УДК 004.31

В.В. Макаров, канд. техн. наук**, В.В. Жабина**

Национальный технический университет Украины
"Киевский политехнический институт", г. Киев

ВЫЧИСЛЕНИЕ ЛОГАРИФМА ПРИ ПОРАЗРЯДНОМ ВВОДЕ И ВЫВОДЕ ИНФОРМАЦИИ

Розглядається метод обчислення логарифмічної функції при порозрядному введенні операнда і виведенні результату, починаючи із старших розрядів. Запропоновано алгоритм обчислення функції, що забезпечує однакову форму представлення аргументу і результату. Показана можливість застосування зазначеного методу для зменшення часу реалізації послідовності операцій, залежних за даними.

The method of calculation of logarithmic function at digit-by-digit input of operand and output of result, beginning from most significant digits is examined. There offered an algorithm of calculation of function which provides the identical form of presentation of argument and result. There shown possibility of the use of the indicated method for diminishing of time realization of sequence of operations, dependent from data.

Введение

При реализации итерационных процессов, вычислении сложных зависимостей методом суперпозиции функций возникает необходимость выполнения цепочек операций, зависимых по данным. В этом случае распараллеливать вычисления на уровне выполнения операций не представляется возможным, так как выполнение очередной операции может начинаться после получения результата предыдущей операции.

Для ускорения вычислений в указанном случае используют операционные устройства (ОУ), которые выполняют операции в неавтономном (on-line) режиме [1]. Последовательность зависимых операций выполняется на аппаратном уровне в режиме частичного совмещения с использованием цепочки взаимодействующих между собой ОУ.

На каждом шаге вычислений в ОУ вводится по одному разряду операнда и формируется один разряд результата, начиная со старших разрядов. Промежуточный результата, полученный на i-м шаге, в одном ОУ, может быть использован на $i+1$ -м шаге в другом ОУ в качестве очередного разряда операнда. В этом случае выполнение каждой последующей операции начинается не после завершения предыдущей, а сразу после получения первого (старшего) разряда результата этой операции.

Известен метод вычисления логарифма [2-4], получивший название «цифра за цифрой», который позволяет формировать результат поразрядно, но требует перед началом вычисления наличия всех разрядов аргумента. Это не позволяют использовать его для вычислений в неавтономном режиме.

Ниже рассматривается модификация метода «цифра за цифрой», позволяющая вычислять функцию при поразрядном вводе и выводе информации. При рассмотрении метода будем исходить из условия совместимости формы представления информации с другими операциями, которые выполняются в смещенных избыточных системах счисления [5].

Теоретическое обоснование метода

При вычислении функции $Y = \log_a X$ аргумент может быть преобразован к виду [4]

$$X = X_H \cdot \prod_{i=1}^{n} (1 + 2^{-i})^{q_i}, \qquad (1)$$

где $X_H < X -$ начальное приближение, $q_i \in \{0,1\}$, а функцию можно получить как

$$Y = \log_a X_H + \sum_{i=1}^{n} q_i \log_a (1 + 2^{-i}). \qquad (2)$$

В данном случае с каждым шагом значение суммы

$$R_j = X_f \prod_{i=1}^{j} (1+2^{-i})^{q_i} \qquad (3)$$

приближается к X, не превышая его, а результатом преобразования аргумента является последовательность $q_1, q_2, ... q_j$. Для преобразования аргумента в параллельном ОУ в каждом цикле достаточно одной итерации [3].

Рассмотрим случай, когда аргумент, представленный в двоичной избыточной системе с цифрами {0,1,2} поступает в ОУ поразрядно, начиная со старших разрядов.

Будем считать, что $1 \le X < a$, $0 \le Y < 1$, $X_f = 1$.

Пусть разряды целой части и первый дробный разряд x_1 известен до начала вычислений, а в каждом i-м цикле поступает разряд x_{i+1}.

Поскольку в i-м цикле преобразования учитываются только поступившие старшие разряды операнда, то одной итерации в цикле может оказаться недостаточно, так как значение операнда может измениться при поступлении следующих разрядов. «Неправильная» итерация i-го цикла должна быть исправлена в последующих $(n-i)$ циклах. Здесь под «неправильной» итерацией понимают такую итерацию, в результате выполнения которой увеличивается разность между значением частичной суммы (3) преобразования операнда и истинным значением операнда на i-м шаге.

Для функции $Y = \log_a X$ условие сходимости метода имеет вид

$$X_i + x_{i+1} 2^{-(i+1)} < R_{i-1}(1+2^{-i})^{(k+1)}, \qquad (4)$$

где X_i – код, содержащий только i старших разрядов операнда, x_{i+1} – очередная цифра операнда, k – максимальное количество итераций, которое необходимо выполнить в i-м цикле, чтобы скомпенсировать остаток предыдущего цикла и цифру операнда, поступившую в текущем цикле.

Разделив обе части неравенства на R_{i-1} и, логарифмируя его, получим необходимое значение k:

$$k > \frac{\ln(X_i / R_{i-1} + x_{i+1} 2^{-(i+1)} / R_{i-1})}{\ln(1+2^{-i}) - 1}. \qquad (5)$$

Для выполнения неравенства (5) наиболее неблагоприятным является случай, когда в числителе максимальная величина. Поскольку операнд представлен в двоичной позиционной избыточной системе счисления с цифрами $\{0,1,2\}$, то $x_{i+1} 2^{-(i+1)} / R_{i-1}$ максимально при $x_{i+1} = 2$, $R_{i-1} = X_f = 1$ и равно 2^{-i}.

Можно записать, что по окончании $(i-1)$-го цикла $X_i < R_{i-1}(1+2^{-(i-1)})$. Примем $X_i = R_{i-1}(1+2^{-(i-1)})$, тогда $X_i / R_{i-1} = (1+2^{-(i-1)})$. Учитывая полученные значения и (5), определим

$$k > \ln(1+3\cdot 2^{-i}) / \ln(1+2^{-i}) - 1.$$

Нетрудно показать, что это неравенство выполняется при $k = 2$, то есть процесс вычислений будет сходящимся и в каждом цикле может быть от 0 до 2-х итераций.

Заметим, что при изменении основания логарифма a максимальное число итераций в каждом цикле не изменяется, т.к. выполнение неравенства не зависит от a.

Преобразование операнда и вычисление функции производится по формулам (1) и (2). При $q_i = 2$ вычисление R_i представляет собой двукратное сложение числа с этим же числом, сдвинутым на i разрядов: $R_i^{'} = R_{i-1} + 2^{-i} R_{i-1}$, $R_i^{''} = R_i^{'} + 2^{-i} R_i^{'}$.

Результат формируется поразрядно, начиная со старших разрядов, причем, для его представления также используется избыточная система счисления с цифрами $\{0,1,2\}$.

Для того, чтобы при вычислении Y_i по формуле (2) результат не превысил максимальное допустимое значение, необходимо вывод результата производить с задержкой на p шагов, которая определяется из условия

$$(y_{i\max} + 1) 2^{-i+p} > (Y_i - Y_{i-1}) + Y_{i-1}^{'}, \qquad (6)$$

где Y'_{i-1} – код, содержащий только $(n-i+p)$ младших разрядов Y_{i-1}, $y_{i\max}$ – максимальное значение i-го разряда результата. В данном случае $y_{i\max}$=2, тогда (6) с учетом (2) будет иметь вид

$$3\cdot 2^{-i+p} > q_i \cdot \log_a(1+2^{-i}) + Y'_{i-1}$$

или при $q_i = 2$, $Y'_{i-1} = 2^{-i+p+1}$ –

$$p > \log_2(2\cdot \ln(1+2^{-i})/2^{-i}\cdot \ln a).$$

С учетом трех старших членов разложения функции $\ln(1+2^{-i})$ в ряд Тейлора получим максимальное значение задержки

$$p =]\log_2(1.7/\ln a)[,$$

где $]A[$ - ближайшее к A большее целое число.

Например, для десятичного и натурального логарифмов задержка p составляет один цикл. Это значит, что разряд результата с весом 2^{-i} формируется в $(i+1)$-м цикле. Для двоичного логарифма $p = 3$.

Максимальная погрешность вычислений составляет $\Delta = \Delta_1 + \Delta_2$, где Δ_1 – погрешность первой части вычислений (преобразование аргумента), Δ_2 – погрешность второй части вычислений (непосредственное вычисление функции).

В свою очередь $\Delta_1 = \varepsilon_1 + \varepsilon_2$, где ε_1 – погрешность, возникающая от замены бесконечного представления аргумента конечным, а ε_2 – погрешность определения первых n разрядов $q_1, q_2, ... q_n$. Погрешность ε_1 составляет [4]

$$\varepsilon_1 < (1/\ln a)2^{-n}. \qquad (7)$$

Погрешность ε_2 возникает из-за выхода младших разрядов одного из слагаемых за пределы разрядной сетки при вычислении $X_i = X_{i-1}\cdot(1+2^{-i})$. В [4] показано, что $\varepsilon_2 \le n\cdot 2^{-n}$. Учитывая, что в данном случае в каждом цикле возможны две итерации, получим

$$\varepsilon_2 \le 2n\cdot 2^{-n}. \qquad (8)$$

Тогда с учетом (7) и (8)

$$\Delta_1 < (1/\ln a + 2n)2^{-n}. \qquad (9)$$

Погрешность Δ_2 возникает из-за того, что константы $\log_a(1+2^{-i})$ представлены с погрешностью, не превышающей $2^{-(n+1)}$. В результате n двойных сложений эта погрешность составит

$$\Delta_2 \le n\cdot 2^{-n}. \qquad (10)$$

Общая погрешность вычислений с учетом (9) и (10) определяется выражением

$$\Delta < (1/\ln a + 3n)2^{-n}.$$

Для уменьшения погрешности можно добавить дополнительные разряды в регистры и сумматоры.

