УДК 004.31

В.М. Опанасенко, О.М. Хіміч, О.М. Лісовий, Т.В. Чистякова

Розв'язання задач з підвищеною точністю обчислень

Рассмотрены аппаратные и программные средства для решения задач с повышенной точностью вычислений. Разработано устройство с поддержкой 128-битной арифметики с плавающей точкой, позволяющее решать плохо обусловленные системы линейных алгебраических уравнений.

Hardware and software means for solving the tasks with the increased precision are considered. A computing device with support of 128-bit arithmetic's with a floating point is developed which makes it possible to solve ill-conditioned systems of linear algebraic equations.

Розглянуто апаратні та програмні засоби для розв'язання задач з підвищеною точністю обчислень. Розроблено пристрій з підтримкою 128-бітної арифметики з плаваючою точкою, який дає можливість розв'язувати погано обумовлені системи лінійних алгебраїчних рівнянь.

Актуальність використання засобів з підвищеною точністю обчислень

Однією із задач, яка набула актуальності з середини минулого століття є забезпечення достатньої точності обчислень на комп'ютері. Розв'язанням цієї задачі було збільшення розрядності даних.

В процесорах серії *Ельбрус* було закладено підтримку 128-бітної арифметики. У компанії *IBM* дослідження підтримки арифметики з підвищеною точністю здійснювались із 1969 року і були впроваджені у системах серії *IBM S*/390 *VM*. В 90-ті роки процесори стали масовим продуктом, і найбільшого розвитку набули комп'ютери загального призначення (персональні), в яких акцент робився не на обчислення високої точності, що й спричинило зупинку розвитку 128-бітної арифметики. Оптимальною було обрано розрядність 80 біт, що сьогодні маємо в переважній більшості процесорів, у яких розрядність співпроцесора з плаваючою точкою дорівнює 80 біт. Це можна пояснити тим, що ринок насамперед орієнтований на бізнес-процеси, мультимедійні застосування, де такої розрядності обчислень достатньо.

Нині у світі апаратну підтримку 128-бітних обчислень мають лише декілька компаній, такі, як *IBM* на базі операційної системи *AIX*, *Hewlett-Packard* на базі операційної системи *HP-UX*, *AMD*-64 сімейства 10*h* (процесори *Phenom* з мікропрограмною підтримкою) [1], *Sun Microsystems* процесорна архітектура *SPARC-V*9 [2]. Деякі операційні системи, серед яких *Solaris*, *IBM* мають програмно реалізовану підтримку 128-бітної арифметики з плаваючою точкою.

Сьогодні розрядності бракує як на робочих станціях науковців, так і на більшості суперкомп'ютерів із кластерною архітектурою, які використовують стандартні масові процесори. Для задач, які потребують обчислення з підвищеною точністю (128 біт та більше), відомі задачі аеродинаміки, розрахунки міцності, проектування конструкцій в авіабудуванні, обробка результатів тощо.

Розв'язання задач з оцінкою вірогідності має декілька шляхів, які можна розділити на *апаратні* та *програмні*. До *апаратних* належать: комп'ютерні засоби на базі процесорів з підтримкою *FPU* 128 біт; комп'ютерні засоби на базі процесорів *IBM POWER*6 з підтримкою десяткового *FPU* 128 біт (*DFP*); комп'ютерні засоби, які використовують арифметичні пристрої з розрядністю обчислень 128 біт або більше і під'єднуються до комп'ютера в одну з стандартних шин (*PCI*, *PCI-e* та ін.), такі пристрої можуть бути розроблені на реконфігуровних платах на базі програмовних логічних інтегральних схем (ПЛІС). До *програмних* належать: використання сучасних пакетів математичних обчислень, які підтримують символьну арифметику (*Maple*, *Matlab* та ін.); використання в програмуванні бібліотек, які дозволяють програмно забезпечити обчислення з довільно заданими форматами (бібліотеки *MPFR*, *GMP* [3]).

Кожний з шляхів має свої переваги й недоліки. Зазначимо деякі вагомі нюанси: висока вартість серверів *IBM POWER*6; суттєві накладні витрати для програмних підходів. Вагомою перевагою використання пристроїв на реконфігуровних платах на базі ПЛІС є можливість реалізації як двійкової, так і десяткової арифметики й інших нестандартних підходів, у тому числі з використанням нестандартних форматів, при цьому розрядність даних може бути 128 біт і більше.

Розв'язання систем лінійних рівнянь з підвищеною точністю

Як випливає з [4], можливість отримання достовірного розв'язку (псевдорозв'язку) задачі залежить від узгодження математичних можливостей комп'ютерів (розрядність, архітектура) з математичними властивостями комп'ютерної задачі (обумовленість, коректність).

Отже, уточнити комп'ютерне розв'язання задачі, знаючи обумовленість матриці системи, очевидно можна, розв'язуючи систему з підвищеною розрядністю. В цьому випадку є принципова можливість досягти будь-якої заданої точності комп'ютерного розв'язку.

Для прогнозу довжини мантиси машинного слова, що забезпечує задану точність для сумісних систем можна користуватися наступним емпіричним правилом, яке випливає з оцінок [5]: кількість правильних десяткових значущих цифр в комп'ютерному розв'язанні приблизно дорівнює $\mu - \alpha$, де μ – кількість цифр в десятковому представленні мантиси числа з плаваючою точкою, α – десятковий порядок числа обумовленості матриці.

Прикладом є задача розв'язання погано обумовлених СЛАР (система лінійних алгебраїчних рівнянь), оскільки, аналізуючи її число обумовленості, можна орієнтуватися, який формат з плаваючою точкою (ФПТ) необхідно використовувати для її розв'язання [4].

