УДК 681.3.042

И.Н. Федотова-Пивень

Черкасский государственный технологический университет, Черкассы

О МНОГООПЕРАНДНОМ СЛОЖЕНИИ С ОГРАНИЧЕННЫМ РАСПРОСТРАНЕНИЕМ ЕДИНИЦ ПЕРЕНОСОВ

В статье приведен краткий сравнительный анализ двух методов многооперандного сложения: метода Уоллеса и метода сложения в линейной избыточной рекуррентной системе счисления третьего порядка. Показано преимущество второго метода благодаря диагональной записи суммы каждого вертикального разрядного среза и применения набора рекуррентных правил сложения. На основе программной модели и экстраполяции расчетных значений установлено многократное возрастание быстродействия и уменьшение количества промежуточных значений второго метода с ростом числа операндов до 300 по сравнению с методом Уоллеса.

Ключевые слова: *многооперандное сложение, рекуррентная система счисления, избыточность, вертикальные разрядные срезы, одновременное сложение.*

Вступление

Постановка проблемы. Широкий класс традиционных вычислительных задач, интенсивное развитие криптографии требуют обработки чисел все большей и большей разрядности. Это стимулирует создание новых вычислительных устройств с повышенным быстродействием обработки массивов больших чисел. При работе с большими числами разрядность таких чисел служит тормозящим моментом даже при выполнении простых операций – сложения и вычитания [1]. Это обусловлено необходимостью учитывать единицы переноса (ЕП) в старшие разряды при каждом сложении, которые в принципе могут двигаться по всей разрядной сетке [1]. Традиционное сложение 2-х N-разрядных чисел в общем случае может требовать порядка N битовых ЕП [1]. Арифметические сумматоры служат основными архитектурными блоками для вычисления всех остальных арифметических операций, увеличение их быстродействия может дать значительный эффект ускорения для общего процесса вычислений [1]. Таким образом, многооперандное сложение (МС) чисел большой разрядности является актуальным направлением исследований.

Анализ предыдущих исследований и публикаций. МС может выполняться либо поочередно по операндам и одновременно по их разрядам, либо одновременно по операндам и поочередно по их разрядам [2]. МС по алгоритму Уоллеса (АУ) выполняется поочередно по группам из нескольких операндов и одновременно по их разрядам и является теоретически самым быстрым, но структура этого алгоритма наиболее трудная для практической имплементации [3]. К тому же соединения различной длины и сложность путей сигналов приводят к логическим опасностям и искажению сигналов в синхронных конструкциях, что будет иметь негативные последствия для параметров мощности и производительности [3].

Сложение в линейной избыточной рекуррентной системе счисления (ЛИРССЧ) 3-го порядка [4], образованной на основе рекуррентного соотношения вида

$$B_n=B_{n-1}+3B_{n-3}+2B_{n-4} \qquad (1)$$

с алфавитом {0, 1} и начальными значениями 1 1 1 1 2 4 8 (B0 = 1, B1 = 1, B2 = 1, B3 = 1, B4 = 2, B5 = 4, B6 =8) происходит одновременно по всем операндам и поочередно по их разрядам. При этом находится сумма единиц в разрядах с одинаковым весом массива двоичных чисел, т.е. сумма единиц вертикального разрядного среза (ВРС). Такая вертикальная организация вычислений имеет преимущество по сравнению с горизонтальной организацией с увеличением количества операндов [5]. Блоки нахождения суммы единиц есть в процессорах суперЭВМ [6].

Однако сравнительного анализа МС методом Уоллеса и методом применения правил сложения в ЛИРССЧ 3-го порядка (1) для различного числа слагаемых, а также получения конкретных расчетных и прогнозных значений, достаточных для выбора метода для эффективного практического внедрения МС проведено не было.

Целью данного исследования является проведение сравнительного анализа МС методом Уоллеса и методом применения правил сложения в ЛИРССЧ (1) для различного числа слагаемых, а также получение конкретных расчетных и прогнозных значений для выбора метода для эффективного практического внедрения МС.

Изложение основного материала

Вычисления в серии из k сложений можно организовать так, чтобы ЕП учитывались только один раз при заключительном сложении [1]. Общая схема сложения такая: многорядный исходный код слагаемых преобразуется в многорядный промежуточный код, далее промежуточный код преобразуется в 2-рядный, а 2-рядный код преобразуется в однорядный код результата. ЕП при каждом сложении не добавляются, а запоминаются в дополнительном регистре [1]. Такое сохранение ЕП в дополнительном регистре предопределяет невозникновение и нераспространение цепочек из ЕП вдоль разрядной сетки на этапе промежуточных вычислений. Цепочки ЕП не распространяются, а накладываются друг на друга [1], т.е. на этом этапе ЕП запоминаются, а не вычисляются. Метод преобразования исходного многорядного кода слагаемых в 2-рядный код определяет количество сохранений ЕП в промежуточном коде.

