УДК 004.222.2 + 004.312.2

О.Я. АНОПРІЄНКО, С.В. ІВАНИЦЯ, С.В. КУЛІБАБА

Донецький національний технічний університет, Донецьк

ПРИНЦИП РОБОТИ, СТРУКТУРА І МОДЕЛЮВАННЯ БЛОКУ ПЕРЕТВОРЮВАЧА ФОРМАТІВ У СКЛАДІ ПОСТБІНАРНОГО СПІВПРОЦЕСОРА

Анотація. Показана необхідність створення постбінарного перетворювача форматів як ключового блоку постбінарного співпроцесора. Розглянуто принцип роботи та структуру цього блоку, а також особливості його апаратної реалізації. Наведено результати синтезу сигналів і моделювання розробленої мовою VHDL спрощеної моделі блоку.
Ключові слова: постбінарний формат даних, тетракод, VHDL, модель, співпроцесор.
Аннотация. Показана необходимость создания постбинарного преобразователя форматов как ключевого блока постбинарного сопроцессора. Рассмотрены принцип работы и структура этого блока, а также особенности его аппаратной реализации. Приведены результаты синтеза сигналов и моделирования разработанной на языке VHDL упрощенной модели блока постбинарного преобразователя форматов.
Ключевые слова: постбинарный формат данных, тетракод, VHDL, модель, сопроцессор.
Abstract. Necessity of creating postbinary formats converter of the key block postbinary coprocessor is shown. Principles of operation and structure of this unit, and features of hardware-based implementation are observed. Results of simulation of signals and modelling of simplified model of convecter unit postbinary formats, which is developed in VHDL,are pointed presented.
Key world: postbinary data format, tetracode, VHDL, model, coprocessor.

Вступ

Вся множина сучасного апаратного забезпечення обчислювальних пристроїв оперує числами, представленими двома форматами: цілочисельними та дійсними. При цьому цілочисельна арифметика реалізована без похибок, на відміну від дійсної, яка не в змозі оперувати нескінченною множиною дійсних значень при наявності кінцевих числових наборів в дійсних форматах. Дійсні числа в комп’ютерних системах представляються в форматах чисел з плаваючою комою. Стандарт чисел з плаваючою комою IEEE 754-2008 [1] передбачає в ряді випадків наближене представлення початкового числа в плаваючих форматах. Похибка при цьому носить накопичувальний характер та залежить від подальших операцій з представленим значенням. Накопичення похибки в деяких випадках може призвести до повної втрати достовірності результату.

Така ситуація призводить до неприйнятних на сучасному етапі розвитку обчислювальної техніки наслідків. Для більшості прикладних задач сучасної точності чисел з плаваючою комою цілком достатньо, але, наприклад, для мультімасштабного моделювання макро- та мікросвіту, такі похибки досягають неприпустимо великих величин [2]. А в тих випадках, коли мова йде про безпеку і життя людини, достовірність результату стає одним з найважливіших критеріїв. Зокрема, директор Інституту математики та її застосувань в Міннеаполісі, США, Дуглас Н. Арнольд стверджує, що цілий ряд найбільших аварій з людськими жертвами та мільярдними збитками пов’язаний із сучасними технологіями комп’ютерних обчислень, в яких дані представлені згідно стандарту IEEE 754-2008 [3].

Перехід від бінарних до **постбінарних форматів** представлення дійсних чисел зумовлений двома важливими моментами:

1. Можливість постбінарного кодування даних (з використанням значень неоднозначності, яка є особливістю тетракоду, як одного з можливих наборів гіперкоду) [4].

2. Зміна структури самого формату, в якому крім стандартних полів знака, порядку та мантиси присутнє поле ідентифікатора (ID), що складається з **коду постбінарного формату CF** (визначає «ширину» формату — кількість розрядів, точність) і **модифікатора постбінарного формату MF** (визначає тип даних, «упакованих» у полі формату) [5, с. 198].

У світлі вищевикладеного представляється актуальною задача проектування вбудованої процесорної системи, що підтримує плаваючі формати чисел і забезпечує коректність їх обчислень, які не призводять до втрати достовірності. Рішення такої задачі можна розбити на два етапи: 1-й етап — модифікація форматів представлення дійсних чисел (корекція представлення даних); 2-й етап — введення додаткових методів контролю точності для стандартних алгоритмів обчислення(корекція обчислень).

