

УДК 681.142.1.

АССОЦИАТИВНЫЙ ПРОЦЕССОР ДЛЯ КРУПНОБЛОЧНОЙ ОБРАБОТКИ ИНФОРМАЦИИ

В.Н.Алеева, Я.И.Фет

Введение

В работе рассматривается однородный ассоциативный процессор широкого назначения, обеспечивающий эффективную крупноблочную обработку информации при решении как вычислительных, так и информационно-логических задач.

В § 1 описана ячейка накопителя (θ -ячейка). В ходе разработки процессора структура и функции θ -ячейки уточнялись на основе анализа ее поведения при решении различных задач. В этом же параграфе приведена блок-схема процессора.

В § 2 рассмотрены 8 микрокоманд процессора, которые составляют основу всех микропрограмм.

§ 3 посвящен подробному содержательному описанию некоторых микропрограмм процессора.

Для каждой микропрограммы приведены оценки времени её выполнения и необходимого расхода памяти.

Особое внимание уделяется организации микропрограммы "быстрого умножения".

Благодаря реализации в ассоциативном варианте известного метода сокращенного сложения с хранением переносов, получена вместо квадратичной линейная (относительно числа разрядов) оценка времени умножения.

Микропрограмма быстрого умножения позволяет эффективно использовать предлагаемый процессор для векторных и матричных вычислений.

в качестве примеров в конце статьи рассмотрены задача параллельного вычисления попарных скалярных произведений двух массивов векторов и задача последовательного (по столбцам) вычисления произведения двух матриц.

1. Описание ассоциативного процессора

Ассоциативный процессор для крупномасштабной обработки информации (АКП) состоит из матрицы-накопителя и нескольких управляющих схем. Накопитель представляет собой двумерную однородную структуру, построенную из θ -ячеек. θ -ячейка (рис. 1)

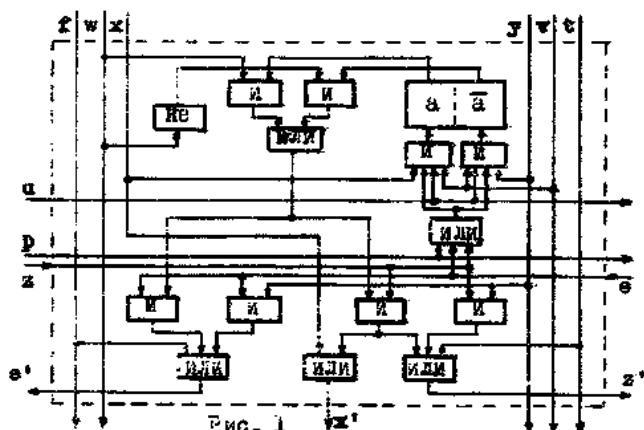


Рис. 1

содержит 1 двоичный запоминающий элемент и 14 логических вентилей и выполняет следующие функции:

$$x' = x \vee x \bar{b}, \quad (1)$$

$$x' = z(bvy)vt, \quad (2)$$

$$v' = e(bvy) \vee f, \quad (3)$$

$$\bar{b} = aw \vee \bar{a}\bar{w}, \quad (4)$$

$$z_3 = x(xvevp)wv, \quad (5)$$

$$z_2 = y(xvevp)wv, \quad (6)$$

3 матрице-накопителе выходы f', t', v', w', x', y' каждой ячейки соединены со входами f, t, v, w, x, y соседней нижней

ячейки, выходы p', u' и z' со входами p, u и z соседней правой ячейки, а выход e' — со входом e соседней левой ячейки. Выходные переменные ведут себя по-разному: f, t, v, w, y — вертикальные шины, p и u — горизонтальные шины. Каждая из шин проводит в обе стороны подаваемые на нее сигналы. Поведение x', z', e' определяется выражениями (1), (2), (3) и зависит от состояния запоминающего элемента, а также от значений поступающих на входы ячейки переменных x, z, e, w, y, t, f .

Кроме матрицы-накопителя, в состав АКП (рис. 2) входят: регистр признака (РП), регистр маски (РМ), регистр индикаторов (РИ), регистр управляющих триггеров (РУТ), выходной регистр (ВР), ЭУ микропрограммы (ЭУМП) и устройство управления (УУ).

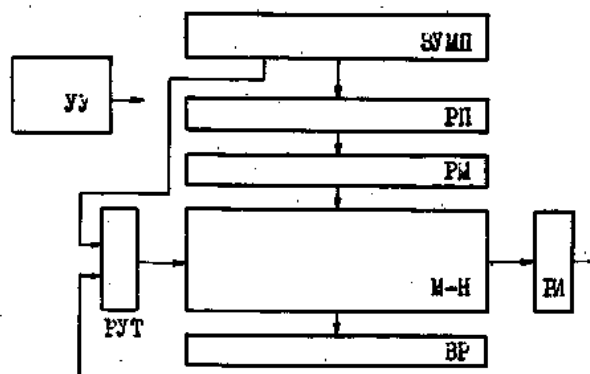


Рис. 2

2. Микрокоманды процессора

В АКП могут выполняться 8 различных микрокоманд: запись горизонтальная (ЗГ), запись вертикальная (ЗВ), чтение обычное (ЧО), чтение дизъюнктивное (ЧД), поиск на равенство (ПР), поиск максимума (ПМ), поиск-запись вправо (ПЗП), поиск-запись влево (ПЗЛ).

В таблице I приведены значения переменных $e, f, p, t, u, v, w, x, y, z$ на границах матрицы-накопителя при выполнении различных микрокоманд.

