УДК 681.325.5

АЛГОРИТМІЧНІ ОСНОВИ ПОБІТОВОЇ ОБРОБКИ КОДІВ ЗОЛОТОЇ ПРОПОРЦІЇ

О.Д. Азаров, О.І. Черняк

Актуальність

При організації паралельних обчислень за рахунок розподіленої обробки виникає проблема, що полягає у необхідності реалізації великої кількості інформаційних зв'язків між окремими пристроями. Одним з відомих підходів вирішення даної проблеми є порозрядна конвеєрна обробка послідовних кодів чисел, починаючи зі старших розрядів, що дозволяє у десятки разів зменшити кількість інформаційних зв'язків без суттєвого зменшення продуктивності. Реалізація пристроїв для конвеєрної порозрядної обробки вимагає використання надлишкових систем числення, оскільки лише в них можливе обмеження довжини перенесення у старші розряди при додаванні і відніманні. Від довжини перенесення залежать апаратні витрати пристроїв порозрядної конвеєрної обробки послідовних кодів. Побітова конвеєрна обробка є окремим випадком порозрядної конвеєрної обробки. При побітовій обробці досягається найменша кількість інформаційних зв'язків між розподіленими пристроями. Тому актуальним є побітове конвеєрне виконання розподілених обчислень з використанням надлишкової системи числення, що має найменшу довжину перенесення.

Аналіз останніх досліджень

Серед відомих надлишкових позиційних систем числення для конвеєрної порозрядної обробки послідовних кодів чисел найбільш широко використовується знакорозрядна система числення [1,2,3,4] з використанням відомих неавтономних алгоритмів обробки. На кожному такті у неавтономній операції беруть участь два чергових розряди кодів операндів і так звана поправка, код якої визначається на попередньому такті. У попередніх наукових працях авторами запропоновано загальний опис класу систем числення, що дозволяють виконувати порозрядну обробку зі старших розрядів [5]. Вони названі АМ-системами числення. Особливістю АМ-систем числення є наявність у них адитивного співвідношення певного типу між вагами розрядів. Тому у цих системах числення можна виконувати адитивні перетворення над кодами чисел, що є узагальненням перенесення і позичання при додаванні і відніманні. За умовою виконання адитивні перетворення діляться на елементарні (ЕА-перетворення), універсальні (UA-перетворення) та повні (FA-перетворення). За напрямком перенесення адитивні перетворення поділяються на перетворення з перенесенням у старші розряди (L-перетворення) та перетворення з перенесенням у молодші розряди (R-перетворення). Окремим випадком АМ-систем числення є коди золотої пропорції, описані у [6,7]. Ці коди дозволяють виконувати побітову обробку.

Мета та постановка задач

Метою статті є опис результатів, отриманих у процесі розробки алгоритмічних основ конвеєрного побітового виконання арифметичних операцій над кодами золотої пропорції. Побітова обробка послідовних кодів золотої пропорції має певні особливості. При побітовому додаванні і відніманні кодів золотої пропорції замість коду поправки використовується код проміжного результату, що являє собою частину розрядів результату, отриманого на черговому такті, через які можливе розповсюдження перенесення або позичання у наступному такті виконання операції. Код проміжного результату формується за допомогою операцій адитивного перетворення кодів золотої пропорції. Алгоритми побітового множення і ділення кодів золотої пропорції принципово відрізняються від відомих неавтономних алгоритмів, оскільки в них взагалі не використовується якийсь певний код числа, що бере участь в операції разом з розрядами кодів операндів. Це викликано тим, що розрядність коду поправки у відомих неавтономних алгоритмах множення і ділення не менша розрядності кодів операндів. Тому визначення поправки на кожному такті побітового множення та ділення послідовних кодів золотої пропорції за допомогою повнорозрядних операцій додавання чи віднімання, як це відбувається у неавтономних алгоритмах, вимагатиме багато часу на розповсюдження перенесення і призведе до зменшення продуктивності обробки. Тобто, в алгоритмах побітового множення і ділення потрібно використовувати побітове додавання і віднімання для формування результату.

Таким чином, при розробці алгоритмічних основ конвеєрного побітового виконання арифметичних операцій над кодами золотої пропорції постають такі задачі:

1. Розробка алгоритмів побітового додавання і віднімання на основі адитивних перетворень та визначення довжини розповсюдження перенесення і позичання.
2. Розробка алгоритмів побітового множення і ділення на основі побітового додавання і віднімання.

Загальні принципи побітової обробки

Спільним у всіх алгоритмах побітової обробки послідовних кодів золотої пропорції є побітове подання кодів операндів, починаючи зі старших розрядів і побітове формування коду результату також зі старших розрядів. Причому, розряди коду результату формуються у темпі подання розрядів кодів операндів, але з деякою сталою затримкою у часі. Позначимо коди операндів X та Y, а код результату Z. На кожному i-му такті виконання побітової операції на вхід пристрою подаються чергові розряди кодів операндів $x_{n-i-1}\cdot\alpha^{n-i-1}$ та $y_{n-i-1}\cdot\alpha^{n-i-1}$ і утворюється черговий розряд коду результату $z_{n-i-1}\cdot\alpha^{n-i+d}$, вага якого більша ваг оброблюваних розрядів на довжину розповсюдження перенесення у старші розряди.

Побітове додавання

При побітовому додаванні кодів золотої пропорції на кожному такті може виникнути перенесення як у старші, так і в молодші розряди. Тому у загальному випадку на кожному такті виконується додавання трьох значень: чергових розрядів кодів доданків і відповідного розряду коду перенесення у молодші розряди. Отже, на кожному такті побітового додавання можливі такі комбінації:

$$\begin{array}{r}0\\+\;0\\\underline{0}\\0\end{array}\quad\begin{array}{r}0\\+\;0\\\underline{1}\\1\end{array}\quad\begin{array}{r}0\\+\;1\\\underline{0}\\1\end{array}\quad\begin{array}{r}1\\+\;0\\\underline{0}\\1\end{array}\quad\begin{array}{r}0\\+\;1\\\underline{1}\\2\end{array}\quad\begin{array}{r}1\\+\;0\\\underline{1}\\2\end{array}\quad\begin{array}{r}1\\+\;1\\\underline{0}\\2\end{array}\quad\begin{array}{r}1\\+\;1\\\underline{1}\\3\end{array}$$

Для ліквідації переповнення виконується послідовність EAR- та EAL-перетворень, що разом утворюють UAL-перетворення. Для обмеження розповсюдження перенесення у старші розряди на кожному такті побітового додавання виконується операція FAL-перетворення, що складається з EA-перетворень. У загальному випадку, після виконання FAL-перетворення у деякій групі розрядів хоча б один з двох старших розрядів цієї групи матиме нульове значення. Тому додаванні в молодших по відношенню до даної групи розрядах перенесення в старші розряди, попадаючи на нульовий розряд, не розповсюджується далі. Для виконання FAL-перетворення вказаної групи розрядів їх потрібно аналізувати та обробляти паралельно. Код результату, отриманого на i-му такті побітового додавання, складається з двох частин: групи розрядів, через які можливе розповсюдження перенесення на наступному такті, і групи розрядів, значення яких в подальшому не буде змінюватись. Частина розрядів, що не змінюватиме в подальшому свого значення являє собою черговий код результату побітового додавання. Частина розрядів, значення яких може змінюватись на наступному такті, є черговим проміжним результатом S(i). Проміжний результат, отриманий на кожному i-му такті побітового додавання, має у своєму коді, крім (n-i-1)-го розряду, ще старші та молодші від нього розряди:

$$S(i) = \alpha^{n-i-1-b} \cdot \sum_{j=0}^{d+b} s(i)_j \cdot \alpha^j$$

Отриманий на даному такті черговий розряд коду суми може бути переданий в інший побітовий пристрій для подальшої обробки, а черговий проміжний результат запам'ятовується у даному пристрої і враховується на наступному такті додавання:

$$(x_{n-i-1} + y_{n-i-1}) \cdot \alpha^{n-i-1} + S(i-1) \to S(i) + z_{n-i+d} \cdot \alpha^{n-i+d} .$$

Отже, загальний алгоритм побітового додавання на кожному такті повинен виконувати такі етапи:

1. Зсув коду попереднього проміжного результату на один розряд вліво;
2. Додавання чергових розрядів кодів доданків до відповідного розряду коду отриманого проміжного результату;
3. Виконання FAL-перетворення отриманого коду проміжного результату та формування значення чергового розряду коду суми.