Метою даної роботи є розробка **блоку перетворювача постбінарних форматів** (ППФ, pFC — postbinary formats converter), що виконує наступні функції:

– виділення формату завантаженого операнда за значенням поля коду формату CF;

– визначення типу даних в форматі за значенням поля модифікатора формату MF;

– реалізація постбінарного кодування (перетворення: двійковий код → тетракод);

– підготовка даних до завантаження в **постбінарний арифметико-логічний пристрій** (ПАЛП, або pALU — postbinary arithmetic logic unit) з розділенням розрядів операнду на поля мантиси, порядку та знака.

Важливість блоку ППФ у складі постбінарної процесорної системи (можливо, постбінарного співпроцесора) очевидна, оскільки на нього покладено функції розпізнавання типу значень, виділення та обробки полів формату і підготовки їх до завантаження у відповідні регістри ПАЛП.

В рамках даної роботи розглянута апаратна реалізація блоку ППФ для підготовки операндів до завантаження в регістри 120-розрядного ПАЛП. Наведено результати синтезу вихідних сигналів, функціональна схема ППФ та результати моделювання.

Принцип роботи та структура перетворювача постбінарних форматів

Розробку блоку ППФ можна позиціонувати як один з перших етапів розробки **постбінарного співпроцесора**, який представляє собою аналог математичного співпроцесора (або модуля операцій з плаваючою комою), здатного обробляти числа в постбінарних форматах. На рис. 1 наведено спрощену функціональну схему постбінарного співпроцесора (pFPU — postbinary floating point unit).

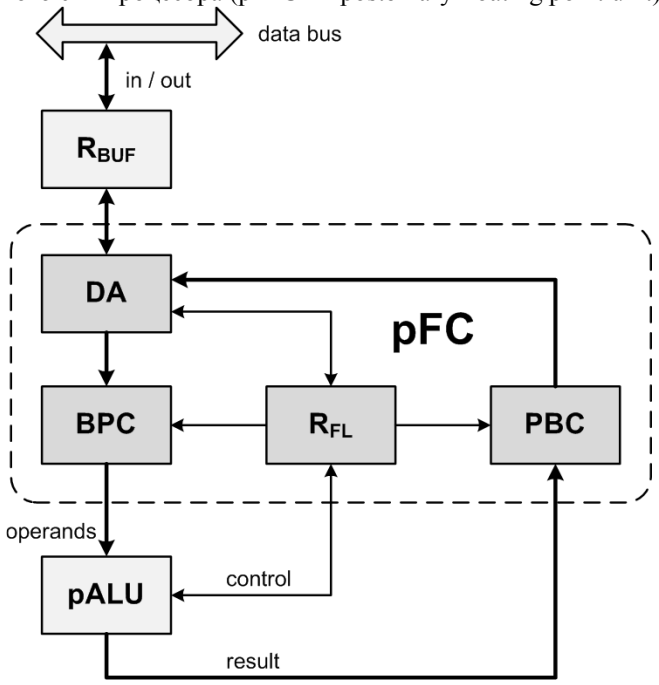

Рисунок 1 − Спрощена функціональна схема постбінарного співпроцесора pFPU з виділеною структурою постбінарного перетворювача форматів pFC (data bus – шина даних; in/out – вхідна/вихідна шина; operands – шина операндів; control – шина керування; result – шина результату)

Дані (бітове поле формату), що надійшли на вхід співпроцесора, а також результат обчислень, що надійшов з постбінарного АЛП (pALU) зберігаються в **буферному регістрі R_{BUF}** (buffer register). **Аналізатор даних DA** (data analyzer) розпізнає формат і тип розміщених у ньому даних, а також виконує виділення з вмісту R_{BUF} інформаційної та службової частин. Інформаційна частина (мантиса, порядок і знак) передається на **блок постбінарного перетворювача BPC** (binary-postbinary converter), а службова частина (код і модифікатор формату) обробляється та записується у **регістр прапорів R_{FL}** (flags register). У блоці BPC (при необхідності, на що вказують відповідні поля регістру RFL) відбувається постбінарне кодування даних, тобто заміна бінарного коду тетракодом (фактично приведення бітів до тетритів [5, с. 65]), після чого постбінарні дані надходять до регістрів операндів pALU. Результат з постбінарного АЛП проходить через **блок бінарного перетворювача PBC** (postbinary-binary converter), в якому дані (при необхідності, на що вказують відповідні поля регістру R_{FL}) перетворюються з тетракоду в двійковий код (фактично представлення кожного тетриту парою біт), і знову надходять на блок DA. У блоці DA дані (що містять тільки інформаційну частину) збираються в первісний вигляд постбінарного формату шляхом формування і додавання службової частини згідно з необхідними полями в R_{FL}. Зібраний формат даних завантажується в регістр R_{BUF}.

