УДК 004.318

ПРОЕКТУВАННЯ КОНВЕЄРНОГО ПРОЦЕСОРА RISC-V АРХІТЕКТУРИ З АПАРАТНИМ СПІВПРОЦЕСОРОМ ЦИФРОВОЇ ОБРОБКИ СИГНАЛІВ

Ваврук Є. Я. – канд. техн. наук, доцент, доцент кафедри електронних обчислювальних машин Національного Університету «Львівська політехніка», Львів, Україна.

Махров В. В. – студент кафедри комп'ютерних систем та мереж ДВНЗ «Ужгородський національний університет», Ужгород, Україна.

Гедеон Г. О. – асистент кафедри комп'ютерних систем та мереж ДВНЗ «Ужгородський національний університет», Ужгород, Україна.

АНОТАЦІЯ

Актуальність. Цифрова обробка сигналів використовується в багатьох сферах науки, техніки та діяльності людини. Одним із шляхів реалізації алгоритмів цифрової обробки сигналів є розробка співпроцесорів, як складової частини відомих архітектур. У випадку розробки конвеєрного пристрою такий підхід дозволить використовувати програмні та апаратні засоби відповідної архітектури, забезпечити швидше виконання алгоритмів обробки сигналів, скоротити кількість тактів та кількість звернень до пам'яті.

Мета роботи – проектування та дослідження характеристик конвеєрного процесора архітектури RISC-V з співпроцесором цифрової обробки сигналів, що виконує швидке перетворення Фур'є.

Метод. Аналіз технічної літератури та існуючих рішень дозволяє оцінити переваги і недоліки сучасних розробок та на основі них сформувати актуальність обраної теми. Побудова моделей і дані симуляцій дозволяють перевірити працездатність моделі, знайти слабкі ланки компонентів та поліпшити параметри моделі.

Результати. Спроектовано конвеєрний процесор архітектури RISC-V, який виконує базовий набір інструкцій. Проаналізовано час виконання простої асемблерної програми на конвеєрному та однотактному процесорах. Згідно результатів, тестова програма на конвеєрному процесорі виконується за 29 тактів, тоді як на однотактному – за 60 тактів. Розроблено структуру співпроцесора виконання алгоритму швидкого перетворення Фур'є та набір процесорних інструкцій, які дозволяють працювати із співпроцесором. Кількість тактів виконання співпроцесором алгоритму швидкого перетворення Фур'є за основою два для 512 точок складає 2358 тактів, а для 1024 точок – 5180 тактів.

Висновки. Проведені дослідження та розрахунки показали, що використання розробленого апаратного співпроцесора зменшує час виконання алгоритму ШПФ та навантаження на процесор під час обчислень.

КЛЮЧОВІ СЛОВА: RISC-V, процесор, цифрова обробка сигналів, швидке перетворення Фур'є, конвеєр, співпроцесор, FPGA.

АБРЕВІАТУРИ

ЦОС – цифрова обробка сигналів;
ШПФ – швидке перетворення Фур'є;
ISA – instruction set architecture;
RISC – reduced instruction set computer;
ARM – advanced RISC machine;
NOP – no operation;
FPGA – field-programmable gate array;
MIPS – million instruction per second;
DMIPS – Dhrystone MIPS;
CPI – clock per instruction;
SIMD – single instruction multiple data;
SSE – streaming SIMD extensions;
AVX – advanced vector extension;
LUT – look-up table;
MUX – multiplexer;
ALU – arithmetic-logic unit.

НОМЕНКЛАТУРА

CPU_e – час виконання алгоритму;
CPU_p – кількість тактів, необхідних для виконання алгоритму;
CPU_t – тривалість одного такту в секундах;
CPI_a – середня кількість тактів для виконання алгоритму;
IC – кількість інструкцій для виконання алгоритму;
M – мільйон операцій за секунду;
$X(k)$ – вектор частотної складової сигналу;
$x(2n)$ – вектор дискретних значень (парні індекси елементів);
$x(2n+1)$ – вектор дискретних значень (непарні індекси елементів);
W_N – повертаючий множник;
f_n – частота сигналу за номером n;
f_S – частота дискретизації;
N – кількість точок алгоритму ШПФ;
P – кількість тактів для виконання алгоритму ШПФ;
L – кількість стадій конвеєру;
K – кількість точок, яку опрацьовує метелик;
Q – кількість метеликів.

ВСТУП

При побудові сучасних комп'ютерних систем та окремих пристроїв розробники орієнтуються на використання відкритих обчислювальних стандартів архі-

DOI 10.15588/1607-3274-2024-1-18

тектур, наприклад, x86–x64, ARM, SPARC. RISC-V є однією з таких відкритих і доступних стандартів архітектур з простим та модульним набором команд (ISA). ISA описує роботу ядра процесора, кількість регістрів, кожну інструкцію машинного рівня та її байт-код.

Доступна та постійно розширювана архітектура дає можливість інженерам проводити різного роду дослідження, демонструвати неперевершені результати в продуктивності та енергоефективності, а також робити конкуренцію, як комерційним виробникам, так і іншим інженерам.

Алгоритми цифрової обробки використовуються у багатьох сферах: наука, медицина, інженерія, виробництво, телекомунікації тощо. У залежності від вимог, обчислення алгоритмів ЦОС може виконуватися безпосередньо процесором, окремим спеціалізованим вузлом чи на базі апаратного співпроцесора, впровадженого у структуру основного процесора. Останній підхід суттєво зменшує вимоги до параметрів основного процесора.

Одним з найважливіших алгоритмів ЦОС є швидке перетворення Фур'є (ШПФ). ШПФ – це алгоритм прискореного обчислення дискретного перетворення Фур'є, що зменшує об'єм обчислень з $O(n^2)$ до $O(nlog_2(n))$. Складність алгоритму говорить про те, що для виконання прямого або оберненого перетворення необхідно n^2 комплексних операцій додавання та множення для класичного алгоритму, а для ШПФ – $nlog_2(n)$. Під час виконання алгоритму безпосередньо процесором необхідно постійно записувати/читати в/з комірки пам'яті, час доступу до якої в кілька разів більший за час тактового сигналу процесора. Крім того, як до складу універсальних процесорів, так і у процесори ЦОС, не входить повноцінний співпроцесор для виконання алгоритму ШПФ.

Тому, є актуальною розробка структури співпроцесора ШПФ та впровадження його у ядро RISC-V архітектури.

Об'єкт дослідження – конвеєрний процесор архітектури RISC-V з співпроцесором цифрової обробки сигналів.

Предмет дослідження – методи та засоби проектування процесора архітектури RISC-V з модулем ШПФ та дослідження його характеристик.

Мета роботи – проектування та дослідження характеристик конвеєрного процесора архітектури RISC-V з співпроцесором цифрової обробки сигналів, що виконує швидке перетворення Фур'є.

1 ПОСТАНОВКА ЗАДАЧІ

Для забезпечення функціонування процесора необхідно:

– розробити модель одноядерного конвеєрного процесора;

– розробити модель співпроцесора ШПФ;

– розробити набір інструкцій, які дозволяють процесору керувати співпроцесором;

– перевірити роботу розроблених моделей, використовуючи мікросхеми FPGA.

Процесор повинен відповідати таким вимогам: частота роботи – не більше 250 МГц, CPI – не більше 2; MIPS при частоті 250 МГц – не менше 200.

Співпроцесор ШПФ повинен відповідати таким вимогам: частота роботи модуля – не більше 250 МГц, кількість точок ШПФ – 512/1024.

