Вычисление математических функций на основе разрядно-параллельных схем[1]

В.М. Хачумов

Аннотация. Показана возможность разрядного представления математических функций для их быстрого вычисления на основе различных подходов. Для быстрого вычисления функций предложены разрядно-параллельные схемы, ориентированные на применение многовходовых сумматоров, приведена архитектура специализированного геометрического процессорного элемента. Даны оценки вычислительной сложности. Рассмотренная разрядная технология может быть применена в бортовых вычислителях, использующих технику FPGA для обработки сигналов и выработки управлений.

Ключевые слова: математические функции, разрядные схемы, параллельные вычисления, алгоритмы CORDIC, многовходовый сумматор, специализированный процессорный элемент.

Введение

Одним из перспективных направлений в авиакосмическом приборостроении является создание специализированных бортовых устройств управления. Одно из важнейших требований к встраиваемым вычислителям в авиакосмических системах – малое энергопотребление, возможность проведения быстрых грубых вычислений, малые затраты памяти. В качестве математического обеспечения подобных систем в последнее время все чаще применяют алгоритмы семейства CORDIC (COordinate Rotation DIgital Computer) [1], которые характеризуются отсутствием «длинных» операций при вычислении функций, простотой аппаратной реализации, небольшими объемами необходимой памяти, возможностью гибкого регулирования точности и времени вычислений. Алгоритмы CORDIC ранее применялись в специализированных устройствах [2, 3], графических калькуляторах [4] и процессорах [5]. Укажем на некоторые современные приложения CORDIC-алгоритмов, которые могут быть полезны в бортовых вычислителях: быстрые (оптимизированные) вычисления тригонометрических функций при одновременном уменьшении сложности и задержек в аппаратных средствах на основе техники FPGA [6, 7]; построение беспроводных сетей WLAN [8]; построение цифровых синтезаторов, модуляторов и конвейеров [9], цифровая обработка сигналов [10]. Компанией Xilinx созданы приложения в виде библиотеки для DSP-процессоров [11], разработана документация на библиотеку CORDIC-алгоритмов [12]. Отметим интерес и отечественных разработчиков ПЛИС (FPGA) к этой технологии. Так, в публикации [13] дается описание архитектуры элементов, предназначенных для вычисления значений тригонометрических и гиперболических функций средствами генератора параметризированных модулей “CORE Generator” и ядра в составе проектируемых устройств. Алгоритмы CORDIC привлекли внимание разработчика электронных схем “Lattice Semiconductor” (NASDAQ: LSCC, США). Построенный им

[1] Работа выполнена при финансовой поддержке проектов РФФИ №16-07-00096а и №15-29-06945 офи_м

специализированный процессор для вычисления функций имеет CORDIC IP-ядро, которое настраивается и поддерживает несколько функций, в том числе «поворот» (“rotation”), «вектор» (“vectoring”), sin(x), cos(x) и arctan(x) и другие функции [14]. В связи с быстрым развитием и активным внедрением в авиакосмические приложения отечественной аппаратной платформы «Мультикор» [15] становится актуальной задача встраивания алгоритмов CORDIC в ее архитектуру, а именно, в состав математического обеспечения DSP-ядра для обеспечения базовых арифметических операций, соответствующих стандарту IEEE-754.

Несмотря на отмеченные преимущества, алгоритмы CORDIC имеют ряд недостатков, не позволяющих эффективно использовать их в параллельных системах, отличных от конвейерных, – итеративный характер вычислений, необходимость коррекции результата и др. Автор настоящей работы на протяжении ряда лет проводил исследования по разрядно-параллельному представлению алгоритмов CORDIC, принципиально отличному от общепринятых представлений. Отметим серию работ [16-26], в которых были изложены основные положения этого подхода. Идея разрядно-аналоговых вычислительных схем принадлежит Г.Е. Пухову [27], но развития теория не получила, очевидно, из-за ограниченности набора реализуемых функций.

Исходя из повышенного интереса к организации встроенного математического обеспечения для систем управления, автор настоящей работы ставит своей задачей обобщение некоторых полученных ранее результатов по построению разрядно-параллельных вычислительных схем различными способами.

1. Разрядно-параллельные вычисления в системах реального времени

Основой разрядного исчисления служат понятия разрядного вектора и матрицы [27]. Вектор $x = 0.x_1x_2...x_n$, компонентами которого являются разряды двоичного числа $x_j \in \{0,1\}$, $j = 1,..,n$, $x_1 = 1$, назовем нормализованным разрядным вектором. Задача отображения результата функционального преобразования $y = f(x)$, где $y = 0.y_1y_2...y_m$, в разрядно-параллельную схему заключается в представлении каждого i-го разряда числа y в виде разрядной формулы: $y_i = f_i(x_1, x_2,..., x_n)$, $y_i \in \{0,1\}$, $i = 1,..,m$, $y_1 = 1$. В ряде случаев, характерных для алгоритмов CORDIC, разряд математической функции может быть представлен в виде функции от связанных с разрядами аргумента переменными, например операторами Волдера $y_i = \gamma_i(\varepsilon_1, \varepsilon_2,..., \varepsilon_n)$. Такие схемы также будем называть разрядно-параллельными при условии одновременной доступности всех операторов.

1.1. Вычисление обратной функции

Рассмотрим операцию $y = 1/x$. Здесь $x = (x)2^{-K}$ – положительное двоичное число, представленное в нормализованном виде, $(x) = (x_1x_2...x_m)$ – мантисса, K – порядок числа x, x_i – значение i-го разряда мантиссы $(x_i \in \{0,1\})$ с собственным весом 2^{m-i}, причем $x_1 = 1$, m – длина разрядной сетки. Результат нахождения обратной величины представляется в виде $y = (y)2^{-[2(m-1)-K]}$, где $(y) = (y_1y_2...y_m)$ – мантисса числа y. Связь между y и x устанавливается с помощью разрядного выражения [27]: $y = [x]^{-1}$, где y – разрядный вектор, $[x]$ – квадратная разрядная матрица, $[x]^{-1}$ – обратная разрядная матрица. Разрядная матрица образуется из разрядных векторов. Для получения обратной разрядной матрицы могут быть использованы все известные методы обращения. Однако в данном конкретном случае удается установить следующие соотношения:

$$
\begin{aligned}
&y_1 = x_1 = 1,\\
&y_2 = -y_1 x_2,\\
&y_3 = -y_1 x_3 - y_2 x_2,\\
&y_4 = -y_1 x_4 - y_2 x_3 - y_3 x_2,\\
&\ldots\ldots\ldots\ldots\\
&y_m = -\sum_{i=1}^{m-1} y_i x_{(m+1)-i}.
\end{aligned}
\qquad (1)
$$

Рассмотрим, без потери общности, порядок подготовки и организации вычислений для $m = 8$. В результате последовательных подстановок, учитывая, что $x_i^2 = x_i$ при условии $x_i \in \{0,1\}$, получим

$$
\begin{aligned}
&y_1 = 1,\\
&y_2 = -x_2,\\
&y_3 = -x_3 + x_2,\\
&y_4 = -x_4 + 2x_2x_3 - x_2,\\
&y_5 = -x_5 + 2x_2x_4 + x_3 - 3x_2x_3 + x_2,\\
&y_6 = -x_6 + 2x_2x_5 + 2x_3x_4 - 3x_2x_4 + x_2x_3 - x_2,\\
&y_7 = -x_7 + 2x_2x_6 + 2x_3x_5 - 3x_2x_5 + x_4 - 6x_2x_3x_4 + 4x_2x_4 - x_3 + x_2x_3 + x_2,\\
&y_8 = -x_8 + 2x_2x_7 + 2x_3x_6 - 3x_2x_6 + 2x_4x_5 - 6x_2x_3x_5 + 4x_2x_5 - 8x_2x_4 - 3x_3x_4 + 12x_2x_3x_4 - x_2,
\end{aligned}
\qquad (2)
$$

где y_i записаны через соответствующие разрядные коэффициенты числа x. Для приведения системы (2) к виду, пригодному для выполнения операции группового суммирования, следует переместить соответствующие элементы из одного разряда в другой с учетом их весов. Таким образом, имеем

$$
\begin{aligned}
&y_1 = 1,\\
&y_2 = -x_2,\\
&y_3 = -x_3 + x_2,\\
&y_4 = -x_4 - x_2,\\
&y_5 = -x_5 + x_3 + x_2x_3 + x_2 + x_2x_5 + x_3x_4,\\
&y_6 = -x_6 + x_2x_4 + x_2x_3 - x_2 + x_2x_6 + x_3x_5 - x_2x_3x_5,\\
&y_7 = -x_7 - x_2x_5 + x_4 - x_3 + x_2x_3 + x_2 + x_2x_7 + x_3x_6 - x_2x_6 + x_4x_5 - x_2x_3x_5 - x_3x_4,\\
&y_8 = -x_8 - x_2x_6 - x_3x_4 - x_2.
\end{aligned}
\qquad (3)
$$

Вычислительная схема является разрядно-параллельной. В общем случае $y_i \notin \{0,1\}$, поэтому для получения окончательного результата необходимо выполнить суммирование всех элементов с учетом весов.

Установим соответствие между схемой (3) и соотношениями операции CORDIC для той же операции. Аргумент функции $y = 1/x$ должен быть представлен в виде $x = 1/\prod_{i=1}^{n}(1+\varepsilon_i 2^{-i})$.

