

*ЖАБИН В.И.,  
ЖАБИНА В.В.,  
СКОРИЧЕНКО А.В.*

## РЕАЛИЗАЦИЯ НЕАВТОНОМНЫХ ВЫЧИСЛЕНИЙ В ИЗБЫТОЧНЫХ СИСТЕМАХ СЧИСЛЕНИЯ НА ПЛИС

Показана возможность реализации неавтономных вычислений с плавающей запятой в вычислительных системах с непосредственными связями между вычислительными модулями. Предложен метод вычисления полиномов с поразрядной обработкой операндов со старших разрядов в симметричных избыточных системах счисления. Использование указанного метода позволяет расширить диапазон представления чисел и сократить необходимый ресурс ПЛИС благодаря малому числу связей между модулями.

Possibility of realization of on-line floating point computations in systems with direct connections between computation units is shown. Method of polynomial calculation with the use of digit-by-digit operand processing from left-hand digits in symmetrical redundant numeric systems is proposed. Implementation of the above-mentioned method enables to extend the range of numbers representation as well as to save FPGA resources due to small number of connections between units.

**Ключевые слова:** вычисление полиномов, неавтономный режим, поразрядная обработка, ресурсы ПЛИС.

### Введение

Особенности работы вычислительных систем в контуре управления объектами и процессами во многих случаях требуют обеспечения высокой скорости реализации алгоритмов с мелкозернистой структурой [1, 2]. Такую структуру имеют, например, алгоритмы траекторных задач, когда необходимо решать системы уравнений, выполнять полиномиальную интерполяцию функций, преобразование координат в многомерном пространстве и т.д. Как известно, для ускорения вычислений применяются параллельные вычислительные системы, обеспечивающие совмещение обработки данных в параллельных ветвях алгоритмов на различных уровнях параллелизма. Для мелкозернистых алгоритмов эффективным может быть преобразование информации на низком уровне параллелизма, например, на уровне обработки машинных слов, то есть выполнения отдельных операций. По сравнению с более высоким уровнем параллелизма это предполагает увеличение числа параллельных ветвей.

Однако, скорость обработки информации зависит не только от времени выполнения операций в параллельных ветвях, но и от затрат времени на обмен информацией между ветвями, то есть между вычислительными модулями (ВМ) параллельной системы. При пересылке инфор-

мации между ВМ на уровне отдельных слов целесообразно использовать процедуры обмена информацией через общую память со сложными процедурами доступа. Уменьшить затраты времени на обмен данными позволяет использование потоковых систем с непосредственными связями (ПНС) между ВМ [3-6].

В ПНС выходы одних ВМ подключаются к входам других ВМ в соответствии с графом потока данных (ГПД). ВМ работают в неавтономном (on-line) режиме. В процессе вычислений данные пересылаются непосредственно от одних ВМ к другим, преобразуясь на каждом шаге в соответствии с операциями, заданными вершинами ГПД. В таком случае отсутствуют сложные процедуры пересылки данных между ВМ, то есть уменьшаются затраты времени на обмен данными между ними.

Достижения в области интегральной технологии позволяют создавать параллельные системы CSoC (Configurable System on Chip – конфигурируемые системы на кристалле), которые реализуются на ПЛИС с использованием интерактивных средств разработки. В общем случае микросхемы содержат вычислительные ядра, память и программируемую логику, что дает потенциальную возможность оптимально адаптировать аппаратные средства к конкретному применению.

ПНС могут быть реализованы на одной или нескольких микросхемах. Это определяется как размерностью задач, так и сложностью вычислительных средств, что зависит от реализуемых ими алгоритмов выполнения операций. Важной проблемой является сокращение аппаратных ресурсов ПЛИС, необходимых для реализации систем [5-7].

Использование нескольких микросхем создает дополнительные проблемы. Недостатком такой реализации является то, что микросхемы могут не располагать требуемым количеством выводов для обеспечения связей между частями системы. При этом часть ресурсов микросхем может оставаться незадействованной.

Учитывая важность проблемы недостатка выводов микросхем, компанией Virtual Machine Works предложена технология VirtualWire (виртуальные соединения) для построения систем на нескольких микросхемах [7]. Идея, заложенная в основе технологии, заключается в использовании незадействованного оборудования для реализации специальных цепей, обеспечивающих поочередное подключение к выводам микросхемы информации из разных источников внутри микросхемы. Для этого внутри каждой ПЛИС должен быть реализован автомат управления потоками данных.

Данная технология, хотя и может решить проблему нехватки выводов микросхем, создаст большие временные задержки продвижения потоков данных, что противоречит самой идее потоковой модели вычислений.

Одним из подходов к решению проблемы уменьшения количества связей между ВМ является использование квазипараллельной арифметики, позволяющей совмещать процессы поразрядного ввода в ВМ операндов и поразрядного формирования результатов [8-10]. Поразрядный обмен данными со старших разрядов позволяет не только уменьшить число связей между ВМ, но и обеспечить выполнение зависимых по данным операций в режиме совмещения. Формализованные методики перехода от ГПД к структуре ПНС известны, например, [11].

Методы неавтономных вычислений лучше исследованы для чисел с фиксированной запятой. Известны алгоритмы выполнения арифметических операций с плавающей запятой (например, [12]), однако в этой области требуются дальнейшие исследования.

Представление чисел с фиксированной запятой приводит к ряду недостатков:

- существенные ограничения на диапазон представления чисел;
- накопление погрешности при выполнении последовательности операций в случае фиксированной разрядности операндов;
- необходимость масштабирования операндов для обеспечения необходимого соотношения их величин в определенном месте цепочки операций (например, при сложении разряды слагаемых должны иметь одинаковый вес);
- необходимость синхронизации поступления разрядов операндов на входы ВМ из разных источников введением в схему элементов задержки.

Ниже на примере вычисления полиномов показана возможность устранения указанных недостатков за счет обработки данных в неавтономном режиме с плавающей запятой.

### Организация аппаратных средств вычислителя полиномов

Для вычисления полиномов воспользуемся методом Горнера первого порядка. Например, полином 5-го порядка можно записать в виде

$$F = (((((a_5x + a_4)x + a_3)x + a_2)x + a_1)x + a_0).$$

Система представляет собой цепочку ВМ, каждый из которых выполняет промежуточную операцию  $Z = XY + A$  (рис. 1).

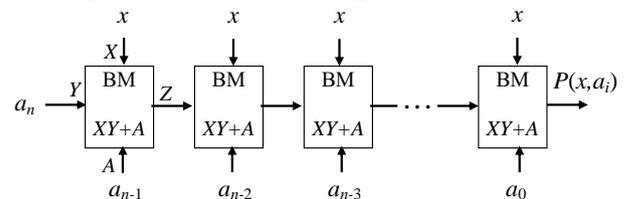


Рис. 1. Система для вычисления полиномов  $P(x, a_i)$  по схеме Горнера первого порядка

ВМ позволяют совмещать выполнение зависимых операций на уровне обработки разрядов слов следующим образом.

На каждом шаге вычислений в ВМ вводится по одному разряду операндов и формируется один разряд результата. При этом разряд промежуточного результата, полученный на  $i$ -м шаге в одном ВМ при выполнении  $j$ -й операции, может быть использован на  $(i+1)$ -м шаге в другом ВМ при выполнении  $(j+1)$ -й операции. При таком режиме вычислений выполнение следующей операции будет начинаться не после завершения выполнения предыдущей операции, а сразу же после получения первого разряда результата этой операции. Режим работы таких ВМ называют неавтономным, так

как для выполнения последовательности операций необходимо несколько ВМ, которые синхронно обмениваются информацией в процессе работы.

Рассматриваемый режим выполнения операций со старших разрядов возможен в избыточных системах счисления, например, в двоичной системе с основанием  $k=2$  и цифрами  $g = \{-1,0,1\}$ .

Хотя на входах и выходах ВМ числа представлены последовательным кодом, такие устройства по внутренней организации ближе к параллельным устройствам. В связи с этим они получили название квазипараллельные [3, 4]. С использованием квазипараллельных ВМ при выполнении последовательности зависимых по данным операций реализуется параллелизм на уровне обработки разрядов операндов.

Как известно [7], в ПЛИС можно сконфигурировать группу программируемых логических блоков для работы в качестве ВМ в виде программных или микропрограммных ядер. С помощью специальных средств в ядрах можно сформировать требуемый набор команд и микроархитектуру ядра. Этот факт является очень важным для построения ВМ со специальными алгоритмами выполнения операций, в том числе, с использованием избыточной системы счисления.

Обобщенная структура ВМ для выполнения операций с плавающей запятой (рис. 2) содержит блок обработки порядков (БОП), блок обработки мантисс (БОМ), буферы мантисс типа FIFO (Буф  $M_X$ , Буф  $M_Y$  и Буф  $M_A$ ) и регистры порядков (Рег  $P_X$ , Рег  $P_Y$  и Рег  $P_A$ ).

Сигналы инициализации и управления ВМ условно не показаны, чтобы не затенять рисунок.

Мантиссы представляются в симметричном избыточном коде с цифрами  $\{-1,0,1\}$ . Для пересылки цифры мантиссы между ВМ необходимо иметь два проводника. Цифры  $-1, 0, 1$  кодируются соответственно парами разрядов  $10, 00, 01$ . С порядками чисел выполняются элементарные операции, поэтому они могут быть представлены в канонической двоичной системе счисления. Для уменьшения связей между ВМ порядки могут пересылаться последовательным кодом по одному проводнику.

Поступление разрядов порядков и мантисс в ВМ тактируется внешними синхросигналами. Темп поступления разных операндов может быть различным. Наличие информации в буферах и регистрах показывают специальные признаки.

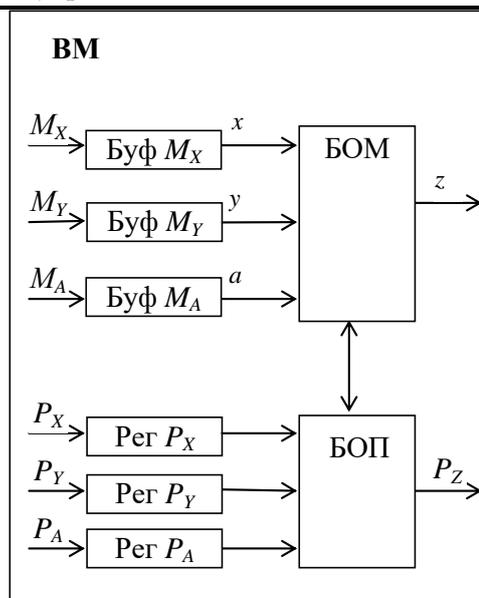


Рис. 2. Вычислительный модуль

Выдача данных из ВМ сопровождается синхросигналами, которые формируются в этом ВМ.

### Алгоритмы обработки мантисс и порядков

Форматы представления чисел с плавающей запятой в избыточных системах счисления имеют свои особенности, которые описаны в [12]. Будем считать, что мантиссы являются нормализованными дробными числами и представлены в форме:

$$X = \sum_{i=1}^n x_i 2^{-i}, \quad Y = \sum_{i=1}^n y_i 2^{-i}, \quad A = \sum_{i=1}^n a_i 2^{-i}, \quad (1)$$

где  $x_i, y_i, a_i \in \{-1,0,1\}$  – цифры операндов,  $n$  – разрядность операндов.

Операнды, содержащие только  $i$  разрядов справа от запятой, обозначим соответственно через  $X_i, Y_i, A_i$ . Например,  $A_i = a_1 a_2 a_3 0 \dots 0$ .

Результат  $Z$  может формироваться поразрядно в системе с такими же цифрами с запаздыванием на некоторое число шагов.

Потребуем, чтобы погрешность результата была знакопеременной и по абсолютной величине не превышала половины веса  $n$ -го разряда после запятой. Это требование будет выполняться, если на  $i$ -м шаге цифру  $z_i$  результата выбирать таким образом, чтобы имело место соотношение

$$Z_i - 2^{-i-1} \leq 2^{-p} (X_i Y_i + A_i) < Z_i + 2^{-i-1}, \quad (2)$$

где  $p$  – число шагов запаздывания формирования цифр результата.

Используя методику в [8] и формулы (1), (2), можно получить алгоритм вычисления  $Z$  в неавтономном режиме. В рассматриваемом

случае  $p = 3$ . Следовательно, для получения  $n$  разрядов результата после запятой необходимо выполнить  $n + 3$  шагов вычисления.

Алгоритм обработки мантисс имеет следующий вид.

1.  $X_0, Y_0, R_0$  присвоить значение 0.
2. Для  $i = \overline{1, n+3}$  выполнять пункты 3-7.
3.  $H_i = 2R_{i-1} + 2^{-3} X_{i-1}y_i + 2^{-3} Y_{i-1}x_i + 2^{-3} a_i + 2^{-3-i} x_i y_i$ .
4.  $X_i = X_{i-1} + x_i 2^{-i}$ .
5.  $Y_i = Y_{i-1} + y_i 2^{-i}$ .
6.  $z_i = \begin{cases} -1, & \text{если } H_i < -2^{-1}; \\ 0, & \text{если } -2^{-1} \leq H_i < 2^{-1}; \\ 1, & \text{если } 2^{-1} \leq H_i. \end{cases}$
7.  $R_i = H_i - z_i$ .

Здесь  $H_i$  и  $R_i$  – вспомогательные переменные.

Порядки являются целыми числами. Они поступают в ВМ вместе с первым разрядом соответствующего операнда и обрабатываются совместно с мантиссами.

Обработка порядков производится следующим образом.

1. Принять порядки  $P_X, P_Y$  и  $P_A$ , установить вспомогательные переменные  $q := 0, j := 0, s := 0$ .
2. Получить предварительный порядок результата  $P_Z$ , выполнив преобразования:  
 $P_W := P_X + P_Y$ ; если  $P_W \geq P_A$ , то  $q := 1$ ;  
 $\Delta P := \max(P_W, P_A) - \min(P_W, P_A)$ ;  
 $P_Z := \max(P_W, P_A) + p$ .
3. Если в Буф  $M_X$ , Буф  $M_Y$  и Буф  $M_A$  есть очередные цифры мантисс операндов  $x, y, a$  (см. рис.1), то перейти к п. 4, иначе выполнить повторно п. 3.
4. Если  $\Delta P = 0$ , то принять в БОМ цифры мантисс  $x, y, a$  и перейти к п. 7.
5. Если  $q = 0$ , то принять в БОМ цифры  $x, y$  и 0 вместо  $a$ , иначе принять  $a$  и нули вместо  $x, y$ .
6.  $\Delta P := \Delta P - 1$ .
7. Сформировать цифру результата  $z$ .
8. Если  $s \neq 0$ , то перейти к п. 11.
9. