УДК 004.31

КОНВЕЄРНА МОДЕЛЬ І РЕАЛІЗАЦІЯ СУМАТОРА ДІЙСНИХ ЧИСЕЛ НА FPGA

І. Я. Зеленьова, Т. В. Голуб, Т. С. Дьячук, А. Є. Діденко
Національний університет «Запорізька політехніка»

Анотація. *Пропонується спосіб підвищення ефективності операції складання дійсних чисел з використанням принципу конвеєризації. Схема спроектована для реалізації на мікросхемах типу FPGA, що дозволяє використовувати архітектурні особливості даного базису, зокрема можливість паралельного виконання задач на одному кристалі. Кожний блок запропонованої конвеєрної структури на FPGA описано та реалізовано, як окремий проект цифрового функціонального вузла. Проаналізовані результати експериментального тестування конвеєрного суматора на різних мікросхемах фірми Altera/Intel.*

Ключові слова: *дійсні числа, конвеєризація, FPGA, продуктивність обчислень, суматор.*

Вступ

Широке використання дійсних чисел обумовлене тим, що при вирішенні значної частини задач фізики, математики або інформатики необхідно представляти результат з якомога більш високою точністю.

Дійсні числа зазвичай представляються у так званому експоненціальному записі. При переведенні дійсного числа в двійкову систему числення виникають певні складності. По-перше, неможливо записати число, яке в десятковій системі числення має нескінченну дробову частину (ірраціональне число). По-друге, деякі раціональні числа не мають в двійковій системі кінцевого представлення, тобто такі числа можуть мати в даній системі нескінченну дробову частину.

Зважаючи на те, що кількість розрядів мантиси може мати тільки кінцеву довжину, в комп'ютері можна представити тільки кінцеву множину дійсних чисел.

Стандарт IEEE Std 754™-2008 визначає формат чисел з рухомою комою та методи арифметичних операцій з ними в обчислювальних машинах [1, 2, 7]. Це найбільш поширений стандарт, який гарантує однаковість виконання операцій з числами з рухомою комою на програмному та апаратному рівнях. Двома основними типами чисел з рухомою крапкою в форматі IEEE Std 754™-2008 є тип float (одинарна точність) та тип double (подвійна точність). В даній роботі розглядаються тільки нормалізовані числа з одинарною точністю, оскільки арифметичні операції над числами float та double не відрізняються.

З метою розширення сфери використання комп'ютерів, а також підвищення якості обчислення, в сучасних комп'ютерних системах широко використовуються математичні співпроцесори (FPU), основними задачами яких є обробка чисел з рухомою комою. На даний момент основна кількість сучасних процесорів мають вбудовані блоки, які виконують складні та різноманітні арифметичні операції над числами з рухомою комою [1, 4, 7].

Поряд з цим розробка цифрових пристроїв на програмованих логічних інтегральних схемах (ПЛІС), зокрема на FPGA, стрімко розвивається та істотно спрощується за рахунок використання спеціальних технічних рішень – IP (Intellectual Property) блоків, тобто модулів, параметри яких настроюються користувачем на конкретні вимоги розроблюваного проекту. Цьому, в свою чергу, сприяє розвиток та впровадження в практику проектування мов опису апаратури (VHDL і Verilog) [3, 9, 13].

Реалізація арифметики над числами з рухомою комою на FPGA, має такі переваги, як гнучкість використання та низькі витрати на виробництво [3, 5]. Пристрій на FPGA можна пристосовувати для конкретних потреб проекту, що зменшує собівартість кінцевого продукту та надає можливість вирішувати задачі, для яких не існує готових рішень у вигляді представлених на ринку пристроїв. На даний момент FPGA стрімко розвиваються, що робить їх більш придатними для використання у сферах, де необхідно проводити значну кількість обчислень з максимальною точністю, а саме в наукових дослідженнях, в сфері цифрової обробки сигналів та ін. [3, 5, 6, 8].

В даній роботі пропонується спосіб підвищення ефективності операції додавання дійсних чисел з використанням конвеєризації [5, 7, 13, 14]. Схема спроектована для реалізації на мікросхемах типу FPGA або ProASIC, що дозволяє використовувати архітектурні особливості дано-

го базису, зокрема можливість паралельного виконання задач на одному кристалі.

1. Аналіз проблеми

В цифровій техніці для зручного представлення дуже великих або дуже малих чисел, а також для уніфікації їх написання та виконання арифметичних операцій використовується експоненціальний запис – представлення дійсних чисел у вигляді мантиси та порядку, а також використовується поняття нормалізованого запису числа [1].

Дійсне число (число з рухомою крапкою) в двійковій системі має наступний вигляд:

$$N = (-1)^S M \cdot B^E, \quad (1)$$

де N – число, що записують; M – мантиса; S – знак числа; E – порядок числа.

Для чисел з рухомою крапкою нормалізація в двійковій системі набуває вигляду:

$$1 \le M < 2 . \quad (2)$$

Таким чином, унікальність представлення дійсного числа підтримується і в двійковій системі числення. Але ж в комп'ютері не можливо виділити під дробову частину нескінченну кількість біт. Тому обмежуються певною точністю *p* (precision). Через такі обмеження при арифметичних операціях може виникати так зване «underflow», тобто число, яке занадто мале, щоб бути представлене в комп'ютері. Для того, щоб наочно представити проблему, введемо поняття одиниці на останній позиції (ulp, unit in the last place) та машинного нуля [1, 2].

Нехай в форматі з рухомою крапкою записане число 1, мантиса якого має *p* біт: $1.000\ldots000_{(2)}$. Найменше число, яке буде більше за 1, буде число $1.000\ldots001_{(2)}$. В загальному вигляді таке число можна представити наступною формулою:

$$1 + 2^{-(p-1)} , \quad (3)$$

де p – кількість біт мантиси (precision).