Здесь ε_i, $\varepsilon_i \in \{+1,-1\}$ – величины, называемые «операторы Волдера» [1, 2], которые вычисляют по следующей формуле $\varepsilon_i = -sign\left(x\prod_{k=1}^{i-1}(1+\varepsilon_k 2^{-k}) - 1 \right)$.

С учетом операторов результат операции получают следующим образом:

$$y = \prod_{i=1}^{n}(1+\varepsilon_i 2^{-i}). \qquad (4)$$

Последовательно раскрывая произведение (4), y можно представить в виде

$$\begin{aligned}
y_0 &= 1,\\
y_1 &= \varepsilon_1,\\
y_2 &= \varepsilon_2,\\
y_3 &= \varepsilon_3 + \varepsilon_1\varepsilon_2,\\
y_4 &= \varepsilon_4 + \varepsilon_1\varepsilon_3,\\
y_5 &= \varepsilon_5 + \varepsilon_1\varepsilon_4 + \varepsilon_2\varepsilon_3,\\
y_6 &= \varepsilon_6 + \varepsilon_1\varepsilon_5 + \varepsilon_2\varepsilon_4 + \varepsilon_1\varepsilon_2\varepsilon_3,\\
y_7 &= \varepsilon_7 + \varepsilon_1\varepsilon_6 + \varepsilon_2\varepsilon_5 + \varepsilon_3\varepsilon_4 + \varepsilon_1\varepsilon_2\varepsilon_4,\\
y_8 &= \varepsilon_8 + \varepsilon_1\varepsilon_7 + \varepsilon_2\varepsilon_6 + \varepsilon_3\varepsilon_5 + \varepsilon_1\varepsilon_2\varepsilon_5 + \varepsilon_1\varepsilon_3\varepsilon_4.
\end{aligned} \qquad (5)$$

Схема (5) выражает результат операции через операторы Волдера, которые могут быть вычислены средствами CORDIC только последовательно. В принципе, операторы можно получить, приравнивая (3) и (5):

$$\begin{aligned}
\varepsilon_0 &= x_1 = 1,\\
\varepsilon_1 &= -x_2,\\
\varepsilon_2 &= -x_3 + x_2,\\
\varepsilon_3 &= -x_2x_3 - x_4,\\
\varepsilon_4 &= -x_2x_4 - x_5 + x_3 + x_2x_5 + x_3x_4 + x_2,\\
\varepsilon_5 &= 2x_2x_3 - x_6 + x_2x_4 + x_2x_6 + x_3x_5 - x_2x_3x_5 + x_2x_3x_4 - x_3x_4,\\
\varepsilon_6 &= 3x_2x_3 - x_2x_3x_4 + x_2x_4 - x_7 - x_2x_5 - x_3x_5 + x_4 + x_2x_7 + x_3x_6 - x_2x_6 + x_4x_5,\\
\varepsilon_7 &= 4x_2x_3 - x_2x_5 - 2x_2x_6 + 2x_2x_4x_5 - x_3x_6 + x_3x_5 + 2x_2x_3x_6 + 2x_2x_3x_4 - 2x_2x_3x_5 - x_4x_5 - x_8.
\end{aligned} \qquad (6)$$

Схема (6) демонстрирует принципиальную возможность получения значений псевдооператоров непосредственно по двоичному аргументу x, однако из практических соображений достаточно воспользоваться готовой схемой (3).

1.2. Извлечение квадратного корня

Пусть $y = 0.0y_1y_2...y_n$ – положительное двоичное число, представленное в нормализованном виде $0.25 < y < 0.5$. Связь между y и x устанавливается с помощью разрядного выражения [27]: $Y = [X]\cdot X$, где $[X]$ – квадратная разрядная матрица (размерности $n \times n$); Y, X – разрядные вектора.. Следуя методике вычисления разрядных коэффициентов [27], можно получить следующие соотношения (для $n = 8$):

$$Y=\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \end{pmatrix},\ X=\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{pmatrix},\quad [X]=\begin{bmatrix} x_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ x_2 & x_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ x_3 & x_2 & x_1 & 0 & 0 & 0 & 0 & 0 \\ x_4 & x_3 & x_2 & x_1 & 0 & 0 & 0 & 0 \\ x_5 & x_4 & x_3 & x_2 & x_1 & 0 & 0 & 0 \\ x_6 & x_5 & x_4 & x_3 & x_2 & x_1 & 0 & 0 \\ x_7 & x_6 & x_5 & x_4 & x_3 & x_2 & x_1 & 0 \\ x_8 & x_7 & x_6 & x_5 & x_4 & x_3 & x_2 & x_1 \end{bmatrix},$$

$$\begin{aligned} &y_1 = x_1 = 1, \\ &y_2 = 2x_2, \\ &y_3 = 2x_3 + x_2^2, \\ &y_4 = 2x_4 + 2x_2x_3, \\ &y_5 = 2x_5 + 2x_2x_4 + x_3^2, \\ &y_6 = 2x_6 + 2x_2x_5 + 2x_3x_4, \\ &y_7 = 2x_7 + 2x_2x_6 + 2x_3x_5 + x_4^2, \\ &y_8 = 2x_8 + 2x_2x_7 + 2x_3x_6 + 2x_4x_5. \end{aligned} \tag{7}$$

Для двоичной системы счисления справедливо: $x_i^2 = x_i$. С учетом этого после преобразований получим разрядно-параллельную вычислительную схему

$$\begin{aligned} &x_1 = y_1 = 1, \\ &x_2 = 0, \\ &x_3 = y_2, \\ &x_4 = y_3, \\ &x_5 = y_4, \\ &x_6 = y_5 - y_2 - y_2y_3, \\ &x_7 = y_6 - y_2y_4, \\ &x_8 = y_7 + y_2 - y_3 - y_2y_5 - y_3y_4 + y_2y_3, \\ &x_9 = y_8 - y_3y_5 + y_2y_4 + y_2y_3, \\ &x_{10} = y_2y_4 - y_2y_6 - y_4 - y_2y_7 - y_3y_6 - y_4y_5 - y_2y_3 - y_2 + y_2y_3y_5, \\ &x_{11} = y_3 + y_2y_3y_5 + y_2y_6 + y_3y_4, \\ &x_{12} = y_2y_6 + y_3y_4 - y_2, \\ &x_{13} = y_2y_3 - y_2y_5 + y_2y_3y_4, \\ &x_{14} = y_2y_3, \\ &x_{15} = y_2y_3 - y_2, \\ &x_{16} = y_2y_4 + y_2y_3. \end{aligned} \tag{8}$$

В соответствии с подходом CORDIC [1, 2], для той же операции $\varepsilon_i = -sign\left(Y\prod_{k=1}^{i-1}(1+\varepsilon_k 2^{-k})^2 - 1 \right)$, $x = \sqrt{y} = y\prod_{i=1}^{n}(1+\varepsilon_i 2^{-i})$, откуда следует

$$
\begin{aligned}
&x_1 = y_1 = 1,\\
&x_2 = y_2 + y_1\varepsilon_1,\\
&x_3 = y_3 + y_2\varepsilon_1 + y_1\varepsilon_2,\\
&x_4 = y_4 + y_3\varepsilon_1 + y_2\varepsilon_2 + y_1(\varepsilon_3 + \varepsilon_1\varepsilon_2),\\
&x_5 = y_5 + y_4\varepsilon_1 + y_3\varepsilon_2 + y_2(\varepsilon_3 + \varepsilon_1\varepsilon_2) + y_1(\varepsilon_4 + \varepsilon_1\varepsilon_3),\\
&x_6 = y_6 + y_5\varepsilon_1 + y_4\varepsilon_2 + y_3(\varepsilon_3 + \varepsilon_1\varepsilon_2) + y_2(\varepsilon_4 + \varepsilon_1\varepsilon_3) + y_1(\varepsilon_5 + \varepsilon_1\varepsilon_4 + \varepsilon_2\varepsilon_3),\\
&x_7 = y_7 + y_6\varepsilon_1 + y_5\varepsilon_2 + y_4(\varepsilon_3 + \varepsilon_1\varepsilon_2) + y_3(\varepsilon_4 + \varepsilon_1\varepsilon_3) +\\
&\quad + y_2(\varepsilon_5 + \varepsilon_1\varepsilon_4 + \varepsilon_2\varepsilon_3) + y_1(\varepsilon_6 + \varepsilon_1\varepsilon_5 + \varepsilon_2\varepsilon_4 + \varepsilon_1\varepsilon_2\varepsilon_3),\\
&x_8 = y_8 + y_7\varepsilon_1 + y_6\varepsilon_2 + y_5(\varepsilon_3 + \varepsilon_1\varepsilon_2) + y_4(\varepsilon_4 + \varepsilon_1\varepsilon_3) +\\
&\quad + y_3(\varepsilon_5 + \varepsilon_1\varepsilon_4 + \varepsilon_2\varepsilon_3) + y_2(\varepsilon_6 + \varepsilon_1\varepsilon_5 + \varepsilon_2\varepsilon_4 + \varepsilon_1\varepsilon_2\varepsilon_3) + y_1(\varepsilon_7 + \varepsilon_1\varepsilon_6 + \varepsilon_2\varepsilon_5 + \varepsilon_3\varepsilon_4 + \varepsilon_1\varepsilon_2\varepsilon_4).
\end{aligned}
\tag{9}
$$

Взаимосвязь разрядов результата и операнда можно установить на основе равенства

$$1/y = \prod_{i=1}^{n}(1+\varepsilon_i 2^{-i})^2. \tag{10}$$

Для обратной функции разрядно-параллельная схема была получена ранее. Раскрывая правую часть равенства (10), запишем с учетом принятых ограничений и весов разрядное представление функции $1/y$ (Табл.1).