Алгоритм вычисления функции

Рассмотрим алгоритм вычисления функции $Y = \ln X$.

Как было показано выше, для этой функции условие (5) удовлетворяется при $k = 2$, а задержка формирования разрядов результата составляет $p = 1$.

Перед началом вычислений известны два разряда целой части и один разряд дробной: $X_0 = x_{-1}\cdot 2^1 + x_0\cdot 2^0 + x_1\cdot 2^{-1}$. В каждом i-м цикле поступает $(i+1)$-й разряд операнда $x_{i+1} \in \{0,1,2\}$.

Ниже приведен алгоритм вычислений $Y = \ln X$ в i-м цикле.

1. $X_i = X_{i-1} + x_{i+1}2^{-(i+1)}$,
   $R'_i = R_{i-1}(1+2^{-i})$.
2. $\begin{cases} Если\ R'_i > X_i,\ то\ q_i = 0,\ R_i = R_{i-1}, \\ \qquad переход\ на\ п.4, \\ Если\ R'_i \le X_i,\ то\ q_i = 1, \\ \qquad R''_i = R'_i\cdot(1+2^{-i}), \\ \qquad Y'_i = Y_{i-1} + \ln(1+2^{-i}). \end{cases}$

3.

$$\begin{cases} \textit{Если } R_i'' > X_i, \textit{ то } q_i = 1,\ R_i = R_i', \\ \qquad\qquad Y_i = Y_i', \\ \textit{Если } R_i'' \le X_i, \textit{ то } q_i = 2,\ R_i = R_i'', \\ \qquad\qquad Y_i = Y_i' + \ln(1+2^{-i}). \end{cases}$$

4. Конец цикла.

Как было показано выше, при $p=1$ разряд результата с весом 2^{-i} выводится в $(i+1)$-м цикле. Значение разряда y_i можно определить на основании анализа i-го разряда r_i слова Y_{i+1} и переноса c_i из него в старший разряд [6]:

$$y_i = \begin{cases} 0, \textit{ если } r_i = 0,\ c_i = 0, \\ 1, \textit{ если } r_i = 1,\ c_i = 0, \\ 2, \textit{ если } c_i = 1. \end{cases}$$

Пусть, например, необходимо вычислить значение функции $Y = \ln X$ при разрядности $n=8$ и значении операнда $X = (1.7931)_{10} = (01.10120211)_2$.

Вычисления иллюстрируются табл. 1. В исходном состоянии (0-й цикл) известны целая часть операнда и один разряд дробной части, т.е. $X_0 = 01.1$. Кроме того, $Y_0 = 0$, $R_0 = X_f = 1$. В колонке Y_i жирным шрифтом отмечен разряд r_i перед началом текущего цикла и после его окончания. На основании значения этого разряда и переноса из него в старший разряд c_i определяется очередной разряд результата y_i.

В результате вычислений на выходах устройства формируется значение $Y = (0.02002110)_2 = (0.5859)_{10}$. Табличное значение функции 0.5939, что соответствует теоретической погрешности.

Хотя результат и аргумент представлены в избыточной системе счисления, устройство может быть построено с использованием обычной двоичной аппаратуры. В его состав должны входить двоичные сумматоры и регистры для формирования и хранения промежуточных переменных, блок постоянной памяти, в котором записаны константы вида $\ln(1+2^{-i})$, а также логические схемы для определения очередных цифр результата.

Максимальное время $t_ц$ выполнения одного цикла вычислений составляет $t_ц = 3t_{сл} + 2t_{сдв} + 2_{ср}$, где $t_{сл}, t_{сдв}, t_{ср}$ – соответственно время сложения, сдвига и сравнения. В параллельном устройстве время выполнения цикла составляет $t_{цп} = 2t_{сл} + t_{сдв}$ [4].

