# ДОДАТОК А

Графічний матеріал кваліфікаційної роботи

Харківський національний університет радіоелектроніки

факультет КІУ
кафедра ЕОМ

# Кваліфікаційна робота

**на тему:**

# Методи прискорення периферійних обчислень на основі FPGA

Керівник проекту:
проф. Коваленко А.А.

Розробив:
ст. гр. СПм-21-2
Ільяшов О.А.

Харків 2023

## Архітектура сучасних периферійних обчислень

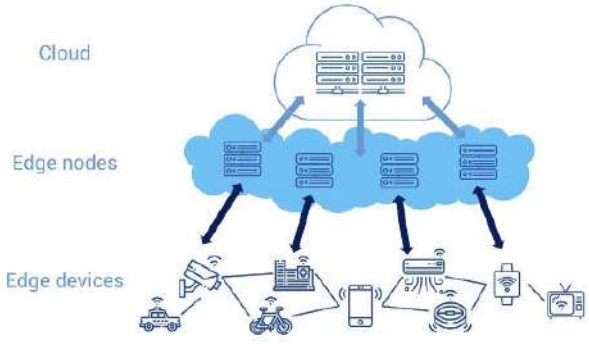



Рисунок 1 - Архітектура периферійних обчислень

Таблиця 1 - Допустимі затримки для різних послуг

| Тип послуги | Прийнятна затримка |
|---|---|
| Онлайн ігри | < 1000 мс |
| Аудіо послуги | < 450 мс |
| Голос через IP | 200 мс |
| Відеосервіси | < 150 мс |
| Відео через IP | 70 мс |
| Дані | < 400 мс |
| Передача медичних даних | 100 - 400 мс |
| Телехірургія | 300 мс |
| Електрокардіограма | 1000 мс |
| Послуги не в реальному часі | Кілька секунд |

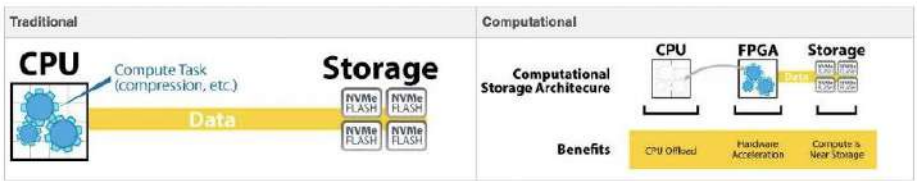



Рисунок 2 – Місце FPGA у апаратному прискоренні обчислень

# Мета та задачі роботи

- У якості **об'єкту** дослідження в нашій роботі виступає обчислювальна система на базі FPGA прискорювачів
- **Предметом** дослідження є методи та інструменти для апаратного прискорення периферійних обчислень з використанням FPGA .
- **Метою** роботи є вивчення та впровадження методів прискорення периферійних обчислень з використанням FPGA

Мета кваліфікаційної роботи досягається вирішенням наступних **задач**, а саме треба:

- провести дослідження придатності FPGA для прискорення периферійних робочих навантажень;
- розробити модифікований метод прискорення периферійних робочих навантажень з використанням FPGA;
- розробити тестові сценарії на OpenCL для досягнення прискорення на архітектурі SoC-FPGA;
- запропонувати структуру, яка використовує архітектуру з кількома чергами в програмному стеку для мінімізації конфліктів між різними програмами для доступу до різних прискорювачів FPGA

3

# Прискорення на основі FPGA


Рисунок 1 - FPGA, що використовуються для прискорення у хмарі

| | CPU | GPU | FPGA | ASIC |
|---|---|---|---|---|
| Overview | Sequential processor for general-purpose applications | Originally designed for graphics; now used in a wide range of computationally intensive applications | Flexible collection of logic elements and IP blocks that can be configured and changed in the field | Custom integrated circuit optimized for the end application |
| Processing | Single- and multi-core processor, plus specialized blocks: FPU, etc. | Thousands of identical processor cores | Configured for application | Application-specific: may include third-party IP cores |
| Programming | huge range of high-level languages (e.g., C, C++, Python, Java, Fortran); assembly language | OpenCL & Nvidia's CUDA API allow general-purpose programming (e.g., C, C++, Python, Java, Fortran) | Traditionally HDL (Verilog, VHDL); newer systems include C/C++ via OpenCL & SDAccel | ... |
| Strengths | Versatility, multitasking, ease of programming | Massive processing power for target applications: video processing, image analysis, signal processing; suitable for data parallel tasks | Configurable for specific application; configuration can be changed after installation; high performance per watt; accommodates massively parallel operation; | Custom-designed for application with optimum combination of performance and power consumption |
| Weaknesses | OS capability adds high overhead; optimized for sequential processing with limited parallelism | High power consumption, not suited to some algorithms; problems must be reformulated to take advantage of parallelism; Sequential execution | Difficult to program; second-longest development time; poor performance for sequential operations; not good for floating-point operations | Longest development time; high cost; cannot be changed without redesigning the silicon |

Рисунок 1.2 - Порівняння різних аспектів апаратних прискорювачів

4

# Огляд архітектури FPGA

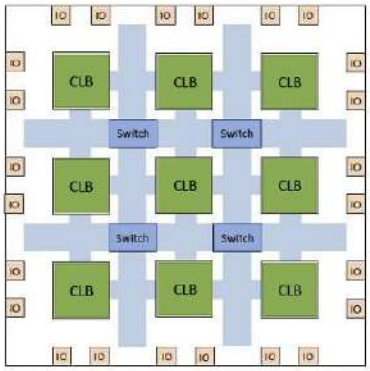



Рисунок 1 - Внутрішня архітектура типової FPGA

У доступних на даний момент системах і платформах існує дві основні платформи для взаємодії FPGA і багатоядерних процесорів:

- з використанням плат FPGA через PCIe.
- з використанням FPGA системи на кристалі (SoC).

5

# Апаратне прискорення на основі FPGA

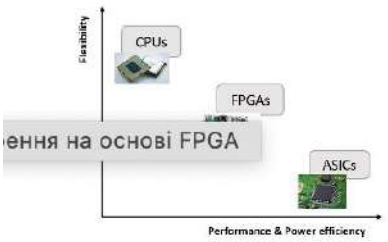



Рисунок 1 - Компроміс гнучкості та енергоефективності для CPU, FPGA та ASIC

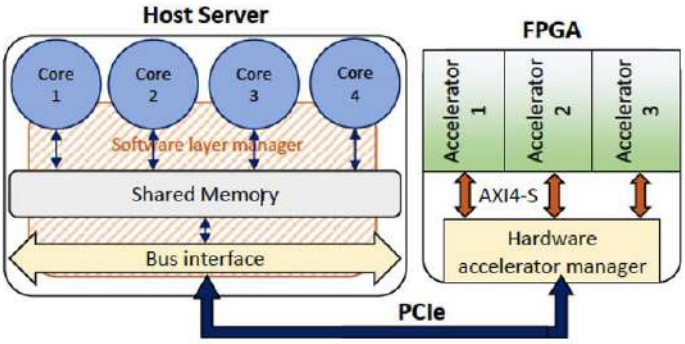



Рисунок 2 - Кілька прискорювачів FPGA використовуються для прискорення в хмарах і центрах обробки даних

6

# Апаратне прискорення обчислень на FPGA з використанням OPENCL

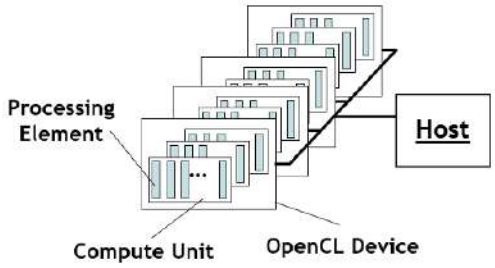



Рисунок 1 - Модель платформи OpenCL

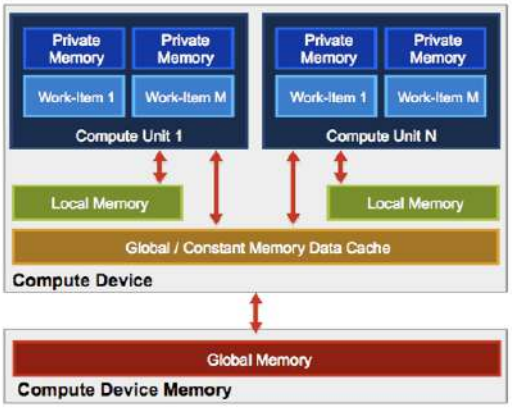



Рисунок 2 - Модель пам'яті OpenCL

7

# Експериментальні дослідження. Схема установки

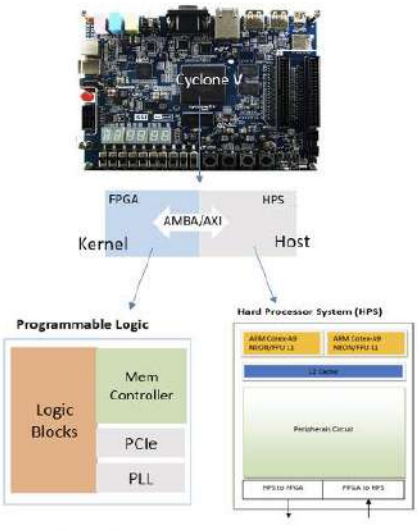



Рисунок 1 – Експериментальне системне обладнання на базі Terasic DE1-SoC з Altera Cyclone V

8

# Експериментальні дослідження. Програмування системи

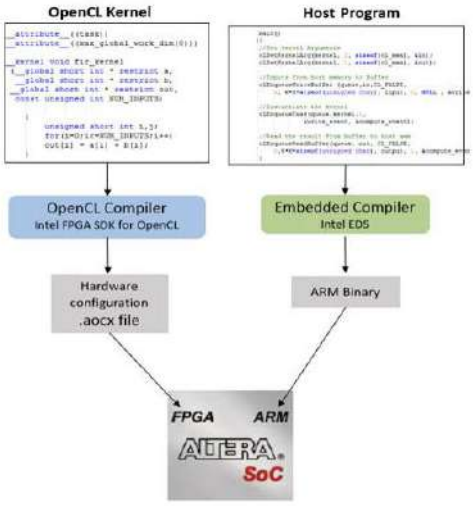



Рисунок 1 - Огляд налаштування системи з кодом OpenCL

9

# Хост-програма та процес розробки ядра пристрою

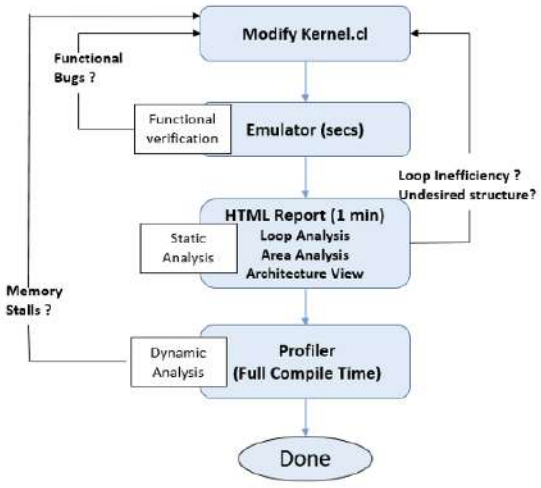



Рисунок 1 - Метод розробки функції ядра

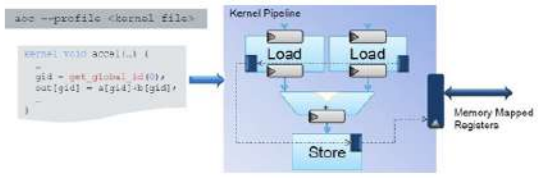



Рисунок 2 - Аналіз за допомогою Profiler, ядро, інтегроване з Flip-flops при завантаженні та збереженні транзакцій

10

# Оптимізація дизайну для прискорення обчислень

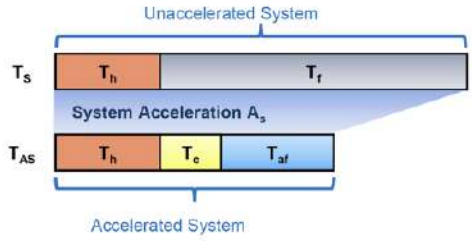



Рисунок 1 - Часова шкала неприскореної системи проти прискореної системи

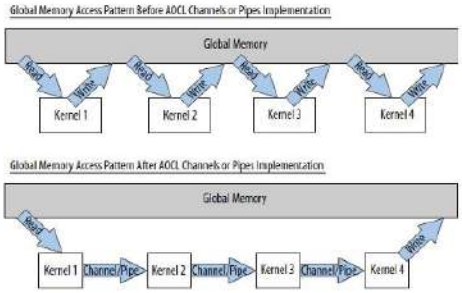



Рисунок 2 - Зв'язок між ядрами: використання глобальної пам'яті проти каналів

Таблиця 1 - Позначення часу

| Позначення часу | Опис |
|---|---|
| $T_S$ | Загальний час, витрачений на систему без прискорення |
| $T_h$ | Час, витрачений на неприскорену систему для головної частини |
| $T_f$ | Час, витрачений у неприскореній системі для функції без прискорення |
| $T_{AS}$ | Загальний час, витрачений на прискорену систему |
| $T_C$ | Час, витрачений на спілкування між хостом і прискорювачем |
| $T_{AF}$ | Час, витрачений на прискорену систему для прискореної функції |
| $A_S$ | Системне прискорення = $T_s/T_{AS}$ |

11

# Архітектура оптимізації

1. **Ядра одного робочого елемента (Single Work Item Kernels).** Уся програма ядра виконується як один потік. У цих конструкціях пропускна здатність досягається за допомогою методу під назвою Loop Pipelining (рисунок 1).
2. **Ядра NDRange.** У ядрі NDRange ефективність обробки даних досягається шляхом впровадження паралелізму рівня даних. Апаратне забезпечення копіюється, щоб забезпечити можливість паралельного виконання. Ядро NDRange може мати кілька обчислювальних блоків (CU) і кілька блоків обробки даних з однією інструкцією (SIMD), які одночасно працюють над різними робочими групами або паралельними робочими елементами.

*Exec time = ((Num iterations * II) + (Loop latency)) * Time period*

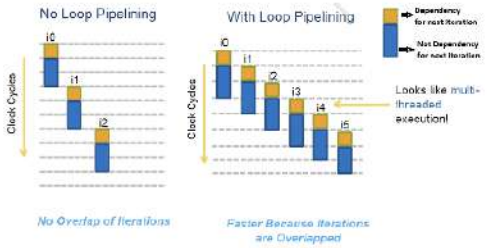



Рисунок 1 - Ітерації конвеєрної обробки без циклу проти конвеєрної обробки циклу

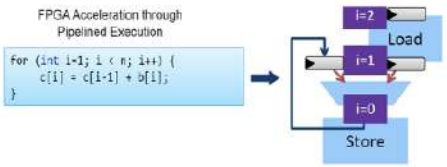



Рисунок 2 - Реалізація одного робочого елемента, що показує зворотний зв'язок даних для обробки залежності

12

# Атрибути оптимізації та результати прискорення

Таблиця 1 - Опис системи бенчмарків

| Еталон | Ядро | Pipeline статус | II | $F_{макс}$ (МГц) | Логіка (%) | Найвище прискорення |
|---|---|---|---|---|---|---|
| Sobel | Одиничне завдання | Так | 2 | 136 | 20 | 7.3 |
| FIR | Одиничне завдання | Так | 1 | 168 | 20 | 0,95 |
| ADPCM | Одиничне завдання | Так | 40 | 158 | 20 | 2.6 |
| Децимація | Одиничне завдання | Так | 1 | 129 | 81 | 2.8 |
| Інтерполяція | Одиничне завдання | Так | 1 | 125 | 28 | 2.8 |
| AES | NDRage CU=2, SIMD=2 | Немає | N/A | 135 | 84 | 6.9 |

- **Sobel**: Sobel - це фільтр зображень 3X3 із розпізнаванням країв, розроблений для 8-розрядних зображень .bmp.
- **FIR**: Фільтр кінцевої імпульсної характеристики на 10 натискань.
- **ADPCM** - адаптивний кодер диференційної імпульсно-кодової модуляції для 16-бітових зразків імпульсно-кодової модуляції (PCM).
- **Децимація** - 5-ступеневий децимаційний фільтр. Він складається з 5 каскадних FIR-фільтрів, де вихід одного фільтра подається на наступний фільтр.
- **Інтерполяція** - 4-ступеневий інтерполяційний фільтр.
- **Шифрування** AES: 16-бітне шифрування даних і 128-бітне шифрування AES.

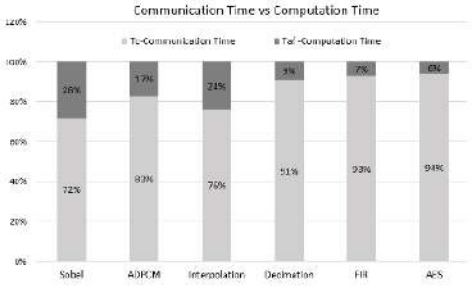



Рисунок 1 - Комунікація проти часу обчислень у прискореній системі (ARM+FPGA)

Таблиця 1 показує системні параметри для різних тестів. Нижче наведені терміни, визначені контрольними тестами:
- час роботи (arm_time) = час обчислення програми лише на процесорі ARM,
- час запису (write_time) = час, витрачений на передачу вхідних даних від хоста до буфера вхідних даних,
- час читання (read_time) = час, витрачений на передачу вихідних даних із вихідного буфера на хост,
- api_time = Час передачі даних між хостом (ARM) та ядром (FPGA),
- comp time = час обчислення програми на ядрі FPGA,
- hw time = api time + comp time,
- acceleration = arm time/hw time,
- api% = (api time / hw time) *100.

13

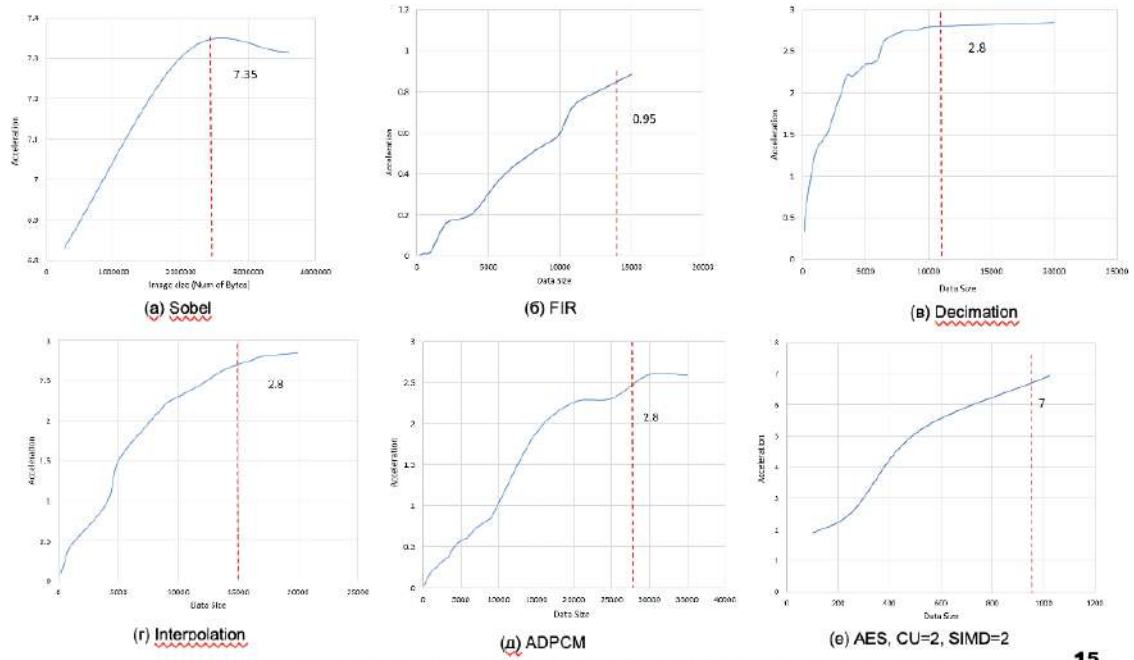
# Результати експериментів (1)

Таблиця 1 - Прискорення AES шляхом зміни кількості атрибутів CU та SIMD для різних розмірів даних, робочих груп = N і робочих елементів = (Кількість входів)/N, де N=2,4

| Робочі групи | Обчислені одиниці | SIMD одиниць | 256 входів | 512 входів | 1024 Входів |
|---|---|---|---|---|---|
| 2 | 1 | 1 | 2.5 | 4.6 | 6.6 |
| 2 | 1 | 2 | 2.7 | 5.4 | 6.6 |
| 2 | 1 | 4 | 2.4 | 5.4 | 6.9 |
| 2 | 2 | 1 | 4.0 | 4.6 | 6.9 |
| 2 | 2 | 2 | 2.6 | 5.2 | 6.9 |
| | (а) Кількість робочих груп = 2 | | | | |
| Робочі групи | Обчислені одиниці | SIMD одиниць | 256 Входів | 512 Входів | 1024 Входів |
| 4 | 1 | 1 | 0,7 | 2.6 | 6.2 |
| 4 | 1 | 2 | 1.0 | 2.7 | 4.6 |
| 4 | 1 | 4 | 3.8 | 5.5 | 5.7 |
| 4 | 2 | 1 | 1.3 | 2.1 | 5.7 |
| 4 | 2 | 2 | 1.2 | 2.0 | 6.5 |
| | (б) Кількість робочих груп = 4 | | | | |

14

# Результати експериментів (2)




(а) Sobel (б) FIR (в) Decimation

(г) Interpolation (д) ADPCM (е) AES, CU=2, SIMD=2

15

Рисунок 1 - Графіки прискорення та розміру вхідних даних для набору тестів

# Висновки

**У кваліфікаційній магістерській роботі нами:**

- проведені дослідження придатності FPGA для прискорення периферійних робочих навантажень;
- розроблений модифікований метод прискорення периферійних робочих навантажень з використанням FPGA;
- розроблені тестові сценарії на OpenCL для досягнення прискорення на архітектурі SoC-FPGA;
- запропоновано структуру, яка використовує архітектуру з кількома чергами в програмному стеку для мінімізації конфліктів між різними програмами для доступу до різних прискорювачів FPGA

16