Рассмотрим методы преобразования исходного многорядного кода слагаемых в 2-рядный код с ограничением распространения цепочек ЕП:

1) метод Уоллеса (программная модель приведена на рис. 1);

2) метод применения правил сложения в ЛИРССЧ (1) (программная модель приведена на рис. 2).

2097152	1048576	524288	262144	131072	65536	32768	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1		
				0	1	0	1	0	1	0	1	1	0	1	1	1	0	0	1	0	1		87781
				0	1	0	0	1	0	1	0	1	1	0	1	0	0	1	1	0	0		76620
				0	1	1	1	0	0	1	0	0	0	1	1	1	0	0	1	1	1		116967
				0	1	0	0	1	0	1	0	0	1	0	0	0	1	1	0	1	0		76058
				0	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1	1		3599
				0	0	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0	0		4976
				0	0	1	1	1	1	1	0	1	1	0	1	1	1	1	1	0	1		64381
				0	0	0	0	0	1	1	1	0	1	0	1	1	0	0	0	1	1		7523
				0	0	1	0	1	0	0	0	1	1	0	1	0	1	1	1	1	1		41823
				0	0	0	0	1	0	1	0	1	0	0	1	1	1	0	0	1	1		10867
				0	1	1	0	1	1	0	1	0	1	0	1	1	1	0	1	1	1		111991
				0	0	1	0	0	1	1	1	1	0	1	0	0	1	0	1	1	0		40598
				0	0	0	0	1	0	1	0	0	1	1	0	0	1	0	0	0	1		10641
				0	0	0	0	1	0	1	1	0	1	0	0	1	1	1	1	0	1		11581
				0	0	0	1	0	0	1	0	0	1	1	0	1	1	1	1	0	1		18877
				0	1	1	1	0	0	1	1	0	1	0	0	1	0	1	1	1	0		118062
				0	1	1	0	1	0	0	0	0	0	0	1	1	1	1	0	1	0		106618
				0	1	0	1	0	1	1	1	0	1	0	0	1	1	1	0	1	0		89402
				0	0	1	1	0	0	1	0	0	1	1	0	0	1	0	0	1	0		51602
				0	1	1	0	1	1	1	0	0	0	1	1	1	1	0	0	0	1		112881
				0	1	1	0	1	1	0	1	0	1	0	1	0	0	1	1	1	0	s1	1162848
			0	1	0	1	0	0	1	0	1	0	1	1	1	0	0	1	0	1	0	c1	
				0	1	0	0	1	1	0	1	0	0	0	1	1	0	0	1	0	1	s2	
			0	0	0	0	0	0	0	1	0	1	1	0	0	0	1	1	0	1	0	c2	
				0	0	0	1	0	0	0	1	0	1	0	1	0	0	0	0	0	1	s3	
			0	0	1	0	1	1	1	0	1	1	0	1	1	1	1	1	1	1	0	c3	
			0	0	1	0	0	0	0	0	0	0	1	1	0	0	1	0	0	1	0	s4	
			0	0	1	0	1	1	1	1	1	0	0	1	1	1	0	1	1	1	0	c4	
			0	0	0	0	1	0	0	1	1	0	1	0	0	0	1	0	0	0	1	s5	
			0	0	0	0	1	0	1	0	0	1	1	0	1	1	1	1	0	1	0	c5	
			0	0	1	0	0	1	1	0	0	0	0	0	1	1	0	1	1	1	0	s6	
			0	1	1	1	0	0	1	1	0	1	0	0	1	1	1	0	1	0	0	c6	
		0	0	1	0	0	0	0	1	0	1	0	0	1	1	1	0	0	0	0	1	s7	
		0	0	1	1	0	1	1	0	1	0	1	0	1	0	0	1	1	1	0	0	c7	
		0	0	0	1	0	0	1	0	0	1	0	1	1	0	0	0	1	0	1	1	s8	
		0	0	0	0	1	0	1	0	1	1	0	0	1	1	1	0	1	0	0	0	c8	
		0	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	1	1	0	1	s9	
		0	0	1	0	1	0	0	1	1	0	1	1	0	0	1	0	0	1	0	0	c9	
		0	0	1	0	1	1	1	1	1	0	0	1	0	1	1	0	0	0	0	0	s10	
		0	0	1	0	0	0	1	0	0	1	0	0	1	1	1	1	1	1	0	0	c10	
		0	0	0	0	0	1	0	1	1	0	1	1	1	1	1	1	0	1	1	0	s11	
	0	0	1	1	0	0	1	0	0	1	0	0	1	0	0	0	1	0	0	1	0	c11	
		0	0	1	0	0	0	0	0	0	1	1	1	1	0	1	0	0	0	0	1	s12	
	0	0	0	0	1	0	1	1	1	0	0	0	0	1	1	0	1	1	0	0	0	c12	
		0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	1	1	1	0	s13	
	0	0	1	0	1	1	1	0	1	0	0	1	1	1	1	1	0	0	0	0	0	c13	
	0	0	1	0	0	0	0	0	1	0	1	0	1	0	1	0	0	0	1	0	1	s14	
0	0	0	1	0	0	1	0	0	1	0	1	1	1	0	1	1	0	0	1	0	0	c14	
	0	0	1	0	0	1	0	1	1	0	1	1	1	0	0	1	1	0	1	1	0	s15	
0	0	0	0	1	0	1	0	1	0	0	0	0	1	1	0	0	1	0	0	0	0	c15	
0	0	0	1	0	0	0	0	1	1	0	1	0	1	0	0	0	1	0	1	1	1	s16	
0	0	1	0	0	1	0	0	1	0	1	1	1	0	1	1	0	0	1	0	0	0	c16	
0	0	1	1	1	1	1	0	1	1	1	0	1	0	0	1	0	0	1	1	1	1	s17	
0	0	0	0	0	0	0	1	0	0	1	0	1	1	0	0	1	0	0	0	0	0	c17	
0	0	1	1	1	0	0	1	0	0	1	0	0	1	1	0	0	1	1	1	1	0	s18	
0	0	0	0	1	1	0	1	1	1	0	1	0	0	1	1	0	0	0	0	1	0	c18	
0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1	0		
0	0	1	1	1	1	0	1	1	1	1	1	0	1	1	1	0	1	1	1	1	0		
30	31	0	0	0	0	36	0	0	0	0	0	42	0	0	0	46	0	0	0	0	51		
0	0	0	0	34	0	0	37	0	0	0	0	0	0	44	0	0	0	0	0	50	0		
0	1	0	0	0	1	1	0	1	1	1	1	1	0	0	1	1	0	0	0	0	0	=	1162848

Рис. 1. Сложение 20-ти целых положительных чисел в обычной ДССЧ по АУ в программной модели

Обе программные модели сложения методом Уоллеса и методом применения правил сложения в ЛИРССЧ (1) созданы с помощью встроенного в Excel 2003 языка программирования VBA.

МС по АУ реализуется при помощи цифровых 3:2 компрессоров (Carry Save Adders), которые складываются в деревья, и сумматора с распространением переносов (CPA-adder) [7]. Минимизация числа цифровых компрессоров и связей между ними путем применения методов оптимизации (теории графов) [8, 9] повышает быстродействие многооперандного сумматора.

На рис. 1 приведена программная модель сложения 20-ти целых положительных чисел в обычной двоичной системе счисления (ДССЧ) по АУ. В ней промежуточный код при сложении нескольких слагаемых каждый раз разделяется на код условной суммы и код переносов. При моделировании работы сумматора с распространением переноса (СРП) применена формула

$$B_n = \sum_{m=k}^{n-1} B_m + B_k \,, \qquad (2)$$

где $k \geq 0;\ k \leq m \leq n-1;\ n \geq 20;\ B_k \neq 0$.

Результаты моделирования работы СРП приведены в последних 3-х строках рис. 1 и рис. 2.

						n=16																			
2097152	1048576	524288	262144	131072	65536	32768	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	1	1	1	вес разряда
					1	0	1	0	1	0	1	1	0	1	1	1	0	0	1	0	0	1	0	0	87781
					1	0	0	1	0	1	0	1	1	0	1	0	0	1	0	0	1	1	1	1	76620
					1	1	1	0	0	1	0	0	0	1	1	1	0	0	1	1	0	1	0	0	116967
					1	0	0	1	0	1	0	0	1	0	0	0	1	1	0	0	1	0	0	1	76058
					0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	1	1	3599
					0	0	0	0	1	0	0	1	1	0	1	1	0	1	1	1	1	0	1	0	4976
					0	1	1	1	1	1	0	1	1	0	1	1	1	1	1	0	0	0	1	0	64381
					0	0	0	0	1	1	1	0	1	0	1	1	0	0	0	1	0	1	0	0	7523
					0	1	0	1	0	0	0	1	1	0	1	0	1	1	1	0	1	1	1	0	41823
					0	0	0	1	0	1	0	1	0	0	1	1	1	0	0	0	0	1	1	1	10867
					1	1	0	1	1	0	1	0	1	0	1	1	1	0	1	1	0	0	0	1	111991
					0	1	0	0	1	1	1	1	0	1	0	0	1	0	1	0	1	0	1	0	40598
					0	0	0	1	0	1	0	0	1	1	0	0	0	1	1	1	1	1	1	0	10641
					0	0	0	1	0	1	1	0	1	0	0	1	1	1	1	0	0	0	0	1	11581
					0	0	1	0	0	1	0	0	1	1	0	1	1	1	1	0	0	0	0	1	18877
					1	1	1	0	0	1	1	0	1	0	0	1	0	1	1	0	0	1	1	0	118062
					1	1	0	1	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	0	106618
					1	0	1	0	1	1	1	0	1	0	0	1	1	1	0	0	1	0	1	0	89402
					0	1	1	0	0	1	0	0	1	1	0	0	1	0	0	0	0	0	1	1	51602
					1	1	0	1	1	1	0	0	0	1	1	1	1	0	0	0	1	0	0	0	112881
					1	1	1	0	0	1	0	0	1	1	1	1	1	1	1	1	1				1162848
				0	0	1	1	0	1	0	0	0	1	1	0	0	0	0	0	0	1				
			0	0	1	0	0	1	0	0	1	1	0	1	1	1	1	1	0	0	0				
		1	1	0	1	1	1	1	1	1	0	1	1	1	1	1	0	1	0	0	0				
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
0	0	1	1	0	1	1	1	0	0	0	1	0	1	0	1	1	0	1	1	1	0				
0	0	0	0	1	1	1	1	1	1	0	1	1	0	1	1	1	1	0	0	1	0				
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0				
0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0				
0	0	1	1	1	1	1	0	1	1	1	1	0	1	1	1	0	1	1	1	0	0				
2	3	0	0	0	0	0	9	0	0	0	0	14	0	0	0	18	0	0	0	22	23				
0	0	0	0	6	0	0	0	0	0	0	0	0	0	16	0	0	0	0	21	0	0				
0	1	0	0	0	1	1	0	1	1	1	1	1	0	0	1	1	0	0	0	0	0				= 1162848

Рис. 2. Одновременное сложение 20-ти целых положительных чисел в ЛИРССЧ (1) в программной модели. Во 2-й строке сверху указан вес разряда

В табл. 1 представлена система правил (СП) одновременного сложения 50-ти целых положительных чисел в ЛИРССЧ (1). Эта ЛИРССЧ обладает простой СП одновременного сложения целых положительных чисел [3] (табл. 1).

Таблица 1

Система правил одновременного сложения 50-ти целых положительных чисел в ЛИРССЧ (1) (n – номер разряда, B_n – вес разряда).

№ п/п	Правило	№ п/п	Правило	№ п/п	Правило
1	2	3	4	5	6
1.	0=0+0	**18.**	$17B_n= B_n+B_{n+4}$	**35.**	$34B_n= B_{n+1}+B_{n+5}$
2.	$B_n=B_n+0$	**19.**	$18B_n= B_{n+1}+B_{n+4}$	**36.**	$35B_n= B_n+B_{n+1}+B_{n+5}$
3.	$2B_n=B_{n+1}$	**20.**	$19B_n= B_n+ B_{n+1}+B_{n+4}$	**37.**	$36B_n= B_{n+2}+B_{n+5}$
4.	$3B_n= B_n+B_{n+1}$	**21.**	$20B_n= B_{n+2}+B_{n+4}$	**38.**	$37B_n= B_n+B_{n+2}+B_{n+5}$
5.	$4B_n=B_{n+2}$	**22.**	$21B_n= B_n+ B_{n+2}+B_{n+4}$	**39.**	$38B_n= B_{n+1}+B_{n+2}+B_{n+5}$
6.	$5B_n= B_n+B_{n+2}$	**23.**	$22B_n= B_{n+1}+B_{n+2}+B_{n+4}$	**40.**	$39B_n=B_n+B_{n+1}+B_{n+2}+B_{n+5}$
7.	$6B_n= B_{n+1}+B_{n+2}$	**24.**	$23B_n= B_n+B_{n+1}+B_{n+2}+B_{n+4}$	**41.**	$40B_n= B_{n+3}+B_{n+5}$
8.	$7B_n= B_n+B_{n+1}+B_{n+2}$	**25.**	$24B_n= B_{n+3}+B_{n+4}$	**42.**	$41B_n= B_n+ B_{n+3}+B_{n+5}$