Таблица 1

Значения переменных на границах θ -матрицы											
Микро-команда	f_{ij}	w_{ij}	x_{ij}	y_{ij}	v_{ij}	t_{ij}	u_{ij}	p_{ij}	z_{ij}	e_{ij}	
ЭГ	0	-	код записи	обратный код записи	маска записи	0	$u_{ij}=1$ в выбр. строках	1	0	0	
ЭВ	0	-	1	0	$v_{ij}=1$ в выбр. строках	0	маска записи	код записи	0	0	
ЧО	0	1	0	1	-	0	0	0	$z_{ij}=1$ в выбр. строке	0	
ЧД	0	1	0	1	-	0	0	0	$z_{ij}=1$ в выбр. строках	0	
ПР	0	код поиска	-	маска поиска	-	0	0	0	- " -	0	
ПМ	0	1	0	$y_{ij}=x_{ij}$	-	0	0	0	- " -	0	
зона опроса	0	код поиска	-	маска поиска	0	$t_{ij}=1$ в правом столбце поля	$u_{ij}=1$ в выбр. строках	1	1	0	
	0	-	код записи	обратный код записи	маска записи			1	1	0	
зона опроса	$f_{ij}=1$ в левом столбце поля	код поиска	-	маска поиска	0	0	$u_{ij}=1$ в выбр. строках	1	0	1	
	зона записи	-	код записи	обратный код записи	маска записи	0		1	0	1	

θ -матрицу можно рассматривать как матричное запоминающее устройство, в котором возможна адресная запись по строкам (запись горизонтальная) или по столбцам (запись вертикальная).

При горизонтальной записи слово $X = x_n, \dots, x_m$, которое подлежит записи, подается на входы x , а обратный его код, т.е. слово $\bar{X} = \bar{x}_n, \dots, \bar{x}_m$, одновременно подается на входы y верхней границы матрицы.

На шины f и t подаются константы $f = t = 0$. Состояние шин w на запись не влияет. На входы z всех ячеек левой границы матрицы в режиме записи подаются константы $z = 0$. Вследствие этого, согласно (1), во всех ячейках $x' = x$. Значит, во все ячейки любого j -го столбца матрицы поступает сигнал $x_{ij} = x_j$. Этот сигнал подается на входы вентиля записи "1" запоминающих элементов данного столбца. В то же время на входы вентиля записи "0" поступает с шины y сигнал \bar{x}_j .

Для записи в любую ячейку служат три шины: горизонтальные шины p и u и вертикальная шина v . На шины v обычно подается сигнал $v = 1$. В тех столбцах, в которых необходимо замаскировать (запретить) запись, подается $v = 0$. С помощью шины u указывается адрес записи, а именно: сигнал $u = 1$ подается только в те строки, в которые надо записать слово X . Разрешение записи осуществляется подачей в выбранную строку сигнала $p = 1$.

Формально процесс записи в запоминающий элемент θ -ячейки описывается выражениями (5) и (6). Заметим, что разрешение записи по горизонтали может осуществляться не только переменной p , но также переменными z и e . Эти виды записи относятся к микрокомандам ПЗИ и ПЗЛ, которые подробно рассматриваются ниже.

При вертикальной записи слово $P = p_n, \dots, p_m$, которое подлежит записи, подается на входы p левой границы матрицы. На шины f, t, z и e подаются константы $f = t = z = e = 0$. Состояние шин w на запись не влияет. На шины x подаются константы 1, а

на шины y - константы 0. На все шины u подается сигнал $u=1$. С помощью шины v указывается адрес записи, а именно: сигнал $v=1$ подается только в те столбцы, в которые надо записать слово P . Если столбец, в который производится запись, предварительно был очищен, то ясно, что в него запишется слово P .

Ч т е н и е о б ы ч н о е (40)

Для считывания произвольного слова, записанного в некоторой строке матрицы-накопителя на все входы f, t и x верхней границы матрицы подаются константы $f=t=x=0$, на входы y и w константы $y=w=1$. Состояние шин v на чтение не влияет. Адрес считываемого слова указывается с помощью переменной z , а именно: сигнал $z=1$ подается на ту строку, содержимое которой необходимо прочитать. Все остальные z должны быть равны нулю.

Поскольку все $y=1$, то, согласно (2), сигнал $z=1$ поступит на все ячейки выбранной строки. Кроме того, во всех этих ячейках $x=0$. Следовательно, в соответствии с (1), $x'=b$. Но так как $w=1$, то, согласно (4), $b=a$. Значит, во всех ячейках данной строки $x'=a$. Поскольку во всех остальных строках $z=0$, то в них $x'=x$. Следовательно, значения переменных a считываемой строки без изменений передаются на выходы x' нижней границы матрицы, а отсюда - на выходной регистр.

Ч т е н и е д и з ь ю н к т и я н о е (4Д)

При этой микрокоманде, в отличие от предыдущего, сигнал $z=1$ подается не на одну строку матрицы, а одновременно на несколько. Из (4) и (1) следует, что в этом случае на выходах x' нижней границы матрицы реализуется поразрядная дизъюнкция всех считываемых строк.

П о и с к н а р а в е н с т в о (ПР)

При поиске на равенство θ -матрица работает, как обычное ассоциативное ЗУ. Код опроса подается на шины w . На шины

f, p, t, u подаются константы $f=p=t=u=0$. Состояние шин x и v на поиск не влияет. На все те строки матрицы, в которых ведется поиск, подаются (на левую границу) сигналы $x=1$. Если во всех ячейках некоторой строки $u=a$, то в этих ячейках, согласно (4) и (1), $v=1$, а $x'=x$. Следовательно, в каждой строке, содержимое которой совпадает с кодом опроса, на правой границе матрицы появится сигнал $x'=1$, который и является признаком совпадения. Маскирование некоторых разрядов при поиске производится путем подачи в соответствующих столбцах сигнала $y=1$. Согласно (2), в таких столбцах $x'=x$ независимо от соотношения переменных a и u .