Машинний нуль (машинний епсілон, ε) є проміжком між числом, яке задається формулою (2) та числом 1:

$$\varepsilon = 2^{-(p-1)} . \quad (4)$$

Одиниця на останній позиції вираховується за формулою:

$$ulp = 2^{-(p-1)} \cdot 2^{E} = \varepsilon \cdot 2^{E}, \quad (5)$$

де E – експонента числа [1].

Щоб подолати описану вище проблему «underflow», вводяться денормалізовані числа (проміжок від -0,5 до 0,5), мантиса яких має значення менше нуля в машинному вигляді. Обробка денормалізованих чисел в обчислювальному приладі потребує ускладнення алгоритму арифметичних операцій, що призводить до ускладнення самої схеми приладу [1, 8].

Широко використовуваний в сучасній обчислювальній техніці формат IEEE Std 754™-2008, встановлює такі обмеження:

– єдине представлення чисел всіма обчислювальними машинами, які використовують стандарт;
– операції округлення чисел;
– види та способи представлення виняткових ситуацій (ділення на нуль та ін.).

На рис. 1 представлені числа з рухомою крапкою у форматі IEEE 754, де

– S – знаковий розряд (1 біт);
– E – розряди експоненти;
– M – розряди мантиси;
– b – кількість біт експоненти;
– m – кількість біт мантиси.

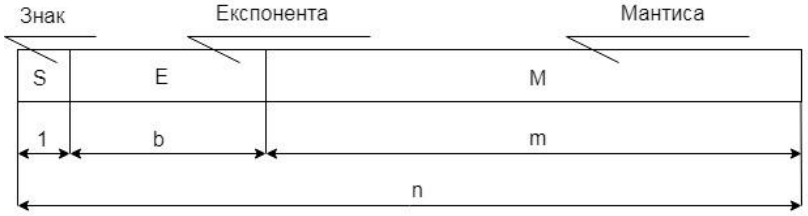


Рис.1. Число з рухомою крапкою в форматі IEEE 754

Для чисел з одинарною точністю n = 32, b = 8, m = 22. Для чисел з подвійною точністю n = 64, b = 11, m = 52.

Потрібно звернути увагу, що в представленні нормалізованих чисел присутня неявна одиниця. Така особливість, по-перше, робить число відповідним умові (1), а по-друге, дозволяє зберігати на один біт більше інформації.

Ще однією особливістю чисел в форматі IEEE Std 754™-2008 є наявність зміщення (bias) у експоненти, що дорівнює 127 (для float) або 1023 (для double). Це дозволяє уникати запису

від'ємної експоненти, а також полегшує порівняння чисел.

Таким чином, підсумовуючи вище сказане, можна дати формулу, яка представляє нормалізоване число з рухомою крапкою з одинарною точністю в форматі IEEE Std 754™-2008:

$$N = (-1)^S \cdot 2^{E-127} \cdot 1.M, \qquad (6)$$

де N – число типу float; M – мантиса без неявної одиниці; S – знак числа; E-127 – експонента зі зміщенням.

Найменше число, яке можна представити в типі float $N_{min}=1.000\ldots000_{(2)}\cdot 2^{-126} = 2^{-126} \approx 1{,}2\cdot 10^{-38}$.

Найбільше число, яке можна представити в типі float $N_{max}=1.111\ldots111_{(2)}\cdot 2^{127} \approx 2^{128} \approx 3{,}4\cdot 10^{38}$.

Оскільки в форматі IEEE 754 для нормалізованих чисел завжди присутня неявна одиниця, то виникає питання, як представити число 0.

Формат IEEE Std 754™-2008 визначає не тільки представлення нуля, а й інших виняткових чисел.

Оскільки результат деякої арифметичної операції над числами з рухомою крапкою може не вміщатися у виділену кількість біт для мантиси, то вдаються до операції округлення.

Алгоритм додавання двох нормалізованих чисел з рухомою крапкою наступний [1, 2]:

1) Порівняти експоненти двох чисел. Менше з чисел зсунути праворуч на кількість біт, що дорівнюють різниці експонент. Якщо число виходить за розрядну сітку, то молодші біти втрачаються.

2) Додати мантиси (зі знаками) двох чисел. Якщо числа негативні, то попередньо представити їх в доповняльному коді, інвертувавши мантису та додавши до молодшого розряду 1.

3) Якщо при додаванні відбувся перенос до знакового розряду, то необхідно виконати зсув на 1 розряд праворуч, збільшивши при цьому експоненту на 1. Інакше, перейти до пункту 4).

4) Якщо результат негативний, то тоді відповідь представлена у доповняльному коді. Необхідно представити число в прямому коді, для чого мантису інвертують і додають до неї 1. Інакше, перейти до пункту 5).

5) Якщо результат складання двох мантис в прямому коді виявився денормалізованим, тобто в старшому біті мантиси знаходиться 0, то необхідно виконати зсув на 1 розряд ліворуч та зменшити експоненту на 1.

6) Якщо після виконання пункту 5) результат залишився денормалізованим, знову повторити пункт 5). Інакше, перейти до пункту 7).

7) Провести операцію округлення.

8) Представити отримане число так, щоб воно відповідало умові (5).