Розглянемо кожен з етапів більш детально. На першому етапі при зсуві коду попереднього проміжного результату S(i-1) на один розряд вліво його молодший розряд встановлюється у нульове значення. Тому код проміжного результату 'S(i), отриманого на даному етапі, визначається за виразом

$$'S(i) = S(i-1) \cdot \alpha .$$

На другому етапі у кожному i-му такті у додаванні приймають участь три розряди: два чергових розряди кодів операндів $x_{n-i-1}\cdot\alpha^{n-i-1}$ та $y_{n-i-1}\cdot\alpha^{n-i-1}$, та i-й розряд $'s(i)_y$ зсунутого коду попереднього проміжного результату. При цьому формується код нового проміжного результату:

$$''S(i) = 'S(i) + (x_{n-1-i} + y_{n-1-i}) \cdot \alpha^{n-1-i} .$$

Отже, на даному етапі можливе переповнення і-го розряду. Для ліквідації можливого переповнення на третьому етапі використовується FAL-перетворення усіх розрядів коду проміжного результату:

$$S(i) = FAL_{d+b}(''S(i)).$$

FAL-перетворення виконується як послідовність EAL-перетворень. При цьому слід враховувати ситуацію, при якій після додавання окремих розрядів значення другого розряду коду проміжного результату може бути більше від одиниці

$$''s(i)_b > 1.$$

Якщо у даній ситуації сусідні розряди коду проміжного результату дорівнюють нулю

$$''s(i)_{b-1} = 0 \text{ і } ''s(i)_{b+1} = 0,$$

то для виконання даного перетворення потрібно спочатку виконати EAR-перетворення з і-го розряду вказаного коду.

Визначимо необхідну розрядність коду проміжного результату S(i). Для цього потрібно визначити значення d довжини розповсюдження перенесення у старші розряди та значення b довжини розповсюдження перенесення у молодші розряди. Спочатку покажемо, що максимальна довжина розповсюдження перенесення у молодші розряди b=2. Припустимо, що кількість старших розрядів d коду проміжного результату достатня для того, щоб не було переповнення за рахунок перенесення у старші розряди при додаванні. Очевидно, що на першому такті побітового додавання b≤2. Покажемо, що якщо на деякому (i-1)-му такті b≤2, то ця умова виконується також на наступному i-му такті. Виконаємо FAL-перетворення коду проміжного результату, отриманого на (i-1)-му такті. Тоді початковий код проміжного результату на і-му такті не матиме двох сусідніх одиниць. Допущення b≤2 для (i-1)-го такту означає, що значення будь-якого розряду коду S(i-1) не більше одиниці. Тому на і-му такті

$$x_{n-i-1} + y_{n-i-1} + 's(i)_b \le 3,$$

тобто, значення переповненого розряду коду проміжного результату може бути тільки два або три. Нехай на і-му такті побітового додавання у другому розряді коду проміжного результату виникло переповнення. Розглянемо два випадки. Перший випадок – перший або третій розряди коду проміжного результату мають одиничне значення. Другий випадок – дані розряди мають нульове значення. У першому випадку початкове значення другого розряду коду проміжного результату матиме нульове значення. Тому у переповненому стані цей розряд може мати значення тільки два. Якщо при цьому одиничне значення має третій розряд коду проміжного результату, то четвертий його розряд матиме нульове значення. EAL-перетворення у четвертий розряд ліквідує переповнення:

0	1	2		
+1	-1	-1		$EAL_3(S)$
1	0	1		

Якщо ж у першому випадку одиничне значення має не третій, а перший розряд, тоді EAL-перетворення у третій розряд також ліквідує переповнення:

	0	2	1	
	+1	-1	-1	$EAL_2(S)$
	1	0	1	

У другому випадку переповнене значення другого розряду коду проміжного результату може бути два або три. Оскільки при цьому нульовий і перший розряди цього коду мають нульове значення, то спочатку виконується EAR-перетворення другого розряду. Це призводить до перенесення у нульовий та перший розряди. Далі виконується EAL-перетворення у третій розряд, а потім, якщо необхідно, виконуються EAL-перетворення у більш старші розряди. Розглянемо обидва варіанти другого випадку:

1.

	0	2	0		
		-1	+1	+1	$EAR_1(S)$
	0	1	1	1	
+1	-1	-1			$EAL_2(S)$
1	0	0	1		

2.

		0	3	0		
			-1	+1	+1	$EAR_1(S)$
		0	2	1	1	
	+1	-1	-1			$EAL_2(S)$
	1	1	0	1		
+1	-1	-1				$EAL_3(S)$
1	0	0	0	1		

Як видно з цих варіантів, EAR-перетворення зменшує на одиницю значення другого розряду коду проміжного результату. Після нього виконується EAL-перетворення отриманого коду у третій розряд і далі, якщо необхідно, у четвертий. Кожне з цих перетворень також зменшує на одиницю значення другого розряду коду проміжного результату. Тому після виконання вказаних EA-перетворень в одиничному значенні може залишитись тільки нульовий розряд даного коду. Отже, перенесення у молодші розряди може формуватися тільки тоді, коли нульовий і перший розряди коду проміжного результату мають нульове значення. Тому воно не розповсюджується далі в молодші розряди. Тобто, максимальна довжина перенесення в молодші розряди дорівнює двом розрядам (b=2).

Покажемо, що перенесення в старші розряди розповсюджується не далі, ніж у третій розряд відносно поточного. Тобто, d=2. При цьому кількість розрядів коду проміжного результату S визначається як d+b+1=5. Очевидно, що максимальне перенесення у коді проміжного результату буде при максимальних значеннях поточних розрядів кодів доданків та максимальному значенні коду попереднього проміжного результату. Можливі два випадки значення попереднього проміжного результату:

1) старший розряд коду проміжного результату на (i-1)-му такті дорівнює одиниці;

2) старший розряд коду проміжного результату на (i-1)-му такті дорівнює нулю.

Розглянемо окремо кожен з цих випадків. У першому випадку п'яти-розрядний код проміжного результату з максимальним значенням матиме вид 10101. Покажемо, що такий код проміжного результату не може виникнути при побітовому додаванні. Визначимо вимоги, яким повинні задовольняти значення поточних розрядів кодів доданків і код попереднього проміжного результату для того, щоб одержати код 10101 нового проміжного результату. Одиничне значення нульового розряду коду нового проміжного результату може бути лише при EAR-перетворенні з другого розряду. Тому в цьому розряді повинна бути хоча б одна одиниця. Виконання EAR-перетворення з другого розряду встановлює перший розряд коду проміжного результату в одиничне значення. Для отримання нульового значення цього розряду при одиничному значенні нульового розряду потрібно виконати EAL-перетворення у третій розряд. Дане перетворення, потребує ще однієї одиниці у другому розряді коду проміжного результату. Тобто значення даного розряду повинно бути не меншим двох. Для того, щоб у другому розряді остаточного коду проміжного результату залишилась одиниця, потрібно щоб після додавання чергових розрядів кодів доданків його значення дорівнювало трьом. Це означає, що на попередньому такті перший розряд коду проміжного результату повинен мати одиничне значення. Очевидно, що завжди після EAR-перетворення з другого розряду коду переповненого проміжного результату на цьому ж такті виконується EAL-перетворення в його третій розряд. Тому нульовий і другий розряди початкового коду чергового проміжного результату, отриманих зсувом коду попереднього проміжного результату на один розряд вліво, повинні мати нульові значення. Позначимо значення третього і четвертого розрядів коду попереднього проміжного результату відповідно як a1 та a2. Значення чергового розряду коду суми позначимо через a3, де a1,a2,a3 $\in \{0,1\}$. Тоді процес додавання двох одиниць чергових розрядів кодів доданків матиме вид:

	a2	a1	0	1	0		
+				1			
				1			
				3			
				-1	+1	+1	$EAR_1(S)$
				2	1	1	
			+1	-1	-1		$EAL_2(S)$
			1	1	0	1	
			...				
	a3	1	0	1	0	1	

У даному випадку можливість отримання коду 10101 проміжного результату означає, що справедлива рівність:

$$a1 \cdot \alpha^5 + a2 \cdot \alpha^4 + 3 \cdot \alpha^2 = a3 \cdot \alpha^5 + \alpha^4 + \alpha^2 + 1. \quad (1)$$

Тотожні перетворення даної рівності приводять до виразу:

$$a1 \cdot \alpha^2 + a2 \cdot \alpha + 1 = a3 \cdot \alpha^2 + \alpha.$$

Розглянемо випадок, при якому a2=0. Тоді рівність (1) матиме вид:

$$a1 \cdot \alpha^2 + 1 = a3 \cdot \alpha^2 + \alpha.$$

Очевидно, що отримана рівність не виконується. Розглянемо випадок, при якому a2=1. Тоді рівність (1) матиме вид:

$$a1 \cdot \alpha^2 + 1 = a3 \cdot \alpha^2.$$

Дана рівність теж не виконується ні при яких значеннях a1 і a3. Тобто, рівність (1) не виконується ні при яких значеннях старших розрядів перетворюваного коду. Це означає, що неможливо отримати на деякому такті побітового додавання код 10101 проміжного результату. Тому кодом проміжного результату з максимальним значенням у першому випадку може бути тільки 10100.