Розв'язання СЛАР на комп'ютерній техніці виконується за допомогою прямих та ітераційних методів [4]. Прямі методи (Гауса, *LU*-розвинення, Холецького та ін.) дають можливість при обчисленні на ідеальному комп'ютері (обчислення та представлення даних без похибок) отримати розв'язок задачі за скінчене число арифметичних операцій. Ітераційні методи (метод простої ітерації, Якобі, Зейделя та ін.) або методи послідовних наближень дають можливість отримати наближений розв'язок, що збігається з розв'язанням задачі за необмеженої кількості ітерацій.

Одним з важливих факторів вибору методу розв'язання конкретної задачі є обчислювальна ефективність методу. Оскільки операція додавання виконується набагато швидше операції множення та ділення, обчислювальна ефективність методу залежатиме від кількості операцій множення та ділення.

Метод Гауса – один з ефективних методів розв'язання СЛАР з не виродженими матрицями загального виду. Для успішного використання алгоритму необхідно, щоб вхідні коефіцієнти були відмінні від нуля, крім того, близькість їх до нуля може приводити до великої похибки розв'язків. Кількість арифметичних дій прямого ходу складає $\approx 2/3n^3$, зворотного $\approx n^2$.

Постановка задачі

Як приклад розглянемо СЛАР (1) з числом обумовленості 3,55868727582796*E*16. Велике число обумовленості вказує на те, що система є машинно вироджена на ФПТ (*Single*, *Double*, *Extended*), тому для отримання достовірного розв'язку не може використовуватися класична комп'ютерна арифметика з плаваючою точкою.

$$A = \begin{bmatrix} 0{,}1348531574394464 & 0{,}1878970588235294 & 0{,}1909117647058824 & 0{,}1779264705882353 \\ 0{,}1878970588235294 & 0{,}262 & 0{,}265 & 0{,}247 \\ 0{,}1909117647058824 & 0{,}265 & 0{,}281 & 0{,}266 \\ 0{,}1779264705882353 & 0{,}247 & 0{,}266 & 0{,}255 \end{bmatrix}$$

$$b = \begin{bmatrix} 0{,}3516 \\ 0{,}4887 \\ 0{,}5105 \\ 0{,}4818 \end{bmatrix}, x = \begin{bmatrix} 6{,}662162162161606738798064490430572168030658239553569314802100940649855 9\ldots E12 \\ -4{,}016891891890723506952166298245477287559781737502963469048697496817269 0\ldots E12 \\ -1{,}665540540539970051894018639784241392354074075505748928758861569054296 6\ldots E12 \\ 9{,}797297297302797072574940696301157235597379312722179179629928555505557 2\ldots E12 \end{bmatrix}. \quad (1)$$

Програмні підходи розв'язання СЛАР

Сьогодні як засоби символьного розв'язування прикладних задач використовуються па-

кети *Maple*, *MatLab*, *MathCad* та ін. Дані програмні продукти мають широкі можливості для форматування математичних текстів, проведення різних розрахунків, експериментів, символьних обчислень і т.ін. Проте не можна не відзначити, що ці середовища є замкнутими, тобто немає можливостей використання документів із цих середовищ у традиційних системах програмування таких, як *Delphi*, *C++*, *Basic*.

Використання пакету Maple. Особливістю використання символьної арифметики є відсутність похибки введення вхідних коефіцієнтів в систему, окрім тих випадків, коли користувач обмежує їх самовільно. Однак похибка обчислень наявна, і для її зменшення потрібно збільшувати кількість десяткових цифр оброблювальних даних: для *Maple* це параметр *Digits*.

Для розв'язання СЛАР (1) розглянемо залежність точності обчислень від параметра *Digits* (табл. 1).

Таблиця 1. Похибки обчислення СЛАР (1) за допомогою пакету *Maple*

Digits	Середньоквадратична похибка, порівняння з еталоном
16	10^{1}
20	10^{-4}
30	10^{-13}
50	10^{-33}
70	10^{-53}

Аналіз середньоквадратичної похибки розв'язань для *Digits* 16 та *Digits* 20 взагалі не гарантує результату, що є наслідком поганої обумовленості матриці СЛАР, яка розв'язується. Для *Digits* 30 гарантовано отримуємо 13 десяткових цифр, використання *Digits* 70 гарантує 53 десяткових цифри.

Переваги використання символьної арифметики виявляються в можливості виконання надзвичайно точних обчислень. Одним, проте вагомим, недоліком символьної арифметики є час, затрачений на розв'язування задачі відносно класичної комп'ютерної арифметики значно збільшується.

Використання бібліотеки MPFR. Ця бібліотека заслуговує на особливу увагу, оскільки програміст, який користується нею, створює власний формат з плаваючою точкою, вказуючи розрядність порядку числа та його мантису. Отже можна створити формати, не описані в сучасних стандартах, проте які виконують класичні арифметичні операції з плаваючою точкою.

Розглянемо залежність точності обчислень від ширини мантиси (табл. 2). Коректні десяткові знаки в розв'язках виділені напівжирним шрифтом.

Відзначимо, що збільшуючи ширину мантиси вдвічі, кількість коректних десяткових знаків розв'язків збільшується вдвічі майже лінійно. Ширина мантиси 512 біт – далеко не межа, є можливість використовувати 1024, 2048 і більше. Наприклад, використовуючи ширину мантиси 40980 біт, кількість коректних десяткових знаків розв'язків дорівнюватиме 12320. Звичайно, час, затрачений для таких обчислень відносно класичної комп'ютерної арифметики, значно збільшується.