Таким чином, робота зв'язки блоків BPC→DA←PCB дозволить отримати на виході pFPU результат в тому форматі, в якому було введено вихідні значення. Це означає, що з боку апаратури постбінарний співпроцесор pFPU ні чим не відрізнятиметься від бінарного співпроцесора FPU, а постбінарне представ-

лення числових значень буде використовуватися виключно для внутрішньої роботи блоків співпроцесора.

У таблиці істинності (табл. 1) блоку ППФ використовуються позначення постбінарних форматів **pbΩ**, **pbΩ/Ψγ**, де:

– pb — скорочений запис постбінарного формату (від англ. **p**ost**b**inary);

– Ω — розрядність формату, що представляє значення Ω /32-арної точності (фактично, необхідна для зберігання формату розрядність);

– Ψ — розрядність значення, що входить до Ω-розрядного формату;

– γ — визначник групи Ψ-розрядного значення, що представляє множину {f, i, p, fp, ip}:

- γ = ∅ – звичайне число Ω-розрядне двійкове число;
- γ = f – дріб виду *a*/*b* (Ψ-розрядні двійкові чисельник *a* та знаменник *b* звичайного дробу в одному полі даних та зі спільним знаковим розрядом, причому Ψ = Ω/2);
- γ = i – інтервал виду $[x_1, x_2]$ (Ψ-розрядні двійкові значення меж інтервалу x_1 і x_2 в одному полі даних, причому Ψ = Ω/2);
- γ = p – постбінарне число (Ψ-розрядний тетракод, виражений Ω-розрядним двійковим кодом, тому Ψ = Ω/2);
- γ = fp – постбінарний дріб виду *a*'/*b*' (Ψ-розрядні постбінарні чисельник *a*' та знаменник *b*' звичайного дробу в одному полі двійкових даних та зі спільним знаковим постбінарним розрядом, тому Ψ = Ω/4);
- γ = ip – постбінарний інтервал виду $[x'_1, x'_2]$ (Ψ-розрядні постбінарні значення меж інтервалу x_1 та x_2 в одному полі двійкових даних, тому Ψ = Ω/4).

Таблиця 1 − Таблиця істинності сигналів-ознак формату

Постбінарний формат	*Модифікатор формату MF*	*Код формату CF*	*Молодший байт формату X*[7:0]	*Вихідні сигнали*[*]		
				CFm	*MFm*	
					Data_type	*Data_form*
pb32	0	0	xxxxxx00	00	0	00
pb32/16p	1		xxxxxx10		1	00
pb64	00	01	xxxx0001	01	0	00
pb64/32f	01		xxxx0101		0	01
pb64/32i	10		xxxx1001		0	10
pb64/32p	11		xxxx1101		1	00
pb128	000	011	00000011	10	0	00
pb128/64f	001		00001011		0	01
pb128/64i	010		00010011		0	10
pb128/64p	011		00011011		1	00
pb128/32fp	100		00100011		1	01
pb128/32ip	101		00101011		1	10
pb256	000	0111	00000111	11	0	00
pb256/128f	001		00010111		0	01
pb256/128i	010		00100111		0	10
pb256/128p	011		00110111		1	00
pb256/64fp	100		01000111		1	01
pb256/64ip	101		01010111		1	10

[*] — *CFm*[1:0] – фіксоване (2 біта) поле коду формату CF (вказує розрядність формату) для зберігання в регістрі прапорів R_{FL}; *MFm*[2:0] – фіксоване (3 біта) поле модифікатора формату MF (вказує тип даних) для зберігання в регістрі прапорів R_{FL}: *Data_type*[0] – кодовий базис (0 – двійкове кодування, 1 – постбінарне кодування); Data_form[1:0] – тип даних (00 – число, 01 – дріб, 10 – інтервал, 11 – резерв).