Окончание табл. 1

1	2	3	4	5	6
9.	$8B_n=B_{n+3}$	**26.**	$25B_n= B_n+ B_{n+3}+B_{n+4}$	**43.**	$42B_n= B_{n+1}+ B_{n+3}+B_{n+5}$
10.	$9B_n= B_n+B_{n+3}$	**27.**	$26B_n=B_{n+1}+B_{n+3}+B_{n+4}$	**44.**	$43B_n=B_n+B_{n+1}+B_{n+3}+B_{n+5}$
11.	$10B_n= B_{n+1}+B_{n+3}$	**28.**	$27B_n= B_n+B_{n+1}+B_{n+3}+B_{n+4}$	**45.**	$44B_n= B_{n+2}+ B_{n+3}+B_{n+5}$
12.	$11B_n= B_n+B_{n+1}+B_{n+3}$	**29.**	$28B_n=B_{n+2}+B_{n+3}+B_{n+4}$	**46.**	$45B_n=B_n+B_{n+2}+B_{n+3}+B_{n+5}$
13.	$12B_n= B_{n+2}+B_{n+3}$	**30.**	$29B_n= B_n+B_{n+2}+B_{n+3}+B_{n+4}$	**47.**	$46B_n=B_{n+1}+B_{n+2}+B_{n+3}+B_{n+5}$
14.	$13B_n= B_n+B_{n+2}+B_{n+3}$	**31.**	$30B_n= B_{n+1}+B_{n+2}+B_{n+3}+B_{n+4}$	**48.**	$47B_n=B_n+B_{n+1}+B_{n+2}+B_{n+3}+B_{n+5}$
15.	$14B_n= B_{n+1}+B_{n+2}+B_{n+3}$	**32.**	$31B_n=B_n+B_{n+1}+B_{n+2}+B_{n+3}+B_{n+4}$	**49.**	$48B_n= B_{n+4}+B_{n+5}$
16.	$15B_n= B_n+B_{n+1}+B_{n+2}+B_{n+3}$	**33.**	$32B_n=B_{n+5}$	**50.**	$49B_n= B_n+B_{n+4}+B_{n+5}$
17.	$16B_n=B_{n+4}$	**34.**	$33B_n= B_n+B_{n+5}$	**51.**	$50B_n= B_{n+1}+B_{n+4}+B_{n+5}$

При нахождении суммы большого количества многоразрядных двоичных чисел вследствие наличия единиц в смежных разрядах возможно возникновение различной длины цепочек ЕП из разрядов с меньшим весом в разряды с большим весом. При этом невозможно заранее ни предвидеть возникновение цепочки ЕП, ни рассчитать ее длину. СП из табл. 1 не допускает неконтролируемое распространение цепочек ЕП благодаря свойству бесконфликтной расстановки единиц сразу в несколько разрядов большего веса.

Эта СП обеспечивает такую расстановку единиц по разрядам разного веса при промежуточных вычислениях и получении результирующей суммы, которая исключает запись нуля или единицы в любой разряд с уже записанной в нем единицей. Тем самым исключается потеря промежуточных значений (ПЗ) при вычислении суммы и обеспечивается бесконфликтное сложение.

В ЛИРССЧ (1) использована диагональная запись [5, 10 – 13] суммы каждого вертикального разрядного среза (ВРС), определяемая СП табл. 1, причем каждому разряду суммы ВРС отвечает свой горизонтальный ряд ПЗ. А именно, для заданного n каждая цифра суммы ВРС записывается в i-м горизонтальном ряду ПЗ (i=0,1,2…m, где m – разрядность суммы ВРС) в столбце, отвечающем весовому значению разряда B_{n+i}. Диагональная запись суммы каждого ВРС в ДССЧ является частным случаем применения правил из табл. 1 для одновременного сложения в ЛИРССЧ (1). Максимальная разрядность суммы ВРС зависит от количества слагаемых и равна числу горизонтальных рядов ПЗ для диагональной записи этой суммы (по одному разряду суммы единиц ВРС на один ряд ПЗ). Например, для диагональной записи суммы ВРС при вычислении суммы 20 слагаемых сначала необходимо 5 рядов разрядов ПЗ (см. рис. 2). Далее эти 5 рядов разрядов рассматриваются как 5 новых чисел увеличенной разрядности, к новым суммам ВРС которых применяются первые 6 правил табл. 1. Для записи этих полученных сумм ВРС требуются уже 3 новых ряда разрядов под ПЗ. Из этих 3 новых рядов разрядов применением первых 4 правил табл. 1 получают двухрядный код. Так СП из табл. 1 применяется многократно для новых наборов ПЗ. На заключительном этапе сложение 20 слагаемых сводится к сложению двух рядов двоичных кодов и получению результирующей двоичной суммы с помощью СРП. Результат одновременного МС в ЛИРССЧ (1) представляется в ДССЧ без дополнительных преобразований.

Длина разрядной сетки каждой суммы в ВРС зависит от числа слагаемых и определяется формулой

$$y = \left\lfloor \frac{\ln(x)}{\ln 2} + 1 \right\rfloor,$$

где x – количество слагаемых в десятичной СЧ, y – разрядность числа x в ДССЧ, $\lfloor\ \rfloor$ обозначает округление с отбрасыванием дробной части числа, поскольку число разрядов является целым. График зависимости числа разрядов для записи суммы единиц в каждом ВРС от количества одновременно суммируемых чисел приведен на рис. 3.

Преимущества суммирования в ЛИРССЧ (1) по сравнению с АУ:

1) Промежуточный код не разделяется на код условной суммы и код переносов.

2) Не используются сумматоры с сохранением переноса при МС.

Из рис. 3 можно сделать такие выводы:

1) есть широкие интервалы одновременно складываемых чисел, не требующие роста разрядности, а, следовательно, и сложности многооперандного сумматора;

2) структура алгоритма одновременного МС не меняется с ростом числа слагаемых;

3) вместо сумматоров с сохранением переноса можно использовать устройство подсчета еди-

ниц в каждом ВРС и регистр для хранения суммы каждого ВРС.

Левая часть каждого интервала одновременно складываемых чисел, не требующего роста разрядности многооперандного сумматора, ограничена числами вида 2^L, а правая часть – числами вида 2^{L+1} , где L=1,2,3… С увеличением количества одновременно складываемых чисел эти интервалы еще больше уширяются, что указывает на практическую возможность построения многооперандных сумматоров с изменяемым в широких диапазонах количеством слагаемых без аппаратного увеличения сложности этих многооперандных сумматоров в пределах одного диапазона. Например, в программной модели сложения в ЛИРССЧ (1) при сложении в интервале от 32 до 63 слагаемых включительно по сравнению со сложением от 16 до 31 слагаемых добавляется только один ряд промежуточных значений из 16 разрядов, тогда как в программной модели АУ добавляется соответственно от 32 до 64 рядов промежуточных значений разрядностью не меньше 16.

МС в избыточной рекуррентной системе счисления за счет применения простой СП во много раз уменьшает количество сохраняемых ПЗ по сравнению со сложением по АУ. Так, при сложении 10-ти 16-разрядных операндов в ЛИРССЧ (1) требуется в 1,86 раза меньше сохраняемых ПЗ, при сложении 20-ти 16-разрядных операндов – в 3,1 раза меньше сохраняемых ПЗ [4], а при сложении 50-ти 16-разрядных операндов – в 7,3 раза меньше сохраняемых ПЗ по сравнению со сложением по АУ. Соответственно при сложении по АУ уменьшается быстродействие МС.

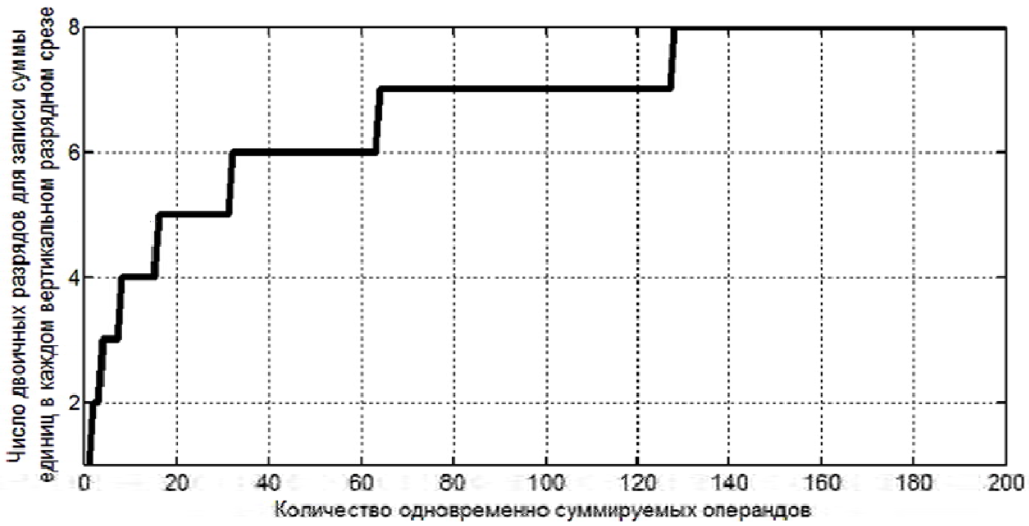

Рис. 3. График зависимости числа разрядов для записи суммы единиц в каждом ВРС от количества одновременно суммируемых чисел