Таблиця 2. Результати та похибки розв'язання СЛАР (1) за допомогою *MPFR*

Результат	**Похибка**
Параметр *precision* (ширина мантиси) 128 біт $X1$ = 6,662162162161606738798**1784272624219630009705819599066713848634080363808…**$\times 10^{12}$ $X2$ = –4,016891891890723506952**2349954529161335077159939636629536717504773690734…**$\times 10^{12}$ $X3$ = –1,665540540539970051894**0471239922038446691827554383941547341135835935688…**$\times 10^{12}$ $X4$ = 9,797297297302797072575**1082504656422558394110540814752187237235414585256…**$\times 10^{12}$	10^{-22}
Параметр *precision* (ширина мантиси) 256 біт $X1$ = 6,662162162161606738798064490430572168030658239553569314802100**1531195230…**$\times 10^{12}$ $X2$ = –4,016891891890723506952166298245477287559781737502963469048697**0219828036…**$\times 10^{12}$ $X3$ = –1,665540540539970051894018639784241392354074075505748928758861**3721717134…**$\times 10^{12}$ $X4$ = 9,797297297302797072574940696301157235597379312722179179629927**3973727147…**$\times 10^{12}$	10^{-62}
Параметр *precision* (ширина мантиси) 512 біт $X1$ = 6,662162162161606738798064490430572168030658239553569314802100940649855**9…**$\times 10^{12}$ $X2$ = –4,016891891890723506952166298245477287559781737502963469048697496817269**0…**$\times 10^{12}$ $X3$ = –1,665540540539970051894018639784241392354074075505748928758861569054296**6…**$\times 10^{12}$ $X4$ = 9,797297297302797072574940696301157235597379312722179179629928555505557**2…**$\times 10^{12}$	10^{-137}

Окрім розглянутих програмних підходів, є можливість використовувати емулятори *DFPU*, які дозволяють з урахуванням формату даних, відмінного від двійкового, та більш складних обчислень отримувати точніші результати відносно *FPU* такої ж розрядності. Емулятор *Decimal Floating-Point Math Library* фірми *Intel* [6] підтримує формати *decimal*32, *decimal*64, та *decimal*128 (32, 64, 128 бітні варіанти).

Перевагою десяткового ФПТ відносно двійкового є більший діапазон представлення чисел, тобто деяка множина задач, яка під час обчислень на двійкових ФПТ приводила до *Nan*-результату, на десяткових ФПТ може бути розв'язана. Іншою перевагою є можливість використовувати арифметику в десятковому форматі на будь-якому персональному комп'ютері платформ *x*86, *x*64.

Апаратна реалізація для розв'язання СЛАР на базі ПЛІС

Для розв'язання СЛАР методом Гауса розроблено пристрої (рис. 1) для наступних форматів даних: *Single*, *Double*, *Extended*, *Quadruple* [6], де БК – блок керування, ПЗП – програмований запам'ятовуючий пристрій, ОЗП – оперативно-запам'ятовуючий пристрій, БОД – блок обробки даних [7], *PCI-e* – шина *PCI-e*, *CE – chip enable* (сигнал початку роботи).

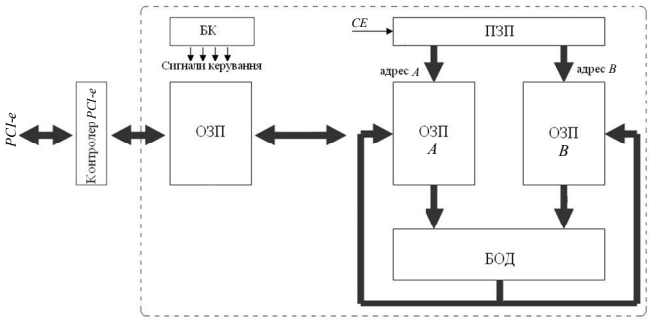

Рис. 1. Загальна блок-схема пристрою

Для кожного з форматів даних всі складові частини схеми окрім ПЗП, повинні мати відповідну розрядність: для *Quadruple* – 128 розрядні, для *Double* – 64 розрядні і т.д. ПЗП – це пам'ять, в якій зберігається запрограмований метод Гауса, однакова для всіх форматів і програмується на етапі формування ПЗП за допомогою *Core Generator*. Для розв'язання СЛАР четвертого порядку достатній об'єм 256 слів по 20 біт. Формат команд, які зберігаються в ПЗП, наведено на рис. 2. Для розв'язання систем більших порядків, окрім збільшення об'єму ПЗП, буде збільшуватися ширина команди, оскільки збільшиться об'єм ОЗП, а отже і поля для їх адресації. Блок ОЗП – пам'ять, яка є буфером при пересиланні даних з комп'ютера в пристрій та навпаки. БК відповідає за перемикання ОЗП з роботи в зовнішньому режимі на роботу у внутрішньому режимі та навпаки, а також за керування роботою всіх блоків пристрою.

Рис. 2. Формат команди

Як приклад, використовується моделююча плата *XC*4*VFX*100 фірми *PLDA* з ПЛІС *Virtex* 4 *FX*100-*ff*11, яка під'єднується до комп'ютера через шину *PCI-e*.

Об'єм ОЗП дорівнює 256 слів, а його ширина відповідає ширині формату даних. Блоки ОЗП *A* та *B* використовуються для зберігання початкових коефіцієнтів, результатів, та проміжних даних під час обчислень. БОД має три необхідні для методу Гауса арифметичні вузли (додавання–віднімання, множення, ділення) відповідної розрядності.

Функціонування БК складається з трьох етапів:

- запис вхідних коефіцієнтів з ОЗП в ОЗП *A* та ОЗП *B*;
- виконання програми, що описує метод Гауса;
- запис результату з ОЗП *B*, в ОЗП; формується сигнал, що символізує закінчення виконання мікропрограми, ОЗП перевмикається на роботу з контролером *PCI-e*.

Функціонування пристрою також складається з трьох етапів:

• запис коефіцієнтів СЛАР в ОЗП по шині *PCI-e*;

• в комірку ОЗП за адресою 160 записується значення «*F*», що слугує сигналом початку роботи ПК, ОЗП перевмикається на роботу з ОЗП *A* та ОЗП *B*;

• зчитування результатів з ОЗП по шині *PCI-e*.

Тестування пристроїв відбувається на етапах розробки окремих блоків, а також пристрою в цілому. Для тестування використовується система моделювання *ModelSim XE* 6,0*d* [8], яка дозволяє виконати верифікацію на всіх етапах проектування. Отримані апаратно-часові характеристики пристроїв наведено у табл. 3.