Поиск максимума (ПМ)

При поиске максимума θ -матрица эквивалентна α -матрице [I]. Действительно, в микрокоманде ПМ все $u=1$, $t=0$. Значит, $b=a$, и выражения (1) и (2) принимают вид:

$$\begin{aligned}x' &= x \vee x a, \\x' &= x (a \vee y).\end{aligned}$$

Кроме того, в этой микрокоманде выход x'_{ij} каждого столбца соединяется (через инвертор) со входом y_{ij} того же столбца.

Следовательно, θ -матрица ведет себя, как α -матрица. В работе [I] показано, что в такой матрице при подаче на левую границу сигналов $x=1$ сигнал $x'=1$ на правой границе появляется в тех и только тех строках, в которых содержатся максимальные (равные) числа. Маскирование некоторых разрядов при поиске максимума производится путем подачи на соответствующие шины y вместо инверсии x' константы $y=1$.

Поиск - запись вправо (ПЗП)

При этой микрокоманде в матрице-накопителе выделяются поля, каждое из которых делится на две зоны: левая - зона опроса и правая - зона записи. Работа ведется параллельно по всем полям. В каждом поле выполняются одновременно поиски на равенство по заданному для этого поля признаку (в зоне опроса) и запись заданного для этого поля кода (в зону записи). Для этого в столбцы матрицы-накопителя, непосредственно примыкающие к левой гра-

ниче каждого из выделенных полей, подается сигнал $\bar{t}=1$, который возбуждает сигналы \bar{x} в ближайшей зоне опроса. Сигналы \bar{x}' на выходе зоны опроса, в соответствии с (2) и (4), возникают только в тех строках, в которых код опроса совпадает с кодом хранимого слова. Распространяясь дальше в зону записи своего поля, эти сигналы, в соответствии с (5) и (6), обеспечивают разрешение записи заданного кода в запоминающие элементы зоны записи.

Поиск - запись влево (ПЗЛ)

Эта микрокоманда аналогична предыдущей с той разницей, что в ней каждая правая зона является зоной опроса, а левая - зоной записи. В столбцы матрицы-накопителя, непосредственно прилегающие к правой границе каждого из полей, подается сигнал $\bar{t}=1$, возбуждающий сигналы e в ближайшей зоне опроса. В соответствии с (3) и (4) в выделенных строках на выходе правой зоны возникают сигналы e' , которые обеспечивают запись в левую зону своего поля.

3. Микропрограммы процессора

В этом разделе будет приведено содержательное описание микропрограмм ввода и вывода информации, алгебраического сложения, умножения, деления, массового сдвига вправо (влево) и быстрого умножения. Арифметические операции выполняются параллельно над M парами операндов ($M \leq N$), причем каждая пара должна быть расположена в одной строке матрицы-накопителя. Предполагается представление чисел с фиксированной запятой в прямых кодах.

Длительность каждой микропрограммы будет оцениваться количеством микрокоманд, необходимых для её выполнения. Более точные оценки, учитывающие индивидуальные длительности микрокоманд, зависят от физической реализации процессора и здесь не могут быть рассмотрены.

Необходимый расход оборудования оценивается количеством θ -ячеек.

3.1. Ввод информации

Ввод информации в матрицу можно осуществлять последовательно по строкам или последовательно по столбцам. Если нужно ввести в некоторые строки или в некоторые столбцы одинаковые слова, то их можно вводить одновременно.

Если запись производится последовательно по строкам, то используется микрокоманда ЗГ. На входы $x(y)$ верхней границы подаются последовательно записываемые слова, а на входы u левой границы в заполняемые строки подаются по очереди сигналы $u = 1$.

Время заполнения матрицы

$$T_{гвв} \leq N.$$

Равенство достигается, если все слова массива различны. В общем случае возможно строгое неравенство за счет параллельной записи одинаковых слов.

Если запись производится последовательно по столбцам, то используется микрокоманда ЗВ. На входы ρ левой границы подаются последовательно записываемые слова, а на входы v верхней границы в заполняемые столбцы подаются по очереди сигналы $v = 1$.

Для ввода массива из N слов требуется время

$$T_{звв} \leq n.$$

Очевидно, что при $n < N$ ввод информации целесообразно производить с помощью микрокоманды ЗВ.

3.2. Вывод информации

Вывод информации можно осуществить последовательно по строкам, либо последовательно по столбцам.

Если вывод производится по строкам, то используется микрокоманда ЧО, адреса выводимых строк ($z = 1$) подаются по очереди на входы z левой границы. Если вывод производится по столбцам, то используется микрокоманда ПР. Адрес ($v = 1$) подаётся на вход v считываемого столбца. На вход $x(y)$ этого столбца подаётся признак $f(0)$. На все входы z левой границы подаются константы $z = 1$. Выходным регистром служит столбец индикаторов АКП.

Для вывода массива из N n -разрядных слов необходимо время

$$T_{\text{выв}} = N \quad (\text{при выводе по строкам})$$

$$T_{\text{выв}} = n \quad (\text{при выводе по столбцам}).$$

Целесообразно использовать вывод по столбцам, если $n < N$.

3.3. Алгебраическое сложение

На рис. 3 показано распределение памяти накопителя при выполнении данной микропрограммы. Результаты заносятся в поле первых операндов и заменяют их там по окончании работы программы. Используется метод последовательных преобразований состояний, описанный в [2].

При выполнении алгебраического сложения в зависимости от знаков операндов и соотношения их абсолютных величин возможны следующие варианты формирования результатов (таблица 2).

Таблица 2.

Знак a	Знак b	Соотношение абсолютных величин	Знак результата	Вид операции
+	+	$ a > b $	+	$ a - b $
+	-	" "	+	$ a - b $
-	+	" "	-	$ a - b $
-	-	" "	-	$ a + b $
+	+	$ a < b $	+	$ b - a $
+	-	" "	-	$ b - a $
-	+	" "	+	$ b - a $
-	-	" "	-	$ a + b $
+	+	$ a = b $	+	$ a + b $
+	-	" "	+	$ a - b $
-	+	" "	+	$ b - a $
-	-	" "	-	$ a + b $



Рис.3

Микропрограмма АЛГЕБРАИЧЕСКОЕ СЛОЖЕНИЕ состоит из трех подпрограмм: сравнение абсолютных величин, определение знака результата и вида операции, собственно сложение.