Табл. 1. Разрядное представление функции $1/y$

2^0	$1+\varepsilon_1$
2^{-1}	$\varepsilon_2 + \varepsilon_1\varepsilon_2$
2^{-2}	$\varepsilon_3 + \varepsilon_1\varepsilon_3 + \varepsilon_1^2$
2^{-3}	$\varepsilon_1^2\varepsilon_2 + \varepsilon_2\varepsilon_3 + \varepsilon_1\varepsilon_2\varepsilon_3 + \varepsilon_4 + \varepsilon_1\varepsilon_4 + \varepsilon_1^2\varepsilon_4$
2^{-4}	$\varepsilon_2^2 + \varepsilon_1\varepsilon_2^2 + \varepsilon_1^2\varepsilon_3 + \varepsilon_2\varepsilon_4 + \varepsilon_1\varepsilon_2\varepsilon_4 + \varepsilon_5 + \varepsilon_1\varepsilon_5$
2^{-5}	$\varepsilon_1^2\varepsilon_2\varepsilon_3 + \varepsilon_1^2\varepsilon_4 + \varepsilon_3\varepsilon_4 + \varepsilon_1\varepsilon_3\varepsilon_4 + \varepsilon_2\varepsilon_5 + \varepsilon_1\varepsilon_2\varepsilon_5 + \varepsilon_6 + \varepsilon_1\varepsilon_6$
2^{-6}	$\varepsilon_2\varepsilon_3 + \varepsilon_1\varepsilon_2^2\varepsilon_3 + \varepsilon_3^2 + \varepsilon_1\varepsilon_3^2 + \varepsilon_1^2\varepsilon_2\varepsilon_4 + \varepsilon_2\varepsilon_3\varepsilon_4 + \varepsilon_1\varepsilon_2\varepsilon_3\varepsilon_4 +$ $+\varepsilon_1^2\varepsilon_5 + \varepsilon_3\varepsilon_5 + \varepsilon_1\varepsilon_3\varepsilon_5 + \varepsilon_2\varepsilon_6 + \varepsilon_1\varepsilon_2\varepsilon_6 + \varepsilon_7 + \varepsilon_1\varepsilon_7$
2^{-7}	$\varepsilon_2\varepsilon_3^2 + \varepsilon_1\varepsilon_2\varepsilon_3 + \varepsilon_2^2\varepsilon_4 + \varepsilon_1\varepsilon_2^2\varepsilon_4 + \varepsilon_1^2\varepsilon_3\varepsilon_4 + \varepsilon_1^2\varepsilon_2\varepsilon_5 + \varepsilon_2\varepsilon_3\varepsilon_5 +$ $+\varepsilon_1^2\varepsilon_6 + \varepsilon_2\varepsilon_7 + \varepsilon_1\varepsilon_2\varepsilon_7 + \varepsilon_8 + \varepsilon_1\varepsilon_8.$

Приравнивая мантиссы полученного результата и результата обратной операции (5), получим значения псевдооператоров для операции извлечения квадратного корня:

$$
\begin{aligned}
&\varepsilon_1 = 0,\\
&\varepsilon_2 = -y_2,\\
&\varepsilon_3 = -y_3 + y_2,\\
&\varepsilon_4 = -y_4 - y_2y_3,\\
&\varepsilon_5 = -y_5 + y_3 + y_2y_5 + y_3y_4 - y_2y_4,
\end{aligned}
$$

$$\varepsilon_6 = -y_6 - y_3y_4 + 2y_2y_3 + y_2y_3y_4 + y_2y_4 - y_2 + y_2y_6 + y_3y_5 - y_2y_3y_5,$$
$$\varepsilon_7 = -y_7 + 4y_2y_3 - y_3y_5 + y_2y_3y_5 - 2y_2y_3y_4 - y_2y_5 - y_2y_6 + y_2y_4 - y_2 + y_4 - y_3 + y_2y_7 + y_3y_6 + y_4y_5.$$

Подставляя полученные псевдооператоры в (9), получим

$$\begin{aligned}
&x_1 = y_1 = 1,\\
&x_2 = y_2,\\
&x_3 = y_3 - y_2,\\
&x_4 = y_4 - y_3,\\
&x_5 = y_5 - y_4 + y_2 - 3y_2y_3,\\
&x_6 = y_6 - y_5 - y_2 - 3y_2y_4 + y_2y_5 + y_3y_4 + y_2y_3,\\
&x_7 = y_7 - y_6 - 2y_2 - y_2y_5 - 3y_3y_4 + 4y_2y_3 + 2y_2y_3y_4 + 2y_2y_4 + y_2y_6 + y_3y_5 - y_2y_3y_5,\\
&x_8 = y_8 - y_7 - 2y_2 - 2y_2y_6 - 3y_3y_5 - 5y_2y_3y_4 + 2y_2y_3y_5 + 2y_3y_4 +\\
&\quad + 3y_2y_4 + 6y_2y_3 + y_2y_7 + y_3y_6 + y_4y_5.
\end{aligned} \qquad (11)$$

Перераспределяя элементы между соседними разрядами, легко получить нормализованную разрядно-параллельную схему.

1.3. Вычисление показательной функции

При вычислении показательной функции $y = a^x$ с основанием a будем считать, что ее аргумент x представляет собой n -разрядное двоичное число. Следовательно, справедливо следующее представление результата в виде произведения

$$y = a^{x_1 2^{-1} + \ldots + x_n 2^{-n}} = \prod_{k=1}^{n} a^{x_k 2^{-k}} = \prod_{k=1}^{n} z_k, \qquad (12)$$

где $z_k = a^{x_k \cdot 2^{-k}}$. Учитывая, что x – вектор, состоящий из нулей и единиц, можно записать

$$z_k = \begin{cases} 1, x_k = 0 \\ \sqrt[2^k]{a}, x_k = 1 \end{cases} = 1 + (\sqrt[2^k]{a} - 1)x_k. \qquad (13)$$

Для определенности положим $a = e$, n =8. Подставляя набор констант

$$\begin{aligned}
&\sqrt[2]{e} = (1.6487...)_{10} = (1.10100110...)_2, \quad \sqrt[4]{e} = (1.2840...)_{10} = (1.01001000...)_2\\
&\sqrt[8]{e} = (1.1331...)_{10} = (1.00100010...)_2 \quad \sqrt[16]{e} = (1.0645...)_{10} = (1.00010000...)_2\\
&\sqrt[32]{e} = (1.0317...)_{10} = (1.00001000...)_2 \quad \sqrt[64]{e} = (1.0157...)_{10} = (1.00000100...)_2\\
&\sqrt[128]{e} = (1.0078...)_{10} = (1.00000010...)_2 \quad \sqrt[256]{e} = (1.0039...)_{10} = (1.00000001...)_2
\end{aligned} \qquad (14)$$

в выражения (13), получим разрядные векторы $z_1,..., z_8$:

$$\begin{aligned}
&z_1 = (1.x_1 0 x_1 00 x_1 x_1 0...)_2\\
&z_2 = (1.0 x_2 00 x_2 000...)_2\\
&z_3 = (1.00 x_3 000 x_3 0...)_2\\
&z_4 = (1.000 x_4 0000...)_2\\
&z_5 = (1.0000 x_5 000...)_2\\
&z_6 = (1.00000 x_6 00...)_2
\end{aligned}$$

$z_7 = (1.000000x_7 0...)_2$

$z_8 = (1.0000000x_8 ...)_2$.

В соответствии с (12) результат $y = 1.y_1 y_2 ... y_8$ представляется в виде следующей схемы:

$$
\begin{aligned}
y_0 &= 1 \\
y_1 &= x_1 \\
y_2 &= x_2 \\
y_3 &= x_1 + x_3 + x_1 x_2 \\
y_4 &= x_4 + x_1 x_3 \\
y_5 &= x_2 + x_5 + x_1 x_2 + x_1 x_4 + x_2 x_3 \\
y_6 &= x_1 + x_6 + x_1 x_2 + x_1 x_3 + x_1 x_5 + x_2 x_4 + x_1 x_2 x_3 \\
y_7 &= x_1 + x_3 + x_7 + x_1 x_2 + x_1 x_4 + x_1 x_6 + x_2 x_5 + x_3 x_4 + x_1 x_2 x_4 \\
y_8 &= x_8 + x_1 x_3 + x_1 x_5 + x_1 x_7 + x_2 x_3 + x_2 x_6 + x_3 x_5 + x_1 x_2 x_3 + x_1 x_3 x_4 + x_1 x_2 x_5 .
\end{aligned}
\qquad (15)
$$

В Табл.2 для сравнения приводятся точные значения экспоненты и вычисленные по разрядным формулам.

Табл. 2. Вычисление экспоненциальной функции

Аргумент x	$\exp(x)$	
	Табличное значение	Разрядная схема
0.1	1.1052	1.0977
0.2	1.2214	1.2148
0.3	1.3499	1.3399
0.4	1.4918	1.4765
0.5	1.6487	1.6484
0.6	1.8221	1.8046
0.7	2.0138	1.9922
0.8	2.2255	2.1992
0.9	2.4596	2.4179

Точность может быть улучшена при переходе к большей разрядности. Данный подход может быть применен для любого другого основания.

В соответствии с подходом CORDIC аргумент функции представляется в виде $x = \sum_{i=1}^{n} \ln(1 + \varepsilon_i 2^{-i})$, знаки операторов находятся из условия $sign(\varepsilon_i) = sign\left(x - \sum_{i=1}^{i-1} \ln(1 + \varepsilon_i 2^{-i}) \right)$, а результат получается в виде $\exp(x) = \prod_{i=1}^{n} (1 + \varepsilon_i 2^{-i})$.

Раскрывая выражение для $\exp(x)$, получим схему поразрядного представления результата:

$$
\begin{aligned}
y_0 &= \varepsilon_0 \\
y_1 &= \varepsilon_1 \\
y_2 &= \varepsilon_2 \\
y_3 &= \varepsilon_3 + \varepsilon_1 \varepsilon_2 \\
y_4 &= \varepsilon_4 + \varepsilon_1 \varepsilon_3 \\
y_5 &= \varepsilon_5 + \varepsilon_1 \varepsilon_4 + \varepsilon_2 \varepsilon_3 \\
y_6 &= \varepsilon_6 + \varepsilon_1 \varepsilon_5 + \varepsilon_2 \varepsilon_4 + \varepsilon_1 \varepsilon_2 \varepsilon_3 \\
y_7 &= \varepsilon_7 + \varepsilon_1 \varepsilon_6 + \varepsilon_2 \varepsilon_5 + \varepsilon_3 \varepsilon_4 + \varepsilon_1 \varepsilon_2 \varepsilon_4 \\
y_8 &= \varepsilon_8 + \varepsilon_1 \varepsilon_7 + \varepsilon_2 \varepsilon_6 + \varepsilon_3 \varepsilon_5 + \varepsilon_1 \varepsilon_2 \varepsilon_5 + \varepsilon_1 \varepsilon_3 \varepsilon_4 .
\end{aligned}
\qquad (16)
$$

В CORDIC имеем проблему одновременного получения значений операторов ε_i. Приравняв выражения при соответствующих коэффициентах в формулах (15) и (16), выразим операторы ε_i через разряды аргумента:

$$\begin{aligned}
&\varepsilon_0 = 1\\
&\varepsilon_1 = x_1\\
&\varepsilon_2 = x_2\\
&\varepsilon_3 = x_1 + x_3\\
&\varepsilon_4 = x_4 - x_1\\
&\varepsilon_5 = x_1 + x_2 + x_5\\
&\varepsilon_6 = x_6 + x_1 x_3\\
&\varepsilon_7 = 2x_1 - x_2 + x_3 + x_7 + x_1 x_2\\
&\varepsilon_8 = x_8 - 2x_1 - 3x_1 x_2 - x_1 x_4.
\end{aligned} \qquad (17)$$

Разумеется, такой результат является полезным только с точки зрения демонстрации принципиальной возможности получения разрядно-параллельных схем на основе CORDIC.

1.4. Логарифмирование

Пусть X – целое положительное двоичное число разрядности n. Представим его в нормализованном виде $x = (x)2^{(n-k)} = (x)2^p$. Здесь (x) – мантисса числа x, $0.5 \le (x) < 1$, n – разрядность, k – число сдвигов для нормализации x, $p = (n-k)$ – величина порядка, предполагается, что запятая фиксирована перед старшим разрядом мантиссы. Тогда для вычисления логарифмической функции $y = \log_a x$ справедлива следующая формула:

$$\log_a(x) = p \cdot \log_a(2) + \log_a(x). \qquad (18)$$

Первая (старшая) часть выражения (18) реализуется стандартными средствами (для основания $a = 2$ умножения не требуется). Пусть (x) – нормализованная мантисса двоичного числа разрядности n. Для нахождения функции воспользуемся представлением [2]:

$$\log_a(x) = -\sum_{i=1}^{n} \log_a(1 + \varepsilon_i 2^{-i}) \qquad (19)$$

Здесь $\varepsilon_i \quad (i = 1,..., n)$ – операторы Волдера, $\varepsilon_i \in \{+1, -1\}$, которые находят по формуле

$$\varepsilon_i = -sign\left[(x)\prod_{j=1}^{i-1}(1 + \varepsilon_j 2^{-j}) - 1 \right],\ i = 2,...,n.$$

Справедливы также соотношения

$$\begin{aligned}
&y_{i+1} = y_i - \varepsilon_i x_i 2^{-i},\\
&x_{i+1} = x_i + \varepsilon_i y_i 2^{-i},\\
&\varepsilon_{i+1} = sign(y_{i+1}),\\
&\varphi_{i+1} = \varphi_i + \log_a(1 + \varepsilon_i 2^{-i}).
\end{aligned}$$

Начальные значения: $x_0 = (x),\ y_0 = 1 - (x),\ \varphi_0 = 0, \varepsilon_0 = 1$.

Конечные значения: $x_n = 1,\ y_n = 0,\ \varphi_n = -\log_a(x)$.

Таким образом, каждому коду (x) можно поставить в соответствие набор операторов $\varepsilon_i \quad (i = 1,..., n)$, хранимых в виде нулей и единиц. Если вычислить заранее и занести в память ло-

гарифмы (натуральные, двоичные, десятичные и т.д.) констант вида $(1+2^{-i})$ и $(1-2^{-i})$, общее число которых для каждого основания составляет $2n$, то вычисление логарифмической функции $y=\log_a(x)$ сведется к операции группового суммирования. Содержимое памяти определяется видом логарифма и требуемой точностью хранения фиксированного числа величин $\log(1+2^{-i})$ и $\log(1-2^{-i})$. Например, при $n=8$, $a=10$ постоянная (для любого основания) часть для некоторых значений x и логарифмов задается Табл. 3 и Табл. 4.

Табл. 3. Фрагмент хранимых величин операторов

x	ε_1	ε_2	ε_3	ε_4	ε_5	ε_6	ε_7	ε_8
0.5	1	1	1	-1	1	-1	-1	1
0.6	1	1	-1	1	-1	-1	1	1
0.7	1	-1	1	1	1	1	1	1
0.8	1	-1	1	1	1	1	1	-1

Табл. 4. Хранимые величины логарифмов $\lg(1+2^{-i})$ и $\lg(1-2^{-i})$

I	$(1+2^{-i})$	$\lg(1+2^{-i})$	$(1-2^{-i})$	$\lg(1-2^{-i})$
1	1.500000	0.1760910	0.500000	-0.3010300
2	1.250000	0.0969100	0.750000	-0.1249390
3	1.125000	0.0511526	0.875000	-0.0579920
4	1.062500	0.0263290	0.937500	-0.0280287
5	1.031250	0.0133640	0.968750	-0.0137883
6	1.015625	0.0067334	0.984375	-0.0068394
7	1.007812	0.0033795	0.992187	-0.0034065
8	1.003906	0.0016930	0.996094	-0.0016997

Таким образом, для входного x из Табл.3 выбираются операторы ε_i, для каждого из которых извлекается одно из чисел в табл.4, в соответствии со знаком оператора.

Рассмотрим пример вычисления десятичного логарифма.

Пусть $x=00001000_{(2)}=8_{(10)}$. После нормализации имеем $x=(.10000000_{(2)})2^4=(.5_{(10)})2^4$, т.е. $(x)=0.5_{(10)}$. Пользуясь таблицами, получим с учетом реальной разрядности хранимых операндов, не превышающей при n = 8 и двух дополнительных двоичных разрядов трех десятичных цифр:

$$\lg x=4\lg 2-(0.176+0.097+0.051+0.013+0.002)+(0.028+0.007+0.003)=1.204-0.300=0.904.$$

Точное значение: $\lg x=0.903$.

Для логарифмической функции нецелесообразно строить разрядно-параллельную вычислительную схему. Здесь достаточно иметь возможность параллельного считывания и суммирования данных хранимых в памяти.

1.5. Вычисление тригонометрических функций $\sin(\varphi),\cos(\varphi)$

В соответствии с CORDIC для вычисления пары указанных тригонометрических функций используются следующие рекуррентные соотношения:

Этап 1:
$$\theta_{i+1}=\theta_i-\varepsilon_i\cdot arctg(2^{-i})$$
$$sign(\varepsilon_i)=sign(\theta_i). \qquad (20)$$

Этап 2:
$$y_{i+1}=y_i-\varepsilon_i 2^{-i}x_i$$
$$x_{i+1}=x_i+\varepsilon_i 2^{-i}y_i,$$

Начальные условия: $\theta_0 = \varphi, y_0 = 0, x_0 = 1/k$. Результат: $y_n = -\sin(\varphi), x_n = \cos(\varphi)$

Основным недостатком данных итерационных формул является необходимость коррекции результата. Коэффициент деформации вычисляется по формуле $k = \prod_{i=0}^{n} \sqrt{1 + 2^{-2i}}$, причем для числа итераций $n \geq 10$ $k \approx 1.65$.

Для компенсации деформации может быть предложен следующий способ. В качестве начального значения примем обратную величину $k^{-1} \approx 0.60725 = (0.10011011...)_2$. Последовательно раскрывая формулы для второго этапа вычислений, получим следующие разрядно-параллельные вычислительные схемы.

Для синуса:

$$
\begin{aligned}
y_0 &= 0 \\
y_1 &= \varepsilon_0 \\
y_2 &= \varepsilon_1 \\
y_3 &= \varepsilon_2 \\
y_4 &= \varepsilon_0 + \varepsilon_3 - \varepsilon_0\varepsilon_1\varepsilon_2 \\
y_5 &= \varepsilon_0 + \varepsilon_1 + \varepsilon_4 - \varepsilon_0\varepsilon_1\varepsilon_3 \\
y_6 &= \varepsilon_1 + \varepsilon_2 + \varepsilon_5 - \varepsilon_0\varepsilon_1\varepsilon_4 - \varepsilon_0\varepsilon_2\varepsilon_3 \\
y_7 &= \varepsilon_0 + \varepsilon_2 + \varepsilon_3 + \varepsilon_6 - \varepsilon_0\varepsilon_1\varepsilon_5 - \varepsilon_0\varepsilon_2\varepsilon_4 - \varepsilon_0\varepsilon_1\varepsilon_2 - \varepsilon_1\varepsilon_2\varepsilon_3 \\
y_8 &= \varepsilon_0 + \varepsilon_1 + \varepsilon_3 + \varepsilon_4 + \varepsilon_7 - \varepsilon_0\varepsilon_1\varepsilon_6 - \varepsilon_0\varepsilon_2\varepsilon_5 - \varepsilon_0\varepsilon_1\varepsilon_3 - \varepsilon_1\varepsilon_2\varepsilon_4 - \varepsilon_0\varepsilon_1\varepsilon_2 - \varepsilon_0\varepsilon_3\varepsilon_4.
\end{aligned}
\qquad (21)
$$

Для косинуса:

$$
\begin{aligned}
y_0 &= 0 \\
y_1 &= 1 \\
y_2 &= -\varepsilon_0\varepsilon_1 \\
y_3 &= -\varepsilon_0\varepsilon_2 \\
y_4 &= 1 - \varepsilon_0\varepsilon_3 - \varepsilon_1\varepsilon_2 \\
y_5 &= 1 - \varepsilon_0\varepsilon_1 - \varepsilon_0\varepsilon_4 - \varepsilon_1\varepsilon_3 \\
y_6 &= -\varepsilon_0\varepsilon_1 - \varepsilon_0\varepsilon_2 - \varepsilon_0\varepsilon_5 - \varepsilon_1\varepsilon_4 - \varepsilon_2\varepsilon_3 \\
y_7 &= 1 - \varepsilon_0\varepsilon_2 - \varepsilon_0\varepsilon_3 - \varepsilon_0\varepsilon_6 - \varepsilon_1\varepsilon_5 - \varepsilon_2\varepsilon_4 - \varepsilon_1\varepsilon_2 + \varepsilon_0\varepsilon_1\varepsilon_2\varepsilon_3 \\
y_8 &= 1 - \varepsilon_0\varepsilon_1 - \varepsilon_0\varepsilon_3 - \varepsilon_0\varepsilon_4 - \varepsilon_0\varepsilon_7 - \varepsilon_1\varepsilon_6 - \varepsilon_2\varepsilon_5 - \varepsilon_1\varepsilon_3 + \varepsilon_0\varepsilon_1\varepsilon_2\varepsilon_4 - \varepsilon_1\varepsilon_2 - \varepsilon_3\varepsilon_4.
\end{aligned}
\qquad (22)
$$

Операторы могут быть вычислены заранее для различных углов и внесены в память. Примеры расчета операторов для различных углов поворота (против часовой стрелки) содержатся в Табл. 5.

Табл. 5. Операторы для углов поворота

φ, град	ε_0	ε_1	ε_2	ε_3	ε_4	ε_5	ε_6	ε_7
0	1	-1	-1	-1	1	-1	1	1
10	1	-1	-1	1	-1	1	1	-1
20	1	-1	1	-1	-1	-1	1	-1
30	1	-1	1	-1	1	1	-1	1
40	1	-1	1	1	1	-1	-1	-1
50	1	1	-1	-1	-1	1	1	1
60	1	1	-1	1	-1	-1	1	-1

Данная таблица значений рассчитывается заранее и хранится в ПЗУ. Ее можно применять также и для выполнения операций «поворот».

1.6. Операции «поворот» и «вектор»

Рассмотрим операцию поворота точки (x, y) вокруг оси O_z на угол φ. Преобразование поворота выполняется по формулам

$$x' = x\cos(\varphi) - y\sin(\varphi),$$
$$y' = y\cos(\varphi) + x\sin(\varphi).$$

Алгоритм CORDIC предполагает следующую процедуру:

$$\varphi_{i+1} = \varphi_i - \varepsilon_i arctg(2^{-i})$$
$$y_{i+1} = y_i - \varepsilon_i x_i 2^{-i}$$
$$x_{i+1} = x_i + \varepsilon_i y_i 2^{-i}$$
$$\varepsilon_i = sign(\varphi_i).$$

Начальные условия: $y_0 = y,\ x_0 = x,\ \varphi_0 = \varphi,\ \varepsilon_0 = \pm 1$ (знак ε_0 определяет направление поворота). Результат: $y_n = ky', x_n = kx'$, где y', x' – новые координаты, n – количество итераций. Заметим, что величины $arctg(2^{-i})$ вычисляют заранее и хранят в памяти. Недостаток алгоритма: линейная деформация (удлинение) вектора в k раз. Проблема может быть решена одним из способов: раздельными поворотами вектора для вычисления новых координат, введением дополнительных компенсирующих итераций или прямой компенсацией деформации.

Если задать начальные условия: $y_0 = 0,\ \varepsilon_0 = 1,\ x_0 = 1/k,\ \varphi_0 = \varphi$, то получим следующий результат: $y_n = -\sin(\varphi),\ x_n = \cos(\varphi)$. Введем вспомогательные величины $A = y - \varepsilon_0 x,\ B = x + \varepsilon_0 y$. Здесь x, y – исходные координаты конца вектора. Результат операции «поворот» (x', y') для разрядности операндов $m = 8$ можно представить в следующем виде:
для координаты x':

$$\begin{aligned}
&x_1 = b_1,\\
&x_2 = b_2 + \varepsilon_1 a_1,\\
&x_3 = b_3 + \varepsilon_1 a_2 + \varepsilon_2 a_1,\\
&x_4 = b_4 + \varepsilon_1 a_3 + \varepsilon_2 a_2 + (\varepsilon_3 a_1 - \varepsilon_1\varepsilon_2 b_1),\\
&x_5 = b_5 + \varepsilon_1 a_4 + \varepsilon_2 a_3 + (\varepsilon_3 a_2 - \varepsilon_1\varepsilon_2 b_2) + (\varepsilon_4 a_1 - \varepsilon_1\varepsilon_3 b_1),\\
&x_6 = b_6 + \varepsilon_1 a_5 + \varepsilon_2 a_4 + (\varepsilon_3 a_3 - \varepsilon_1\varepsilon_2 b_3) + (\varepsilon_4 a_2 - \varepsilon_1\varepsilon_3 b_2) + [\varepsilon_5 a_1 - (\varepsilon_1\varepsilon_4 + \varepsilon_2\varepsilon_3) b_1],\\
&x_7 = b_7 + \varepsilon_1 a_6 + \varepsilon_2 a_5 + (\varepsilon_3 a_4 - \varepsilon_1\varepsilon_2 b_4) + (\varepsilon_4 a_3 - \varepsilon_1\varepsilon_3 b_3) + [\varepsilon_5 a_2 - (\varepsilon_1\varepsilon_4 + \varepsilon_2\varepsilon_3) b_2] +\\
&+ [(\varepsilon_6 - \varepsilon_1\varepsilon_2\varepsilon_3) a_1 - (\varepsilon_1\varepsilon_5 + \varepsilon_2\varepsilon_4) b_1],\\
&x_8 = b_8 + \varepsilon_1 a_7 + \varepsilon_2 a_6 + (\varepsilon_3 a_5 - \varepsilon_1\varepsilon_2 b_5) + (\varepsilon_4 a_4 - \varepsilon_1\varepsilon_3 b_4) + [\varepsilon_5 a_3 - (\varepsilon_1\varepsilon_4 + \varepsilon_2\varepsilon_3) b_3] +\\
&+ [(\varepsilon_6 - \varepsilon_1\varepsilon_2\varepsilon_3) a_2 - (\varepsilon_1\varepsilon_5 + \varepsilon_2\varepsilon_4) b_2] + [(\varepsilon_7 - \varepsilon_1\varepsilon_2\varepsilon_4) a_1 - (\varepsilon_1\varepsilon_6 + \varepsilon_2\varepsilon_5 + \varepsilon_3\varepsilon_4) b_1];
\end{aligned} \tag{23}$$

для координаты y':

$$
\begin{aligned}
&y_1 = a_1,\\
&y_2 = a_2 - \varepsilon_1 b_1,\\
&y_3 = a_3 - \varepsilon_1 b_2 - \varepsilon_2 b_1,\\
&y_4 = a_4 - \varepsilon_1 b_3 - \varepsilon_2 b_2 - (\varepsilon_3 b_1 + \varepsilon_1\varepsilon_2 a_1), Y_4 = a_4 - \varepsilon_1 b_3 - \varepsilon_2 b_2 - (\varepsilon_3 b_1 + \varepsilon_1\varepsilon_2 a_1),\\
&y_5 = a_5 - \varepsilon_1 b_4 - \varepsilon_2 b_3 - (\varepsilon_3 b_2 + \varepsilon_1\varepsilon_2 a_2) - (\varepsilon_4 b_1 + \varepsilon_1\varepsilon_3 a_1),\\
&y_6 = a_6 - \varepsilon_1 b_5 - \varepsilon_2 b_4 - (\varepsilon_3 b_3 + \varepsilon_1\varepsilon_2 a_3) - (\varepsilon_4 b_2 + \varepsilon_1\varepsilon_3 a_2) - [\varepsilon_5 b_1 + (\varepsilon_1\varepsilon_4 + \varepsilon_2\varepsilon_3) a_1],\\
&y_7 = a_7 - \varepsilon_1 b_6 - \varepsilon_2 b_5 - (\varepsilon_3 b_4 + \varepsilon_1\varepsilon_2 a_4) - (\varepsilon_4 b_3 + \varepsilon_1\varepsilon_3 a_3) - [\varepsilon_5 b_2 + (\varepsilon_1\varepsilon_4 + \varepsilon_2\varepsilon_3) a_2] +\\
&+[(\varepsilon_1\varepsilon_2\varepsilon_3 - \varepsilon_6) b_1 - (\varepsilon_1\varepsilon_5 + \varepsilon_2\varepsilon_4) a_1],\\
&y_8 = a_8 - \varepsilon_1 b_7 - \varepsilon_2 b_6 - (\varepsilon_3 b_5 + \varepsilon_1\varepsilon_2 a_5) - (\varepsilon_4 b_4 + \varepsilon_1\varepsilon_3 a_4) - [\varepsilon_5 b_3 + (\varepsilon_1\varepsilon_4 + \varepsilon_2\varepsilon_3) a_3] +\\
&+[(\varepsilon_1\varepsilon_2\varepsilon_3 - \varepsilon_6) b_2 - (\varepsilon_1\varepsilon_5 + \varepsilon_2\varepsilon_4) a_2] + [(\varepsilon_1\varepsilon_2\varepsilon_4 - \varepsilon_7) b_1 - (\varepsilon_1\varepsilon_6 + \varepsilon_2\varepsilon_5 + \varepsilon_3\varepsilon_4) a_1].
\end{aligned}
\tag{24}
$$

Таким образом, мантисса результата записывается через разрядные коэффициенты двоичных операндов A и B. Координаты (x', y') конца повернутого вектора оказываются увеличенными в k раз, что требует их нормализации. Для параллельного выполнения (23)-(24) операторы $\{\varepsilon_1,...,\varepsilon_7\}$, $\varepsilon_i \in \{+1,-1\}$ необходимо вычислить заранее для требуемых углов поворота (Табл. 5). Для параллельного представления операторов Волдера воспользуемся соотношением $\lim\limits_{n\to\infty}(\varphi - \sum\limits_{i=0}^{n} \varepsilon_i arctg(2^{-i}) = 0$, откуда при ограниченном значении n следует

$$\varphi = \sum_{i=0}^{\infty} \varepsilon_i arctg(2^{-i}) \approx \sum_{i=0}^{n} \varepsilon_i arctg(2^{-i}). \tag{25}$$

Величины $arctg(2^{-i})$ являются табличными константами (Табл. 6).

Табл. 6. Константы $arctg(2^{-i})$

i	$arctg(2^{-i})$ град (decimal)	$arctg(2^{-i})$ град(binary)
0	45.0	101101.10
1	26.6	011010.10
2	14.0	001110.00
3	7.2	000111.00
4	3.6	000011.10
5	1.8	000001.11
6	0.9	000000.11
7	0.45	000000.01

Умножая на ε_i двоичные представления $arctg(2^{-i})$ и суммируя в соответствии с (25), получим разрядное представление угла поворота $\varphi = \varphi_1\varphi_2...\varphi_n$:

$$
\begin{aligned}
&\varphi_1 = \varepsilon_0,\\
&\varphi_2 = \varepsilon_1,\\
&\varphi_3 = \varepsilon_0 + \varepsilon_1 + \varepsilon_2,\\
&\varphi_4 = \varepsilon_0 + \varepsilon_2 + \varepsilon_3,\\
&\varphi_5 = \varepsilon_1 + \varepsilon_2 + \varepsilon_3 + \varepsilon_4,\\
&\varphi_6 = \varepsilon_0 + \varepsilon_3 + \varepsilon_4 + \varepsilon_5,\\
&\varphi_7 = \varepsilon_0 + \varepsilon_1 + \varepsilon_4 + \varepsilon_5 + \varepsilon_6,\\
&\varphi_8 = \varepsilon_5 + \varepsilon_6 + \varepsilon_7.
\end{aligned}
\tag{26}
$$

Используя представления (26), получим псевдооператоры:

$$\begin{aligned}
&\varepsilon_0 = \varphi_1,\\
&\varepsilon_1 = \varphi_2,\\
&\varepsilon_2 = \varphi_3 - \varphi_1 - \varphi_2,\\
&\varepsilon_3 = \varphi_4 + \varphi_2 - \varphi_3,\\
&\varepsilon_4 = \varphi_5 + \varphi_1 - \varphi_4 - \varphi_2,\\
&\varepsilon_5 = \varphi_6 - 2\varphi_1 + \varphi_3 - \varphi_5,\\
&\varepsilon_6 = \varphi_7 + \varphi_4 - \varphi_6 - \varphi_3,\\
&\varepsilon_7 = \varphi_8 + 2\varphi_1 + \varphi_5 - \varphi_7 - \varphi_4.
\end{aligned} \qquad (27)$$

С учетом одновременной доступности псевдооператоров (27) формы (23) и (24) следует считать разрядно-параллельными.

Операция «вектор» (вычисляет длину вектора и угол положения вектора, выходящего из начала координат)

$$y_{i+1} = y_i - \varepsilon_i x_i 2^{-i}$$
$$x_{i+1} = x_i + \varepsilon_i y_i 2^{-i}$$
$$\varphi_{i+1} = \varphi_i + \varepsilon_i arctg(2^{-i})$$
$$\varepsilon_i = sign(y_i)$$

Начальные условия: $y_0 = y,\ \ \varepsilon_0 = 1,\ \ x_0 = x,\ \ \varphi_0 = 0$. Результат: $x_n = k\sqrt{x^2 + y^2}$, $\varphi_n = -arctg(y/x)$.

Для реализации операции «вектор», в соответствии с которой должны быть вычислены длина вектора и угол положения, можно использовать схемы (23), (24). В этом случае достаточно, например, приравнять нулю выражения для всех разрядов y', что равносильно повороту вектора до его совмещения с осью абсцисс. Полученная при этом система псевдооператоров должна быть подставлена в выражения (23) для вычисления длины вектора и в выражение (26) для вычисления угла положения.

2. Способ вычисления и оценка временной сложности разрядно-параллельных схем

Рассмотрим технику работы с разрядно-параллельными схемами на примере вычисления обратной функции. Для получения окончательного результата на основе схемы (3) сформируем две группы двоичных операндов в соответствии с их знаками и весами. Вариант такого представления представлен ниже в Табл. 7.

В данной вычислительной схеме имеется всего двенадцать 8-разрядных операндов (6 положительных и 6 отрицательных), представленных разрядными срезами (столбцами Табл. 7). Получение окончательного результата путем их суммирования не вызывает затруднений. Операция алгебраического суммирования n операндов на основе каскадной схемы соединения параллельных сумматоров теоретически может быть выполнена за число тактов $t_1 = \lceil \log_2(n) \rceil + 1$, в то время как последовательное суммирование займет $(n-1)$ тактов. Такт здесь равен времени одной операции суммирования m - разрядных операндов. Оценим время выполнения операции «поворот». Количество слагаемых в схемах (23) и (24) для m-разрядных операндов оценивается формулой $S(m) = \sum_{i=1}^{m} N(i)$, где: $N(i) = 1 + [i/2] + [i/3] + ... + [i/k]$, $k = [-1/2 + 1/2\sqrt{1 + 8 \cdot i}]$. Здесь $[x]$ – целая часть x. Некоторые оценочные значения сложности (в тактах) вычислительных схем для разрядности операндов m приведены в Табл. 8.

Табл. 7. Разрядно-параллельная схема для вычисления $y = 1/x$

Вес	Положительные операнды						Отрицательные операнды					
	1	2	3	4	5	6	1	2	3	4	5	6
2^7	1	0	0	0	0	0	0	0	0	0	0	0
2^6	0	0	0	0	0	0	x_2	0	0	0	0	0
2^5	x_2	0	0	0	0	0	x_3	0	0	0	0	0
2^4	0	0	0	0	0	0	x_4	x_2	0	0	0	0
2^3	x_3	x_2x_3	x_2	x_2x_5	x_3x_4	0	x_5	0	0	0	0	0
2^2	x_2x_4	x_2x_3	x_2x_3	x_3x_5	0	0	x_6	$x_2x_3x_5$	x_2	0	0	0
2^1	x_4	x_2x_3	x_2	x_2x_7	x_3x_6	x_4x_5	x_7	$x_2x_3x_5$	x_3	x_2x_6	x_2x_5	x_3x_4
2^0	0	0	0	0	0	0	x_8	x_2x_6	x_2	x_3x_4	0	0

Табл. 8. Оценка сложности вычислений некоторых функций (в тактах)

Функция:	«rotation», «vectoring»		$\ln(x)$		1/X		$\sqrt{x}$	
m	$n = S(m)$	t_1	$n = m$	t_1	$n = 2m$	t_1	$n = m$	t_1
8	29	6	8	4	16	5	8	4
16	142	9	16	5	32	6	16	5
32	707	10	32	6	64	7	32	6
64	3476	12	64	7	128	8	64	7

Величина ускорения вычислений, достигаемая разрядно-параллельным представлением функций, растет с увеличением разрядности операндов, но требует существенного увеличения аппаратурных затрат. Интерес представляет технология суммирования на основе ассоциативных многовходовых суммирующих устройств (МСУ). Быстродействие определяется, в зависимости от конструкции, по формулам: $t_2 = m + \lceil \log_2(n) \rceil + 2$, $t_3 = m/2 + \lceil \log_2(n) \rceil + 2$, $t_4 = \lceil m/2 \rceil$, причем такт соответствует времени считывания операнда из постоянной памяти. Современная технология дает возможность сделать величину такта равной менее 10 нс, что ведет к увеличению производительности до величины порядка 10^7 крупных операций в секунду. При многократном вычислении функций можно дополнительно привлечь конвейерность, тогда среднее время выполнения операции будет примерно равным $\lceil \log_2(n) \rceil$, где n – ширина разрядно-параллельного слоя, определяемая количеством операндов.

3. Процессорные элементы с набором крупных операций

В работе [14] приведена схема вычислителя, основанного на применении CORDIC IP-ядра, которое настраивается и поддерживает несколько функций, в том числе «поворот», вычисление cos(x), arctan(x) и другие. В устройстве поддерживается параллельный режим, в котором выходные данные обрабатываются в одном цикле тактовой частоты, и последовательный, в котором выходные данные вычисляются за несколько тактов. IP-ядро поддерживает переменную точность и нескольких вариантов алгоритмов округления. Структурная схема устройства представлена на Рис. 1.

В работе [13] предлагается конвейерная архитектура элемента, предназначенного для вычисления тригонометрических и гиперболических функций алгоритмами CORDIC (Рис. 2).

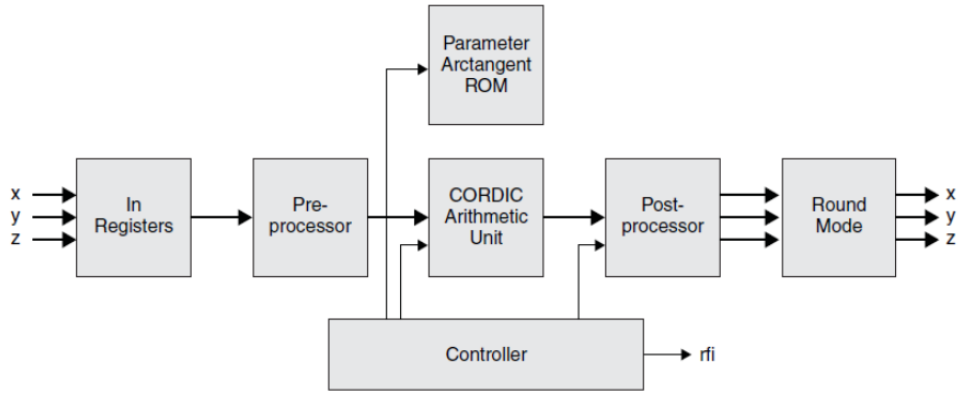

Рис. 1. Блок-схема ядра вычислителя (CORDIC IP Core)

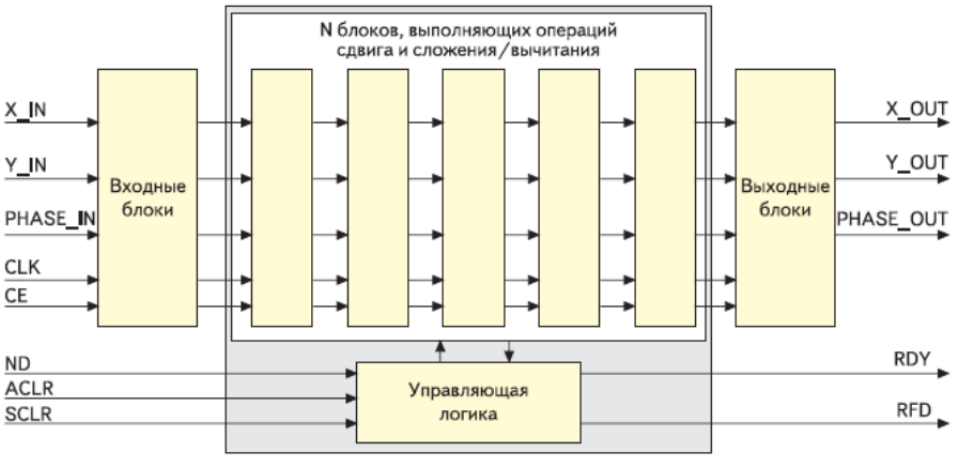

Рис. 2. Структура конвейерного элемента для вычисления функций CORDIC

Элемент характеризуется поддержкой:

− генерации описаний элементов с конвейерной и параллельной архитектурой;

− применения в составе формируемых элементов блока «грубого» поворота вектора и расширения диапазона изменения аргумента вычисляемых тригонометрических функций;

− коррекции результата при выполнении операций поворота и преобразования координат;

− технологии Xilinx Smart-IP, обеспечивающей наилучшую реализацию генерируемого элемента в выбранном кристалле.

На основе разработанных разрядно-параллельных схем предложена структура специализированного геометрического процессорного элемента (ГПЭ), операционная часть которой представлена на Рис. 3 [19].

Основной упор сделан на таблично-алгоритмические вычисления, причем значительная роль отводится ПЗУ, в которых хранится вся предварительно подготовленная информация. Алгоритмы семейства [27] выглядят более эффективными и компактными для операций типа 1/x и $\sqrt{x}$, в то время как алгоритмы CORDIC хороши для выполнения составных геометрических операций, поэтому целесообразно объединить в ГПЭ лучшие стороны обоих направлений. Каждая операция реализуется в устройстве путем соответствующей настройки его структуры. Программирование структуры осуществляется соответствующей коммутацией (назначением) отдельных составляющих ГПЭ с помощью управляющих сигналов $C_1, C_2, ..., C_{12}$, которые формируются после расшифровки кода команды устройством управления. Использование разрядно-параллельных алгоритмов

позволяет строить ГПЭ на единой методологической основе с применением простого и удобного математического аппарата группового суммирования и однородной элементной базы FPGA. Ограниченная система команд ГПЭ представлена в Табл. 9.

Таблица предполагает дальнейшее расширение за счет включения новых команд. ГПЭ аппаратно поддерживает структурное программирование алгоритмов машинной графики. Входными и выходными параметрами команд ГПЭ служат координаты, которые, например, задают положение точек изображения на экране дисплея, в рабочем поле графопостроителя или в других системах отображения.

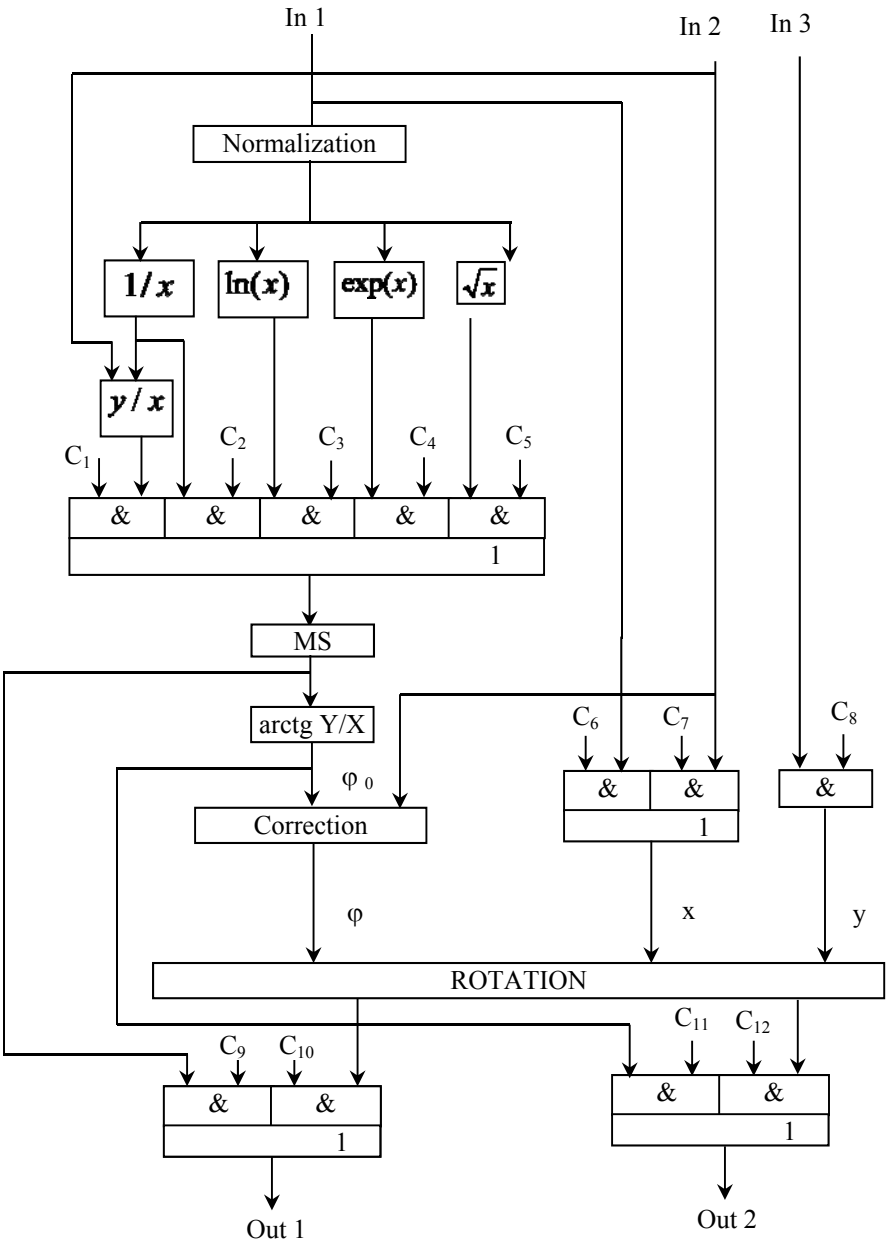

Рис. 3. Операционная часть ГПЭ

Табл. 9. Система команд геометрического процессорного элемента

Операции	Код	Вх.1	Вх.2	Вх.3	Вых.1	Вых.2
«rotation»	0000	x	y	φ	$x' = x\cos(\varphi) - y\sin(\varphi)$	$y' = y\cos(\varphi) - x\sin(\varphi)$
«vectoring»	0001	x	y		$\sqrt{x^2 + y^2}$	$\varphi = arctg(y/x)$
y/x	0010	x	y		y/x	$\varphi = arctg(y/x)$
$(y/x)b$	0011	x	y	b	$(y/x)b$	$\varphi = arctg(y/x)$
$\sin(\varphi),\cos(\varphi)$	0100	1	0	φ	$\cos(\varphi)$	$\sin(\varphi)$
$\ln(x)$	0101	x			$\ln(x)$ (порядок)	$\ln(x)$ (мантисса)
$\sqrt{x}$	0110	x			$\sqrt{x}$ (порядок)	$\sqrt{x}$ (мантисса)
$\exp(x)$	0111	x			$\exp(x)$ (порядок)	$\exp(x)$ (мантисса)

Заключение

В настоящей работе выполнено обобщение ряда работ и показана принципиальная возможность разрядно-параллельного представления результатов вычисления математических функций, полученных на основе разрядных представлений Пухова Г.Е., CORDIC и других алгоритмов. Такая реализация удобна для организации вычислений в системах машинной графики и обработки сигналов, решения задач управления и моделирования процессов реального времени. Оценки вычислительной сложности показывают, что разрядно-параллельные схемы в сочетании с методами группового суммирования операндов позволяют достичь показателей быстродействия, превышающих аналогичные показатели арифметических и RISC-процессоров, использующих метод разложения в ряд Тейлора.

Литература

1. Volder J.E. The Cordic trigonometric computing technique. – IRE Transactions on Electronic Computers, Vol. 8, pp. 330-334, 1959.
2. Байков В.Д., Смолов В.Б. Специализированные процессоры. Итерационные алгоритмы и структуры. – М.: Радио и связь, 1985. – 288 с.
3. Байков В.Д., Вашкевич С.Н. Решение траекторных задач в микропроцессорных системах ЧПУ. – Л.: Машиностроение, 1986. – 106 с.
4. Trig functions – how do TI products calculate them? –http://education.ti.com/product/tech/30xa/faqs/faq83086.html
5. Yuen A.K. Intel's Floating-Point Processors. – Electro/88 Conference Record, 1998, pp. 48
6. Rui Xu, Zhanpeng Jiang, Hai Huang, Changchun Dong. An Optimization of CORDIC Algorithm and FPGA Implementation. – International Journal of Hybrid Information Technology Vol.8, No.6 (2015), pp.217-228. – URL: http://www.sersc.org/journals/IJHIT/vol8_no6_2015/21.pdf
7. Shrugal Varde, Dr. Nisha Sarwade, Richa Upadhyay. Hardware Implementation Of Hyperbolic Tan Using Cordic On FPGA International Journal of Engineering Research and Applications (IJERA), Vol.3, Issue 2, March-April 2013, pp.696-699. – URL: http://www.ijera.com/papers/Vol3_issue2/DK32696699.pdf
8. Gadekar S.R. CORDIC Algorithm for WLAN. – Int. Journal of Engineering Research and Applications, Vol. 5, Issue 8, (Part 2) August 2015, pp.01-04. URL: http://www.ijera.com/papers/Vol5_issue8/Part%20-%202/A58020104.pdf
9. Godbole B.B., Nikam R.H. FPGA implementation of CORDIC algorithm used in DDS based modulators. – International Journal of Advanced Research in Computer and Communication Engineering, Vol.4, Issue 1, January 2015, pp.94-97. – URL: http://www.ijarcce.com/upload/2015/january/IJARCCE2D.pdf
10. Токарев В.А., Хлуденев А.В. CORDIC-алгоритм. – Оценка эффективности алгоритмов цифровой обработки сигналов при конвейерной реализации. – Огарев on-line. Электронное периодическое издание для студентов и аспирантов, Опубликовано 16.10.2015, с.1-4. – URL: http://journal.mrsu.ru/tag/cordic-algoritm/, http://journal.mrsu.ru/wp-content/uploads/2015/10/statya-tokarev-orenburg-.pdf
11. Nikhil Dhume, Ramakrishnan Srinivasakannan. Parameterizable CORDIC-Based Floating-Point Library Operations (XAPP552 (v1.0) June 1, 2012). – URL: http://www.xilinx.com/support/documentation/application_notes/xapp552-cordic-floating-point-operations.pdf

12. CORDIC v6.0. LogiCORE IP Product Guide- Vivado Design Suite, PG105 November 18, 2015. URL: http://www.xilinx.com/support/documentation/ip_documentation/cordic/v6_0/pg105-cordic.pdf
13. Зотов В. Разработка базовых компонентов цифровых устройств, реализуемых на базе ПЛИС FPGA фирмы Xilinx®, с помощью генератора параметризированных модулей CORE Generator. − Компоненты и технологии, №2, 2008, с. 49-56.
14. CORDIC IP Core User's Guide. © 2012 Lattice Semiconductor Corp. – IPUG81_1.3, August 2012.
15. «МУЛЬТИКОР» – отечественные микросхемы ОАО НПЦ «ЭЛВИС». – URL: http://multicore.ru/
16. Артамонов Е.И., Кокаев О.Г., Исмаилов Ш-М.А., Хачумов В.М. Параллельный алгоритм Волдера для операции поворота и его применение в машинной графике. – Управляющие системы и машины, 1990, №1, с.106-109.
17. Хачумов В.М. Разрядно-параллельное представление алгоритмов Волдера. – Информатика и вычислительная техника. Теория и приложения. Часть II. – Махачкала: Даг.отд. Международной акад. информат.,1995, с.19-28.
18. Khachumov V.M. Disparalleling problems in Volder algorithms. – Материалы Международной конференции "Интерактивные системы: Проблемы человеко- компьютерного взаимодействия", Ульяновск (27-31 августа 1995 г.). – Ульяновск: УГТУ,1995, с.56-57.
19. Khachumov V.M. Bit-parallel algorithms and devices. – Conference Proceedings of the 11-th International Conference on Computer Graphics & Vision GRAPHICON'2001 (September 10-15, 2001, Nizny Novgorod), 2001, pp224-226.
20. Хачумов В.М. Разрядно-параллельные структуры для обработки и анализа изображений. – Материалы 6-ой Международной конференции «Распознавание образов и анализ изображений: новые информационные технологии» РОАИ-6-2002 (21-26 октября 2002). – Великий Новгород: Изд-во Нов. ГУ, Том 2, 2002, с.597-601.
21. Khachumov V.M. Bit-Parallel Structures for Image Processing and Analysis. – Pattern Recognition and Image Analysis, Vol. 13, No 2, 2003, pp. 214-216.
22. Khachumov V.M. Bit-Parallel Structures for Image Processing and Analysis. – Pattern Recognition and Image Analysis, Vol. 13, No 4, 2003, pp. 633-639.
23. Захаров А.В., Хачумов В.М. Разрядно-параллельные вычисления в системах реального времени. – Математика, информатика: теория и практика. Сб. трудов, посвященный 10-летию Университета г. Переславля, под ред. Айламазяна А.К., издательство «Университет города Переславля», 2003, с. 97-104.
24. Захаров А.В., Хачумов В.М. Алгоритмы CORDIC. Современное состояние и перспективы // Программные системы: теория и приложения. – М.: Физматлит, май 2004, с.353-372.
25. Хачумов В.М. Разрядно-параллельные вычислительные схемы. – Труды Всероссийской научно конференции «Методы и средства обработки информации» (Москва, 5-7 октября 2005 г.). – М.: МГУ, 2005, с.288-294.
26. Хачумов В.М. Разрядно-параллельные алгоритмы оценки местоположения и ориентации космического аппарата. – Авиакосмическое приборостроение, №6, 2010, с.18-24.
27. Пухов Г.Е., Евдокимов В.Ф., Синьков М.В. Разрядно-аналоговые вычислительные системы. – М.: Сов. Радио, 1978. – 256 с.

Хачумов Вячеслав Михайлович. Заведующий лабораторией ИСА ФИЦ ИУ РАН. Окончил Ленинградский кораблестроительный институт в 1971 году. Доктор технических наук, профессор. Автор более 200 печатных работ. Область научных интересов: вычислительные алгоритмы, распознавание образов, методы обработки сигналов и изображений, искусственный интеллект. E-mail: vmh48@mail.ru