Таблиця 3. Апаратно-часові характеристики пристроїв на *Virtex* 4 *FX*100-*ff*11

Використані ресурси	Формати даних	
	Double	*Quadruple*
Кількість *Slices*	2335 (4%)	5007 (11%)
Кількість *DSP*48	13 (8%)	49 (30%)
Кількість *RAMB*16	11 (4%)	13 (4%)
Період *Clk*, нс	8	64

Розв'язана СЛАР (1). Після проведення обчислень з відповідними коефіцієнтами *A* та *b* пристроєм, отримуємо наступні результати (табл. 4).

Зауважимо, що обробка даних з використанням форматів *Single* та *Double* не дає коректних розв'язків, а *Extended* гарантує лише дві перші цифри розв'язків. Розв'язання на *Quadruple* гарантує 17 десяткових цифр (ширина мантиси в форматі *Quadruple* 113 біт).

Отже для розв'язання СЛАР (1) є наступні шляхи обчислень: використання символьної арифметики з заданою великою точністю обчислень (більше 60 десяткових знаків), використання комп'ютерних систем з апаратною підтримкою (*FPU*, *DFPU* 128 біт), використання пристроїв на реконфігуровних платах та програмних емуляторів широкоформатних обчислень з плаваючою точкою (128 біт та більше).

Вплив типу округлення в арифметичних компонентах на кінцевий результат

Тип округлення в арифметичних компонентах впливає на точність результатів.

В арифметичних компонентах використовується два типи округлення:

• якщо старший біт частини вектору, що перевищує розрядну сітку, має значення «1», то виконується запис «1» в молодший біт мантиси;

• якщо старший біт частини вектору, що перевищує розрядну сітку, має значення «1», то виконується додавання «1» до мантиси.

В табл. 5 наведено значення розв'язків з різними типами округлення.

Висновки. При розв'язанні СЛАР класичним комп'ютерним способом отримані розв'язки не завжди зберігають машинну суть, тому коректне розв'язання СЛАР може бути виконане за допомогою програмно-алгоритмічних та апаратних засобів.

Для розв'язання задач з підвищеною точністю розроблено пристрої на базі ПЛІС. Апробація цього підходу показала, що збільшення

Таблиця 4. Результати розв'язку СЛАР апаратним шляхом

Формат	Пристрій, що використовує розроблені арифметичні блоки	Пристрій, що використовує *IP-Core* фірми *Xilinx*
Single	***x*1=–64153,664062500000000000** ***x*2= 38681,507812500000000000** ***x*3= 16039,548828125000000000** ***x*4=–9434,2568359375000000000**	***x*1=–27856,2812500000000** ***x*2= 16796,2792968750000** ***x*3= 6965,29150390625000** ***x*4=–4096,46044921875000**
Double	***x*1= 2264503194623,36000000000000000** ***x*2=–1365362220253,90000000000000000** ***x*3=–566125798647,22700000000000000000** ***x*4= 333015175637,34700000000000000000**	***x*1= 3343472363451,5000000** ***x*2=–2015917160209,6700000** ***x*3=–835868090854,30700000** ***x*4= 491687112298,69300000**
Extended	***x*1**= 665**55584185991,900460000000000000** ***x*2**=–401**2925759200,165400000000000000** ***x*3**=–166**3896046497,543230000000000000** ***x*4**= 97**8762380293,5580170000000000000**	***x*1**= 66**57982426662,6836300** ***x*2**=–401**4371757251,6669800** ***x*3**=–166**4495606665,2391300** ***x*4**= 979**115062745,14385500**
Quadruple	***x*1**=6662162162161,6067**16600787977930271704018 6** ***x*2**=–4016891891890,723**49356851428335587820531 0** ***x*3**=–1665540540539,9700**46344699511658272177594** ***x*4**=979729729730,27970**39931887001441233735096 4**	Відсутні арифметичні блоки

Таблиця 5. Вплив типу округлення в арифметичних компонентах

Формат	Округлення першого типу	Округлення другого типу
Quadruple	***x*1= 402983C9F3C87C66D471DD94C71D8BDC** **(**6662162162161,6067**16600787977930271704 01)** ***x*2= C028D3A0B508595C9B6FEFD14DA782CC** **(**–4016891891890,723**49356851428333558782053)** ***x*3= C02783C9F3C87BF854F50E16B806BFB4** **(**–1665540540539,9700**46344699511658272 1775)** ***x*4= 4026C838E291848F355C9E9CB5B70D1A** **(**979729729730,2797**03993188700144123373509)**	***x*1= 402983C9F3C87C66D4894D6545FEACA7** **(**6662162162161,6067**22188550605705216379 62)** ***x*2= C028D3A0B508595C9B8C32E3501156C2** **(**–4016891891890,723**49693760645598480656 67)** ***x*3= C02783C9F3C87BF8550C7DE736E7E52D** **(**–1665540540539,9700**47741640168602262032 8)** ***x*4= 4026C838E291848F35783137A55685D6** **(**979729729730,2797**04814918498346507068 56)**

двійкової розрядності може забезпечити достовірність розв'язку. В арифметичних компонентах доцільно використовувати округлення другого типу.

Універсальність розробленої схеми пристроїв дозволить використовувати її для розв'язання задач більшої розмірності та більш широких форматів даних, які можуть бути стандартизовані або довільні.

Отож для розв'язання задач з підвищеною точністю обчислень доцільно використовувати один з розглянутих засобів. Вибір засобу залежатиме від поставленої задачі та вимог до точності результату.