Максимальне перенесення у старші розряди коду проміжного результату буде при додаванні до коду 10100 двох одиниць поточних розрядів кодів доданків:

		1	0	1	0	0	S(i-1)
+					1		
					1		
	1	0	1	2	0	0	
		+1	-1	-1			$EAL_3(S)$
	1	1	0	1	0	0	S(i)

Отже, у даному випадку перенесення розповсюджується не далі, ніж у четвертий розряд коду проміжного результату. На і-му такті проміжний результат матиме такий самий код, як і на (і-1)-му такті, тому максимальне перенесення в ньому теж буде таким, як на попередньому такті. Тобто, переповнення коду проміжного результату у даному випадку не буде. Якщо ж на (і-1)-му такті проміжний результат матиме значення менше ніж 10100, то також не виникне переповнення цього результату на і-му такті. Інакше це означало б, що на даному такті при меншому початковому значенні коду проміжного результату і однакових значеннях чергових розрядів кодів доданків результатом додавання є код значення якого більше ніж значення коду, отриманого на попередньому такті з більшим початковим значенням проміжного результату. Таким чином, для врахування перенесення у старші розряди при одиничному значенні старшого розряду коду проміжного результату на попередньому такті достатньо двох розрядів.

У другому випадку, якщо старший розряд коду попереднього проміжного результату дорівнюватиме нулю, то код його максимального значення матиме вид 01010. Максимальне перенесення у старші розряди даного коду буде при додаванні до нього двох одиниць чергових розрядів кодів доданків:

		0	1	0	1	0	S(i-1)
+					1		
					1		
	0	1	0	3	0	0	
				-1	+1	+1	$EAR_1(S)$
	0	1	0	2	1	1	
			+1	-1	-1		$EAL_2(S)$
	0	1	1	1	0	1	
	+1	-1	-1				$EAL_4(S)$
	1	0	0	1	0	1	

У даному випадку перенесення розповсюджується у п'ятий розряд коду проміжного результату. Отже, у всіх випадках максимальне перенесення у старші розряди, що виникає на кожному такті додавання, розповсюджується не далі, ніж у третій розряд коду проміжного результату. Тобто, $d \leq 2$. Покажемо, що при $d<2$ виникає переповнення проміжного результату. Для цього розглянемо приклад додавання кодів 1111+1111:

	+	1	1	1	1		
		1	1	1	1		
0	0	2	0	0			
0	0	1	1	1			
0	1	0	0	1			
1	1	0	2	1	0		
1	1	1	1	0	0		
2	1	1	1	2	0	0	
2	1	1	1	1	1	1	
3	1	1	1	1	3	1	0

З прикладу видно, що на третьому такті побітового додавання виникає переповнення проміжного результату, що не може бути ліквідовано AL-перетворенням. Тому кількість паралельно оброблюваних розрядів коду проміжного результату на кожному такті побітового додавання послідовних кодів золотої пропорції дорівнює п'яти. Тобто, d=2.

Таким чином, при побітовому додаванні кодів золотої пропорції код проміжного результату повинен мати п'ять розрядів: один розряд – для формування розрядної суми, два розряди – для запам’ятовування перенесення у старші розряди і два розряди – для запам’ятовування перенесення у молодші розряди.

У процесі побітового додавання кодів золотої пропорції на кожному такті виконується адитивне перетворення коду отриманого проміжного результату. Це перетворення складається з $EAR_2(S)$-перетворення та послідовності $EAL_1(S)$-, $EAL_2(S)$-, $EAL_3(S)$- та $EAL_4(S)$-перетворень, починаючи з молодшого. В кінці адитивного перетворення проміжного результату виконується ще одне $EAL_1(S)$-перетворення. Граф-схема алгоритму побітового додавання подана на рис. 1.

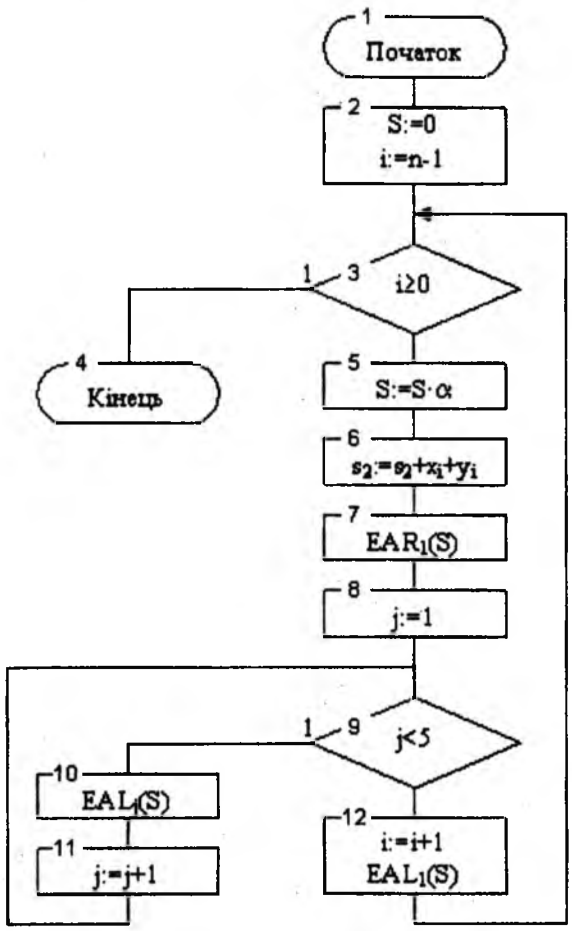


Рис. 1 - Граф-схема алгоритму побітового додавання кодів золотої пропорції

Існує альтернативний варіант виконання операції побітового додавання кодів золотої пропорції, при якому EAR-перетворення третього розряду коду проміжного результату відбувається тільки у випадку переповнення другого розряду коду отриманого проміжного результату і нульових значеннях першого і третього його розрядів. Альтернативний варіант дозволяє зменшити кількість елементарних адитивних перетворень для, проте він вимагає визначення додаткової умови для виконання EAR-перетворення першого розряду коду проміжного результату. Граф-схема алгоритму побітового додавання за альтернативним варіантом подана на рис. 2.

Побітове віднімання

При повнорозрядній обробці паралельних кодів чисел для виконання віднімання використовується додавання доповняльних кодів, при якому іноді виконується перенесення у знаковий розряд для забезпечення коректного визначення знаку результату. Побітова обробка базується на обмеженості довжини перенесення у старші розряди. Однак, при побітовій обробці зі старших розрядів знак результату може бути не відомим протягом кількості тактів, що перевищує довжину розповсюдження перенесення для заданої системи числення. У таких випадках використання доповняльних кодів призводить до неправильного визначення знаку результату. Одним з можливих негативних наслідків використання доповняльних кодів при побітовій обробці є неправильне перетворення результатів побітових операцій з доповняльного у звичайний код. Тому для побітового віднімання кодів золотої пропорції потрібно використовувати не доповняльні, а прямі коди. Це призводить до необхідності розробки окремої операції віднімання.