Поле *CFm* на відміну від поля коду формату CF має фіксовану ширину, що дозволяє економити реєстрову пам'ять і визначати розрядність вхідного числа, як 32·(CFm + 1) біта. Поле *MFm* представляє собою об'єднання сигналів *Data_type* і *Data_form*. Така організація службової частини формату спрощує її передачу та зберігання, та водночас спрощує апаратну реалізацію (виключення зі схеми шифраторів і дешифраторів для об'єднання і розбиття цих сигналів в блоках ППФ).

В результаті синтезу було отримано наступні залежності:

$$CFm[1] = \left(\overline{x}_7\overline{x}_6\overline{x}_3x_2x_1x_0\right) \vee \left(\overline{x}_7\overline{x}_5\overline{x}_3x_1x_0\right) \vee \left(\overline{x}_4\overline{x}_2x_1x_0\right) \vee \left(\overline{x}_5\overline{x}_2x_1x_0\right);$$

$$CFm[0] = \left(\overline{x}_7\overline{x}_5\overline{x}_3x_2x_0\right) \vee \left(\overline{x}_7\overline{x}_6\overline{x}_3x_2x_0\right) \vee \left(\overline{x}_1x_0\right);$$

$$Data\_type = \left(\overline{x}_7\overline{x}_6x_5x_4\overline{x}_3x_2x_1\right) \vee \left(\overline{x}_7x_6\overline{x}_5\overline{x}_3x_2x_1\right) \vee \left(\overline{x}_5x_4x_3\overline{x}_2x_1\right) \vee \left(x_5\overline{x}_4\overline{x}_2x_1\right) \vee \left(x_3x_2\overline{x}_1x_0\right) \vee \left(x_1\overline{x}_0\right);$$

$$Data\_form[1] = \left(\overline{x}_7\overline{x}_6x_5\overline{x}_4\overline{x}_3x_2x_1x_0\right) \vee \left(\overline{x}_7x_6\overline{x}_5x_4\overline{x}_3x_1x_0\right) \vee \left(\overline{x}_5x_4\overline{x}_3\overline{x}_2x_1x_0\right) \vee \left(x_5\overline{x}_4x_3\overline{x}_2x_0\right) \vee \left(x_3\overline{x}_2\overline{x}_1x_0\right);$$

$$Data\_form[0] = \left(\overline{x}_7x_6\overline{x}_5\overline{x}_4\overline{x}_3x_2x_0\right) \vee \left(\overline{x}_7\overline{x}_6\overline{x}_5x_4\overline{x}_3x_2x_0\right) \vee \left(x_5\overline{x}_4\overline{x}_3\overline{x}_2x_1x_0\right) \vee \left(\overline{x}_5\overline{x}_4x_3\overline{x}_2x_1x_0\right) \vee \left(\overline{x}_3x_2\overline{x}_1x_0\right).$$

Функціональність ППФ (зокрема, блок DA), крім перерахованих вище операцій, повинна виконувати ряд процедур, а саме:

1. Забезпечити послідовне завантаження операндів у регістри pALU.
2. При роботі з «складовими» форматами (для дробів та інтервалів) забезпечити парціальне завантаження даних в pALU у вибраному порядку (наприклад, спочатку обробляються знаменники операндів-дробів, потім їх чисельники, або спочатку ліві межі операндів-інтервалів, потім — праві).
3. Забезпечити виконання операції приведення типів. З двох операндів різних типів операції приведення підлягає операнд з меншою розрядністю. При неможливості приведення типів генерується відповідна помилка (помилка приведення типів постбінарних форматів).

Детальний опис перерахованих процедур, реалізація алгоритмів їхньої роботи і обробка виключень виходять за рамки теми даної роботи.

Апаратна реалізація блоку ППФ

Далі розглянуто апаратну реалізацію блоку ППФ для забезпечення узгодження постбінарних форматів зі 120-розрядним ПАЛП (pALU120), що оперує даними, представленими у тетракодах (рис. 2). Формати, що містять такі дані, надходять на вхід ПАЛП без змін, а дані, представлені двійковим кодом, приводяться до тетракоду в блоці бінарно-постбінарного кодування BPC. Для інших даних, що не підходять під вказані на рис. 2 формати, генерується сигнал помилки *err*:

$$err = \left(x_5x_4\overline{x}_2x_1x_0\right) \vee \left(x_6x_5x_2x_1x_0\right) \vee \left(x_7x_2x_1x_0\right) \vee \left(x_3x_2x_1x_0\right).$$

На відміну від рис. 1, на рис. 2 робота блоку BPC виглядає очевидніше і полягає в перетворенні 60-бітної двійкової шини в 120-розрядний тетракод. Це відбувається таким чином: кожен біт представляється парою двійкових розрядів ($0 \rightarrow 01$ та $1 \rightarrow 10$), тим самим стаючи вже значеннями тетритів 0 та 1.

Якщо вхідне число має «тетракодове» представлення, то воно передається в постбінарний АЛП без перетворень.

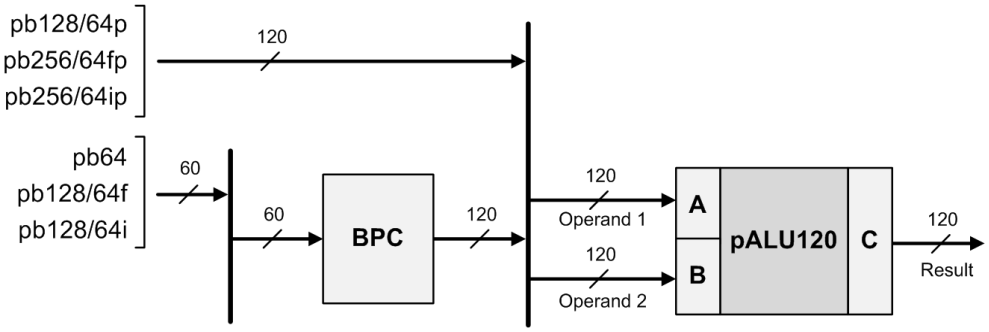

Рисунок 2 − Схема взаємодії постбінарного перетворювача (BPC) та 120-розрядного постбінарного АЛП (pALU120) із зазначенням допустимих постбінарних форматів

На рис. 3 представлена спрощена модель блоку ППФ, що реалізована мовою опису апаратури інтегральних схем VHDL [6]. У спрощеній моделі виконана повна реалізація блоку BPC та часткова — блоку DA (визначення розрядності формату і типу даних в ньому, формування та занесення до R_{FL} сигналів-ознак, підготовка інформаційної частини формату до обробки в pALU).

На вхід моделі блоку ППФ по 256-розрядній шині даних подаються дані в передбачуваному постбінарному форматі. На виході — шини знаку, порядку та мантиси постбінарного числа, отриманого (у разі виправдання припущення) з ідентифікованого і обробленого формату. При цьому також формуються описані раніше сигнали-ознаки, у тому числі й сигнал помилки розпізнавання формату.

Модель блоку ППФ складається з тактованих функціональних модулів U1-U5:

– U1 — 256-розрядний буферний регістр RB (аналог блоку R_{BUF} на рис. 1) для зберігання поля формату вхідного операнду;

– U2 — модуль data_modificator, виконує визначення коду CF і модифікатора MF вхідного формату, і формування їхніх еквівалентів (*CFm*, *MFm*) на шинах bus_cfm, bus_data_form та bus_data_type, або помилку поля формату (якщо у вхідному регістрі дані не pb-форматів) на шині bus_DM_err.

– U3 — постбінарний перетворювач BPC (повна відповідність блоку BPC на рис. 1).

– U4 — мультиплексор data_MX, що вибирає на вихідну шину busOut значення з буферного регістра або з перетворювача PBC в залежності від рівня сигналу вибірки на шині bus_convert.

– U5 — блок convert_analizator (реалізація блоку DA (рис. 1) з обмеженою функціональністю), що визначає, чи потрібно виконувати перетворення вхідного числа в постбінарне представлення: в результаті аналізу сигналів *CFm* і *MFm* формує відповідний керуючий сигнал вибірки на шині bus_convert, або сигнал помилки представлення формату на шині bus_CA_error.

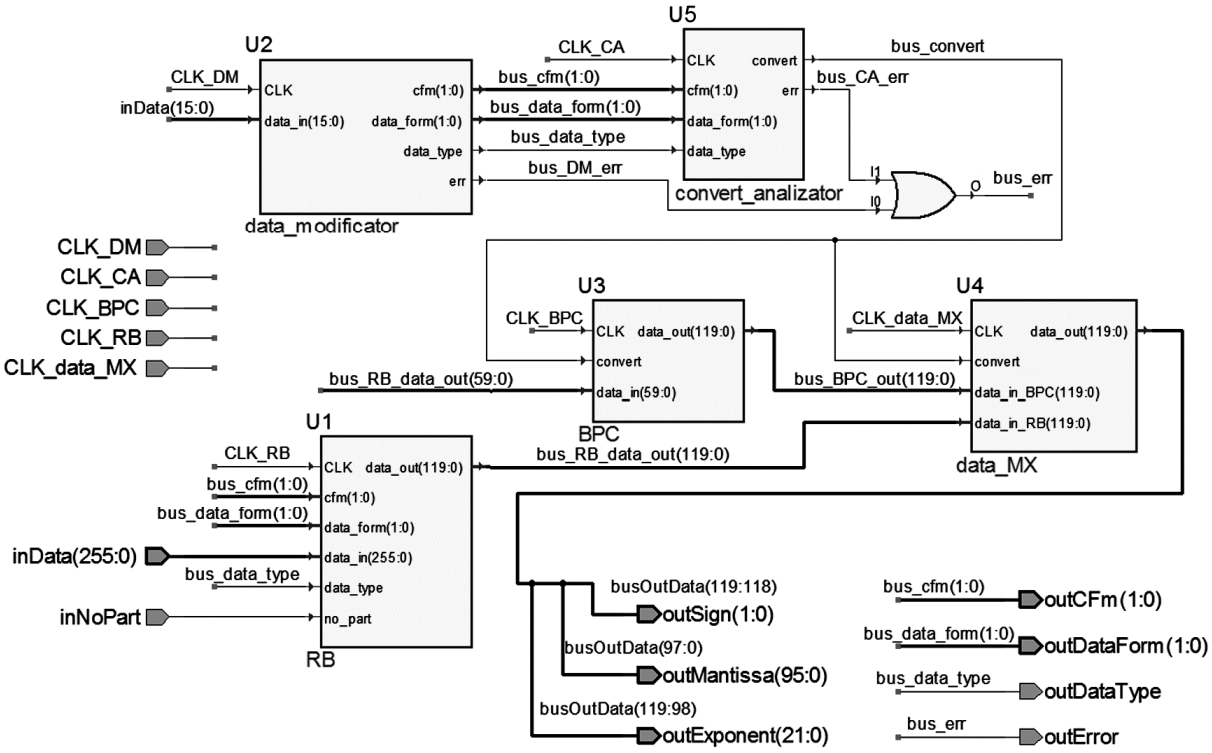

Рисунок 3 − Функціональна схема спрощеної моделі блоку ППФ

Робота моделі блоку ППФ організована за принципом роботи операційного автомата, але в даному випадку на кожен модуль подається власний сигнал синхронізації (CLK). Дана концепція заснована на можливості реалізації покрокового процесу моделювання для поточної моделі і пізніше буде переглянута для загальної моделі постбінарного співпроцесора, що складається з операційної та керуючої частин.

У табл. 2 приведені назви і призначення вхідних і вихідних даних моделі ППФ.

На рис. 4 зображено тимчасову діаграму перетворення формату числа з поля даних, що надійшли на вхід inData перетворювача. Розглянемо роботу моделі по кроках:

Крок 1 — формування значень на виході модуля U2 (data_modificator):

– outError: = 0h — дані розпізнані й справді містять поле постбінарного формату з плаваючою комою;

– outCFm: = 1h — довжина формату 64 біта;

– outDataForm: = 0h — формат представлений бінарним кодом, що означає необхідність постбінарного перетворення;

– outDataType = 0h — формат містить в собі одне число, отже це формат pb64.

Таблиця 2 − Вхідні та вихідні дані моделі ППФ

Назва	*Розряд-ність*	*Опис*
		Вхідні дані
inData	256	Вхідні дані для завантаження в буферний регістр та подальшого аналізу
inNoPart	1	Вказівник номеру частини числа, що підлягає вибірці в форматах дробових та інтервальних чисел (для дробів: 0 – чисельник, 1 – знаменник; для інтервалу: 0 – ліва межа, 1 – права межа). Для форматів одного числа – завжди дорівнює 0. У загальній моделі pFC (рис. 1) зберігається в регістрі R_{FL}
		Вихідні дані
outCFm	2	Фіксоване поле коду формату для завантаження в регістр прапорів. Несе інформацію про розрядності перетвореного формату (сигнал *CFm*, табл. 1)
outDataForm	2	Сигнал *Data_form* (табл. 1) для завантаження в регістр прапорів
outDataType	1	Сигнал *Data_type* (табл. 1) для завантаження в регістр прапорів
outError	1	Помилка роботи моделі – сигнал *err* – завантаження формату числа, що не підтримується (0 – формат розпізнано, дані отримані; 1 – аварійне завершення роботи блоку, очікування наступної вибірки)
outSign	2	Постбінарне значення знаку отриманого з формату числа для завантаження в pALU
outExponent	22	Постбінарне значення порядку отриманого з формату числа для завантаження в pALU
outMantissa	96	Постбінарне значення мантиси отримане з формату числа для завантаження в pALU

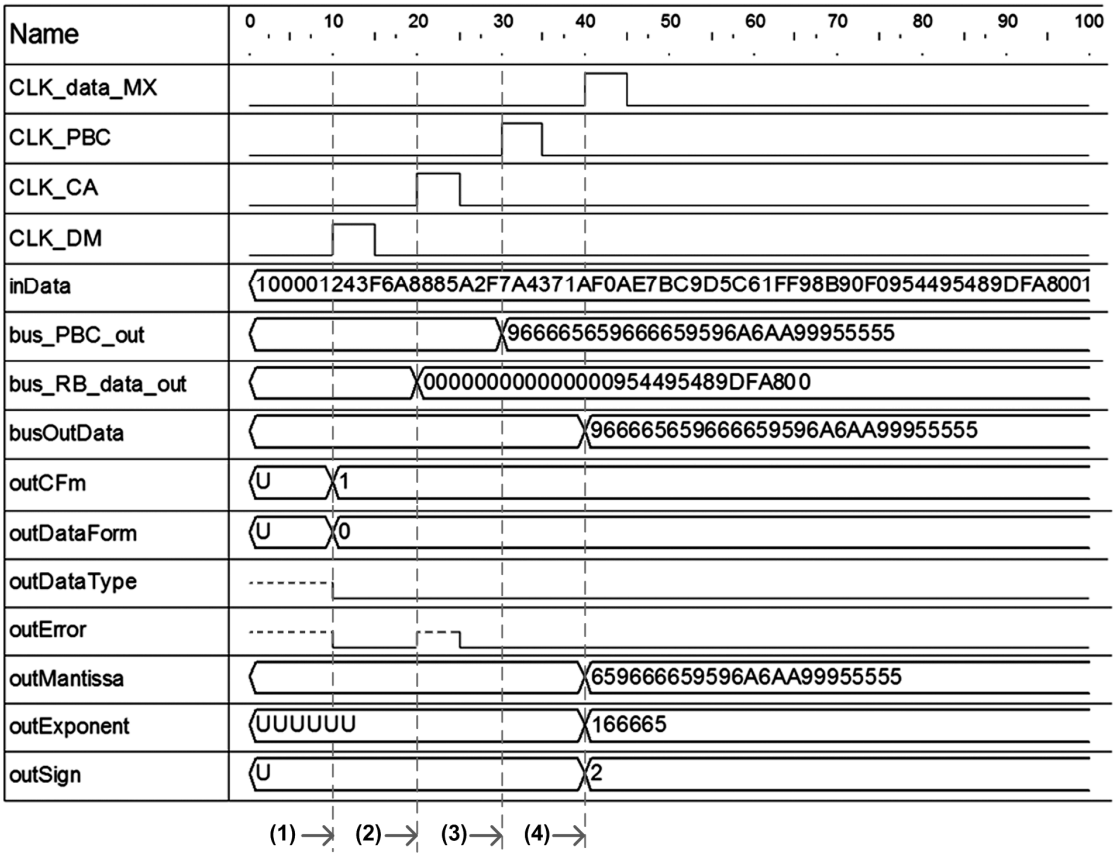

Рисунок 4 − Часова діаграма роботи спрощеної моделі ППФ

Крок 2 — формування сигналу convert := 1h модуля U5 (convert_analizator) — команда модулю PBC про виконання постбінарного перетворення і модулю data_MX для вибірки "постбінарної" шини. На

шину bus_RB_data_out подається інформаційна частина розпізнаного формату: bus_RB_data_out := 954495489DFA800h.

Крок 3 — перетворення бінарного коду з шини bus_RB_data_out в постбінарний, та його передача на шину bus_BPC_data_out, оскільки сигнал convert модуля U5 (convert_analizator) дорівнює 1h — команда модулю РВС про виконання постбінарного перетворення.

Крок 4 — Робота модуля U4 (data_MX) яка полягає у формуванні даних на вихідній шині busOutData по керуючому сигналу convert:

$$\text{convert} = \begin{cases} 0 \rightarrow \text{busOutData} := \text{bus\_PB\_data\_out}, \\ 1 \rightarrow \text{busOutData} := \text{bus\_BPC\_data\_out}. \end{cases}$$

Оскільки convert := 1h, то на вихідну шину надходять дані з модуля ВРС.

В результаті моделювання, з 256-розрядного бінарного поля даних було розпізнано та вибрано число, представлене постбінарним форматом pb64:

954495489DFA8001h ≈ –3,2055650328029E–206.

На відповідних виходах моделі сформовано знак, мантису та порядок цього числа у вигляді тетракоду.

Висновки

Таким чином, показана необхідність створення блоку постбінарного перетворювача форматів pFC як ключового блоку співпроцесора pFPU та описана його реалізація.

Представлена в роботі модель має обмежену функціональність, оскільки виконує частину функцій, покладених на блок перетворювача форматів (рис. 1) постбінарного математичного співпроцесора. Подальша реалізація моделі блоку pFC передбачає:

1) підтримку дробових та інтервальних чисел, з організацією черговості обробки частин цих чисел (чисельника і знаменника дробу, меж інтервалу);

2) реалізацію блоку постбінарно-бінарного конвертора РВС.

Модель розроблено мовою опису апаратури VHDL, що дозволяє виконати апаратну реалізацію інтегральної схеми будь-якої складності на ПЛІС [7]. Така реалізація обрана як альтернатива створення дорогих ВІС та НВІС, а також на користь спрощення модифікації і налагодження окремих частин пристрою.

Дану роботу можна вважати першим кроком створення прототипу pFPU — модуля для виконання широкого спектру математичних операцій над числами, представленими в постбінарних форматах, структура та принцип роботи яких докладно розглянуті в монографії [5].

Перелік джерел

1. IEEE 754: Standard for Binary Floating-Point Arithmetic. [Електроний ресурс]. — Режим доступу: http://grouper.ieee.org/groups/754/.

2. Introduction to Programming in Java. Floating point // Computer Science Departament of Princeton University http://introcs.cs.princeton.edu/java/91float

3. Юровицкий В.М. IEEE754-тика угрожает человечеству. МФТИ, РГСУ, Москва. [Електроний ресурс]. — Режим доступу: http://www.yur.ru/science/computer/IEEE754.htm.

4. Аноприенко А.Я. Тетралогика и тетракоды. / В кн. «Сборник трудов факультета вычислительной техники и информатики». Вып. 1. Донецк, ДонГТУ, 1996.

5. Аноприенко А.Я., Иваница С.В. Постбинарный компьютинг и интервальные вычисления в контексте кодо-логической эволюции. — Донецк: ДонНТУ, УНИТЕХ, 2011. — 248 с., ил.

6. VHDL coding tips and tricks. Get interesting tips and tricks in VHDL programming. [Електроний ресурс]. — Режим доступу: http://vhdlguru.blogspot.com/.

7. ПЛИС — Программируемые Логические Интегральные Схемы. [Електроний ресурс]. — Режим доступу: http://www.fpga-cpld.ru/.

Стаття надійшла: 23.11.2012.

Відомості про авторів

Анопрієнко Олександр Якович – к.т.н., проф., декан факультету КНТ ДонНТУ, Україна, 83001, м. Донецьк, вул. Артема, 58, тел. 050-67-77-270, e-mail: anoprien@gmail.com.

Іваниця Сергій Васильович – аспірант кафедри КІ ДоНТУ, Україна, 83001, м. Донецьк, вул. Артема, 58, тел. 050-258-71-18, e-mail: ivanitsa-serg@rambler.ru.

Кулібаба Сергій Володимирович – магістрант кафедри КІ ДоНТУ, Україна, 83001, м. Донецьк, вул. Артема, 58, тел. 095-387-80-55, e-mail: kulibaba @kulibaba.net.