1. *Family* 10h AMD Phenom Processor Product Data Sheet. – http://www.amd.com.
2. *http*://www.sun.com
3. *http*://www.mpfr.org
4. *Химич А.Н.*, *Молчанов И.Н.*, *Попов А.В.* Параллельные алгоритмы решения задач вычислительной математики – К.: Наук. думка, 2008. – 248 с.
5. *Николаевская Е.А.*, *Чистякова Т.В.* Программно-алгоритмические методы повышения точности компьютерных решений // Кибернетика и системный анализ. – 2009. – № 6. – С. 172–176.
6. *IEEE* 754R Decimal Floating-Point Arithmetic. – http://www.intel.com/technology/itj/2007/ v11i1/s2-decimal/1-sidebar.htm
7. *Опанасенко В.М.*, *Лісовий О.М.* Реалізація проблемно-орієнтованих цифрових пристроїв на кристалах FPGA // Радіоелектронні і комп'ютерні системи. – 2009. – № 5. – С. 176–183.
8. *ModelSim* Xilinx. User's Manual. Version 6.0. – http://www.xilinx.com

Поступила 26.12.2010

Тел. для справок: (044) 530-7091, 424-8257, 402-9341, 503-7677 (Киев)
E-mail: *vlopanas@ukr.net*, *opanasenko@incyb.kiev.ua*, *dept*150*@insyg.kiev.ua*, *Lan-Sasha@yandex.ru*

●

В.Н. Опанасенко, А.Н. Химич, А.Н. Лисовый, Т.В. Чистякова

Решение задач с повышенной точностью вычислений

Актуальность использования средств с повышенной точностью вычислений

Одна из задач, ставшая актуальной с середины прошлого столетия – обеспечение достаточной точности вычислений на компьютере. Результат решения этой задачи – увеличение разрядности обрабатываемых данных.

В процессорах серии *Эльбрус* была заложена поддержка 128-битной арифметики. В компании *IBM* исследования поддержки арифметики с повышенной точностью велись с 1969 года и были внедрены в системах серии *IBM S*/390 *VM*. В 90-х годах процессоры стали массовым продуктом и самое широкое развитие начали приобретать компьютеры общего назначения (персональные), в которых не акцентируется на вычислениях с высокой точностью, что и послужило причиной остановки развития 128-битной арифметики. Оптимальной была определена разрядность сопроцессора с плавающей точкой 80 бит, что и наблюдается в подавляющем большинстве процессоров в настоящее время. Это можно объяснить тем, что рынок прежде всего ориентирован на бизнес-процессы и мультимедийные приложения, где такой разрядности вычислений вполне достаточно.

Сегодня аппаратную поддержку 128-битных вычислений имеют лишь несколько компаний, таких, как *IBM* – на базе операционной системы *AIX*, *Hewlett-Packard* – на базе операционной системы *HP-UX*, *AMD*-64 семейства 10*h* (процессоры *Phenom* с микропрограммной поддержкой) [1], *Sun Microsystems* – процессорная архитектура *SPARC-V*9 [2]. Некоторые операционные системы, среди которых *Solaris*, *IBM*, имеют программно-реализо-

ванную поддержку 128-битной арифметики с плавающей точкой.

В настоящее время недостаток точности ощущается как на рабочих станциях научных работников, так и на большинстве суперкомпьютеров с кластерной архитектурой, которые используют стандартные процессоры общего назначения. Среди задач, требующих вычислений с повышенной точностью (128 бит и больше), известны задачи аэродинамики, расчеты прочности, проектирование конструкций в авиастроении, обработка результатов и др.

Проблема решения задач с оценкой достоверности имеет несколько путей решения, которые можно разделить на *аппаратные* и *программные*. К аппаратным относятся: компьютерные средства на базе процессоров с поддержкой *FPU* 128 бит; компьютерные средства на базе процессоров *IBM POWER*6 с поддержкой десятичного *FPU* 128 бит (*DFP*); компьютерные средства, использующие арифметические устройства с разрядностью вычислений 128 бит или больше и подключаются к компьютеру через одну из стандартных шин (*PCI*, *PCI-e* или др.), такие устройства могут быть разработаны на основе реконфигурируемых плат на базе программируемых логических интегральных схем (ПЛИС). К программным принадлежат: пакеты математических вычислений, поддерживающие символьную арифметику (*Maple*, *Matlab* и др.); библиотеки, позволяющие программно обеспечить вычисления с произвольно заданными форматами (библиотеки *MPFR*, *GMP* [3] и др.).

Каждый подход имеет свои преимущества и недостатки. Отметим некоторые весомые нюансы: высокая стоимость серверов *IBM POWER*6; существенные накладные затраты для программных подходов; преимущество использования устройств на реконфигурируемых платах на базе ПЛИС состоит в возможности реализации как двоичной, так и десятичной арифметики и других нестандартных подходов, в том числе с использованием нестандартных форматов, при этом разрядность данных может быть 128 бит и больше.

Решение систем линейных алгебраических уравнений с повышенной точностью

Как следует из [4], возможность получения достоверного решения задачи зависит от согласования математических возможностей компьютеров (разрядность, архитектура) с математическими свойствами компьютерной задачи (обусловленность, корректность).

Таким образом, уточнить компьютерное решение задачи, зная обусловленность матрицы системы, очевидно, можно путем решения системы с повышенной разрядностью. В этом случае появляется принципиальная возможность достичь любой заданной точности компьютерного решения.

Для прогноза длины мантиссы машинного слова, обеспечивающего заданную точность для совместных систем, можно использовать следующее эмпирическое правило, которое следует из оценок [4]: количество корректных десятичных значащих цифр в компьютерном решении приблизительно равняется $\mu - \alpha$, где μ – количество цифр в десятичном представлении мантиссы числа с плавающей точкой, α – десятичный порядок числа обусловленности матрицы.

Рассмотрим задачу решения плохо обусловленной СЛАУ (система линейных алгебраических уравнений), анализируя ее число обусловленности; при этом можно ориентироваться на выбор формата с плавающей точкой (ФПТ), который необходимо использовать для ее решения [5].

Решение СЛАУ на компьютерной технике выполняется с помощью прямых и итерационных методов [4]. Прямые методы (Гаусса, *Lu*-разложения, Халецкого и др.) дают возможность, при вычислении на идеальном компьютере (вычисление и представление данных без погрешностей), получить решение задачи за конечное число арифметических операций. Итерационные методы (метод простой итерации, Якобе, Зейделя и др.) или методы последовательных приближений дают возможность получить приближенное решение, совпадающее с решением задачи при неограниченном количестве итераций.

Одним из важных факторов выбора того или другого метода при решении конкретной задачи есть вычислительная эффективность метода. Поскольку операция сложения выполняется гораздо быстрее операций умножения и деления, вычислительная эффективность метода будет зависеть от количества операций умножения и деления.

Метод Гаусса – один из эффективных методов решения СЛАУ с невырожденными матрицами общего вида. Для успешного использования алгоритма необходимо, чтобы входные коэффициенты были отличны от нуля, кроме того, близость их к нулю может приводить к большой погрешности решений. Количество арифметических действий прямого хода составляет $\approx(2/3)n^3$, обратного $\approx n^2$.

Постановка задачи

В качестве примера, рассмотрим СЛАУ (1) с числом обусловленности 3,55868727582796*E*16. Большое число обусловленности указывает на то, что система есть машинно-выродженной на ФПТ (*Single*, *Double*, *Extended*), поэтому для получения достоверного решения не может быть использована классическая компьютерная арифметика с плавающей точкой.

$$A=\begin{bmatrix} 0{,}1348531574394464 & 0{,}1878970588235294 & 0{,}1909117647058824 & 0{,}1779264705882353 \\ 0{,}1878970588235294 & 0{,}262 & 0{,}265 & 0{,}247 \\ 0{,}1909117647058824 & 0{,}265 & 0{,}281 & 0{,}266 \\ 0{,}1779264705882353 & 0{,}247 & 0{,}266 & 0{,}255 \end{bmatrix} \qquad (1)$$

$$b=\begin{bmatrix} 0{,}3516 \\ 0{,}4887 \\ 0{,}5105 \\ 0{,}4818 \end{bmatrix}, x=\begin{bmatrix} 6{,}662162162161606738798064490430572168030658239553569314802100940649855 9\ldots E12 \\ -4{,}016891891890723506952166298245477287559781737502963469048697496817269 0\ldots E12 \\ -1{,}665540540539970051894018639784241392354074075505748928758861569054296 6\ldots E12 \\ 9{,}797297297302797072574940696301157235597379312722179179629928555505557 2\ldots E12 \end{bmatrix}.$$

Как эталон решения системы (1) будем использовать числовое решение системы, полученное с использованием мантиссы шириной 512 бит.

Программные подходы решения СЛАУ

В настоящее время в качестве средства символьного решения прикладных задач используются пакеты *Maple*, *Matlab*, *Mathcad* и др. Эти программные продукты имеют большие возможности для форматирования математических текстов, различных расчетов, экспериментов, символьных вычислений и др. Однако нельзя не отметить, что эти средства – замкнутые, т.е. нет возможности использования их совместно с традиционными системами программирования, такими, как *Delphi*, *C++*, *Basic*.

Использование пакета Maple. Особенность использования символьной арифметики – отсутствие погрешности введения входных коэффициентов в систему, кроме тех случаев, когда пользователь ограничивает их принудительно. Однако погрешность вычислений наблюдается, и для ее снижения необходимо увеличивать количество десятичных цифр в обрабатываемых данных: для *Maple* это параметр *Digits*.

Для решения СЛАУ (1) рассмотрим зависимость точности вычислений от параметра *Digits* (табл. 1).

Таблица 1. Погрешности вычисления СЛАУ (1) с помощью пакету *Maple*

Digits	Среднеквадратичная погрешность, сравнение с эталоном
16	10^{1}
20	10^{-4}
30	10^{-13}
50	10^{-33}
70	10^{-53}

При анализе среднеквадратичной погрешности решений при *Digits* 16 и *Digits* 20 результат некорректен – это следствие плохой обусловленности матрицы решаемой СЛАУ. Для *Digits* 30 гарантированно получаем 13 десятичных цифр, использование *Digits* 70 гарантирует уже 53 десятичных цифры.

Преимущество использования символьной арифметики проявляется в возможности выполнения чрезвычайно точных вычислений. Существенный недостаток символьной арифметики состоит в большом времени решения задачи, которое по отношению к классической компьютерной арифметике значительно увеличено.

Использование библиотеки MPFR. Данная библиотека заслуживает особого внимания, так как программист, использующий ее, создает собственный формат с плавающей точкой, указывая разрядность порядка числа и его мантиссы. Таким образом, можно создавать форматы, не описанные в современных стандартах, однако выполнять классические арифметические операции с плавающей точкой.

Рассмотрим зависимость точности вычислений от ширины мантиссы (табл. 2). Корректные десятичные знаки в решениях выделены полужирным шрифтом.

Отметим, что, увеличивая ширину мантиссы вдвое, количество корректных десятичных знаков решений увеличивается вдвое почти линейно. Ширина мантиссы 512 бит – далеко не предел, есть возможность использовать 1024, 2048 и больше. Например, используя ширину мантиссы 40980 бит, количество корректных десятичных знаков решений будет равно 12320. Временные затраты для таких вычислений, по отношению к классической компьютерной арифметике, значительно увеличиваются.

Кроме рассмотренных программных подходов, существуют эмуляторы *DFPU*, позволяющие с учетом формата данных, отличного от двоичного, выполнять и более сложные вычисления, получая более точные результаты по отношению к *FPU* такой же разрядности. Эмулятор *Decimal Floating-Point Math Library* фирмы *Intel* [6] поддерживает форматы *decimal*32, *decimal*64 и *decimal*128 (32, 64, 128 битные варианты соответственно)

Преимуществом десятичного ФПТ по отношению к двоичному состоит в большем диапазоне представления чисел, т.е. некоторое множество задач, приводившее в процессе вычислений на двоичных ФПТ к *Nan*-Результату, на десятичных ФПТ может быть разрешено. Другое преимущество эмуляторов *DFPU* – возможность использовать их на любом компьютере платформ *x*86, *x*64.