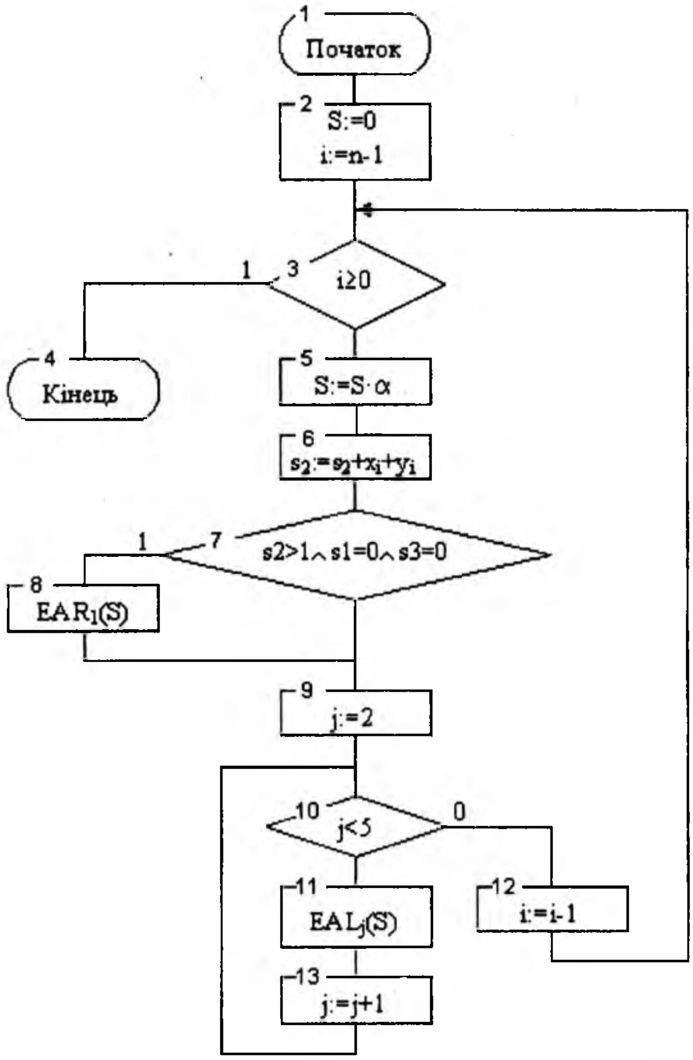


Рис. 2 - Граф-схема алгоритму побітового додавання кодів золотої пропорції зі зменшеною кількістю елементарних адитивних перетворень

Конвеєрне побітове віднімання кодів золотої пропорції виконується за принципами, аналогічними додаванню, і основане на відніманні окремих розрядів. Розглянемо випадок віднімання додатних кодів Z=X-Y, причому будемо вважати, що X≥Y. При побітовому відніманні зі старших розрядів на кожному i-му такті віднімаються цифри чергових розрядів кодів доданків x_{n-i-1} і y_{n-i-1}. Правила віднімання цифр окремих розрядів звичайні для будь-якої системи числення з двійковими цифрами. Навіть якщо X>Y, можлива ситуація, при якій $x_{n-i-1}<y_{n-i-1}$. Тобто, при відніманні одиниць в однойменних розрядах може виникнути від'ємне значення у відповідному розряді коду різниці. За умови X≥Y для ліквідації від'ємного значення у розряді використовується позичання, що здійснюється за рахунок послідовності EAR-перетворень з більш старших розрядів. Отже, при побітовому додаванні кодів золотої пропорції на кожному такті може виникнути перенесення у молодші розряди, що враховується на наступному такті. У загальному випадку на кожному такті виконується операція над трьома значеннями: черговими розрядами кодів зменшуваного і від'ємника та відповідним розрядом коду перенесення у молодші розряди. Перенесення у молодші розряди може мати як додатне, так і від'ємне значення. Тобто, відповідний розряд коду перенесення у молодші розряди може мати три значення: -1, 0 та +1. Оскільки X≥Y, то неможливе одночасне надходження нульового значення чергового розряду коду зменшуваного, одиничного значення чергового розряду коду від'ємника і від'ємного перенесення у відповідний молодший розряд. Тому можливі такі комбінації віднімання у черговому розряді:

0	0	1	0	1	1	0	1	1	1
0	-1	-1	0	0	-1	0	0	-1	0
-1	0	-1	0	-1	0	1	0	1	1
-1	-1	-1	0	0	0	1	1	1	2

Для того, щоб позичання в одному такті не розповсюджувалось на велику відстань, на кожному такті віднімання потрібно виконувати EAR-перетворення старшого розряду коду проміжного результату, отриманого на попередньому такті. Це призводить до перенесення у молодші розряди навіть тоді, коли в них немає від'ємного значення. Внаслідок такого перенесення хоча б один з розрядів коду нового проміжного результату матиме одиничне значення. Тому при виконанні віднімання в молодших по відношенню до даної групи розрядах позичання потрапляє у розряд з одиничним значенням і не розповсюджується далі у старші розряди.

Позначимо код зменшуваного через X, код від'ємника через Y, а код результату через Z. Розглянемо випадок віднімання додатних кодів Z=X-Y, при якому зменшуване не менше ніж від'ємник, тобто, X>=Y. Тоді результат віднімання теж буде додатнім. При побітовому відніманні, починаючи зі старших розрядів, перше ненульове значення результату буде додатнім. Але у подальших тактах віднімання може виникати від'ємне значення результату. Наприклад, при відніманні 10-розрядного коду Y=0000001010 від коду X=1000000000 на нульовому такті побітового віднімання у розряді коду результату з вагою α^9 утворюється одиничне значення. На 6-му і 8-му тактах результат віднімання окремих розрядів буде від'ємним. Ліквідація від'ємних значень у розрядах коду результату з вагою α^3 і α^1 здійснюється за допомогою операцій відповідних EAR-перетворень, що і є позичанням. Отже, процес побітового віднімання вказаних послідовних кодів вимагає побітового виконання послідовності EAR-перетворень. Оскільки EAR-перетворення з деякого розряду коду призводить до збільшення двох молодших розрядів цього коду, то при наявності одиниць у молодших розрядах коду зменшуваного X відбувається їх додавання до одиниць, що утворилися від EAR-перетворення.

Слід відзначити, що від'ємні значення у розрядах при відніманні можуть ліквідовуватись не тільки за рахунок EAR-перетворення зі старших розрядів, але також за рахунок EAL-перетворення з молодших розрядів. З прикладу видно, що хоча у цілому зменшуване не менше ніж від'ємник, але внаслідок надлишковості системи числення частина коду від'ємника з деякого k-го розряду по (n-1)-й розряд може мати значення більше ніж та ж частина коду зменшуваного:

$$\left\{\begin{array}{l} \sum\limits_{i=0}^{n-1} x_i \cdot \alpha^i \ge \sum\limits_{i=0}^{n-1} y_i \cdot \alpha^i; \\ \sum\limits_{i=k}^{n-1} x_i \cdot \alpha^i < \sum\limits_{i=k}^{n-1} y_i \cdot \alpha^i. \end{array}\right\} \qquad (2)$$

У цьому випадку на k-му такті побітового віднімання різниця буде від'ємною. Як відомо з [5], для чисел золотої пропорції виконується

$$\alpha^k > \sum_{i=0}^{k-2} \alpha^i ,$$

тому з (2) слідує, що

$$\sum_{i=k}^{n-1} x_i \cdot \alpha^i - \sum_{i=k}^{n-1} y_i \cdot \alpha^i \ge -1 \cdot \alpha^k . \qquad (3)$$

Тобто, на будь-якому такті k-му такті побітового віднімання меншого числа від більшого абсолютне значення від'ємної різниці не більше одиниці k-го розряду. При відніманні меншого за абсолютним значенням числа від більшого від'ємне значення різниці є тимчасовим і у наступних тактах різниця приймає додатне або нульове значення.

Аналогічно додаванню на кожному i-му такті побітового віднімання код різниці складається з двох частин: групи розрядів, через які можливе розповсюдження позичання на наступному такті, і групи розрядів, значення яких в подальшому не буде змінюватись. Частина розрядів, що не змінюватиме в подальшому свого значення являє собою черговий код різниці Z(i). Частина розрядів, значення яких може змінюватись на наступному такті, є так званим черговим проміжним результатом S(i). При побітовому відніманні кодів золотої пропорції виробляються перенесення як у старші, так і у молодші розряди. Проміжний результат, отриманий на кожному i-му такті побітового віднімання, матиме у своєму коді, крім (n-i-1)-го розряду, ще старші та молодші від нього розряди. Процес побітового віднімання кодів золотої пропорції полягає у тому, що на кожному такті відбувається віднімання чергового розряду коду від'ємника від чергового розряду коду зменшуваного і додавання до коду проміжного результату, отриманого на попередньому такті. На кожному такті формується черговий розряд коду різниці і черговий проміжний результат:

$$(x_{n-i-1} - y_{n-i-1}) \cdot \alpha^{n-i-1} + S(i-1) \rightarrow S(i) + z_{n-i+d} \cdot \alpha^{n-i+d}.$$

Визначимо довжину коду проміжного результату S. Як було показано раніше, довжина перенесення у молодші розряди не перевищує два розряди. Тому потрібно визначити максимальну довжину розповсюдження позичання зі старших розрядів d. Покажемо спочатку, що довжина розповсюдження позичання повинна бути більшою двох розрядів, тобто d>1. Для цього розглянемо приклад віднімання кодів 101000-000111 при d=1:

		5	4	3	2	1	0		
		1	0	1	0	0	0		
-		0	0	0	1	1	1		
0		1	0	0					
1		1	0	0	0				
2		1	0	1	0	0			
3		1	0	1	-1	0	0		
3		1	0	0	0	1	0		
4		1	0	0	0	0	0	0	
5		1	0	0	0	0	-1	0	0

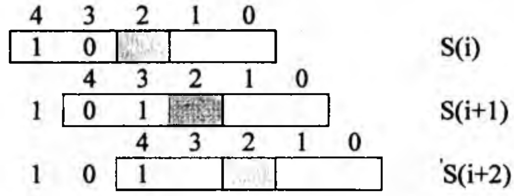
З прикладу видно, що на другому такті віднімання значення проміжного результату менше ніж α^3. Оскільки $\alpha^2+\alpha+1>\alpha^3$, то при подальшому надходженні послідовності нулів зменшуваного і послідовності одиниць від'ємника даного значення недостатньо, щоб компенсувати надходження від'ємних одиниць, хоча у цілому зменшуване більше від'ємника. Тому на п'ятому такті проміжний результат стає від'ємним. Вирішенням проблеми може бути EAR-перетворення на другому такті обох одиниць. Але, оскільки на цьому такті невідомо значення розрядів кодів операндів, що поступлять на третьому такті, то це може призвести до переповнення у інших випадках, наприклад, при відніманні кодів 101111-000000:

		1	0	1	1	1	1
-		0	0	0	0	0	0
0		1	0	0			
1		1	0	0	0		
2		0	1	1	1	1	
3			1	1	2	1	0

У даному прикладі на другому такті виконується EAR-перетворення обох одиниць коду проміжного результату, що призводить до нульового резерву для перенесення. Тому при подальшому надходженні послідовності одиниць зменшуваного і послідовності нулів від'ємника на третьому такті виникає переповнення проміжного результату. Отже, неможливо побітово виконувати віднімання кодів золотої пропорції при d=1. Це пов'язано з тим, що на деякому такті при надходженні i-х розрядів кодів операндів потрібно аналізувати умову EAR-перетворення одиниці з (i+3)-го розряду коду проміжного результату.

Покажемо, що при d=2 можна побітово виконувати віднімання кодів золотої пропорції. Враховуючи, що b=2, код проміжного результату матиме п'ять розрядів. Переповнення коду проміжного результату може виникнути лише при надходженні чергового одиничного розряду коду зменшуваного внаслідок додавання його до проміжного результату, отриманого на попередньому такті. Неможливість такого переповнення при розрядності коду проміжного результату п'ять показана при аналізі побітового додавання. Для аналізу можливості встановлення старших розрядів коду проміжного результату у від'ємне значення розглянемо окремо два випадки: перший – на деякому такті побітового віднімання старший розряд коду проміжного результату встановлюється в одиничне значення і другий – на жодному з тактів старший розряд коду проміжного результату не встановлюється в одиничне значення.

У першому випадку на початку деякого i-го такту старший четвертий розряд коду проміжного результату S(i) буде встановлено в одиничне значення. Розглянемо процес побітового віднімання у цьому випадку. Якщо на (i+1)-му такті виконується умова EAR-перетворення четвертого розряду коду проміжного результату S(i), то відбувається його EAR-перетворення у третій і четвертий розряди коду проміжного результату S(i+1). Внаслідок такого перетворення четвертий розряд коду проміжного результату на наступному такті знову встановлюється в одиничне значення. Якщо ж умова EAR-перетворення не виконується, то на наступному такті в результаті зсуву коду проміжного результату на один розряд вліво ця одиниця повинна бути "виштовхнута" з коду проміжного результату у черговий розряд коду різниці. Але "виштовхування" виконується тільки при невиконанні умови EAR-перетворення у молодші розряди з урахуванням чергових розрядів кодів зменшуваного і від'ємника, тобто, якщо у третьому чи четвертому розрядах коду проміжного результату S(i+1) на (i+1)-му такті встановлюється одиниця. Найбільш критичною є ситуація, коли на (i+1)-му такті третій розряд коду проміжного результату має одиничне значення. Проте, у даній ситуації на початку (i+2)-го такту побітового віднімання старший розряд початкового коду проміжного результату S(i+2) знову матиме одиничне значення:



Отже, якщо на початку деякого такту побітового віднімання старший розряд коду проміжного результату має одиничне значення, то за рахунок EAR-перетворення зі старшого розряду коду проміжного результату знову забезпечується одиничне значення старшого розряду коду цього результату на початку одного з наступних тактів. Це означає, що у даному випадку не буде встановлення у від'ємне значення старшого розряду коду проміжного результату.

У другому випадку не відбувається "виштовхування" одиниць з проміжного результату. Тобто, всі отримані розряди коду результату мають нульове значення. На кожному i-му такті побітового віднімання значення проміжного результату дорівнює різниці частин кодів зменшуваного і від'ємника з (n-i-1)-го по (n-1)-й розряди. Як слідує з (3), така різниця може бути від'ємною, але її абсолютне значення при цьому не більше одиниці (n-i-1)-го розряду, що відповідає другому розряду коду проміжного результату. Отже, і у даному випадку третій і четвертий розряди коду проміжного результату не матимуть від'ємних значень. Тому при розрядності коду проміжного результату п'ять побітове віднімання виконується правильно. Граф-схема алгоритму побітового віднімання подана на рис. 3.

Побітове алгебраїчне додавання

Побітове алгебраїчне додавання – це додавання послідовних кодів операндів з урахуванням їх знаків. Воно являє собою фактично комбінацію додавання і віднімання абсолютних значень операндів, що обираються в залежності від знаків цих операндів. Отже, пристрій алгебраїчного додавання повинен мати два режими виконання операції: додавання та віднімання. При побітовому алгебраїчному додаванні кодів чисел X та Y на деякому i-му такті встановлюється чергова i-та сума Z(i+d)=X(i+d)+Y(i+d) та проміжний результат S(i) розрядністю (d+y+1).

Оскільки знаки операндів не завжди відомі з приходом старших розрядів кодів цих операндів, то встановлення потрібної операції може відбуватись всередині формату числа. Але при цьому все ж можливе забезпечення коректного виконання операції через те, що знак кожного операнда завжди стає відомим безпосередньо перед надходженням першого значущого розряду коду даного операнда. До встановлення знаку операнда на вхід пристрою поступають тільки нульові значення цього операнда. Тому код результату буде однаковим незалежно від того, віднімаються чи додаються розряди з нульовим значенням. При надходженні знаку одного з операндів встановлюється відповідний знак проміжного результату. Для правильного встановлення операції потрібно враховувати знаки обох операндів. У загальному випадку ці знаки стають відомими на різних тактах. Тобто, може виникнути ситуація, в якій на деякому такті відомо знак тільки одного операнда. На наступних тактах на вхід пристрою поступають значущі розряди коду даного операнда. Якщо при цьому знак другого операнда ще невідомий, то можливе неправильне виконання алгебраїчного додавання. Наприклад, при побітовому додаванні 16-розрядних кодів чисел X=(+0100000000000000) і Y=(-0000000000001100) на другому такті на вхід пристрою поступає розряд коду першого операнда з одиничним значенням у той час, коли знак другого операнда може бути ще невідомим. Оскільки на другому такті відповідний розряд коду другого операнда має нульове значення, то відсутність інформації про знак цього операнда не впливає на значення результату. Але на деякому наступному i-му такті виконання операції відбувається "виштовхування" одиниці з коду проміжного результату S. У цьому випадку S(i) встановлюється у нульове значення. З приходом одиничного значення розряду коду від'ємного числа код проміжного результату стає від'ємним, що у подальшому призводить до неправильного формування коду результату Z. Для запобігання такій ситуації до тих пір, доки не будуть відомі знаки обох операндів, встановлюється примусове віднімання. Це потребує виконання EAR-перетворення старшого "виштовхуваного" розряду коду проміжного результату, що забезпечує ненульове значення даного коду на наступному такті алгебраїчного додавання.

При надходженні знаку другого операнда у залежності від знаків обох операндів встановлюється дійсна операція Оп. Нульове значення Оп означає додавання, а одиничне – віднімання. Якщо знаки операндів однакові, то встановлюється додавання, якщо різні – віднімання. Протягом виконання дійсної операції потрібно враховувати знаки операндів і знак проміжного результату. Оскільки кількість старших розрядів коду проміжного результату не менша трьох, то при побітовому алгебраїчному додаванні кодів золотої пропорції знак результату формується за декілька тактів до появи першого значущого розряду коду результату.