Из экстраполяции имеющихся из программных моделей значений полиномом 4-го порядка (рис. 4) с доверительной вероятностью 95% следует, что с увеличением числа операндов отношение числа со-

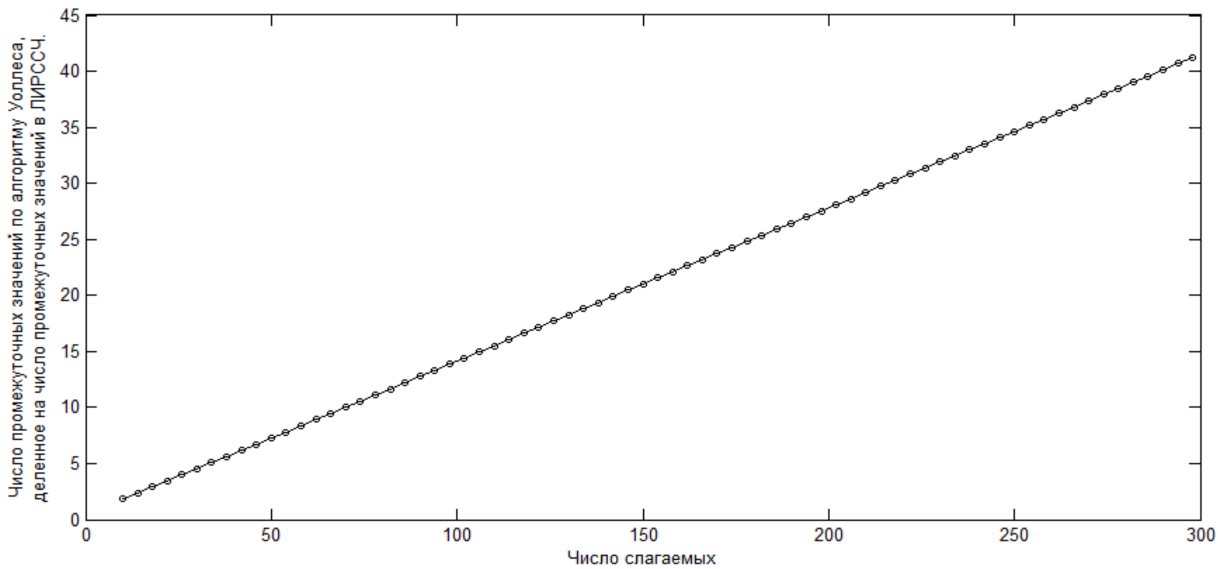

Рис. 4. Экстраполяция имеющихся из программных моделей значений полиномом 4-го порядка

храняемых ПЗ при сложении по АУ к числу сохраняемых ПЗ при сложении в ЛИРССЧ (1) еще больше возрастает. Из рис. 4 следует, что при суммировании 300 чисел по АУ потребуется в 41,5 раза больше ПЗ, чем при суммировании в ЛИРССЧ. Вследствие этого во столько же раз усложнится аппаратная реализация сложения по АУ и соответственно снизится быстродействие.

Выводы

В результате проведенных исследований найдено, что программные модели одновременного многооперандного сложения в ЛИРССЧ (1) для различного количества 16-разрядных целых положительных чисел требуют во много раз меньше разрядов для хранения промежуточных результатов вычислений и во много раз быстрее, чем соответствующие модели сложения по алгоритму Уоллеса в обычной двоичной системе счисления.

Из экстраполяции имеющихся из программных моделей данных следует, что при суммировании 300 чисел по АУ потребуется в 41,5 раза больше ПЗ, чем при суммировании в ЛИРССЧ. Вследствие этого во столько же раз усложнится аппаратная реализация сложения по АУ и соответственно снизится быстродействие.

На практике ЛИРССЧ 3-го порядка (1) может быть использована для ускорения работы цифровых фильтров.

Список литературы

1. *Анисимов А.В. Сложение без единиц переноса / А.В. Анисимов // Кибернетика и системный анализ. – 1996. – №2. – С. 3-16.*
2. *Мартинюк Т.Б. Рекурсивні алгоритми багатооперандної обробки інформації: [Монографія] / Т.Б. Мартинюк. – Вінниця: "Універсум-Вінниця", 2000. – 216 с. – ISBN 966 – 7199 – 98 - 3.*
3. *Balasubramanian P. Self-Timed Multi-Operand Addition / P. Balasubramanian, D.A. Edwards, W.B. Toms // International Journal of circuits, systems and signal processing. – 2012. –Volume 6, Issue 1. – P. 1-11.*
4. *Федотова-Пивень И.Н. Программное моделирование совмещенного во времени сложения двадцати целых положительных чисел в избыточной рекуррентной системе счисления третьего порядка / И.Н. Федотова-Пивень, О.Б. Пивень // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка: Зб. наук. пр. – К.: Век+, – 2013. – № 58. – С. 131-137.*
5. *Ромм Я.Е. Метод вертикальной обработки потока целочисленных групповых данных. I. Групповые арифметические операции / Я.Е. Ромм // Кибернетика и системный анализ. – 1998. – №3. – С. 123-151.*
6. *Супер ЭВМ. Аппаратная и программная организация / Под ред. С. Фернбаха. – М.: Радио и связь, 1991.*
7. *Ajay K. Verma. Automatic synthesis of compressor trees: reevaluating large counters / Ajay K. Verma, Paolo Ienne // DATE'07 Proceedings of the conference on Design, automation and test in Europe. EDA Consortium San Jose, CA, USA. – 2007. – P.443-448. – ISBN:978-3-9810801-2-4.*
8. *Ienne P. Arithmetic transformations to maximise the use of compressor trees. / P. Ienne, A.K. Verma // Second IEEE International Workshop on Electronic Design, Test and Applications, DELTA 2004, Perth, Australia, 28-30 January, P. 219-224.*
9. *Um J. Optimal allocation of carry-save adders in arithmetic optimization / J. Um, T. Kim, C.L. Liu // 1999 InternationaL Conference on Computer Aided Design (ICCAD'99), San Jose, CA, 7-11 November, 1999. – P.410-413.*
10. *Карцев М.А. Вычислительные системы и синхронная арифметика / М.А. Карцев, В.А. Брик. – М.: Радио и связь, 1981. – 358 с.*
11. *Храпченко В.М. Об одном способе преобразования многорядного кода в однорядный / В.М. Храпченко // Доклады АН СССР. – 1963. – Т. 148, №2. – С. 296-299.*
12. *Храпченко В.М. Методы ускорения арифметических операций, основанные на преобразовании многорядного кода / В.М. Храпченко // Вопросы радиоэлектроники. Сер. УП ЭВТ. – 1965. – Вып. 8. – С. 121-144.*
13. *Dadda L. Some schemes for parallel multipliers / L. Dadda // Alta Frequenza. – 1965, May. – Vol.34. – P. 349-356.*

Поступила в редколлегию 19.01.2015

Рецензент: д-р техн. наук, проф. В.Н. Рудницкий, Черкасский государственный технологический университет, Черкассы.

ПРО БАГАТООПЕРАНДНЕ ДОДАВАННЯ З ОБМЕЖЕНИМ ПОШИРЕННЯМ ОДИНИЦЬ ПЕРЕНОСІВ

І.М. Федотова-Півень

У статті наведено короткий порівняльний аналіз двох методів багатооперандного додавання: методу Уоллеса і методу додавання в лінійній надлишковій рекурентній системі числення третього порядку. Показано перевагу другого методу завдяки діагональному запису суми кожного вертикального розрядного зрізу і застосуванню набору рекурентних правил додавання. На основі програмної моделі та екстраполяції розрахункових значень встановлено багаторазове зростання швидкодії та зменшення кількості проміжних значень другого методу із зростанням числа операндів до 300 в порівнянні з методом Уоллеса.

Ключові слова: *багатооперандне додавання, рекурентна система числення, надлишковість, вертикальні розрядні зрізи, одночасне додавання.*

ABOUT A MULTI-OPERAND ADDITION TO THE LIMITED PROPAGATION OF TRANSFER UNITS

I.N. Fedotova-Piven

The paper presents a brief comparative analysis of the two methods for multi-operand addition: Wallace method and the method of addition in linear redundant recurrent number system of the third order. The advantage of the second method due to the diagonal entries sum of each bit vertical slice and the use of a set of recurrence rules of addition. Based on the programming model and the extrapolation of the calculated values found multiple increase performance and reduce the number of intermediate values of the second method with the number of operands to 300 compared with the method of Wallace.

Keywords: *multi-operand addition, recurrent number system, redundancy, vertical bit slices, simultaneous addition.*