I-я подпрограмма - с р а в н е н и е а б с о л ю т н ы х в е л и ч и н выполняется параллельно над всеми парами аргументов и последовательно по разрядам (начиная со старшего).

Перед началом сравнения все разряды столбцов ℓ_1 и ℓ_2 поля меток находится в состоянии "00".

На первом шаге сравнения старшие разряды полей A и B оперируются признаком IO и в выделенных строках заносится метка IO в столбцы ℓ_1 и ℓ_2 поля меток. Затем те же разряды оперируются признаком OI и в выделенных строках заносится метка OI. В результате в столбцах ℓ_1 и ℓ_2 поля меток оказываются отмеченными парой IO (OI) те строки, в которых $a_{im} > b_{im}$ ($a_{im} < b_{im}$). В остальных строках, очевидно, $a_{im} = b_{im}$, и в столбцах ℓ_1 и ℓ_2 поля меток остается пара 00.

Выделенные строки дальнейшего просмотра не требуют, поскольку строгое неравенства в старших разрядах обеспечивают однозначный результат сравнения независимо от соотношения следующих разрядов. Поэтому на каждом шаге проверке подвергаются только те строки, которые имеют метки 00.

Сравнение оканчивается после опроса младших разрядов полей A и B . При этом все строки оказываются разделенными на три подмножества: метками IO отмечены строки, где $|a_i| > |b_i|$, метками OI, где $|a_i| < |b_i|$, метками 00, где $|a_i| = |b_i|$.

Для удобства реализации подпрограммы определения знака результата и вида операции все пары третьего подмножества присоединяются к парам первого и второго подмножеств. При этом ко

второму подмножеству относятся только те пары, у которых знак a_i - минус, а знак b_i - плюс. Все остальные пары третьего подмножества относятся к первому. Практически это выполняется следующими командами:

1. опрос (в третьем подмножестве) знаковых разрядов комбинацией 10;
2. запись в выделенных строках метки I в столбце ℓ_2 поля меток;
3. опрос столбцов ℓ_1 и ℓ_2 поля меток комбинацией 00;
4. запись в выделенных строках метки I в столбце ℓ_1 .

2-я подпрограмма - определение знака результата и вида операции - реализует таблицу 2.

В двоичной форме, с учетом результата работы 1-й подпрограммы, эта таблица может быть представлена следующим образом:

Таблица 3

Знак a	Знак b	Соотношение абсол. велич.		Знак суммы	Вид операции		
					$ a + b $	$ a - b $	$ b - a $
		ℓ_1	ℓ_2		ℓ_3	ℓ_4	ℓ_5
0	0	I	0	0	I	0	0
0	I	I	0	0	0	I	0
I	0	I	0	I	0	I	0
I	I	I	0	I	I	0	0
0	0	0	I	0	I	0	0
0	I	0	I	I	0	0	I
I	0	0	I	0	0	0	I
I	I	0	I	I	I	0	0

Очевидно, работа подпрограммы определения знака результата и вида операции состоит в том, что все строки матрицы-накопителя опрашиваются по очереди 8 раз признаками, соответствующими кодам первых четырех столбцов таблицы 3. После каждого опроса в столбцы знака результата и меток ℓ_3 , ℓ_4 , ℓ_5 выделенных строк заносятся коды, соответствующие последним четырём столбцам данной строки таблицы 3.

3-я подпрограмма - собственно сложение - состоит из трех частей, соответствующих операциям $A+B$, $A-B$, $B-A$, выполняемых последовательно над каждой парой разрядов слагаемых, причем эти части применяются к подмножествам строк, полученным в результате разбиения, произведенного 2-й подпрограммой.

Правила подстановок для каждой из трех частей подпрограммы следуют из приведенной ниже таблицы истинности (таблица 4). В этой таблице a и b - значения переменных, c - значение переноса (займа) из предыдущего разряда, c'_1 - перенос в следующий разряд, c'_2 и c'_3 - заем из следующего разряда.

Таблица 4

a	b	c	$a+b$	c'_1	d_1	$a-b$	c'_2	d_2	$b-a$	c'_3	d_3
0	0	0	0	0	-	0	0	-	0	0	-
0	0	1	1	0	*	1	1	*	1	1	*
0	1	0	1	0	*	1	1	*	1	0	*
0	1	1	0	1	-	0	1	-	0	0	*
1	0	0	1	0	-	1	0	-	1	1	*
1	0	1	0	1	*	0	0	*	0	1	*
1	1	0	0	1	*	0	0	*	0	0	*
1	1	1	1	1	-	1	1	-	1	1	-

В графах d_1 , d_2 , d_3 знаком "*" отмечены строки, в которых значения функций не совпадают со значениями аргументов. Очевидно, что в этих и только в этих строках необходимо производить подстановки.

В процессе обработки каждой пары разрядов при каждой подстановке (кроме последней) все строки, к которым была применена подстановка, отмечаются единицами в столбце меток e_c . Это необходимо для того, чтобы исключить эти строки из дальнейшей обработки данных разрядов. После последней подстановки столбец e_c очищается, то есть подготавливается к следующему циклу.

Время выполнения микропрограммы АЛГЕБРАИЧЕСКОЕ СЛОЖЕНИЕ

$$T_{AC} = 27m + 12.$$

Необходимые размеры матрицы-накопителя

$$S_{Ac} = (2m + 9) \times N.$$

3.4. Умножение

На рис. 4 показано распределение памяти накопителя при 32-полнении микропрограммы УМНОЖЕНИЕ.

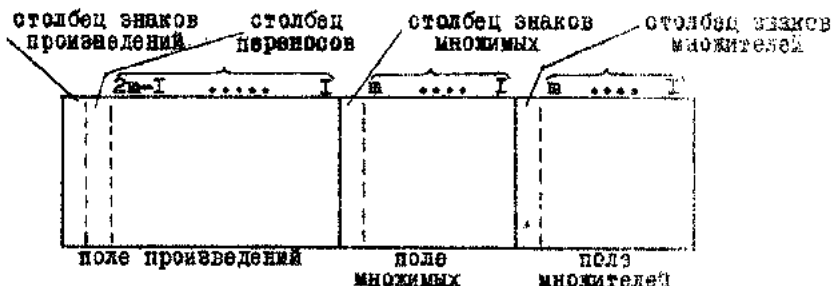


Рис. 4

Предполагается точное вычисление произведений (разрядность произведения $2m$).

Перед началом собственно умножения определяются знаки произведений путем опроса знаковых столбцов полей множителей и множимых признаками 01 и 10. Во всех выделенных при этом строках заносится 1 (код знака " - ") в столбец знаков произведений.

Затем выполняется m циклов собственно умножения.

i -й цикл заключается в следующем.

Опрашивается признаком 1 i -й разряд поля множителей (остальные разряды поля множителей замаскированы). В выделенных строках производится сложение множимых с содержанием поля произведений, причем j -й разряд множимого суммируется с $(j+i-1)$ -м разрядом поля произведений. Это соответствует сдвигу множимого на i разрядов влево относительно произведения.

Отметим, что необходимая в каждом цикле умножения операция группового сложения может быть выполнена значительно проще, чем описанная выше операция алгебраического сложения. Поскольку здесь производится сложение абсолютных величин множимых и

частичных произведений, то достаточно выполнять только часть $A+B$ третьей подпрограммы микропрограммы СЛОЖЕНИЕ.

Время выполнения микропрограммы УМНОЖЕНИЕ

$$T_y = 7m^2 + 3.$$

Необходимые размеры матрицы-накопителя

$$S_y = (4m + 3) \times N.$$

3.5. Деление

На рис. 5 показано распределение памяти накопителя при выполнении микропрограммы ДЕЛЕНИЕ.

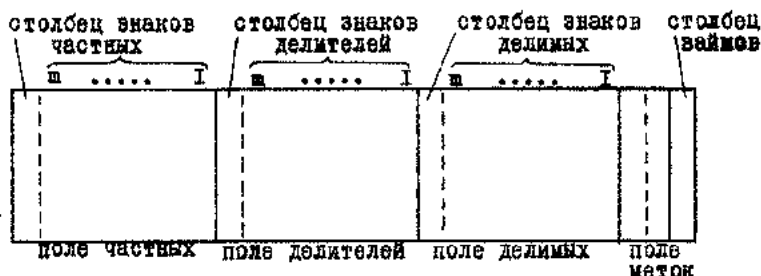


Рис. 5

Предполагается вычисление частных с округлением младшего разряда.

Определение знаков результатов производится точно так же, как и при умножении. Знаки частных записываются в специальный столбец.

Цифры частных записываются в поле частных. Разрядность каждого частного $E_z = m - E_g + 1$, где E_g - разрядность делителя.

Предполагается, кроме того, что все делители - ненулевые. В противном случае надо было бы перед началом деления выполнить в поле делителей поиск на равенство нулю, отметить выделенные строки в специальном столбце меток (не показанном на рис. 5) и исключить эти строки из обработки при делении.

Перейдем к описанию собственно деления. В основе используемого алгоритма - обычное деление "столбиком".

На каждом шаге в процесс деления вовлекаются новые пары аргументов, причем на k -м шаге вовлекаются такие пары, у которых делители по абсолютной величине меньше 2^k .

П е р в ы й ш а г - вычисление старшего разряда частного.

Выделяются те строки, в которых младший разряд делителя равен 1, а все остальные - нулю. В каждой из этих строк производится сравнение абсолютной величины младшего разряда делителя ($b_{i,1}$) со старшим разрядом делимого ($a_{i,m}$). Сравнение выполняется точно так же, как в I-й подпрограмме микропрограммы СЛОЖЕНИЕ.

Результаты сравнения записываются в поле меток.

Метка 00 соответствует строкам, где $a = b$,

"- 01 " " " " $a > b$,

"- 10 " " " " $a < b$.

Во всех строках, отмеченных метками 00 и 01, производится вычитание $a_{i,m} - b_{i,m}$, а в первый столбец поля частных записывается 1 (значение старшего разряда частного).

Отметим, что групповая операция вычитания может выполняться аналогично части А-В третьей подпрограммы микропрограммы СЛОЖЕНИЕ.

Предварительное сравнение делимых с делителями позволяет оставить неизменными те пары, которые при вычитании дали бы отрицательную разность. Это избавляет от необходимости "восстановления остатков".

Используемый метод оправдывается тем, что сложение (для восстановления остатков) требует b_m ассоциативных команд, в то время как сравнение - всего $4m$ команд.

k-й ш а г - вычисление k -го разряда частного.

Выделяются строки, в которых длина значащей части делителя меньше или равна k . Для этого производится опрос поля делителей признаком $\underbrace{00 \dots 0}_{m-k} \underbrace{MM \dots M^*}_k$.

В выделенных строках производится сравнение k старших разрядов делимых $a_{i,m} \dots a_{i,m-k+1}$ с k -разрядными двоичными кодами $b_{i,k} \dots b_{i,1}$, причем для каждой строки, которая

и) Символом М обозначены замаскированные разряды.

впервые включена в процесс обработки, $b_{ik} \dots b_{i1}$ совпадает с делителем; для каждой строки, включенной на $(k-1)$ -м шаге, $b_{ik} = 0$, а $b_{i,k+1} \dots b_{i1}$ совпадает с делителем и т.д.

Сравнение выполняется описанным выше способом. Затем во всех строках, отмеченных метками ОО и ОI, производится вычитание $a_{i,m} \dots a_{i,m-k+1} - b_{ik} \dots b_{i1}$, а в k -й столбец поля частным записывается 1 (значение текущего разряда частного).

После выполнения m -го шага производится округление результатов. Для этого во всех строках сравниваются коды $b_{i,m} \dots b_{i1}$ и $a_{i,m+1} \dots a_{i1}$, и к частному прибавляется 1 в младшем разряде, если строка оказалась помеченной меткой ОО или ОI. При сложении для переносов можно использовать столбец займов. На этом микропрограмма ДЕЛЕНИЕ заканчивается.

Время выполнения микропрограммы ДЕЛЕНИЕ

$$T_d = 5,5m^2 + 17,5 + 3.$$

Необходимые размеры матрицы-накопителя

$$S_d = (3m + 6) \times N.$$

В описанных выше микропрограммах АЛГЕБРАИЧЕСКОЕ СЛОЖЕНИЕ, УМНОЖЕНИЕ, ДЕЛЕНИЕ были использованы только микрокоманды ЗГ, ЧО и ПР. Вообще говоря, эти микрокоманды могут выполняться в ассоциативной матрице, построенной на более простых ассоциативных ячейках (см., например, [3], стр. 17).

Использование других микрокоманд АКП (см. таблицу I), как будет показано ниже, позволяет получить существенное ускорение вычислений.

3.6. Сдвиг вправо (влево)

Массовый сдвиг вправо (влево) выполняется с помощью микрокоманд ИЗП (ПЗЛ). Пусть исходная информация записана во всех нечетных столбцах матрицы. Четные столбцы, расположенные непосредственно справа (слева) от них, очищаются перед сдвигом и используются для промежуточного хранения информации.

Сдвиг выполняется в два этапа. На первом этапе в качестве s -го поля поиска-записи принимается при сдвиге вправо пара столбцов с номерами $2s-1$ и $2s$ ($1 \leq s \leq \frac{n}{2}$), при сдвиге вле-

во - пара столбцов с номерами $2s+1$ и $2s$. Левые (правые) столбцы каждого поля рассматриваются как зоны поиска, а правые (левые) - как зоны записи. Поиск ведется во всех зонах признаком 1.

После выполнения микрокоманды ПЭП (ПЭЛ) содержимое всех левых (правых) столбцов оказывается сдвинутым параллельно в соседние (промежуточные) столбцы. Затем производится очистка всех зон поиска.

На втором этапе в качестве 3 -го поля принимается при сдвиге вправо пара столбцов с номерами $2s$ и $2s+1$ ($1 \leq s \leq \frac{N}{2}$), при сдвиге влево - пара столбцов с номерами $2s$ и $2s-1$. После выполнения микрокоманды ПЭП (ПЭЛ) исходная информация оказывается сдвинутой на 2 столбца вправо (влево).

Время выполнения микропрограммы СДВИГ ВПРАВО (ВЛЕВО) для массива из N m -разрядных слов

$$T_{СП} = T_{СЛ} = 4.$$

Необходимые размеры матрицы-наконечника

$$S_{СП} = S_{СЛ} = 2m \times N.$$

Если необходимо сдвинуть не все, а только некоторые из строк, то на шины u тех строк, содержимое которых не сдвигается, подается константа 0.

Заметим также, что ширина поля между сдвигаемыми столбцами может быть больше единицы и различной для разных s , но всегда для выполнения сдвига между сдвигаемыми столбцами должен существовать очищенный столбец, который используется для промежуточного хранения информации.

3.7. Быстрое умножение

Выше была описана микропрограмма алгебраического сложения. Эта микропрограмма позволяет выполнять алгебраическое сложение двух массивов чисел за $27m+12$ микрокоманд. Если все слагаемые имеют одинаковые знаки, то сложение двух массивов можно выполнить за $6m$ микрокоманд (см. подпрограмму $4+3$ микропрограммы АЛГЕБРАИЧЕСКОЕ СЛОЖЕНИЕ). Поскольку эта оценка не зависит от длины массивов, то, вообще говоря, такое (со-

социативное) сложение весьма эффективно по сравнению со сложением на обычных сумматорах.

Однако, если при выполнении на ассоциативном процессоре операций умножения и комбинированных операций (например, скалярного умножения векторов) использовать для сложения описанные выше микропрограммы, то получается оценка времени порядка m^2 . Покажем, что порядок оценки можно снизить, если использовать при массовом умножении известный метод сокращенного сложения с хранением переносов (см., например, [4]). Предполагается вычисление произведений с удвоенной точностью.

Пусть требуется вычислить покомпонентное произведение векторов A и B , и каждая компонента этих векторов есть m -разрядное двоичное число с фиксированной запятой.

Распределение памяти АП при выполнении этой операции описываемым способом приведено на рис. 6.

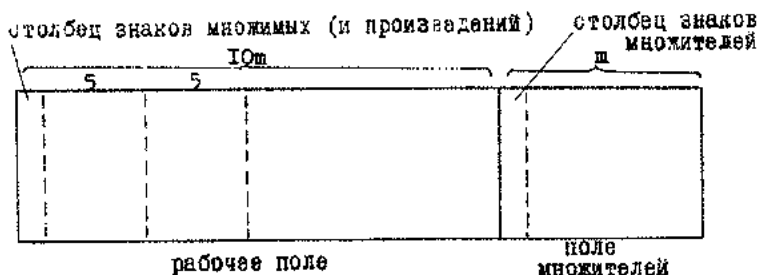


Рис. 6

Перед началом умножения в поле множителей записывается вектор B , причем в i -ю строку поля записывается i -я компонента вектора B .

Рабочее поле разбивается на $2m$ зон по 5 столбцов.

Обозначим пятёрку ячеек i -й строки j -й зоны символами $R_{ij}^2, S_{ij}^2, S_{ij}^1, R_{ij}^1, C_{ij}$. Перед началом умножения в рабочее поле записывается вектор A , причем в ячейки C_{ij} записываются j -е разряды i -й компоненты вектора A . Таким образом, каждая компонента оказывается записанной в ячейках $C_{i,m+1}, \dots, C_{i,2m}$ рабочего поля, причем младшие разряды — справа. Знаки произведений определяются так же, как в описанной выше микропрограмме умножения, и замещают знаки множимых в знаковом столбце рабочего поля.

Умножение чисел $A = a_1 \dots a_m$ и $B = b_1 \dots b_m$ равносильно вычислению суммы

$$\sum_{\ell=1}^m b_{m-\ell+1} (\underbrace{0 \dots 0}_{m-\ell+1} a_1 \dots a_m \underbrace{0 \dots 0}_{\ell-1}). \quad (7)$$

Очевидно, последовательные значения вторых сомножителей в выражении (7) могут быть получены, если после каждого шага умножения выполнять в рабочем поле сдвиг содержимых ячеек C_{ij} в ячейки $C_{i,j-1}$. Этот сдвиг осуществляется микропрограммой СДВИГ ВЛЕВО, причем в качестве вспомогательных ячеек для промежуточного хранения информации используются ячейки R_{ij}^e .

Алгебраическое покомпонентное умножение векторов A и B состоит из m циклов сокращенного сложения и заключительной ассимиляции переносов.

ℓ -й цикл заключается в следующем. В соответствии с алгоритмом сокращенного сложения в процессе вычислений результат хранится в виде двух кодов: S - код суммы по $\text{mod } 2$ и R - код переносов в соседние старшие разряды. В каждом цикле сложения в каждом разряде суммируются три кода: S_{ij}^1, R_{ij}^1 - коды предыдущих результатов и C_{ij} - код очередного слагаемого и вырабатываются новые результаты S_{ij}^2 и R_{ij}^2 . Ячейки $S_{ij}^1, R_{ij}^1, C_{ij}$ всех зон опрашиваются параллельно (в соответствии с таблицей 5), а запись ведется в ячейки S_{ij}^2 и R_{ij}^2 .

Таблица 5.

S_{ij}^1	R_{ij}^1	C_{ij}	S_{ij}^2	R_{ij}^2
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Между двумя последовательными циклами сложения в каждой зоне производится необходимый для подготовки следующего цикла

сдвиг содержимых ячеек S_{ij}^2 и R_{ij}^2 соответственно в ячейки S_{ij}^1 и R_{ij}^1 . Сдвиг S_{ij}^2 в S_{ij}^1 производится микрокомандой ПЭП, а R_{ij}^2 в R_{ij}^1 - микрокомандой ПЭЛ.

Кроме того, между циклами производится сдвиг влево содержимых ячеек C_{ij} (в ячейки $C_{i,j-1}$), то есть очередной сдвиг множителей.

Перед началом l -го цикла ($l=1, \dots, m$) производится анализ l -х разрядов множителей.

Полученное в результате анализа содержимое регистра индикаторов передается через регистр управляющих триггеров на входы u_{ij} матрицы-накопителя. В результате сложение фактически выполняется только в тех строках рабочего поля, которым соответствуют единицы обрабатываемого в данном цикле разряда множителей.

После m -го цикла в ячейках $S_{i1}^1, \dots, S_{i,m}^1$ и $R_{i1}^1, \dots, R_{i,m}^1$, очевидно, находятся коды S_i и R_i произведения $A_i B_i$. Для окончательного формирования произведения необходимо выполнить заключительную ассимиляцию переносов (полное сложение кодов S_i и R_i).

В описываемом АКП эта операция осуществляется следующим образом. Сначала очищаются все ячейки C_{ij} рабочего поля, затем выполняется m циклов сокращенного сложения, причем во всех строках матрицы $u = 1$. В результате в ячейках $S_{i1}^1, \dots, S_{i,m}^1$ получается истинное значение произведения $A_i B_i$.

Время выполнения микропрограммы БЫСТРОЕ УМНОЖЕНИЕ

$$T_{бу} = 18m + 8.$$

Необходимые размеры матрицы-накопителя

$$S_{бу} = (11m + 2) \times N.$$

Реализация сложения с хранением переносов в ассоциативной структуре была рассмотрена еще в работе [5], где получена оценка времени умножения $23m + 34$. Однако предложенный в работе [5] ассоциативный процессор должен иметь индивидуальный комплект управляющих схем для каждой из N пар сомножителей ("группы"). Рассматриваемый здесь АКП обеспечивает умножение N пар чисел с оценкой $18m + 8$ при использовании только одного комплекта управляющих схем для всех N пар сомножителей.

Выше была рассмотрена микропрограмма быстрого умножения для чисел с фиксированной запятой. Отметим, что нетрудно организовать аналогичный процесс быстрого умножения для чисел с плавающей запятой. При этом сложение порядков может производиться параллельно с умножением мантиис. Следовательно, временная оценка не увеличится.

Описанный процесс быстрого массового умножения можно использовать при реализации различных групповых операций. В качестве примера рассмотрим организацию параллельного вычисления на АКП попарных скалярных произведений двух массивов векторов. На рис. 7 приведено распределение памяти. В отличие от описанного выше умножения двух векторов, здесь аргументами являются два массива векторов: массив векторов-множимых A_i и массив векторов-множителей B_i .

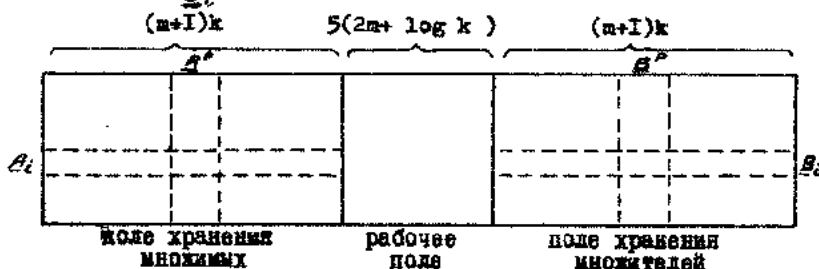


Рис. 7

Алгоритм вычисления попарных скалярных произведений состоит из k шагов (k — размерность векторов) и заключительной ассимиляции переносов.

Сначала рассмотрим случай, когда все компоненты — положительные числа.

p -й шаг ($p = 1, \dots, k$) заключается в том, что в рабочем поле выполняется программа покомпонентного умножения векторов A^p и B^p (но без заключительной ассимиляции переносов). Здесь $A^p(B^p)$ — столбец p -х компонент всех векторов массива $\hat{A}(\hat{B})$. Перед каждым шагом все ячейки C_{ij} рабочего поля очищаются, и в эти ячейки переписываются компоненты очередного вектора массива \hat{A} . После p -го шага в ячейках $S_{i1}^1, \dots, S_{i, (2m + \lceil \log_2 k \rceil)}^1$ и $R_{i1}^1, \dots, R_{i, (2m + \lceil \log_2 k \rceil)}^1$ находится код S_i и R_i суммы произведений $\sum_{j=1}^k a_{ij} b_{ij}$.

*) $\lceil \log_2 k \rceil$ — целая часть числа $\log_2 k$.

Для окончательного формирования массива скалярных произведений необходимо (после k -го шага) в каждой строке рабочего поля выполнить заключительную ассимиляцию переносов.

После этого результат (скалярное произведение $A_i \times B_i$) содержится в ячейках $S'_{i1}, \dots, S'_{i(2m + \lceil \log_2 P \rceil)}$ рабочего поля.

В общем случае, когда компоненты векторов A_i и B_i имеют произвольные знаки, можно использовать сложение в дополнительных кодах. Алгоритм сокращенного сложения в АКП для чисел в дополнительных кодах по существу не отличается от описанного выше для прямых кодов. Знаки произведений определяются на каждом шаге путем анализа знаков очередных компонент и для тех строк, в которых знак произведения отрицательный, в рабочем поле вместо прямого кода множимого записывается его обратный код.

Время выполнения микропрограммы вычисления попарных скалярных произведений двух массивов векторов

$$T_{\text{счп}} = 13m^2 + 11m + 6 \lceil \log_2 k \rceil.$$

Необходимые размеры матрицы-накопителя

$$S_{\text{счп}} = \{5(2m + \lceil \log_2 k \rceil) + 3km - 1\} \times N.$$

Аналогично можно вычислять на АКП последовательно столбцы произведения двух матриц.

Пусть требуется вычислить произведение двух матриц $\hat{Q} = \hat{Q}_1 \times \hat{Q}_2$. Рассмотрим для простоты умножение квадратных матриц размерами $k \times k$. Распределение памяти такое же, как при вычислении попарных скалярных произведений (см. рис. 7).

Массив \hat{A} представляет матрицу \hat{Q}_1 , а массив \hat{B} — транспонированную матрицу $\hat{Q}_2(\hat{Q}_2^T)$. В массиве \hat{B} необходимо использовать вспомогательную $(N + 1)$ -ю строку, индикатор которой может быть по сигналу из устройства управления соединен параллельно со всеми входами u_{i1} . Вычисление \hat{Q}^p (\hat{Q}^p — p -й столбец матрицы \hat{Q}) заключается в том, что в рабочем поле выполняется программа вычисления попарных скалярных произведений массива векторов \hat{A} и массива из k одинаковых векторов $(\hat{Q}_2^T)_p$. Чтобы не расходовать лишнюю память для хранения массива из k одинаковых векторов $(\hat{Q}_2^T)_p$, вектор $(\hat{Q}_2^T)_p$

предварительно считывается во вспомогательную строку массива \vec{B} .

По окончании цикла результат (p -й столбец матрицы \hat{A}) содержится в рабочем поле, причем i -я его компонента находится в ячейках $S_{i1}^1, \dots, S_{i, (2m + \lceil \log_2 k \rceil)}$.

Перед вычислением следующего ($p+1$)-го столбца матрицы \hat{A} необходимо выполнить считывание во внешнее ЗУ p -го столбца, вычисленного в предыдущем цикле, а затем очистить все ячейки рабочего поля и вспомогательную строку массива \vec{B} .

Время выполнения микропрограммы вычисления произведения матриц

$$T_{PM} = (13m^2 + 12m + 6\lceil \log_2 k \rceil + 2) \times k.$$

Необходимые размеры матрицы-накопителя

$$S_{PM} = \{5(2m + \lceil \log_2 k \rceil) + 2k(m+1)\} (k+1) \times N.$$

Л и т е р а т у р а

1. ФЕТ Я.И. Вычислительные среды с простыми ячейками. - В кн.: Вычислительные системы. Вып.54, Новосибирск, 1973, с. 48-66.
2. Fuller R.H. Associative parallel processing. AFIPS Confer. Proc., 1967 SJCC, v.30, p. 471-475.
3. ПРАНГИШВИЛИ И.В. и др. Однородные микровзлектронные ассоциативные процессоры. М., "Советское радио", 1973.
4. ИЛОВАЙСКИЙ И.В., ФЕТ Я.И. Об одном способе организации бы-
стродействующего арифметического устройства. - "Изв. АН
СССР. Сер. технич. наук.", 1966, № 6, вып. 2, с. 60-68.
5. Crane G.A., Githens J.A. Bulk processing in distributed logic
memory. IEEE Trans., 1965, EC-14, N 2, p. 186-196.

Поступила в ред.-изд. отд.

10. IX. 1974 г.