Оскільки дані вимоги пов'язані із часовими параметрами, розглянемо основні з них. Вираз (1) пов'язує найпростіші показники з часом виконання алгоритму (кількість тактів та довжина тактового сигналу):

$$CPU_e = CPU_p \times CPU_t. \quad (1)$$

Вираз (1) не включає жодних посилань на кількість інструкцій, необхідних для алгоритму; показує, що покращити продуктивність можна шляхом зменшення кількості тактових сигналів, необхідних для виконання алгоритму, або скороченням довжини тактового сигналу [1].

Вираз (2) враховує параметр кількості інструкцій, необхідних для виконання алгоритму:

$$CPU_e = IC \times CPI_a \quad (2)$$

Для одноядерного однотактного процесора архітектури RISC-V показник CPI складає 4–5, тоді як для конвеєрного – 1–1.5.

Формула (3) вираховує показник MIPS (M), визначає продуктивність, обернену до часу виконання:

$$M = \frac{IC}{CPU_e \times 10^6}. \quad (3)$$

При розробці інтегральної схеми на FPGA, в даному випадку для RISC-V ядра та співпроцесора ШПФ, використовують термін – рівень логіки. Рівень логіки означає кількість послідовних логічних елементів між початковим (вхідним) вузлом та кінцевим (вихідним). Даний параметр є важливим при проектуванні, оскільки від нього залежить частота роботи модуля та енергетичні витрати при його роботі.

2 ОГЛЯД ЛІТЕРАТУРИ

На базі RISC-V архітектури розроблений ряд процесорів і мікропроцесорів, IP-ядер. Найближчі за задачами та функціями до розроблюваного процесора наведені нижче.

Проект NEORV32 розрахований для мікросхем програмованої логіки. Містить дві стадії конвеєру, кеш пам'ять для інструкцій та даних, два привілейовані режими ядра, 32 входи переривань, периферія: UART, SPI, DMA, CRC і т.д. Мова програмування – VHDL [2]. Максимальна частота процесора – не більше 150 МГц. Це, по-перше, зв'язано з тим, що процесор містить довгі ланцюги логічних елементів на ста-

DOI 10.15588/1607-3274-2024-1-18

дії виконання, які приводять до відчутних часових затримок. По-друге, цифрові компоненти процесора, як: АЛП, модуль керування, модуль генерації констант, модуль вирішення конфліктних ситуацій, використовують вбудовані функції мови VHDL. З однієї сторони компілятор намагається побудувати операції множення, ділення, зсуви та ін. на базових комбінаційних елементах LUT (які мають значні транспортні затримки). З іншої сторони – алгоритм, що написаний операторами, умовними та циклічними блоками мови VHDL, не є оптимізований при оптимально налаштованих параметрах.

The Potato – це проект процесора RISC-V архітектури, який підтримує базовий набір інструкцій та розроблений на мові VHDL. Процесор підтримує один привілейований рівень, апаратний таймер, вісім незалежних сигналів переривань, Wishbone шину та кеш пам'ять інструкцій та даних. Ядро містить п'ять стадій конвеєру. Проект можна застосовувати на різних типах мікросхем програмованої логіки, але робоча частота такого процесора буде не більшою за 150 МГц [3].

При обчисленні алгоритму ШПФ, програми використовують принцип SIMD, а саме в архітектурі Intel це розширення SSE та AVX, в ARM – SVE. Дані розширення застосовуються при обчисленні арифметичних операцій над чотирма-вісьмома числами рухомої коми одинарної точності. Проте програмні забезпечення з даними розширеннями мають обмеження, оскільки обчислення ШПФ проводиться над комплексними числами, а кількість комплексних чисел, яку можна обробити однією інструкцією, становить два-чотири відповідно до обраного розширення.

Під час виконання алгоритму ШПФ, процесор звертається до пам'яті, в якій містяться дані та таблиця повертаючих множників. Якщо процесор не містить кеш пам'яті, час читання даних із зовнішньої пам'яті складає 7–20 нс [4]. При роботі процесора з кеш пам'яттю час читання даних складає від 1 до 4 нс [5].

У портативних системах, де переважно використовуються дешеві ARM Cortex-M ядра порівняно з Cortex-A, використовують ШПФ для стиснення та декодування аудіо сигналу формату MP3, спектрального аналізу сигналу в осцилографах та ін. Такі ядра не мають SIMD розширення для чисел з рухомою комою та виконують алгоритм ШПФ безпосередньо ядром, а час виконання залежить від робочої частоти ядра, обраного формату даних та кількості даних. В дослідженнях [6–7] наведено швидкодію програмної реалізації алгоритму на ядрах Cortex-M4 та Cortex-M7. Час виконання алгоритму ШПФ для кількості точок 512/1024 з числами з рухомою комою одинарної точності для ядра Cortex-M4 при частоті 180 МГц становить 547 мкс, кількість тактів – 98624, тоді як для Cortex-M7 при частоті 216 МГц час виконання – 339 мкс, кількість тактів – 73292.

У наведених матеріалах [8–10] будова модуля ШПФ подібна за структурою: пам'ять, де зберігається ціла та уявна частина комплексних чисел; таблиця повертаючих множників; схема метелика за основою два та контролюючий модуль. Робоча частота модуля – не більше 100 МГц. Модуль [9] максимально опрацьовує 256 точок на одному метелику за основою два, що не є достатнім для більшості прикладних задач.

Проект модуля Radix-2 FFT [11] представляє реалізацію 16-ти точкового ШПФ, написаний на мові VHDL. На вхід даного модуля подаються 32 сигнали у форматі IEEE-754, які формують 16 комплексних значень. Модуль виконує обчислення за один такт. Складність такої схеми полягає у великих обчислювальних ресурсах: для модуля необхідно 64 комплексних помножувачів, 128 комплексних суматорів.

TMS320VC5505/TMS320C5505/TMS320C5515 – це процесори ЦОС, які містять вбудований співпроцесор для ШПФ (HWAFFT). Дані процесори побудовані на ядрі C55x, RISC архітектури, із розрядністю даних 16 бітів. Показник MIPS складає 800 мільйонів операцій за секунду при частоті 400 МГц [12].

Співпроцесор HWAFFT складається з одного метелика за основою два, який виконує алгоритм прорідження за часом. Апаратний модуль підтримує двоетапний режим, в якому два яруси ШПФ обчислюються за один прохід. У даному режимі, керуючий модуль направляє результати обчислень з першого етапу на вхід метелика для обчислення другого етапу. Це забезпечує прискорення обчислень для великої кількості точок. На вхід модуля надходять 16-ти розрядні числа, формат числа – фіксована крапка. Апаратна частина співпроцесора HWAFFT містить дві стадії конвеєру. Комплексне множення з повертаючим множником виконується на першій стадії конвеєра, а комплексне додавання та віднімання – на другій. Результати отримуються через кілька тактів із моменту надходження вхідних даних: п'ять тактів затримки для одноетапного режиму, дев'ять тактів для двоетапного режиму.

Інтерфейс між процесором та співпроцесором реалізований через набір інструкцій, які дозволяють виконувати операції ініціалізації, завантаження/ читання результату та виконання самого алгоритму. Вхідні дані повинні надходити у біт-реверсивному порядку.

Процесор виконує перетворення 512 точок за 3740 тактів, а 1024 – за 7315 тактів [13].

3 МАТЕРІАЛИ ТА МЕТОДИ

Для покращення продуктивності процесора необхідно зменшити кількість тактів на виконання програми або скоротити довжину тактового сигналу. Конвеєрний підхід проектування процесора дозволяє зменшити кількість тактів на виконання алгоритму.

Розробники RISC-V архітектури пропонують розроблювати конвеєрний процесор, який складається з 5 стадій: Fetch (читання інструкції з пам'яті програм), Decode (декодування зчитаної інструкції), Execute (виконання декодованої інструкції), Memory (читання/запис даних з пам'яті даних), WriteBack (запис результату в регістр) [1].

DOI 10.15588/1607-3274-2024-1-18

При роботі процесорів із конвеєрною архітектурою виникають конфліктні ситуації, які призводять до неможливості виконання чергових інструкцій. Налічують три класи конфліктів: конфлікт між даними, конфлікт керування, структурний конфлікт.

Конфлікт між даними виникає, коли для виконання інструкції, необхідні дані від попередньо отриманого результату. Існує два методи усунення конфлікту:

– вставлення NOP операції в конвеєр. При цьому збільшується час виконання інструкції;

– використання методів пересилання. Пересилання відбувається зі стадій Memory та WriteBack до стадії Execute.

Конфлікт керування виникає, коли процесор виконує команду переходу, а інструкція, яка йде після неї, – залишається в конвеєрі. Щоб вирішити даний конфлікт, необхідно передати NOP операцію зі стадії Fetch до стадії Decode коли виконується перехід.

Структурний конфлікт виникає у суперскалярних та багатотактових процесорах, коли інструкції використовують один і той же апаратний ресурс (пам'ять) в однаковий момент часу. Єдиний спосіб уникнути такого конфлікту – виконати NOP операцію та дати можливість конвеєру, з яким виникає конфлікт, завершити свою роботу.

Згідно рекомендацій [1] розробників RISC-V архітектури розроблено функціональну схему конвеєрного процесора (рис. 1). Згідно схеми, конвеєр складається з п'яти стадій та містить наступні компоненти: програмний лічильник (PC), пам'ять інструкцій (SRAM), модуль керування (Control Unit), регістрова пам'ять для цілих чисел (GPRs), регістрова пам'ять для дійсних чисел (FGPRs), модуль генерації констант (IMM.GEN), модуль усунення конфліктів при роботі з пам'яттю (Hazard Detection Unit), АЛП виконання базових команд (ALU), АЛП для команд множення/ділення (M Extension), АЛП дійсних чисел (FPU), модуль керування АЛП (ALU Control Unit), блок виконання порівняння (Branch Unit), модуль пересилання даних (Forwarding Unit ALU), пам'ять даних (DRAM). Процесор виконує 37 базових інструкцій та вісім інструкцій множення/ділення.

При виконанні алгоритмів ШПФ використовують алгоритми прорідження за частотою та прорідження за часом. Для розкладу дискретних даних в спектральний ряд за алгоритмом ШПФ за основою два, використовується наступний вираз [14–15]:

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(2n) \times W_N^{2nk} + \sum_{n=0}^{\frac{N}{2}-1} x(2n+1) \times W_N^{k(2n+1)}. \qquad (4)$$

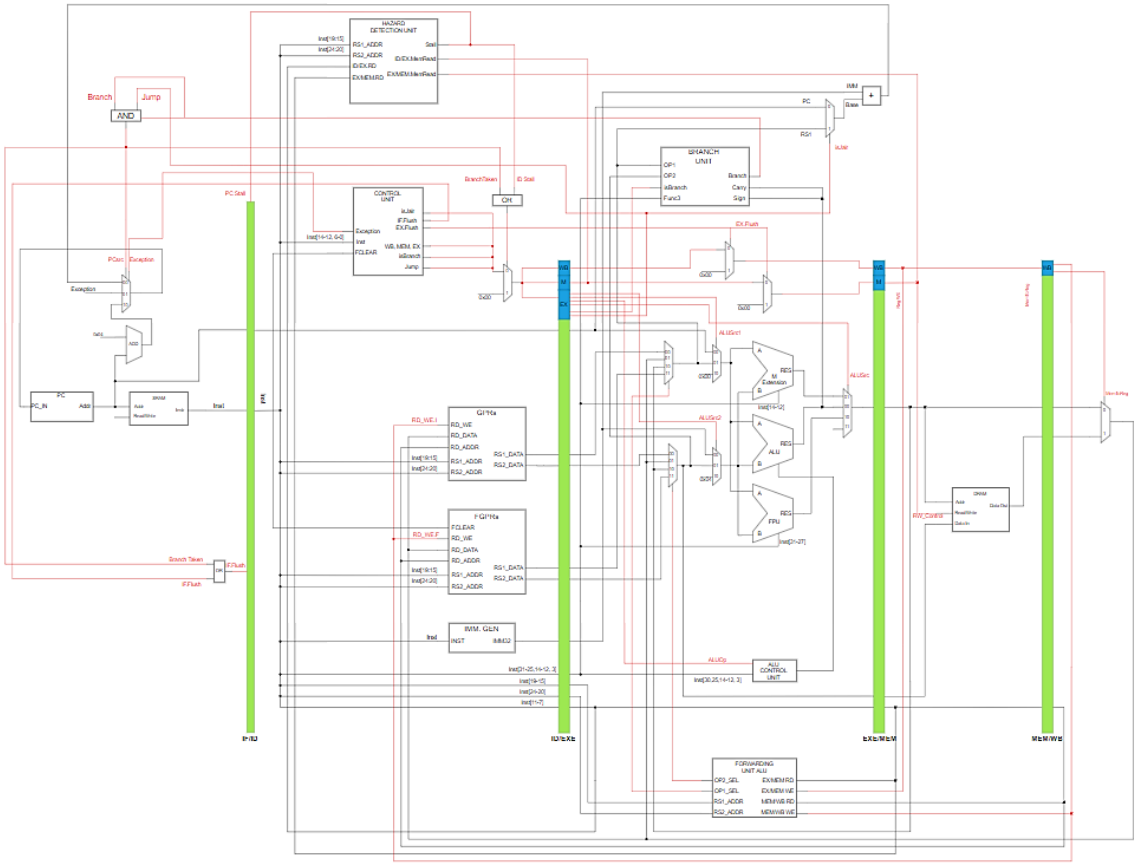


Рисунок 1 – Функціональна схема конвеєрного RISC-V ядра

DOI 10.15588/1607-3274-2024-1-18

Складність обох алгоритмів однакова, але є різниця у порядку вхідних та вихідних даних. Для алгоритму прорідження за часом вхідні дані розміщені в бітреверсивному порядку, а вихідні – в прямому. Зворотній підхід використаний і для прорідження за частотою.

Основним елементом ШПФ є метелик (рис. 2). На рисунку 2 наведено схеми метелика за основою два, які містять дві операції додавання та одну операцію множення комплексних змінних. Стрілка на рисунку 2а означає множення повертаючого множника на різницю двох комплексних величин, а на рисунку 2б – множення повертаючого множника на комплексну величину *b*.

Результатом виконання ШПФ є вектор обчислених даних, який містить номери частот з яких утворена функція. Щоб знайти частоту, маючи номер частоти, використовують наступний вираз:

$$f_n = \frac{f_S \times n}{N}. \quad (5)$$

а б

Рисунок 2 – Схема метелика ШПФ:
а – з прорідженням за частотою, б – з прорідженням за часом

У рамках дослідження схеми модуля [10], авторами даної статті проведено аналіз, зроблено висновки стосовно його працездатності та визначено компоненти, які можна покращити, що в результаті дозволить пришвидшити процес обчислень.

Модуль [10] складається з пристрою генерації адрес AGU (Address Generation Unit); таблиці повертаючих множників Twiddle factor ROM; одного метелика за основою два; двох модулів пам'яті для зберігання дійсної та уявної частини комплексних змінних; з'єднаннями, які з'єднують дані компоненти та утворюють працюючий комплекс.

Пам'ять для комплексних значень та пам'ять для констант повертаючих множників має затримку читання один такт. Модуль AGU контролює генерацію адрес для читання/запису вмісту пам'яті в/з метелика.

Кількість тактів, яка необхідна модулю, щоб виконати операцію на одному метелику і записати результат в пам'ять, рівна десяти [10]. На основі вище описаних параметрів наведено загальну кількість тактів для виконання алгоритму (табл. 1).

Таблиця 1 – Кількість тактів виконання алгоритму ШПФ на однотактному метелику за основою два

Кількість точок	Кількість тактів
32	160
512	23040
1024	51200
2048	92160

Схема метелика виконує алгоритм з частотним прорідженням, що не є повним функціоналом, оскільки можуть виникнути випадки, в яких необхідно саме алгоритм прорідження за часом.

Для перевірки працездатності модуля за описаним принципом [8–10] побудовано математичну модель у системі MathCAD, яка виконує пряме ШПФ над 16 точками. Дана модель складається з матриці повертаючих множників, матриці перестановки повертаючих множників, матриці перестановки точок, функції метелика за основою два та основної функції ШПФ. Функція метелика має можливість виконувати алгоритм, як за прорідженням за часом, так і за частотою.

На рисунку 3 наведено схему виконання алгоритму прямого та оберненого ШПФ (функція FFT16), де позначено: *X* – вхідні дискретні дані; *W* – матриця повертаючих множників; *PermXI* – матриця перестановки точок; *PermW* – матриця перестановки повертаючих множників; *InPerm* – ознака порядку вхідних даних (біт-реверсивний або прямий); *OutPerm* – ознака порядку вихідних даних (біт-реверсивний або прямий); *Inverse* – ознака виконання алгоритму (прорідження за часом або частотою).

Перший крок функції FFT16 – визначення параметрів *a* (збільшення або зменшення ітерацій) та *s* (ітераційний лічильник) за допомогою вхідного параметру *InPerm*. Основний алгоритм виконується в двох операторах циклу. Перший цикл описує ітерацію по чотирьом ярусам, другий – обчислення даних на одному метелику. Після завершення обчислень функція повертає вектор з елементами, розміщеними у бітреверсивному порядку при *OutPerm* рівне одиниці, в інакшому випадку – повертає обчислений вектор.

Результати функції FFT16 порівняно із вбудованою функцією MathCAD – fft, яка приймає один параметр – вектор дискретних даних. Для перевірки використано функцію синуса з такими параметрами: значення амплітуди – 16, частота – 5 Гц, початкова фаза – 0 радіан.

Побудовано амплітудно-частотний графік функції синуса (рис. 4). Згідно графіку, номер частоти сигналу – один, сама частота сигналу за п'ятим виразом – 5 Гц. Отже, обчислена частота рівна частоті дискретного сигналу, що підтверджує правильність обчислень.

DOI 10.15588/1607-3274-2024-1-18

Процес можна прискорити, використовуючи конвеєрний підхід. Модуль ШПФ [10] виконує обчислення двох комплексних значень на одному метелику за десять тактів. Для покращення характеристик авторами даної статті пропонується побудувати шести стадійний конвеєр. Тоді загальна кількість тактів для виконання алгоритму за даним принципом рівна:

$$P = \left(\frac{N}{Q \times K} + L \right) \times \log_K (N). \quad (5)$$

З виразу 6 обчислено загальну кількість тактів, необхідну для виконання алгоритму, використовуючи один конвеєрний метелик (Q = 1) за основою два (K = 2) (табл. 2).

Таблиця 2 – Кількість тактів виконання алгоритму на конвеєрному метелику

Кількість точок	Кількість тактів
32	110
512	2358
1024	5180
2048	11330

На рисунку 5 наведено функціональну схему конвеєрного шести стадійного метелика. В метелику виконується обчислення, як за алгоритмом з прорідження за часом, так і з прорідження за частотою. Згідно рисунку 5, X та Y – вхідні дані, W – повертаючий множник, Ctrl – ознака нормалізації вихідних даних, Mode – ознака виконання алгоритму (прорідження за часом або частотою) CLK – тактовий сигнал для D-тригеру.

Функціональна схема співпроцесора ШПФ наведена на рисунку 6. Даний співпроцесор складається з модуля керування (Control Unit), модуля генерації адреси (Address Generation Unit), пам'яті даних (Data Memory), пам'яті повертаючих множників (Exponent Memory), метелика ШПФ за основою два (Radix-2). Обмін інформацією між процесором та співпроцесором відбувається за допомогою вхідних та вихідних сигналів модуля керування.

```
FFT16(X, W, PermXI, PermW, InPerm, OutPerm, Inverse) :=
    a ← −1  if  InPerm = 1
    a ← 1  otherwise
    s ← 0  if  InPerm = 0
    s ← 3  otherwise
    for  i ∈ 0.. 3
        for  j ∈ 0.. 7
            (A  B) ← Radix_2[X(PermXI2·j,s), X(PermXI2·j+1,s), W(PermWj,s), InPerm, Inverse]
            X(PermXI2·j,s) ← A
            X(PermXI2·j+1,s) ← B
        s ← s + a
    for  k ∈ 0.. 15
        rk ← PermXIk,0
    X ← Permutation(X,r)  if  OutPerm = 1
    X  otherwise
```

Рисунок 3 – Схема алгоритму ШПФ

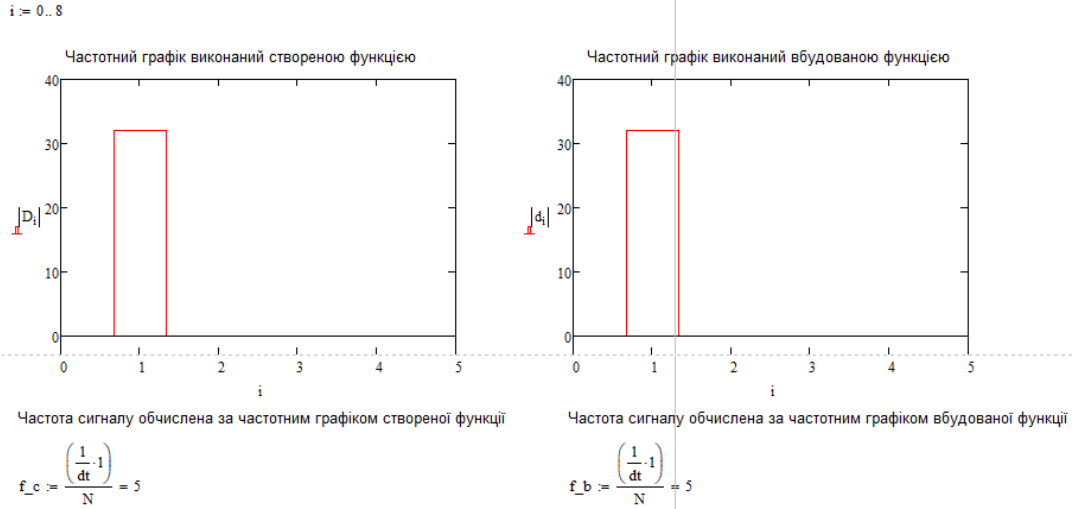


Рисунок 4 – Амплітудно-частотний графік функції синуса

DOI 10.15588/1607-3274-2024-1-18

Пам’ять повертаючих множників має об’єм $\frac{N}{2}\times 4$ байти.

Для керування модулем ШПФ використовуються наступні сигнали: Command – шина команд керування модулем; User Data WR – шина вхідних даних; User Data RD – шина вихідних даних; User Address – шина адреси читання/запису, Enable – сигнал активації модуля.

Розроблений співпроцесор з одним метеликом за основою два та об’ємом пам’яті 4 кБ є невитратним рішенням для покращення процесора. Збільшення пам’яті співпроцесора або основи метелика збільшить складність апаратної розробки та апаратних модулів.

Модуль можна модифікувати збільшенням розміру пам’яті (Data Memory), що дозволить обробити більше точок. При збільшені розміру пам’яті модифікується модуль генерації адрес (Address Generator Unit) та керуючий модуль (Control Unit). Для більш оптимізованого рішення варто використовувати пам’ять розмірністю 2^n, $n \in N$. Кількість точок, яку може обробити співпроцесор, обмежена розмірністю пам’яті.

Введення у схему додаткового метелика за основою два дозволить зменшити час виконання алгоритму. При впровадженні додаткового метелика додається один порт до пам’яті, модифікується модуль генерації адрес (Address Generator Unit) та керуючий модуль (Control Unit). Кількість метеликів, яку можливо встановити у автомат, залежить від кількості портів входу/виходу, яку підтримує пам’ять.

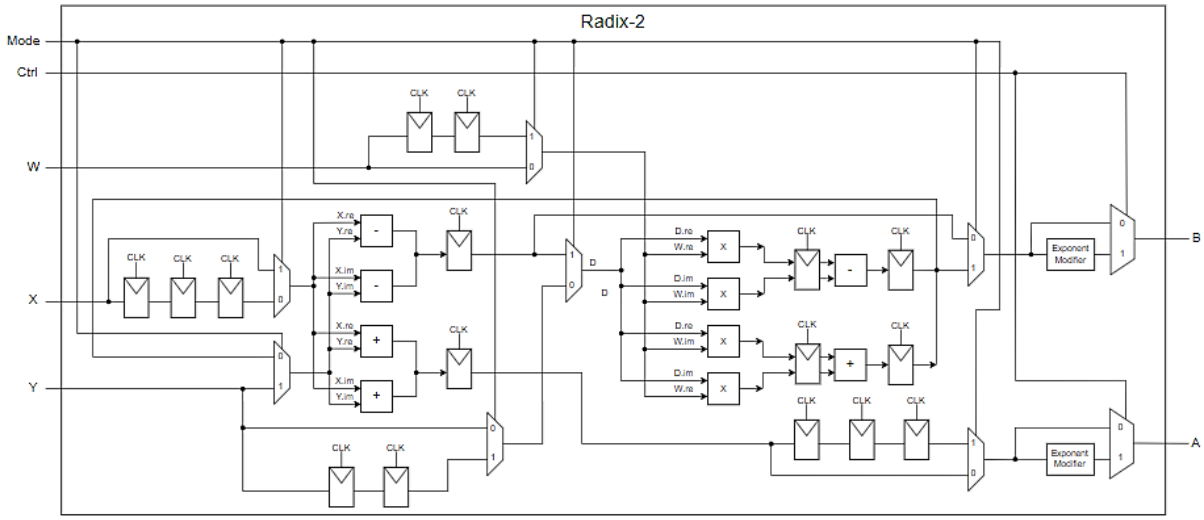


Рисунок 5 – Функціональна схема конвеєрного метелика

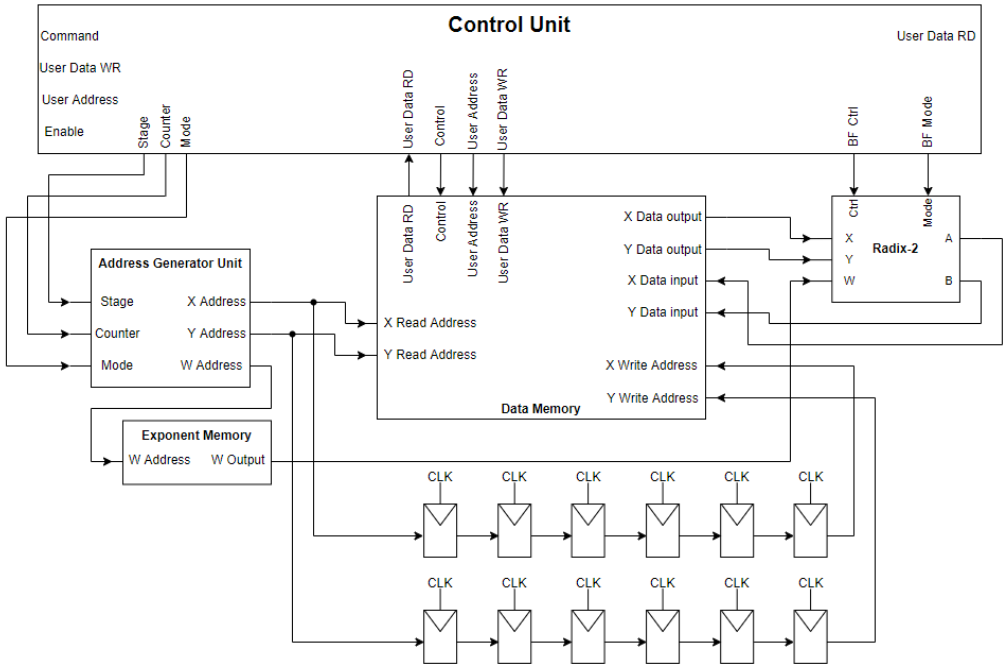


Рисунок 6 – Функціональна схема автомату ШПФ

DOI 10.15588/1607-3274-2024-1-18

На рисунку 7 наведено функціональну схему процесора RISC-V архітектури з впровадженим співпроцесором ШПФ. Співпроцесор знаходиться у стадії виконання (Execute). На вхід мультиплексора під'єднані шина User Data RD разом з іншими АЛП. Інформація для шин Command, User Data RW, User Address, Enable генерується модулем керування ядра процесора (Control Unit) та посилається до входу станового регістра EX/MEM (рис. 1).

Формат інструкцій процесора для роботи з модулем ШПФ наведено на рисунку 8. Команди схожі до формату R-типу, оскільки працюють із двома регістрами і мають певний ідентифікатор інструкції у полі func3 та func7. Створення нового формату інструкцій призвело б до часткової модифікації архітектури та її ускладнення.

Введено 11 інструкцій, чотири з яких дозволяють зчитувати дані з пам'яті дійсної або уявної частини в регістри дійсних або цілих чисел. Наступні чотири інструкції дозволяють записувати дані у пам'ять дійсної або уявної частини з регістрів дійсних або цілих чисел. Останні три команди дозволяють керувати модулем, а саме: старт виконання алгоритму (fftstart), скидання автомату в початкове значення (fftreset), отримання інформації про співпроцесор (fftstatus).

Читання даних з пам'яті та запис їх процесором у співпроцесор ШПФ відбувається один раз для одного виконання перетворення Фур'є. Індексація запису/читання даних в/з пам'яті співпроцесора є прямою або біт-реверсивною при вказанні параметра інструкцією співпроцесора завантаження/читання.

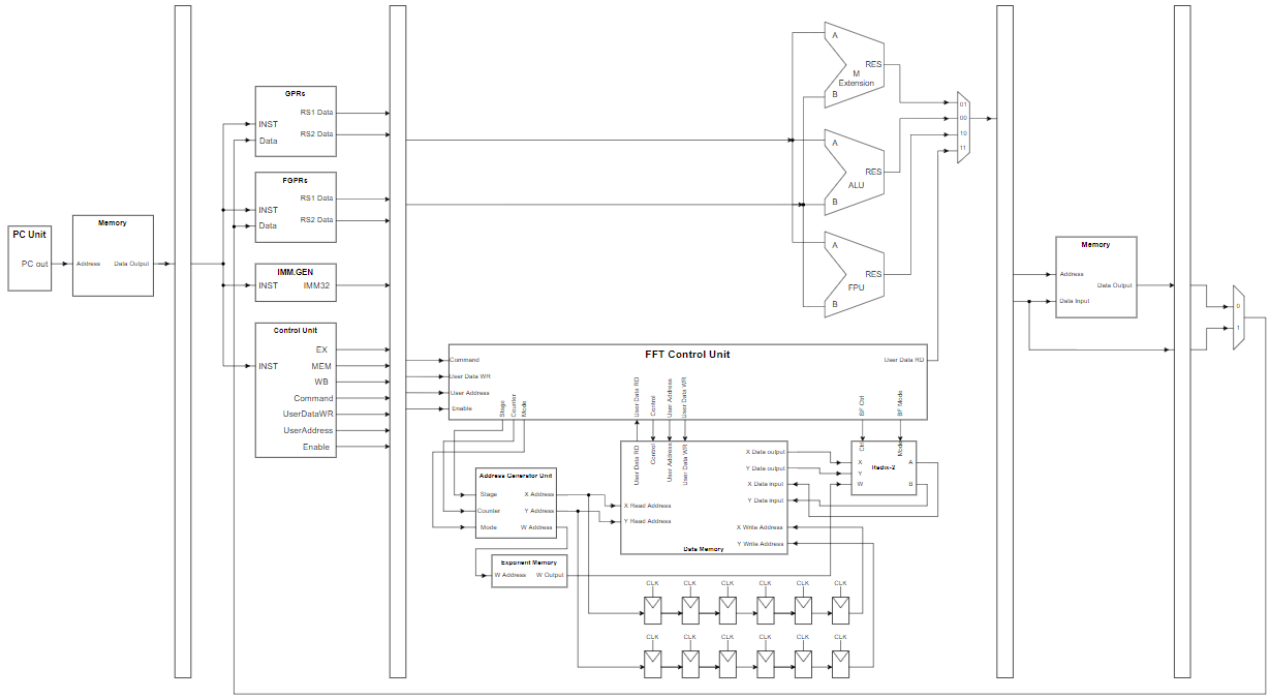
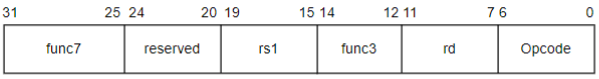


Рисунок 7 – Функціональна схема конвеєрного процесора RISC-V архітектури з впровадженим модулем ШПФ

FFT INSTRUCTION FORMAT



Inst	Name	Opcode	func3	func7
fftstoreRe.f	Store data to FFT from FP register	1011011	000	0x0
fftstoreIm.f	Store Imaginary part to FFT from FP register	1011011	000	0x1
fftstoreRe.i	Store data to FFT from Integer register	1011011	001	0x0
fftstoreIm.i	Store Imaginary part to FFT from Integer register	1011011	001	0x1
fftloadRe.f	Load Real part from FFT to FP register	1011011	010	0x0
fftloadIm.f	Load Imaginary part from FFT to FP register	1011011	010	0x1
fftloadRe.i	Load Real part from FFT to Integer register	1011011	011	0x0
fftloadIm.i	Load Imaginary part from FFT to Integer register	1011011	011	0x1
fftstart	Start execution	1011011	100	
fftreset	Reset module	1011011	101	
fftstatus	Get module status	1011011	110	

Рисунок 8 – Формат інструкцій процесора для роботи з модулем ШПФ

DOI 10.15588/1607-3274-2024-1-18

4 ЕКСПЕРИМЕНТИ

Модель запропонованого процесора розроблена на мікросхемі програмованої логіки Xilinx SPARTAN 6. Дана мікросхема має достатній набір програмованих логічних елементів (LUT), однопортових та багатопортових елементів пам'яті (RAMB16BWER), тригерів, модулів ЦОС (DSP48A1) та інших вбудованих компонентів [16].

Для розробки апаратного забезпечення на мікросхемі Xilinx SPARTAN 6 використовується програмний додаток Xilinx ISE Suite, який дозволяє розробляти логічні схеми різної складності, використовуючи дві мови опису апаратури: VHDL та Verilog. Даний програмний додаток дає можливість оцінити складність моделі, визначити її максимальну частоту, переглянути цифрову схему розробленої моделі та на цій основі провести удосконалення моделі. Обрано мову опису апаратури – VHDL.

Для знаходження параметру CPI, ядра RISC-V архітектури, розроблено програму мовою асемблера для архітектури RISC-V, яка виконує чотири арифметичні операції та одну операцію переходу. Виконуючи програму, необхідно знайти кількість інструкцій та загальну кількість тактів на її виконання.

Для знаходження параметру MIPS, ядра RISC-V архітектури, розроблено програму мовою асемблера множення двох цілочисельних матриць із розрядністю елемента 32 біти. По завершенню виконання програми протягом однієї секунди, необхідно визначити кількість інструкцій, які виконалися за даний час.

Для визначення швидкодії виконання алгоритму ШПФ, написано асемблерну програму, яка передає дискретні дані синуса амплітудою в 3 одиниці, частотою 44100 Гц, початковою фазою – 0 радіан співпроцесору ШПФ та виконує прямий розклад 512/1024 точок у частотний спектр. По завершенню виконання даної програми необхідно визначити кількість тактів для виконання модулем алгоритму та виконати порівняння із готовими результатами досліджень.

5 РЕЗУЛЬТАТИ

Дослідження проведені у симуляторі ISim, який є частиною додатку Xilinx ISE. В симуляторі налаштовано тактову частоту процесора на 250 МГц. Знайдено параметр CPI ядра RISC-V та порівняно його з однотактним процесором. Програма на конвеєрному процесорі виконалась за 29 тактів, а на однотактному – 60. Відповідно CPI конвеєрного процесора – 1.6, а однотактного – 4.6.

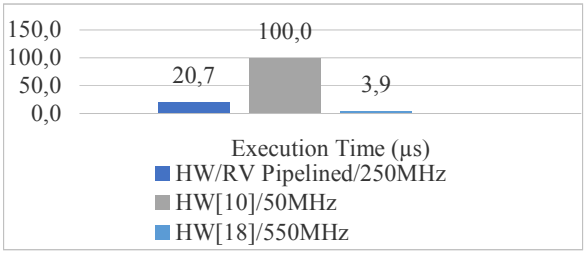
Алгоритм множення двох матриць розмірами 10 на 10 виконано за час 14.224 мкс з виконаною кількістю інструкцій – 3456. Знайдено параметр MIPS конвеєрного процесора – 242, тобто 242 мільйонів інструкцій за секунду.

Схема метелика використовує вбудований апаратний модуль DSP на якому створено 64-х бітний помножувач [17].

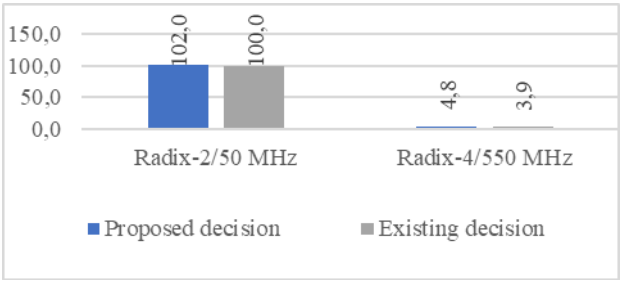
Знайдено кількість тактів, необхідну для виконання алгоритму ШПФ для 512 та 1024 точок (діаграма 3). Для 512 точок модулю необхідно 2358 тактів, а час виконання при частоті 250 МГц – 9.4 мкс. Для 1024 точок кількість тактів складає 5180, а час виконання – 20.7 мкс.

На діаграмі 1 зображено порівняльну характеристику часу виконання ШПФ для 1024 точок для розробленого співпроцесора ШПФ та існуючих рішень. Розроблений співпроцесор виконує алгоритм ШПФ за основою два при тактовій частоті роботи 250 МГц у п'ять разів повільніше у порівнянні з модулем [18] при частоті 550 МГц; модуль [10] при частоті 50 МГц виконує ШПФ за 100 мкс, тоді як розроблений співпроцесор – за 20.7 мкс при тактовій частоті роботи 250 МГц.

На діаграмі 2 наведено порівняльну характеристику часу виконання алгоритму ШПФ спроектованим співпроцесором (Proposed decision) та іншими рішеннями (Existing decision) при частотах 50 МГц і 550 МГц. Кількість точок для алгоритму ШПФ – 1024. Згідно діаграми, співпроцесор виконує алгоритм на 2 мкс повільніше у порівнянні з модулем [18] при тактовій частоті роботи 50 МГц та на 0.9 мкс повільніше за модуль [18] при частоті 550 МГц.



Діаграма 1 – Порівняльна характеристика часу виконання (мкс) ШПФ для 1024 точок



Діаграма 2 – Порівняльна характеристика часу виконання (мкс) алгоритму ШПФ співпроцесором

У таблиці 3 наведено дані використаних ресурсів мікросхеми Xilinx SPARTAN 6.

Таблиця 3 – Використані ресурси системи

LUT	DSP48A1	RAMB16BWER	DFF
7232 з 9112	24 з 32	6 з 32	9872 з 18224

6 ОБГОВОРЕННЯ

Аналізуючи результати досліджень, що проводилися для мікросхеми програмованої логіки, запропонований метод з апаратним співпроцесором ЦОС демонструє прискорення в обчисленнях алгоритму ШПФ у порівнянні з програмними реалізаціями для ядер ARM Cortex-M4 та ARM Cortex-M7 [6–7].

DOI 10.15588/1607-3274-2024-1-18

Час виконання запропонованого методу дещо поступається існуючим апаратним рішенням [8–10] із-за складності апаратних модулів, які складаються з довгих ланцюгів LUT елементів, апаратних вбудований модулів DSP, що збільшують час виконання інструкцій розширення.

Основною перевагою запропонованого методу є його модульність, що дозволяє збільшувати кількість метеликів, змінювати основу метелика, збільшувати кількість точок для опрацювання для зменшення часу виконання без модифікації структури. Варто зазначити, що даний співпроцесор працює з числами з рухомою комою одинарної точності, які забезпечують точність обчислень.

Таким чином, удосконалено процесор RISC-V архітектури, впровадженням співпроцесора ШПФ, без модифікації самої архітектури.

ВИСНОВКИ

Розглянуто відкриту архітектуру набору команд RISC-V, що дозволяє освітнім, науково-дослідним та промисловим організаціям працювати над розробками на основі даних процесорів.

На основі програмованої логіки, використовуючи розроблену модель, розроблена схема конвеєрного процесора RISC-V архітектури. Використовуючи симулятор Isim, проведено дослідження працездатності та продуктивності процесора. Співпроцесор виконує алгоритм ШПФ розмірністю 512/1024 точки. Час виконання співпроцесором алгоритму на одному метелику за основою два для 512 точок склав 9.4 мкс, а для 1024 точок – 20.7 мкс при тактовій частоті роботи 250 МГц.

Проведено дослідження часу виконання алгоритму ШПФ розробленим співпроцесором на частотах 50 МГц і 550 МГц. Згідно результатів, час виконання алгоритму співпроцесором з метеликом за основою два є близьким до значення, наведеного у джерелі [18], тоді як з метеликом за основою чотири, – розроблений співпроцесор виконує ШПФ на 0.9 мкс повільніше.

Подальші дослідження розробленого процесора спрямовані на визначення продуктивності процесора з використанням тестової програми Dhrystone. Це пов'язано з тим, що параметр MIPS не дозволяє об'єктивно оцінити продуктивність процесорів різних архітектур, тоді як DMIPS є стандартизованим та використовується виробниками обчислювальних машин. Для правильності розрахунків продуктивності необхідно при тестуванні використовувати ідентичні версії синтетичного тесту.

Науковою новизною роботи є покращення архітектури RISC-V, розширенням системи команд та впровадженням апаратного співпроцесора ШПФ.

Практичне значення розробленого процесора з апаратним модулем ШПФ полягає у зменшенні часу виконання алгоритму ШПФ, покращенням архітектури RISC-V та можливістю використання апаратних модулів виконання інших алгоритмів ЦОС.

DOI 10.15588/1607-3274-2024-1-18

ПОДЯКИ

Роботу виконано в рамках науково-дослідної теми кафедри комп'ютерних систем та мереж Ужгородського національного університету «Методи і засоби апаратної та програмної реалізації високопродуктивних комп'ютерних систем та мереж» (номер державної реєстрації 0121U110031). Автори вдячні Горвату Петру Петровичу, доценту кафедри комп'ютерних систем та мереж ДВНЗ «УжНУ» за цінні та конструктивні поради під час написання роботи.

ЛІТЕРАТУРА

1. Patterson D. A. Computer Organization and Design. The Hardware/Software Interface: RISC-V Edition / D. A. Patterson, J. L. Hennessy. – California : Morgan Kaufmann Publishers, 2018. – 1665 p.
2. The NEORV32 RISC-V Processor [Electronic resource]. – Access mode: https://github.com/stnolting/neorv32
3. The Potato Project [Electronic resource]. – Access mode: https://github.com/skordal/potato
4. Keeth B. DRAM Circuit Design : Fundamental and High-Speed Topics / B. Keeth, R. J. Baker, B. Johnson, F. Lin. – Hoboken : John Wiley & Sons, 2007. – 440 p.
5. Memory Performance in a Nutshell [Electronic resource]. – Access mode: https://www.intel.com/content/www/us/en/developer/articles/technical/memory-performance-in-a-nutshell.html
6. The DSP capabilities of ARM Cortex-M4 and Cortex-M7 Processors [Electronic resource] / T. Lorenser. – Access mode: https://community.arm.com/cfs-file/__key/communityserver-blogs-components-weblogfiles/00-00-00-21-42/7563.ARM-white-paper-_2D00_-DSP-capabilities-of-Cortex_2D00_M4-and-Cortex_2D00_M7.pdf
7. Digital signal processing for STM32 microcontrollers using CMSIS // STMicroelectronics. – 2018. – AN4841 Rev 2. – 25 P.
8. Sapiecha K. Modular architecture for high performance implementation of FFT algorithm / K. Sapiecha, R. Jarocki // ACM SIGARCH Computer Architecture New. – 1986. – Vol. 14, № 2. – P. 261–270. DOI: 10.1145/17356.17387
9. A portable hardware design of a FFT algorithm / [C. Gonzalez-Concejero, V. Rodellar, A. Alvarez-Marquina et al.] // Latin American applied research. – 2007. – Vol. 37, № 1. – P. 79–82.
10. The Fast Fourier Transform in Hardware: A Tutorial Based on an FPGA Implementation [Electronic resource] / G. Slade. – Access mode: https://web.mit.edu/6.111/www/f2017/handouts/FFTtutorial121102.pdf
11. Radix-2 FFT – VHDL Implementation [Electronic resource]. – Access mode: https://github.com/bugratufan/radix2-fft-vhdl
12. TMS320C55x Technical Overview [Electronic resource]. – Access mode: https://www.ti.com/lit/ug/spru393/spru393.pdf?ts=1693041737254
13. FFT Implementation on the TMS320VC5505, TMS320C5505, and TMS320C5515 DSPs [Electronic resource] / M. McKeown. – Access mode: https://www.ti.com/lit/an/sprabb6b/sprabb6b.pdf?ts=1695025017385
14. Lyons R. G. Understanding Digital Signal Processing, Third Edition / R. G. Lyons. – Boston : Pearson Education, 2011. – 858 p.
15. Ifeachor E. C. Digital Signal Processing: A Practical Approach, Second Edition / E. C. Ifeachor, B. W. Jervis. – Boston : Addison Wesley, 1993. – 779 p.