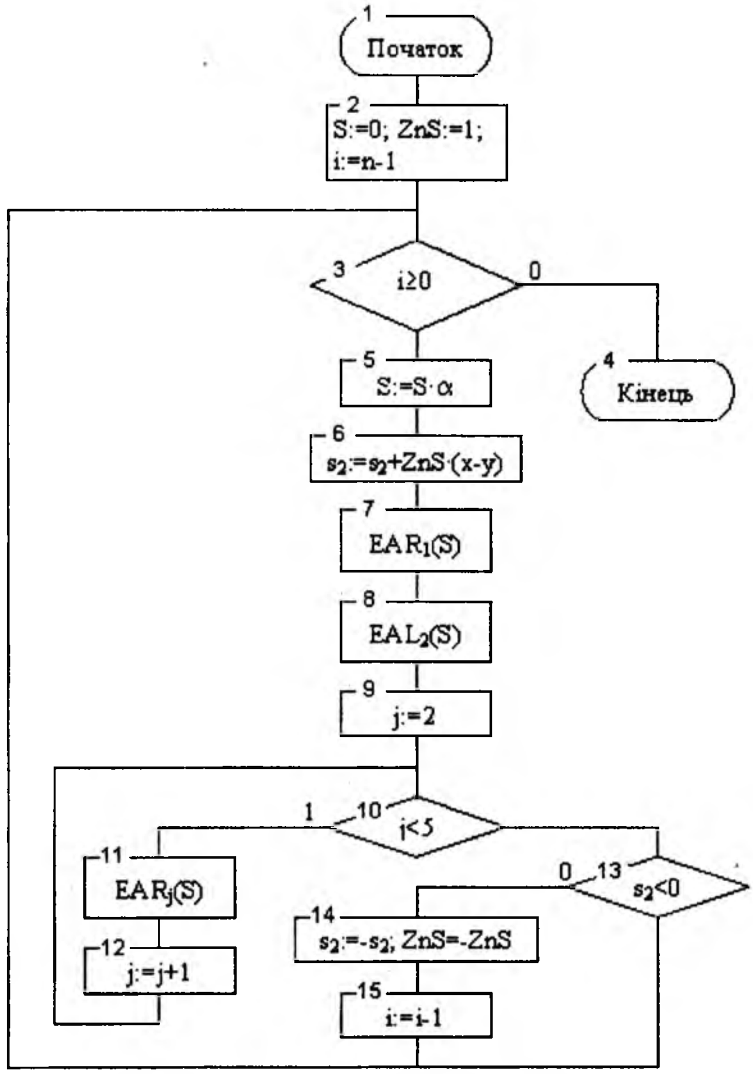
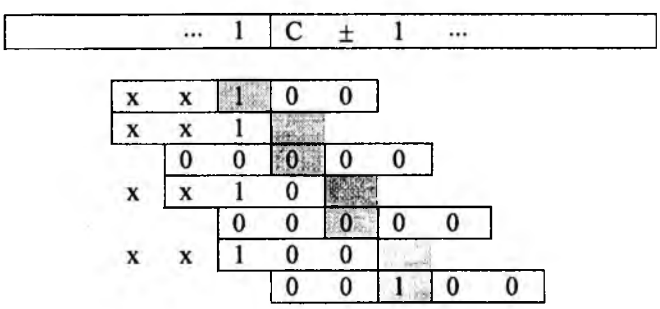


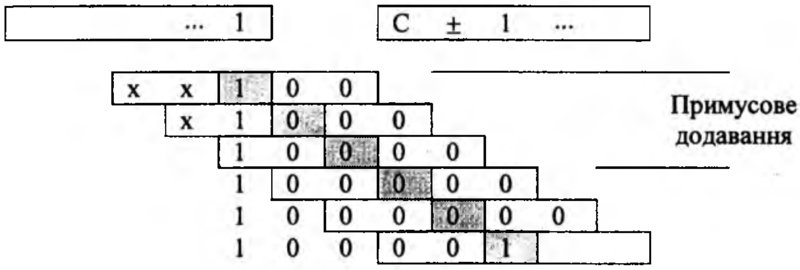
Рис. 3 - Граф-схема алгоритму побітового віднімання кодів золотої пропорції

Це дозволяє вставити знак у формат коду результату на місце розряду з нульовим значенням. Для ідентифікації знаку числа на приймальному кінці безпосередньо перед ним у формат числа вставляється одиниця строба знаку:

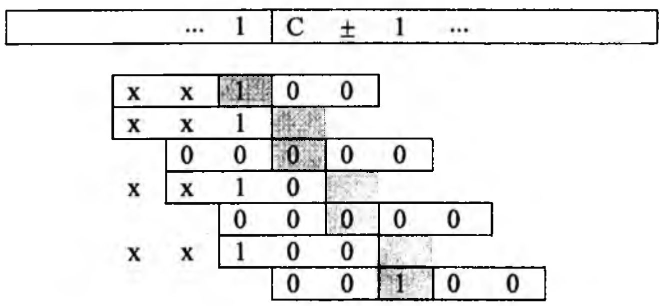


Якщо після надходження знаків обох операндів дійсною операцією встановлюється віднімання абсолютних значень кодів, то на наступному після приходу останніх розрядів кодів операндів такті потрібно виконувати додавання для заборони подальшого EAR-перетворення старшого розряду коду проміжного результату. Інакше це призведе до похибки в обчисленні молодшого розряду коду результату. Альтернативним варіантом може бути перезапис двох старших розрядів коду проміжного результату у три-розрядний регістр зсуву, вихід старшого розряду якого через логічну схему АБО разом з виходом чергового розряду коду суми формує остаточне значення цієї суми.

Розглянемо переваги і недоліки кожного варіанту. У першому варіанті примусове додавання не вимагає додаткового обладнання, але воно виконується протягом додаткових чотирьох тактів після закінчення подавання кодів чергових операндів. Під час примусового додавання не можна подавати на вхід пристрою коди наступних операндів тому, що для них потрібно встановлювати примусове віднімання. Тобто, між кодами операндів потрібно вставляти чотири додаткових розряди. Фактично, це призводить до збільшення на чотири розрядності кодів операндів. Два з них є стробом знаку і знаком. Оскільки істина операція може бути встановлена у одному такті з надходженням знаку, то між кодами достатньо вставляти тільки два розряди. Наприклад:



В альтернативному варіанті надходження сигналу початку нового числа перезаписує другий, третій і четвертий розряди коду проміжного результату у допоміжний регістр зсуву, після чого проміжний результат встановлюється у нульове значення. Розряди коду результату формуються за допомогою логічного АБО над старшим розрядом регістру зсуву і чергового розряду z_i. Даний варіант потребує додаткових три розряди для зберігання старшої частини коду проміжного результату у допоміжному регістрі зсуву, але він дозволяє не вставляти додаткові розряди між кодами операндів. Наприклад:



Таким чином, процес побітового алгебраїчного додавання двох послідовних кодів операндів складається з трьох частин:

1. Після приходу знаку одного з операндів і до приходу знаку другого операнда встановлюється примусове віднімання з метою розгортки старшого розряду коду проміжного результату;
2. Після приходу знаку другого операнда і до приходу останніх розрядів кодів операндів встановлюється дійсна операція;
3. Після приходу останніх розрядів кодів операндів на наступному такті встановлюється примусове додавання для заборони розгортки старшого розряду коду проміжного результату або відбувається перезапис двох старших розрядів коду проміжного результату у допоміжний дворозрядний регістр зсуву.