Реалізація даного алгориму пов'язана із великою кількістю однотипних дій, які виконуються ітераційно. З одного боку, це вимагає багато машинного часу, а з іншого – певна регулярність обчислень добре підходить для реалізації на відповідних регулярних структурах, притаманних програмованим логічним інтегральним схемам, зокрема FPGA. Внутрішня архітектура FPGA в багатьох випадках дозволяє швидше і краще оптимізувати проекти з урахуванням різних критеріїв [3, 5, 13]. Для високої продуктивності виконання таких обчислень і алгоритмів, котрі дозволяють застосовувати широкий паралелізм, FPGA можуть дати багато переваг, тому в новітній літературі розглянуті такі застосування апаратних прискорювачів [9-14, 18], для яких, по-перше, висока продуктивність є основним завданням і основне правило «чим швидше, тим краще»; по-друге, можуть бути запропоновані високопаралельні та конвеєрні архітектури; по-третє, класичні відомі методи не можуть бути використані, тому що цільові вимоги є спеціалізованими.

2. Постановка задачі та мета роботи

Таким чином, актуальною є задача розробки цифрового пристрою, який виконує арифметичні дії над числами з рухомою крапкою, а саме додавання та віднімання, а також має бути портативним, високопродуктивним та відносно недорогим. Метою даної роботи є розробка конвеєризованої структури та ефективних функціональних блоків пристрою, які раціонально використовують ресурси FPGA і забезпечують високу швидкодію, що дозволяє використовувати пристрій в схемах, де необхідна висока продуктивність обчислень за умови мінімальних апаратних витрат.

3. Розробка базової структури суматора дійсних чисел на FPGA

Базова структура суматора чисел з рухомою крапкою, розроблена відповідно до стандартного алгоритму обробки чисел в форматі IEEE Std 754™-2008, представлена на рис. 2. Структура орієнтована на проектування в базисі FPGA, тому має чітко виділені функціональні блоки, котрі можна поведінково визначити з використанням мови опису апаратури VHDL. Розглянемо сутність кожного блоку базової структури.

Блок генерування сигналу виняткової ситуації необхідний для виявлення чисел, з якими не можна проводити арифметичні операції. Він сигналізує про події, пов'язані з появою денормалізованого числа, нескінченності або особливого значення числа з рухомою комою (NaN – Not-a-Number) під час виконання додавання/віднімання

та при запису результату [1, 2].

Блок порівняння експонент необхідний для виявлення більшої з експонент та визначення різниці, на яку треба виконати зсув числа з меншою експонентою.

Блок зсуву мантис приводить два числа до одного порядку, попередньо розширивши число для зберігання знаку, неявної одиниці та бітів, необхідних для округлення.

Рис. 2. Структурна схема суматора дійсних чисел на FPGA

Блок інвертування мантис призначений для операцій з від'ємними числами. Він виконує операцію доповнення до 2, інвертуючи мантису та додаючи до молодшого розряду 1 (доповняльний код). Якщо два числа є від'ємними, то мантиси цих двох чисел не відображаються у доповняльному коді, а представляються як додатні, але знак результату при цьому завжди буде від'ємним.

Суматор представляє собою пристрій для складання двох двійкових чисел. Схема генерує сигнал переповнення, тобто виходу старшого біту результату в знакову частину.

Блок нормалізації результату необхідний для приведення результату до стандартного вигляду, у якому в старшому розряді присутня одиниця. При цьому від'ємні числа попередньо інвертуються (переводяться в прямий код). У цьому блоці також відбувається визначення знаку результату.

Блок округлення використовується для запису результату з максимальною точністю. Число округлюється в залежності від трьох останніх додаткових бітів.

В блоці запису результату вихідне число набуває вигляду, який описує стандарт IEEE Std 754™-2008. Якщо в першому блоці був згенерований сигнал про виняткову ситуацію або сам результат став числом, яке не можна представити у даному форматі, то при записі вихідне число набуде відповідного вигляду (нуль, нескінченність, NaN).

4. Конвеєризація базової структурної схеми

На рис. 2 представлена схема, яка має тільки послідовні кроки виконання операції складання/віднімання на суматорі чисел з рухомою крапкою. Деякі операції не потребують послідовного виконання, а можуть паралельно виконуватися з іншими [7]. Виходячи з цього, запропоновано конвеєризовану структурну схему (рис. 3), котра є функціонально адекватною базовій структурі (рис.2), але розділена на окремі рівні, які виконуються паралельно, а результат виконання операції кожного рівня не залежить від результату виконання операції попереднього рівня в один і той самий момент часу. Така організація роботи дозволяє виконувати операції складання/віднімання на декількома числами одночасно, що окрім підвищення швидкості роботи, також збільшує пропускну здатність суматора. Порівнюючи схему на рисунку 2 та розроблену конвеєризовану схему на рис. 3, можна визначити наступні відмінності.

По-перше, блок генерації сигналу випадкової події розділений на два окремі блоки. Перший блок, блок перевірки на виняткову подію, необхідний для опрацювання виняткової події перед початком обчислень. Другий блок генерації сигналу виняткової події необхідний для перевірки результату обчислень. Блок порівняння експонент та блок перевірки на виняткову подію знаходяться на одному рівні, бо результат обчислень одного блоку не впливає на результат обчислень другого. За цим принципом було побудовано й інші рівні в схемі.

Блок зсуву, блок інвертування мантис та суматор мають ті ж самі функції, що й відповідні блоки в схемі на рисунку 2.

Знаковий блок виконує обчислення знаку результату.

Блок нормалізації був розділений на 3 окремі блоки: блок зсуву результату, блок обробки експоненти та LOD.

LOD (leading one detector, детектор старшої одиниці) – пристрій, що підраховує, на скільки позицій ліворуч потрібно зсунути мантису для того, щоб вона прийняла нормалізований вигляд.

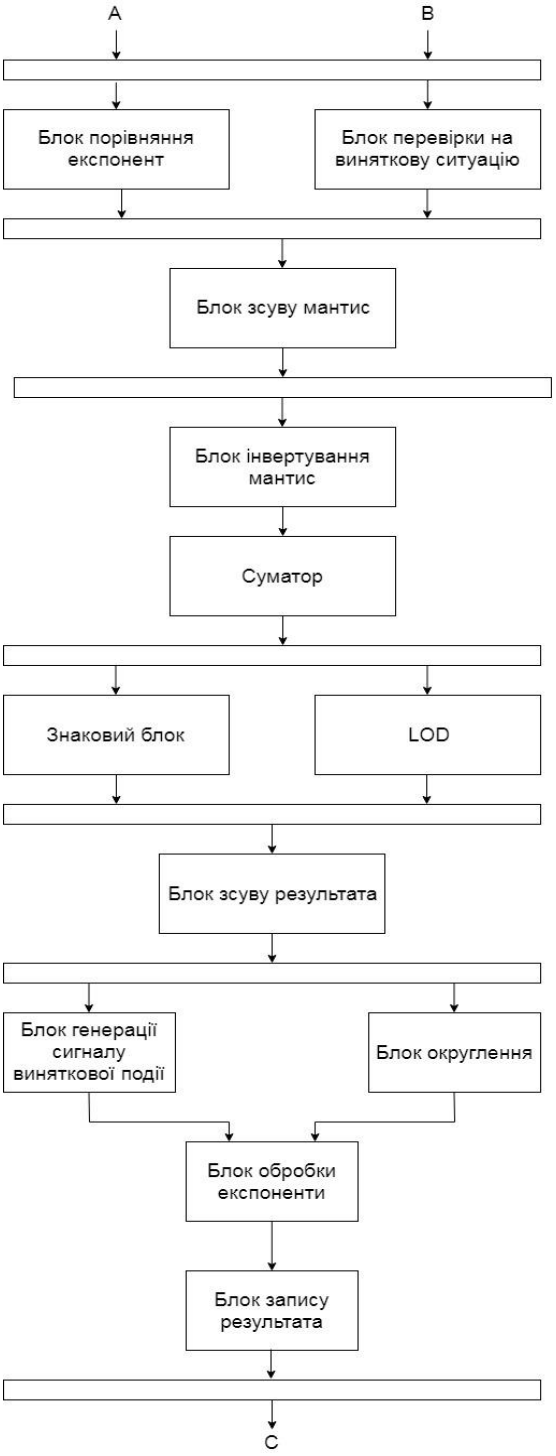


Рис. 3. Конвеєризована схема суматора чисел з рухомою крапкою

Блок зсуву виконує збільшення експоненти на одиницю у разі, коли результат не вміщається в розрядну сітку, або зменшення при операції зсуву експоненти ліворуч.

Блок округлення виконує округлення результату для досягнення максимальної точності.

Блок запису результату необхідний для належного представлення числа в форматі з рухомою комою, який визначається стандартом IEEE Std 754™-2008. Якщо при виконанні обчислень було згенеровано сигнал про виняткову подію, тобто число не може бути представлене в форма-

ті з рухомою комою, то даний блок представляє результат обчислень в одному із зарезервованих виглядів.

Принцип побудови та функціонування внутрішньої структури окремих блоків розглянемо, зокрема, на прикладі блоку детектору старшої одиниці (рис. 4), до складу якого також входить інвертор від'ємного результату.

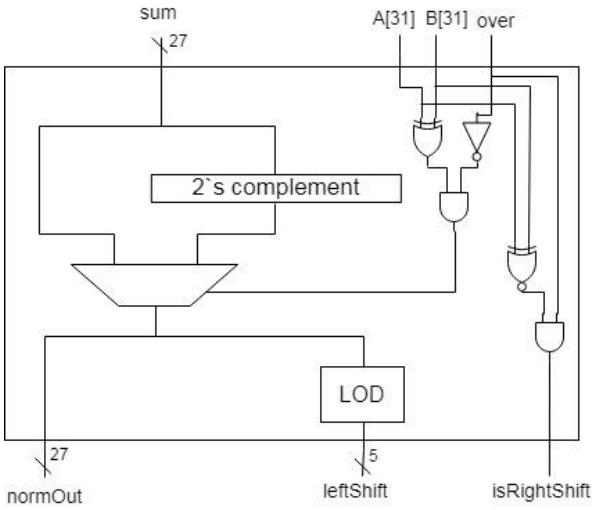


Рис.4. Блок детектору старшої одиниці

На входи даного блоку надходять знаки обох чисел (A[31], B[31]), сигнал переносу в знакову частину «over» та сума «sum».

Головною складовою даного блоку є детектор старшої одиниці (LOD). Його головною функцією є визначення кількості нулів в старших розрядах, тобто визначення величини зсуву ліворуч для нормалізації мантиси (вихід «leftShift»).

За основу функціонального опису LOD був взятий програмний код на VHDL з роботи [19]. В цій схемі використовується 30 логічних елементів. Для опису LOD в даній роботі використано авторський, оптимізований код, за яким синтезується схема, що має кращі характеристики, а саме – використовується 26 логічних елементів, що на 13% менше базового варіанта.

Для нормалізації результату складання (віднімання) було розроблено блок зсуву, що виконує операцію логічного зсуву ліворуч або праворуч в залежності від значення на вході «isRightShift».

Для визначення знаку результату було розроблено знаковий блок, що обчислює знак результату в залежності від входів «sa», «sb», «over». Знак результату буде від'ємним тільки тоді, коли обидва числа від'ємні, або коли одне з чисел від'ємне та по модулю більше за додатне (просигналізує відсутність сигналу переносу).

Наступним є блок округлення чисел до найближчого парного. Даний блок має також вихід «round_sh», що сигналізує про переповнення після округлення. На одному рівні з цим блоком в конвеєрі знаходиться блок генерації сигналу виняткової події.

Далі в схемі конвеєра використовується блок обробки експонент, котрий функціонує наступним чином. На вхід «toLeft» подається кількість бітів, на яку необхідно зменшити експоненту результату. На вхід toRight подається сигнал про збільшення експоненти результату на одиницю, що може виникнути, коли старша одиниця результату вийшла в знаковий біт. Збільшення експоненти спостерігається, коли після округлення виникло переповнення, про що буде сигналізувати вхід «round_to_right». В іншому випадку експонента зменшується на відповідну кількість біт (при цьому саме число зсувається ліворуч).

Виходи «underflow» та «overflow» необхідні для позначення виняткової події.

Якщо при зрушенні числа праворуч та збільшенні експоненти на 1 вона стала дорівнювати 255 (11111111 в двійковому вигляді), генерується сигнал «overflow», який повідомляє про те, що результат виявився занадто великим, щоб поміститися в розрядну сітку.

Якщо при зрушенні числа ліворуч та зменшенні експоненти на відповідну кількість біт вона стала від'ємною, то генерується сигнал «underflow», який повідомляє про те, що результат виявився занадто малим, щоб бути представленим в форматі IEEE754.

При «overflow» результат приймає значення нескінченності, а при «underflow» – значення нуля.

Наступний блок - блок генерації сигналу ви-

няткової події, необхідний для визначення чисел, які не можна опрацьовувати в форматі з рухомою крапкою, а саме з нескінченністю та NaN.

Кожний блок запропонованої конвеєрної структури на FPGA реалізовано, як окремий проект цифрового функціонального вузла, і таким чином, загальна задача розділена на окремі підзадачі, що полегшує експериментальне тестування та відлагодження складових усього пристрою.

5. Результати експериментів

Експериментальні дослідження виконані з використанням пакету EDA Quartus II [9, 17] Розроблена схема була промодельована на FPGA сімейства Stratix III [15]. Аналогом розробленої схеми є функціонально схожий пристрій фірми Altera [16, 19]. Порівняльна характеристика наведена у таблиці 1.

З таблиці 1 видно, що при однаковій кількості тактів, було зменшено кількість ALUT на 8.6%, а тактову частоту збільшено на 37.6%.

Розроблена схема була також промодельована на FPGA сімейства Cyclone III. Порівняльна характеристика власного пристрою та пристрою фірми Altera наведена у таблиці 2.

З таблиці 2 видно, що при однаковій кількості тактів, тактова частота розробленого пристрою була збільшена на 12.3%.

Таблиця 1

Порівняльна характеристика синтезованого конвеєрного пристрою на FPGA Stratix III та серійного пристрою фірми Altera

Пристрій	cycles	ALUTs	Dedicated registers	Fmax, MHz
ALTFP_ADD_SUB megafunction	7	592	376	218
Синтезований пристрій	7	541	373	300

Таблиця 2

Порівняльна характеристика синтезованого конвеєрного пристрою на FPGA Cyclone III та серійного пристрою фірми Altera

Пристрій	cycles	Logic elements	Dedicated logic registers	Fmax, MHz
ALTFP_ADD_SUB megafunction	7	822	346	154
Синтезований пристрій	7	841	344	173

На рис. 5 зображено результати роботи схеми, отримані під час симуляції в програмі Quartus II.

В прикладі на рис. 5 складаються два числа 2.67852483148 та 5.78333123848. В результаті отримується число 8.46185588837. Абсолютна похибка з реальним результатом в 8.46185606996 складає 0.00000018159, а відносна – приблизно 0.00000215%.

На рис. 6 зображено результат віднімання чисел 2.67852483148 та 5.78333123848.

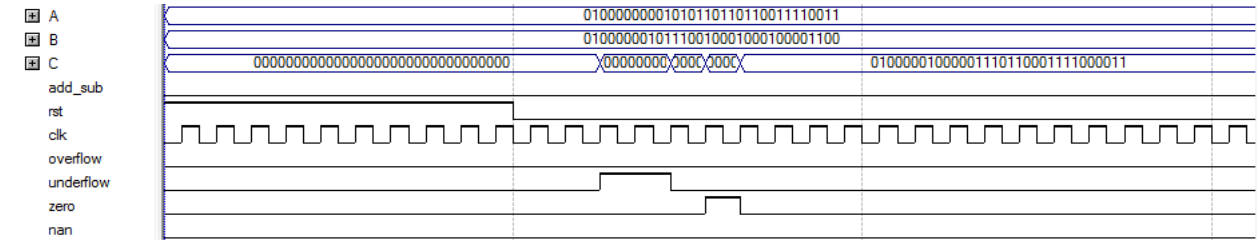


Рис. 5. Додавання двох чисел з рухомою комою

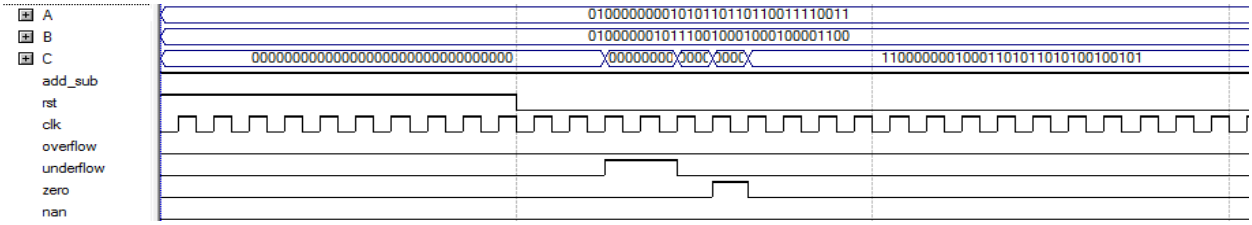


Рис. 6. Віднімання двох чисел з рухомою комою

В результаті отримується число - 3.10480618477. Абсолютна похибка з реальним результатом в -3.10480640700 складає 0.00000022223, а відносна – приблизно 0.00000716%.

Висновки

Таким чином, в результаті виконання роботи було спроектовано пристрій, який виконує операції складання та віднімання над числами з рухомою комою, представленими в форматі IEEE754.

Для покращення характеристик суматора, його було конвеєризовано, тобто розбито на рівні, кожен з яких виконує певну дію над числами. Це дозволило не тільки збільшити пропускну здатність, а й зробило суматор придатним для використання в сучасних синхронних схемах.

В порівнянні з пристроєм фірми Altera, що має такий самий функціонал, було збільшено тактову частоту на 37.6% на FPGA сімейства Stratix III та на 12.3% на FPGA сімейства Cyclone III.

Розроблений суматор має широку сферу використання, оскільки на даний час майже жоден з обчислювальних пристроїв не обходиться без обробки чисел з рухомою крапкою. На сучасному ринку представлений широкий вибір складних функціональних блоків, таких як модулі IP (Intellectual Property) [20]. Але ж при їх використанні в проектах виникають певні проблеми, пов'язані як із ускладненням маршруту проектування, так і з підвищенням кінцевої вартості пристрою [5, 21]. Запропонована модель суматора є досить простою в реалізації на FPGA та може бути використана при розробці бібліотечного IP-модуля у проектах, в яких складний функціонал пристроїв IP core, існуючих на ринку, є надлишковим для конкретної задачі.

Обчислювальні функціональні вузли, реалізовані на FPGA, мають певні переваги порівняно з мікропроцесорами, в тому сенсі, що вони можуть бути спроектовані для конкретної задачі, що сприяє прискоренню обробки даних та зменшенню собівартості готового пристрою.

Список використаної літератури

1. Michael, L. Overton, Numerical Computing with IEEE Floating Point Arithmetic [Text] / Overton, Michael L // Society for Industrial and Applied Mathematics. – Philadelphia, 2001. – p.14.

2. IEEE Std 754™-2008 (Revision of IEEE Std 754-1985) [Text] / IEEE Standard for Floating-Point Arithmetic. – NY, USA, 2008 – 70p.

3. Castillo, Arturo Barrabés, Design of single precision float adder (32-bit numbers) according to IEEE754 standard using VHDL: Master Thesis [Text] / Arturo Barrabés Castillo – Slovenská Technická Univerzita, Bratislava, 2012. – 30p.

4. Baranov, S. Logic and System Desing of Digital Systems [Text] / S. Baranov. – Tallinn, 2008 – 267p.

5. Танненбаум, Э. Архитектура компьютера [Текст] / Э.Танненбаум – М.: Питер, 2009. – 844с.

6. Synthesis, and optimization of FPGA based systems [Text] / Sklyarov V, Skliarova I, Barkalov A, Titarenko L – Springer, Berlin, 2014 – 432p.

7. Barkalov, A. Implementing on the field programmable gate array of combined finite state machine with counter [Text] / A. Barkalov, L. Titarenko, I. Zeleneva, S. Hrushko // Conference Proceedings of 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies DESSERT'2018 Ukraine, Kyiv, May 24-27, 2018. – P.274–282.

8. Pedroni, V Compact Hamming-comparator-based rank order filter for digital VLSI and FPGA implementations [Text] / V. Pedroni // Proceedings of the IEEE international symposium on circuits and systems—ISCAS'2004, Vancouver, BC, Canada. – May 2004, – P.585–588

9. Проектування комп'ютерних систем на основі мікросхем програмованої логіки [Текст] / С. Іванець, Ю. Зубань, В. Казимир, В. Литвинов; – Суми, 2007. – 312 с.

10. A practical reconfigurable hardware accelerator for Boolean satisfiability solvers [Text] / JD Davis, Z Tan, F Yu, L Zhang // Proceedings of the 45th ACM/IEEE design automation conference — DAC'2008, Anaheim, California, USA, June 2008, – P.780–785.

11. Баркалов, А. А. Исследование автомата с программируемой логикой в составе цифровой информационно-управляющей системы на FPGA [Text] / А. А. Баркалов, Л. А. Титаренко, И. Я. Зеленева, С. С. Грушко// ПРОБЛЕМЫ РЕГИОНАЛЬНОЙ ЭНЕРГЕТИКИ (специальный выпуск). 2019. – (SI) 2019, № 1–2. – С.36–45.

12. Forecast overview: semiconductors, worldwide. 2018 update [Electronic Resource]/ Gartner Inc. – Access Mode: https://www.gartner.com/en/documents/3889873/forecast-overview-semiconductors-worldwide-2018-update

13. Sklyarov, V Synthesis of circuits and systems from hierarchical and parallel specifications [Text] / V. Sklyarov // Proceedings of the 12th Biennial Baltic electronics conference – BEC'2010, – Tallinn, Estonia, Oct 2010, – P. 389–392

14. Sklyarov, V Parallel processing in FPGA-based digital circuits and systems [Text] / V. Sklyarov, I. Skliarova // Tallinn: "TUT Press",

2013, – 346с.

15. Intel Corp. Stratix V device overview [Electronic Resource]. – Access Mode: https://www.intel.com/content/dam/www/ programmable/us/en/pdfs/literature/hb/stratix-v/stx5_51001.pdf.

16. Altera documentation. 2018. [Electronic Resource]. – Access Mode: www.altera.com/ support/literature /lit- index. html.

17. Описание САПР Quartus II, Основные этапы проектирования СБИС ПЛ - Формирователь OFDM сигнала на ПЛИС стандарта 802.16d – [Електронний ресурс] Режим доступу: http://www.techexpose.ru/texob-544.html

18. Xilinx Inc. (2016) 7 Series FPGAs configurable logic block [Electronic Resource]. – Access Mode: https://www.xilinx.com/support/documentation/user_guides/ug474_7Series_CLB.pdf

19. Altera, Floating-Point Megafunctions, User Guide [Text] / San Jose, 2011. – p.8.

20. The reference community for Free and Open Source gateware IP cores [Electronic Resource]. – Access Mode: https://opencores.org/

21. Палагин, А. В., Особенности проектирования компьютерных систем на кристалле ПЛИС / А. В. Палагин, Ю. С. Яковлев // Математичні машини і системи, 2017 – № 2 – с.3–14. ISSN 1028-9763.

References

1. Michael, L. Overton, (2001), "Numerical Computing with IEEE Floating Point Arithmetic", *Society for Industrial and Applied Mathematics*, Philadelphia, p. 14.

2. IEEE Std 754™-2008 (Revision of IEEE Std 754-1985) (2008) *IEEE Standard for Floating-Point Arithmetic*, IEEE 3 Park Avenue New York, NY 10016-5997, USA, 2008, 70p.

3. Castillo, Arturo Barrabés, (2012) "Design of single precision float adder (32-bit numbers) according to IEEE754 standard using VHDL: Master Thesis" Slovenská Technická Univerzita, Bratislava, 2012. – p. 30.

4. Baranov, S., (2008) *Logic and System Desing of Digital Systems*, Tallinn, 267 p.

5. Tannenbaum, E (2009),. *Computer architecture* [*Arkhitektura komp'yutera*], Piter, M., 844p.

6. Sklyarov, V, Skliarova, I, Barkalov, A, Titarenko, L., (2014) *Synthesis and optimization of FPGA based systems*, Springer, Berlin, 432 p.

7. Barkalov, A., Titarenko, L., Zeleneva, I., Hrushko, S. (2018) "Implementing on the field programmable gate array of combined finite state machine with counter", *Proceedings of 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies DESSERT'2018, May 24-27, 2018*, Kyiv, Ukraine, pp. 274–282.

8. Pedroni, V (2004) "Compact Hamming-comparator-based rank order filter for digital VLSI and FPGA implementations", *Proceedings of the IEEE international symposium on circuits and systems—ISCAS'2004, May 2004*, Vancouver, BC, Canada. pp. 585–588.

9. Ivanets′, S., Zuban′, YU., Kazymyr, V., Lytvynov, V. (2007) "Computer systems design based on programmable logic circuits" ["Proektuvannya komp'yuternykh system na osnovi mikroskhem prohramovanoyi lohiky"], Sumy, 312 p.

10. Davis, JD, Tan, Z, Yu, F, Zhang, L, (2008) "A practical reconfigurable hardware accelerator for Boolean satisfiability solvers" *Proceedings of the 45th ACM/IEEE design automation conference — DAC'2008, June 2008*, Anaheim, California, USA, pp. 780–785.

11. Barkalov, A. A., Titarenko, L. A., Zeleneva, I. Ya., Hrushko, S. S. (2019) "Research of the Finite State Machine with Programmable Logic as a Part of Digital Information and Control System based on FPGA" [Issledovaniye avtomata s programmiruyemoy logikoy v sostave tsifrovoy informatsionno-upravlyayushchey sistemy na FPGA] *Problemele energeticii regionale*, №1–2.–pp. 36–45.

12. Forecast overview: semiconductors, worldwide. 2018 Gartner Inc., available at: https://www.gartner.com/en/documents/3889873/forecast-overview-semiconductors-worldwide-2018-update (accessed 2 March 2020)

13. Sklyarov, V (2010) "Synthesis of circuits and systems from hierarchical and parallel specifications" *Proceedings of the 12th Biennial Baltic electronics conference – BEC'2010, Oct 2010*, Tallinn, Estonia, pp. 389–392

14. Sklyarov, V, Skliarova, I., (2013) *Parallel processing in FPGA-based digital circuits and systems*, TUT Press, Tallinn, 346 p.

15. Intel Corp. Stratix V device overview, available at: https://www.intel.com/content/dam/www/ programmable/us/en/pdfs/literature/hb/stratix-v/stx5_51001.pdf. (accessed 2 March 2020)

16. Altera documentation. (2018), available at: https://www.altera.com/ support/literature /lit- index. html. (accessed 2 March 2020)

17. Description of CAD Quartus II, The main stages of the design of VLSI submarine - OFDM signal generator on the 802.16d standard [Opisaniye SAPR Quartus II, Osnovnyye etapy proyektirovaniya SBIS PL - Formirovatel' OFDM signala na plis standarta 802.16d], available at: http://www.techexpose.ru/texob-544.html (accessed 2 March 2020)

18. Xilinx Inc. (2016) "7 Series FPGAs configurable logic block", available at: https://www.xilinx.com/support/documentation/user_guides/ug474_7Series_CLB.pdf (accessed 2 March 2020)

19. Altera, Floating-Point Megafunctions, User Guide (2011), San Jose, p. 8.

20. The reference community for Free and Open Source gateware IP cores [Electronic Resource]. – Access Mode: https://opencores.org/ (accessed 2 March 2020)

21. Palagin, A. V., Yakovlev, Yu. S. (2017) "Features of designing computer systems on FPGA crystal [Osobennosti proektirovaniya kompyuternih system na kristale PLIS]", *Mathematical Machines and Systems [Mathematical Machines and Systems]*, № 2, p.3–14.

CONVEYOR MODEL AND IMPLEMENTATION OF THE REAL NUMBERS ADDER ON FPGA

I. Ya. Zeleneva, T. V. Golub, T. S. Diachuk, A. Ye. Didenko
Zaporizhzhia Polytechnic National University

Abstract. *The purpose of these studies is to develop an effective structure and internal functional blocks of a digital computing device – an adder, that performs addition and subtraction operations on floating-point numbers presented in IEEE Std 754™-2008 format. To improve the characteristics of the adder, the circuit uses conveying, that is, division into levels, each of which performs a specific action on numbers. This allows you to perform addition / subtraction operations on several numbers at the same time, which increases the performance of calculations, and also makes the adder suitable for use in modern synchronous circuits.*

Each block of the conveyor structure of the adder on FPGA is synthesized as a separate project of a digital functional unit, and thus, the overall task is divided into separate subtasks, which facilitates experimental testing and phased debugging of the entire device. Experimental studies were performed using EDA Quartus II. The developed circuit was modeled on FPGAs of the Stratix III and Cyclone III family. An analogue of the developed circuit was a functionally similar device from Altera. A comparative analysis is made and reasoned conclusions are drawn that the performance improvement is achieved due to the conveyor structure of the adder.

Implementation of arithmetic over the floating-point numbers on programmable logic integrated circuits, in particular on FPGA, has such advantages as flexibility of use and low production costs, and also provides the opportunity to solve problems for which there are no ready-made solutions in the form of standard devices presented on the market. The developed adder has a wide scope, since most modern computing devices need to process floating-point numbers. The proposed conveyor model of the adder is quite simple to implement on the FPGA and can be an alternative to using built-in multipliers and processor cores in cases where the complex functionality of these devices is redundant for a specific task.

Key words: *real numbers, conveying, FPGA, computing performance, adder.*

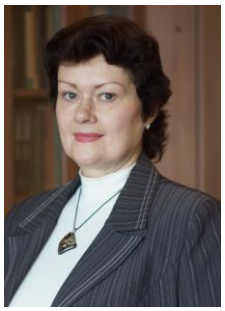
КОНВЕЙЕРНАЯ МОДЕЛЬ И РЕАЛИЗАЦИЯ СУММАТОРА ДЕЙСТВИТЕЛЬНЫХ ЧИСЕЛ НА FPGA

И. Я. Зеленёва, Т. В. Голуб, Т. С. Дьячук, А. Е. Диденко
Национальный университет «Запорожская политехника»

Аннотация. *Предлагается способ повышения эффективности операции сложения действительных чисел с использованием принципа конвейеризации. Схема спроектирована для реализации на микросхемах типа FPGA, что позволяет использовать архитектурные особенности данного базиса, а именно возможность параллельного выполнения задач на одном кристалле. Каждый блок предложенной конвейерной структуры на FPGA описан и реализован как отдельный проект цифрового функционального узла. Проанализированы результаты экспериментального тестирования конвейерного сумматора на разных микросхемах фирмы Altera/Intel.*

Ключевые слова: *действительные числа, конвейеризация, FPGA, производительность вычислений, сумматор.*

Отримано 03.03.2020



Зеленьова Ірина Яківна, Національний університет «Запорізька політехніка», кандидат технічних наук, доцент, доцент кафедри комп'ютерних систем і мереж, вул. Жуковського, 64, Запоріжжя, Україна, E-mail: irina.zeleneva@gmail.com, тел. +38-061-768-92-49

Irina Zeleneva, Zaporizhzhia Polytechnic National University, Ph.D. of Science, docent, assistant professor of the Department of computer systems and networks, Zhukovs′ka str., 64, Zaporizhzhia, Ukraine, E-mail: irina.zeleneva@gmail.com, tel. +38-061-768-92-49

ORCID ID: 0000-0002-4042-4540

Голуб Тетяна Василівна, Національний університет «Запорізька політехніка», асистент кафедри комп'ютерних систем і мереж, вул. Жуковського, 64, Запоріжжя, Україна, E-mail: golub.tv6@gmail.com, тел. +38-061-768-92-49

Tetiana Golub, Zaporizhzhia Polytechnic National University, assistant of the Department of computer systems and networks, Zhukovs′ka str., 64, Zaporizhzhia, Ukraine, E-mail: golub.tv6@gmail.com, tel. +38-061-768-92-49

ORCID ID: 0000-0001-6024-008X

Дьячук Тетяна Сергіївна, Національний університет «Запорізька політехніка», асистент кафедри комп'ютерних систем та мереж. вул.Жуковського, 64, Запоріжжя, Україна, E-mail: diacht@gmail.com, тел. +38-061-768-92-49

Tetiana Diachuk, National University Zaporizhzhia Polytechnic, assistant of the Department of computer systems and networks, Zhukovs′ka str., 64, Zaporizhzhia, Ukraine, E-mail: diacht@gmail.com, tel. +38-061-768-92-49

ORCID ID: 0000-0002-2478-0588

Діденко Артем Євгенович, Національний університет «Запорізька політехніка», студент кафедри комп'ютерних систем і мереж, вул. Жуковського, 64, Запоріжжя, Україна, E-mail: didenko.art121@gmail.com, тел. +38-061-768-92-49

Artem Didenko, Zaporizhzhia Polytechnic National University, student of the Department of computer systems and networks, Zhukovs′ka str., 64, Zaporizhzhia, Ukraine, E-mail: didenko.art121@gmail.com, tel. +38-061-768-92-49

ORCID ID: 0000-0001-7556-2276