16. Spartan-6 Family Overview [Electronic resource]. – Access mode: https://docs.xilinx.com/v/u/en-US/ds160
17. Spartan-6 FPGA Data Sheet: DC and Switching Characteristics [Electronic resource]. – Access mode: https://docs.xilinx.com/v/u/en-US/ds162
18. High Throughput and Mixed Radix N-Point Parallel Pipelined FFT VLSI Architectures for Advanced Wireless Communication / [K. Vijayakanthan, K. Hemachandran, M. Anand et al.] // International Journal of Grid and Distributed Computing. – 2020. – Vol. 13, № 1. – P. 400–411.

Стаття надійшла до редакції 11.12.2023.
Після доробки 30.01.2024.

UDC 004.318

THE DESIGN OF THE PIPELINED RISC-V PROCESSOR WITH THE HARDWARE COPROCESSOR OF DIGITAL SIGNAL PROCESSING

Vavruk Y. Y. – PhD, Associate Professor, Associate Professor of Electronic computing department of Lviv Polytechnic National University, Lviv, Ukraine.

Makhrov V. V. – Student of the Department of Computer Systems and Networks of Uzhhorod National University, Uzhhorod, Ukraine.

Hedeon H. O. – Assistant of the Department of Computer Systems and Networks of Uzhhorod National University, Uzhhorod, Ukraine.

ABSTRACT

Context. The digital signal processing is applied in many fields of science, technology and human activity. One of the ways of implementing algorithms of digital signal processing is the development of coprocessors as an integral part of well-known architectures.

In the case of developing a pipelined device, the presented approach will allow to use software and hardware tools of the appropriate architecture, provide the faster execution of signal processing algorithms, reduce the number of cycles and memory accesses.

Objective. Objectives are design and characterization study of a pipelined RISC-V processor and coprocessor of digital signal processing which performs fast Fourier transform.

Method. Analyzing technical literature and existing decisions allow to assess advantages and disadvantages of modern developments and on the basis of which to form the relevance of the selected topic. Model designing and simulation results allow to examine a model efficiency, to determine weak components' parts and to improve model parameters.

Results. The pipelined RISC-V processor has been designed which executes a basic set of instructions. Execution time of assembly program on the single-cycled and the pipelined processors have been analyzed. According to the results, the test program on the pipelined processor is executed in 29 cycles, while on the single-cycle processor it takes 60 cycles. The structure of the coprocessor for the fast Fourier transform algorithm and a set of processor instructions that allow working with the coprocessor have been developed. The number of cycles of the coprocessor based on Radix-2 fast Fourier transform algorithm for 512 points is 2358 cycles, and for 1024 points is 5180 cycles.

Conclusions. Conducted researches and calculations have showed that the application of the developed hardware coprocessor reduces the fast Fourier transform algorithm execution time and the load of the pipelined processor during calculations.

KEYWORDS: RISC-V, processor, digital signal processing, fast Fourier transform, pipelined, coprocessor, FPGA.

REFERENCES

1. Patterson D. A., Hennessy J. L. Computer Organization and Design. The Hardware/Software Interface: RISC-V Edition. California, Morgan Kaufmann Publishers, 2018, 1665 p.
2. The NEORV32 RISC-V Processor. Access mode: https://github.com/stnolting/neorv32
3. The Potato Project. Access mode: https://github.com/skordal/potato
4. Keeth B., Baker R. J., Johnson B., Lin F. DRAM Circuit Design : Fundamental and High-Speed Topics. Hoboken, John Wiley & Sons, 2007, 440 p.
5. Memory Performance in a Nutshell. Access mode: https://www.intel.com/content/www/us/en/developer/articles/technical/memory-performance-in-a-nutshell.html
6. Lorenser T. The DSP capabilities of ARM Cortex-M4 and Cortex-M7 Processors. Access mode: https://community.arm.com/cfs-file/__key/communityserver-blogs-components-weblogfiles/00-00-00-21-42/7563.ARM-white-paper-_2D00_-DSP-capabilities-of-Cortex_2D00_M4-and-Cortex_2D00_M7.pdf
7. Digital signal processing for STM32 microcontrollers using CMSIS, *STMicroelectronics*, 2018, AN4841 Rev 2, 25 p.
8. Sapiecha K., Jarocki R. Modular architecture for high performance implementation of FFT algorithm, *ACM SIGARCH Computer Architecture New*, 1986, Vol. 14(2), pp. 261–270. DOI: 10.1145/17356.17387
9. Gonzalez-Concejero C., Rodellar V., Alvarez-Marquina A., Icaya E. M. de, Gomez-Vilda P. A portable hardware design of a FFT algorithm, *Latin American applied research*, 2007, Vol. 37(1), pp. 79–82.
10. Slade G. The Fast Fourier Transform in Hardware: A Tutorial Based on an FPGA Implementation. Access mode: https://web.mit.edu/6.111/www/f2017/handouts/FFTtutorial121102.pdf
11. Radix-2 FFT – VHDL Implementation. Access mode: https://github.com/bugratufan/radix2-fft-vhdl
12. TMS320C55x Technical Overview. Access mode: https://www.ti.com/lit/ug/spru393/spru393.pdf?ts=1693041737254
13. McKeown M. FFT Implementation on the TMS320VC5505, TMS320C5505, and TMS320C5515 DSPs. Access mode: https://www.ti.com/lit/an/sprabb6b/sprabb6b.pdf?ts=1695025017385
14. Lyons R. G. Understanding Digital Signal Processing, Third Edition. Boston, Pearson Education, 2011, 858 p.
15. Ifeachor E. C., Jervis B. W. Digital Signal Processing: A Practical Approach, Second Edition. Boston, Addison Wesley, 1993, 779 p.
16. Spartan-6 Family Overview. Access mode: https://docs.xilinx.com/v/u/en-US/ds160
17. Spartan-6 FPGA Data Sheet: DC and Switching Characteristics. Access mode: https://docs.xilinx.com/v/u/en-US/ds162
18. Vijayakanthan K., Hemachandran K., Anand M., Hemachandran K. High Throughput and Mixed Radix N-Point Parallel Pipelined FFT VLSI Architectures for Advanced Wireless Communication, *International Journal of Grid and Distributed Computing*, 2020, Vol. 13(1), pp. 400–411.

DOI 10.15588/1607-3274-2024-1-18