При выполнении последовательности из L зависимых по данным операций с помощью параллельных ОУ необходимо выполнить $Q_1 = nL$ циклов, а в рассматриваемом случае – $Q_2 = (\sum_{j=1}^{L} p_i + n - 1)$ циклов [1], где p_i –задержка при формировании старшего разряда результата в каждом ОУ. С учетом длительности микроопераций в каждом конкретном случае можно определить эффективность применения устройств различного типа. Например, при длине цепочки операций $L=5$, p_i=4 и разрядности операндов $n=32$ можно показать, что время вычислений при использовании ОУ с поразрядным вводом и выводом информации сокращается в 1,4 раза. С увеличением L и n выигрыш в быстродействии возрастает.

Выводы

Использование предлагаемого метода создает потенциальную возможность уменьшения времени выполнения последовательности зависимых по данным операций в режиме частичного совмещения. Благодаря поразрядному вводу и выводу данных разряд промежуточного результата, полученный на i-м шаге, в одном ОУ, может быть использован на $i+1$ -м шаге в другом ОУ в качестве очередного разряда операнда. При этом выполнение каждой последующей операции начинается не после завершения предыдущей, а сразу после получения старшего разряда результата этой операции.

Таблица 1

i	x_{i+1}	X_i	R_i	Y_i	c_i	r_i	y_i
0	0	01.10000000	01.000000000	0.00000000			
1	0	01.10000000	01.000000000 +00.100000000 01.100000000 +00.110000000 10.010000000	0.00000000 +0.01101000 0.01101000			
2	1	01.10000000 +00.00100000 01.10100000	01.100000000 00.110000000 10.010000000	0.**0**1101000	0	0	0
3	2	01.10100000 +00.00100000 01.11000000	01.100000000 +00.001100000 01.101100000 +00.001101100 01.101110000	0.0**1**101000 +0.00011110 0.1**0**000110	1	0	2
4	0	01.11000000	01.101100000 +00.000110110 01.110010110	0.10**0**00110	0	0	0
5	2	01.11000000 00.00001000 01.11001000	01.101100000 +00.000011011 01.101111011 +00.000011011 01.110010110	0.100**0**0110 +0.00001000 0.100**0**1110	0	0	0
6	1	01.11001000 +00.00000010 01.11001010	01.101111011 +00.000001101 01.110001000 +00.000001110 01.110010110	0.1000**1**110 +0.00000100 0.1001**0**010	1	0	2
7	1	01.11001010 +00.00000001 01.11001011	01.110001000 +00.000000111 01.110001111 +00.000000111 01.110010110	0.10010**0**10 +0.00000010 0.10010100 +0.00000010 0.10010**1**10	0	1	1
8	-	01.11001011	01.110010110 +00.000000011 01.110011001	0.10010111**0**	0	1	1
9	-			0.10010111**0**	0	0	0

Благодаря последовательному вводу и выводу информации сокращается необходимое число внешних выводов ОУ, что дает дополнительные преимущества по сравнению с ОУ параллельного типа. В частности, при использовании ПЛИС сокращается необходимое число ячеек ввода-вывода. В свою очередь, увеличение числа свободных выводов ПЛИС повышает возможности использования функционального ресурса микросхемы для другой аппаратуры.

Список литературы

1. *Жабин В.И., Корнейчук В.И., Тарасенко В.П.* Некоторые машинные методы вычисления рациональных функций многих аргументов // Автоматика и телемеханика. – 1977. – № 12. – С. 145-154.
2. *Volder J.E.* The CORDIC trigonometric computing technigue // IRE Trans. on Electr. Comp. – 1959. – Vol. 8, No 3. – P. 330-334.

3. *Байков В.Д., Смолов В.Б.* Специализированные процессоры: итерационные алгоритмы и структуры. – М.: Радио и связь, 1985. – 288 с.
4. *Лапыгин Е.Д.* Аппаратные методы ускорения вычисления некоторых элементарных функций // Вопросы радиоэлектроники. – 1964. – Сер. 11, № 7. – С. 3-17.
5. *Дичка И.А., Жабина В.В.* Совмещение зависимых операций на уровне обработки разрядов операндов // Искусственный интеллект. – 2008. – №3. – С. 649-654.
6. А.с. 662937 (СССР) Устройство для вычисления функции $y = e^x$ / Жабин В.И., Корнейчук В.И., Макаров В.В., Тарасенко В.П.// Открытия. Изобретения. – 1979. – № 18.

.

V.V. Makarov, V.V. Zhabina

CALCULATION OF LOGARITHM AT DIGIT-BY-DIGIT INPUT AND OUTPUT OF INFORMATION

Рассматривается метод вычисления логарифмической функции при поразрядном вводе операнда и выводе результата, начиная со старших разрядов. Предложен алгоритм вычисления функции, обеспечивающий одинаковую форму представления аргумента и результата. Показана возможность применения указанного метода для уменьшения времени реализации последовательности операций, зависимых по данным.