Розглянемо більш детально другу частину, в якій виконується дійсна операція. Дійсна операція виконується над абсолютними значеннями кодів. При цьому на кожному такті до коду проміжного результату додається розряд коду операнда, що має той самий знак і віднімається розряд коду операнда, що має протилежний знак. У випадку, коли дійсною є операція віднімання послідовних кодів золотої пропорції, знак результату визначається знаком більшого за абсолютним значенням операнда. Оскільки знаки операндів необхідно зберігати протягом всієї операції, то більш доцільним є зберігання не знаку проміжного результату, а ознаки Хб. Дана ознака вказує, який з операндів є більшим за абсолютним значенням. Одиничне значення Хб означає, що $|X|\geq|Y|$, а нульове – що $|X|\leq|Y|$. Разом із знаками операндів ця ознака визначає знак проміжної суми, а також дійсну операцію. Через надлишковість системи числення можлива ситуація, в якій у процесі надходження розрядів кодів чисел спочатку за абсолютним значенням більшим буде один операнд, а потім – інший. У цій ситуації побітове алгебраїчне додавання призводить до зміни знаку проміжного результату, що може відбуватись декілька разів у процесі виконання операції. Тому необхідно на кожному такті запам'ятовувати ознаку Хб. Слід відзначити, що зміні значення Хб завжди передує нульове значення проміжного результату. Якщо у процесі віднімання на деякому такті встановлюється в одиничне значення третій розряд коду проміжного результату, то ніяка комбінація наступних розрядів кодів операндів не буде більшою за модулем від цієї одиниці. Це означає, що у подальшому знак проміжного результату не буде змінюватись. Тому у даному випадку на основі знаків операндів та ознаки Хб формується дійсний знак результату, що подається на наступний побітовий пристрій. Граф-схема алгоритму виконання дійсної операції подана на рис. 4.

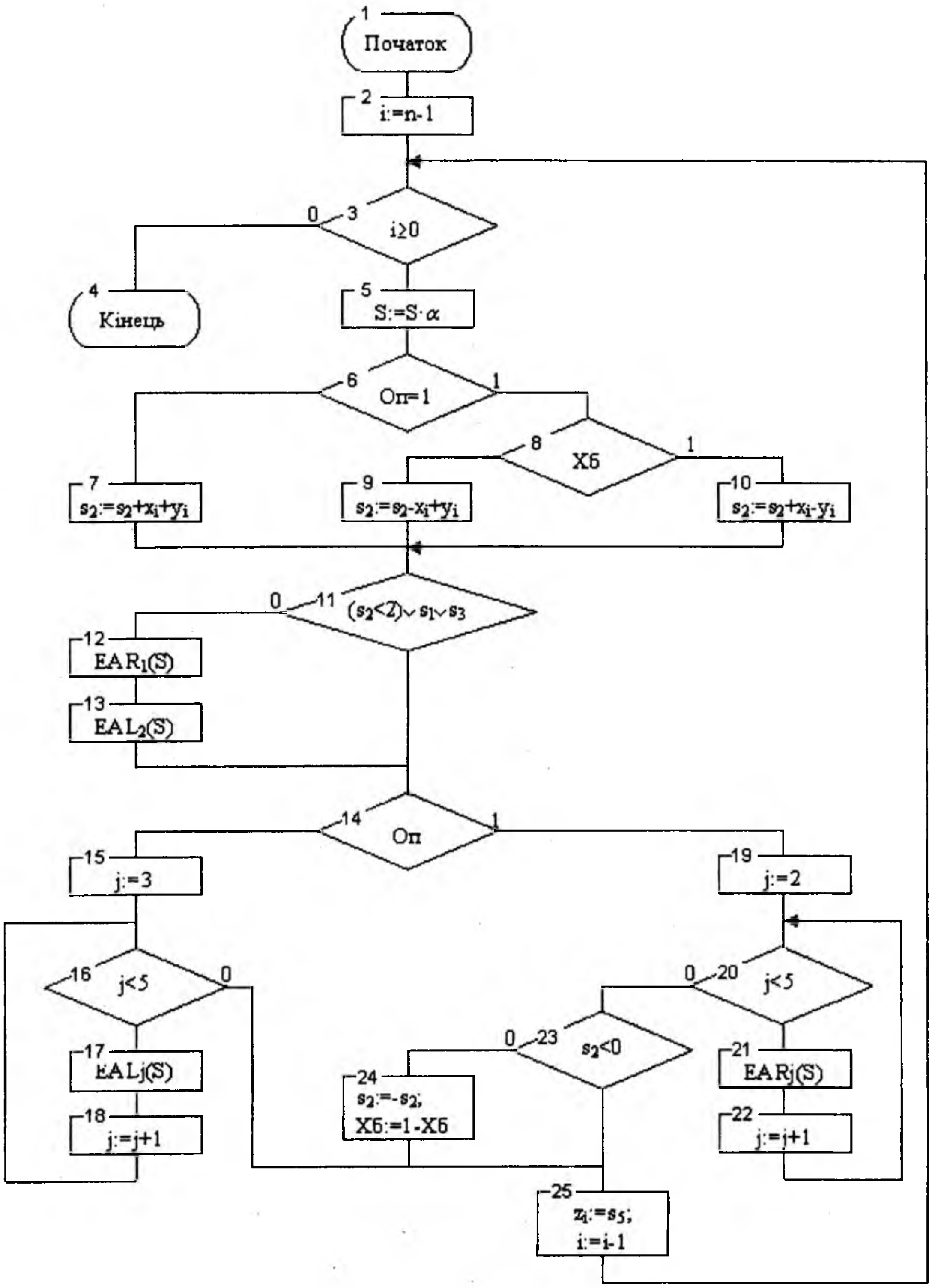


Рис. 4 - Граф-схема алгоритму виконання дійсної операції при побітовому алгебраїчному додаванні кодів золотої пропорції

Побітове множення

Побітове множення послідовних кодів золотої пропорції виконується, починаючи зі старших розрядів. Код добутку формується також зі старших розрядів. Особливістю запропонованого підходу є невелика фіксована затримка потоку кодів добутків відносно потоку кодів співмножників. В основі алгоритму виконання такого множення покладено спосіб отримання часткових добутків, починаючи зі старших розрядів, у вигляді декількох пар послідовних кодів. Кожна з пар кодів утворюється із затримкою на один такт відносно попередньої. Це дозволяє побітово додавати відповідну пару кодів до послідовного коду результату, отриманого при додаванні попередньої пари кодів. Для отримання послідовного коду суми чергової пари кодів і результату додавання попередньої пари кодів використовується три-входовий побітовий суматор послідовних кодів золотої пропорції.

Розглянемо процес утворення часткових добутків при виконанні побітового множення послідовних кодів зі старших розрядів. Визначимо добуток кодів двох чисел Z=X×Y у вигляді суми часткових добутків

$$Z=\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} x_i\cdot y_j\cdot\alpha^{i+j},$$

де x_i – значення i-го розряду коду співмножника X; y_j – значення j-го розряду коду співмножника Y; i,j=0,1,2...– номери розрядів; n – розрядність. Будемо вважати, що обидва співмножники мають значення менші одиниці. Тоді послідовність утворення часткових добутків x_iy_i при множенні відповідних розрядів співмножників можна представити у вигляді:

	x_{n-1}	x_{n-2}	...	x_1	x_0				
×	y_{n-1}	y_{n-2}	...	y_1	y_0				
					$x_{n-1}y_0$	$x_{n-2}y_0$	...	x_1y_0	x_0y_0
				$x_{n-1}y_1$	$x_{n-2}y_1$	...	x_1y_1	x_0y_1	
			...	...	...	...	...		
		$x_{n-1}y_{n-2}$	$x_{n-2}y_{n-2}$	...	x_1y_{n-2}	x_0y_{n-2}			
	$x_{n-1}y_{n-1}$	$x_{n-2}y_{n-1}$	...	x_1y_{n-1}	x_0y_{n-1}				

Код добутку утворюється при додаванні в усіх стовпчиках, отриманих часткових добутків розрядів. Коди співмножників надходять послідовно, починаючи зі старших розрядів. На першому такті при надходженні (n-1)-х розрядів кодів співмножників утворюється частковий добуток $x_{n-1}y_{n-1}$. На другому такті при надходженні (n-2)-х розрядів кодів співмножників утворюються часткові такі добутки:

$x_{n-1}y_{n-2}$	$x_{n-2}y_{n-2}$
$x_{n-2}y_{n-1}$	

При надходженні (n-3)-х розрядів кодів співмножників утворюються такі часткові добутки:

$x_{n-1}y_{n-3}$	$x_{n-2}y_{n-3}$	$x_{n-3}y_{n-3}$
	$x_{n-3}y_{n-2}$	
$x_{n-3}y_{n-1}$		

При надходженні нульових розрядів кодів співмножників утворюються такі часткові добутки:

$x_{n-1}y_0$	$x_{n-2}y_0$	...	x_1y_0	x_0y_0
			x_0y_1	
		...		
	x_0y_{n-2}			
x_0y_{n-1}				

Отже, добуток С можна представити у вигляді виразу:

$$Z = \sum_{i=1}^{n}\left(\left(\sum_{j=1}^{i}(x_{n-i}\cdot y_{n-j})+\sum_{j=1}^{i-1}(x_{n-j}\cdot y_{n-i})\right)\cdot\alpha^{n-i-j}\right).$$

На першому такті виконується множення над найстаршими розрядами множників. На останньому такті – над наймолодшими. Нижче наведена послідовність формування всіх часткових добутків при побітовому множенні n-розрядних кодів, починаючи зі старших розрядів:

	x_{n-1}	x_{n-2}	...	x_{n-i}	...	x_1	x_0
y_{n-1}	1	2		i		n-1	n
y_{n-2}	2	2		i		n-1	n
...				i		n-1	n
y_{n-i}	i	i	i	i		n-1	n
...						n-1	n
y_1	n-1	n-1	n-1	n-1	n-1	n-1	n
y_0	n	n	n	n	n	n	n

Тут: 1 - відповідні часткові добутки при надходженні перших розрядів кодів співмножників; 2 - відповідні часткові добутки при надходженні других розрядів кодів співмножників; і - відповідні часткові добутки при надходженні і-х розрядів кодів співмножників.