Таблица 2. Результаты и погрешности решения СЛАУ (1) с помощью *MPFR*

Результат	Погрешность
параметр *precision* (ширина мантиссы) 128 бит ***X*1** = 6,662162162161606738798**17842726242196300097058195990667138486340803638 08**…× **10^{12}** ***X*2** = –4,016891891890723506952**2349954529161335077159939636629536717504773690734**…× **10^{12}** ***X*3** = –1,665540540539970051894**0471239922038446691827554383941547341135835935688**…× **10^{12}** ***X*4** = 9,797297297302797072575**1082504656422558394110540814752187237235414585256**…× **10^{12}**	10^{-22}
параметр *precision* (ширина мантиссы) 256 бит ***X*1** = 6,662162162161606738798064490430572168030658239553569314802100**1531195230**…× **10^{12}** ***X*2** = –4,016891891890723506952166298245477287559781737502963469048697**0219828036**…× **10^{12}** ***X*3** = –1,665540540539970051894018639784241392354074075505748928758861**3721717134**…× **10^{12}** ***X*4** = 9,797297297302797072574940696301157235597379312722179179629927**3973727147**…× **10^{12}**	10^{-62}
параметр *precision* (ширина мантиссы) 512 бит ***X*1** = 6,662162162161606738798064490430572168030658239553569314802100940649855 9…× **10^{12}** ***X*2** = –4,016891891890723506952166298245477287559781737502963469048697496817269 0…× **10^{12}** ***X*3** = –1,665540540539970051894018639784241392354074075505748928758861569054296 6…× **10^{12}** ***X*4** = 9,797297297302797072574940696301157235597379312722179179629928555505557 2…× **10^{12}**	10^{-137}

Аппаратная реализация для решения СЛАУ на базе ПЛИС

Для решения СЛАУ методом Гаусса разработаны устройства (обобщенная блок-схема приведена на рис. 1) для следующих форматов данных: *Single*, *Double*, *Extended*, *Quadruple* [6], где УУ – устройство управления, ПЗУ – программирующее запоминающее устройство, ОЗУ – оперативно-запоминающее устройство, БОД – блок обработки данных [7], *PCI-e* – шина *PCI Express*, *CE* – *chip enable* (сигнал начала работы).

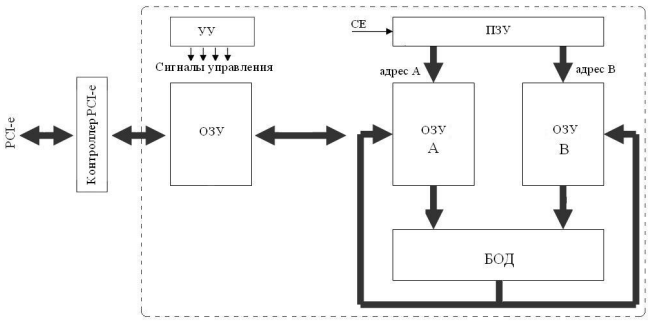

Рис. 1. Обобщенная блок-схема устройства

Для каждого из форматов данных все составные компоненты схемы, кроме ПЗУ, должны иметь соответствующую разрядность: для *Quadruple* – 128 разрядов, для *Double* – 64 разряда и т.д. В ПЗУ хранится запрограммированный алгоритм решения СЛАУ методом Гаусса. Микропрограмма, одинаковая для всех форматов, программируется на этапе формирования ПЗУ с помощью *Core Generator*. Для решения СЛАУ четвертого порядка необходим объем памяти 256 слов по 20 бит. Формат команды, которая хранится в ПЗУ, приведен на рис. 2. Для решения систем больших порядков, кроме увеличения объема ПЗУ, будет увеличиваться ширина команды, так как увеличится объем ОЗУ, а потому и поля для их адресации. Блок памяти ОЗУ есть буфером при пересылке данных из компьютера в устройство и наоборот. УУ отвечает за переключение режима работы ОЗУ (из работы во внешнем режиме на работу во внутреннем режиме и наоборот), а также управляет работой всех блоков устройства.

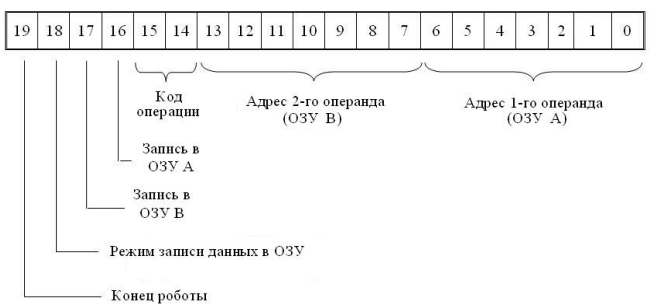

Рис. 2. Формат команды

В качестве примера использована моделирующая плата *XC*4*VFX*100 фирмы *PLDA* с кристаллом ПЛИС типа *Virtex* 4 *FX*100-*ff*11, которая подключается к компьютеру через шину *PCI-e*.

Объем ОЗУ равен 256 словам, а его ширина соответствует ширине формата данных. Блоки ОЗУ *A*, *B* используются для хранения начальных коэффициентов, результатов и промежуточных данных при вычислении. Блок БОД имеет три необходимых для метода Гаусса арифметические ядра (сложение–вычитание, умножение, деление) соответствующей разрядности. Функционирование УУ включает в себя три этапа:

- запись входных коэффициентов из ОЗУ в ОЗУ *A* и ОЗУ *B*;
- выполнение микропрограммы, описывающей метод Гаусса;
- запись результата из ОЗУ *B* в ОЗУ; формируется сигнал, символизирующий окончание выполнения микропрограммы, ОЗУ переключается на работу с контроллером *PCI-e*.

Функционирование устройства состоит также из трех этапов:

- запись коэффициентов СЛАУ в ОЗУ по шине *PCI-e*;
- в ячейку ОЗУ по адресу 160 записываются значения *F*, что служит сигналом начала работы УУ, при этом ОЗУ переключается на работу из ОЗУ *A* и ОЗУ *B*;
- считывание результатов из ОЗУ по шине *PCI-e*.

Тестирование устройства происходит на этапах разработки отдельных блоков, а также устройства в целом. Для тестирования используется система моделирования *Modelsim XE* 6,0*d* [8], которая позволяет выполнить верификацию на всех этапах проектирования. Полученные аппаратно-временные характеристики устройства приведены в табл. 3.

Таблица 3. Аппаратно-временные характеристики устройств на *Virtex* 4 *FX*100-*ff*11

Использованные ресурсы	Форматы данных	
	Double	*Quadruple*
Количество *Slices*	2335 (4%)	5007 (11%)
Количество *DSP*48	13 (8%)	49 (30%)
Количество *RAMB*16	11 (4%)	13 (4%)
Период *Clk*, нс	8	64

Решение СЛАУ (1). После вычислений с соответствующими коэффициентами *A* и *b*, получим следующие результаты (табл. 4).

Отметим, что обработка данных, использующая форматы *Single* и *Double* не дает корректных решений, а *Extended* гарантирует только две первые цифры решений. Решение на *Quadruple* гарантирует 17 десятичных цифр (ширина мантиссы в формате *Quadruple* 113 бит).

Итак, для решения СЛАУ (1) имеются следующие пути: использование символьной арифметики с заданной большой точностью вычислений (больше 60 десятичных знаков); использование компьютерных систем с аппаратной поддержкой (*FPU*, *DFPU* 128 бит); использование устройств на основе реконфигурированных плат и

Таблица 4. Результаты решения СЛАУ аппаратным путем

Формат	Устройство, использующее разработанные арифметические блоки	Устройство, использующее *IP-Core* фирмы *Xilinx*
Single	***x*1 =–64153,664062500000000000** ***x*2 = 38681,507812500000000000** ***x*3 = 16039,548828125000000000** ***x*4 = –9434,2568359375000000000**	***x*1=–27856,2812500000000** ***x*2= 16796,2792968750000** ***x*3= 6965,29150390625000** ***x*4=–4096,46044921875000**
Double	***x*1 = 2264503194623,36000000000000000** ***x*2 = –1365362220253,90000000000000000** ***x*3 = –566125798647,227000000000000000** ***x*4 = 333015175637,347000000000000000**	***x*1= 3343472363451,5000000** ***x*2=–2015917160209,6700000** ***x*3=–835868090854,30700000** ***x*4= 491687112298,69300000**
Extended	***x*1** = 6655584185991,900460000000000000 ***x*2** = –401**2925759200,165400000000000000** ***x*3** = –166**3896046497,543230000000000000** ***x*4**= 97**8762380293,5580170000000000000**	***x*1**= 665**7982426662,6836300** ***x*2**=–401**4371757251,6669800** ***x*3**=–166**4495606665,2391300** ***x*4**= 979**115062745,14385500**
Quadruple	***x*1** = 6662162162161,6067**1660078797793027170401 86** ***x*2** = –4016891891890,723**4935685142833558782053 10** ***x*3** = –1665540540539,9700**4634469951165827217759 4** ***x*4** = 979729729730,2797**0399318870014412337350964**	Отсутствуют арифметические блоки

программных эмуляторов широкоформатных вычислений с плавающей точкой (128 бит и больше).

В табл. 5 приведены значения решений СЛАУ с разными типами округления.

Заключение. При решении СЛАУ классическим компьютерным путем полученные решения не всегда сохраняют машинную суть, поэтому корректное решение СЛАУ может быть получено с помощью программно-алгоритмических и аппаратных средств.

Для решения задач с повышенной точностью разработаны устройства на базе ПЛИС. Апробация этого подхода показала, что увеличение двоичной разрядности может обеспечить достоверность решения СЛАУ. В арифметических компонентах целесообразно использовать округление второго типа.

Таблица 5. Влияние типа округления в арифметических компонентах

Формат	Округление первого типа	Округление второго типа
Quadruple	***x*1 = 402983C9F3C87C66D471DD94C71D8BDC** **(**6662162162161,6067**1660078797793027170401)** ***x*2 = C028D3A0B508595C9B6FEFD14DA782CC** **(**–4016891891890,723**4935685142833558782053)** ***x*3 = C02783C9F3C87BF854F50E16B806BFB4** **(**–1665540540539,9700**4634469951165827217759 4)** ***x*4 = 4026C838E291848F355C9E9CB5B70D1A** **(**979729729730,2797**0399318870014412337350964)**	***x*1 = 402983C9F3C87C66D4894D6545FEACA7** **(**6662162162161,6067**2218855060570521637962)** ***x*2 = C028D3A0B508595C9B8C32E3501156C2** **(**-4016891891890,723**4969376064559848065667)** ***x*3 = C02783C9F3C87BF8550C7DE736E7E52D** **(**-1665540540539,9700**4774164016860226203 28)** ***x*4 = 4026C838E291848F35783137A55685D6** **(**979729729730,2797**0481491849834650706856)**

Влияние типа округления в арифметических компонентах на конечный результат

Тип округления в арифметических компонентах влияет на точность результатов.

В арифметических компонентах используется два типа округления:

- если старший бит части вектора, который не помещается в разрядную сетку, имеет значение «1», то выполняется запись «1» в младший бит мантиссы;
- если старший бит части вектора, который не помещается в разрядную сетку, имеет значение «1», то к мантиссе добавляется «1» младшего бита.

Универсальность разработанной схемы устройств позволит использовать ее в случае необходимости для решения задач большей размерности и более широких форматов данных, которые могут быть стандартизированы или заданы произвольно.

Таким образом, для решения задач с повышенной точностью вычислений целесообразно использовать одно из рассмотренных средств. Выбор средства будет зависеть от поставленной задачи и требований к точности результата.