеціальна проміжна мова HSAIL (Heterogeneous System Architecture Intermediate Language) та фіналізатор, що забезпечує її трансляцію в машинний код, специфічний для різного обладнання. Фіналізатор може використовуватися як для статичної компіляції HSAIL під час зборки додатку, так і в процесі виконання або завантаження додатку. Компоненти для використання HSA реалізовані для різних високорівневих мов програмування, в тому числі для C, C++, Java та Python.

Висновки. Поки що не існує єдиного рішення в області гетерогенних систем, яке б підходило до всіх завдань і задовольняло всі потреби. Гетерогенні обчислення за останні кілька років перет-

ворилися в окрему наукову область, а гетерогенні архітектури можна знайти в багатьох областях обчислювальної техніки - від кластерних суперкомп'ютерів і високопродуктивних серверів до малопотужних пристроїв, що вбудовуються, включаючи мобільні телефони та планшети. Основною проблемою для ефективного використання будь-гетерогенної архітектури є складність її проектування в процесі розробки як апаратного, так і програмного забезпечення.

Перелік джерел посилання:

1. Колпаков А.А. Повышение производительности гетерогенных компьютерных систем обработки данных. Москва ; Берлин: Директ-Медиа, 2019. 121 с.

2. Светличный А.Н. Краткий обзор достижений в области гетерогенных вычислений // Молодой ученый. 2016. №1. С. 213-216.

3. Таненбаум ЭС., Бос Х. Современные операционные системы. Питер, 2015, (4-е изд), 1120 с.

4. Гетерогенная архитектура для CPU, GPU и DSP [Электронный ресурс] URL: http://www.osp.ru/os/2013/08/13037850

5. Начало новой эпохи? Обзор архитектуры HSA [Электронный ресурс] URL: http://www.ferra.ru/ru/system/review/AMD-HSA-architecture/#.VvGrokcXWC4

6. AMD OpenCL™ Zone – Accelerate Your Applications [Electronic resource] URL: http://developer.amd.com/tools-and-sdks/opencl-zone

7. AMD oficial web site [Electronic resource] URL: http://www.amd.com/en-gb (дата звернення: 01.12.2019)

8. Heterogeneous System Architecture или о встрече CPU и GPU [Electronic resource] URL:https://geektimes.ru/company/amd/blog/266930

9. HSA Foundation [Electronic resource] URL: http://www.hsafoundation.com

10. OpenCL [Electronic resource] URL: https://ru.wikipedia.org/wiki/OpenCL

Надійшла до редакції 13.12.2019