Тому добуток С можна представити у вигляді виразу:

$$Z = \sum_{i=1}^{n}\left(\sum_{j=i}^{n}\left(x_{n-i}\cdot y_{n-j}\cdot\alpha^{n-i-j}\right)+\sum_{j=i+1}^{n}\left(x_{n-j}\cdot y_{n-i}\cdot\alpha^{n-i-j}\right)\right).$$

Оскільки код результату утворюється, починаючи зі старших розрядів, то затримка потоку кодів добутків відносно потоку кодів співмножників складає декілька розрядів і не залежить від розрядності.

Побітове ділення

Розглянемо ділення послідовних кодів золотої пропорції, що надходять, починаючи зі старших розрядів. Нехай потрібно виконати ділення Z=X/Y. Коди X та Y приведені до нормалізованого виду:

$$X=0.x_{n-1}x_{n-2}x_{n-3}\ldots x_0;$$

$$Y=0.y_{n-1}y_{n-2}y_{n-3}\ldots y_0.$$

Тому код частки буде меншим від одиниці:

$$Z=0.z_{n-1}z_{n-2}x_{n-3}\ldots z_0.$$

Коди діленого та дільника надходять зі старших розрядів тобто, починаючи з x_{n-1} та y_{n-1}. Для визначення n розрядів коду частки Z потрібно послідовно виконати n операцій. У кожній з цих операцій визначається черговий розряд частки за допомогою конвеєрного побітовою віднімання зсунутого на відповідну кількість розрядів коду дільника від коду черговою залишку. та подальшого зсуву коду дільника праворуч на один розряд за допомогою затримки його на один такт. Черговий залишок являє собою попередній залишок, або результат віднімання, якщо цей результат є більшим від нуля. В залежності від цього черговий розряд коду частки встановлюється відповідно рівним одиниці або нулю.

Основною операцією при діленні є віднімання. З алгоритму віднімання відомо, що знак результату встановлюється як мінімум за два такти до одержання на виході віднімача одиниці. При цьому, якщо числа, що надходять на входи віднімача, відрізняються тільки молодшими розрядами, знак результату може встановлюватися наприкінці формату чисел. Якщо два числа відрізняються в старших розрядах, то знак результату встановлюється на другому такті після початку віднімання. Якщо на першому віднімачі утворився додатній знак, у старшому розряді коду частки буде одиниця і на другий віднімач надходить код залишку та код діленого Y, затриманий на один такт. Якщо на першому віднімачі утворився від'ємний знак, у старшому розряді коду частки буде нуль і на другий віднімач надходить код вихідного числа Z та, затриманий на один такт, код діленого Y. Для підвищення швидкодії пристрою ділення послідовних кодів золотої пропорції, знак на виході віднімача необхідно визначити через три такти після надходження перших одиниць на його входи.

Нехай X>Y. Розглянемо три можливих випадки, що виникають при діленні послідовних кодів золотої пропорції.

Якщо X значно більше Y, то на другому такті після прибуття першої значимої цифри діленого на вхід віднімача, на його виході встановлюється додатній знак, і значення розряду коду частки дорівнює одиниці. Послідовний код залишку надходить на вхід наступного віднімача, на другий вхід якого надходить затримане на один такт (в α разів менше) значення коду дільника.

Якщо X приблизно дорівнює Y, то істинне значення знака на виході віднімача встановлюється не відразу. На виході знака суми буде присутній від'ємний знак. На третьому такті після прибуття першої одиниці на вхід діленого або дільника буде зафіксоване від'ємне значення знака, що відповідає нульовому значенню розряду коду частки. На вхід наступного віднімача надходить послідовний код діленого, на другий вхід якого надходить затримане на один такт значення коду дільника. При цьому вже $X_1>Y_1$, і відбувається визначення значення наступного розряду коду частки.

Якщо X трохи більше Y, тобто відрізняються вони в середніх або молодших розрядах послідовних кодів, то у першому розряді коду частки буде зафіксований нуль, а в двох наступних молодших розрядах коду частки - одиниці. Результат 100 у коді золотій пропорції ідентичний 011, тому що це і є дві форми представлення того самого числа.

З наведеного опису виконання ділення видно, що всі розряди коду частки утворюються після закінчення віднімання на послідовних віднімачах. Робота віднімачів організується у конвеєрному (потоковому) режимі. При цьому затримка одержання останнього розряду коду частки дорівнює 3n тактів.

Всі розряди частки утворюються окремо шляхом аналізу знаків на віднімачах. На першому віднімачі утворюється старший розряд частки. На наступних віднімачах – молодші розряди. Це дозволяє формувати послідовний код результату, починаючи зі старших розрядів, шляхом запису отриманих розрядів частки у регістр зсуву. Таке формування коду частки дозволяє зменшити апаратні витрати при побудові пристроїв побітового ділення кодів золотої пропорції.

Висновки

У даній статті описані результати, отримані у процесі розробки алгоритмів побітового виконання додавання, віднімання, множення та ділення послідовних кодів золотої пропорції.

1. Вперше описано алгоритми побітового додавання, віднімання та алгебраїчного додавання кодів золотої пропорції на основі елементарних адитивних перетворень. При цьому досягається найменша довжина перенесення і позичання, що дозволяє зменшити апаратні витрати при побудові пристроїв.

2. Вперше описано алгоритмічні основи виконання побітового множення на основі побітового додавання кодів золотої пропорції. Відмінністю такого множення є незалежна від розрядності стала затримка потоку результату, що дозволяє підвищити його продуктивність.

3. Вперше описано алгоритмічні основи побітового ділення кодів золотої пропорції що дозволяють зменшити апаратні витрати при побудові пристроїв побітового ділення кодів золотої пропорції за рахунок формування коду частки у регістрі зсуву.

Список літератури

1. Avizenis A. Binary-compatible signet-digit arithmetic. IN: AFIPS Conf Proc. – Vol. 26 – P1. – 1964 – P.663.
2. Самофалов К.Г., Луцкий Г.М. Основы построения конвейерных ЭВМ.- Киев: Вища школа, 1981. – 234 с.
3. Каляев А.В. Многопроцессорные системы с программируемой архитектурой. – М.: Радио и связь, 1984. – 240 с.
4. Ch. Frougny, On-line finite automata for addition in some numeration systems. Theoretical Informatics and Applications 33 (1999), 79–101.
5. Стахов А.П. Коды золотой пропорции. – М.: Радио и связь, 1983. – 152с., ил. – (Кибернетика).
6. Системи числення з адитивними та мультиплікативними співвідношеннями між вагами розрядів / О.Д.Азаров, О.І.Черняк, П.О.Черняк // Вісник ВПІ. – 2001. - №1. – С. 58-64.
7. Методи конвеєрної порозрядної обробки послідовних кодів золотої пропорції / О.І. Черняк, О.Д. Азаров // Вісник ВПІ. – 1996. - №1. – С. 14-17.

Азаров Олексій Дмитрович, д.т.н., проф., директор інституту інформаційних технологій та комп'ютерної інженерії, зав. кафедри обчислювальної техніки Вінницького національного технічного університету, 21021, м. Вінниця, вул. Порика, 16, кв. 13. т.(0432)43-90-02 дом. 43-75-07, azarov@lili.vstu.vinnica.ua.

Черняк Олександр Іванович, ст. викладач кафедри обчислювальної техніки Вінницького національного технічного університету, 21021, м. Вінниця, вул. Келецька, 94, кв. 40, т.(0432)43-90-02, дом. 46-37-16, chernyak@vstu.vinnica.ua.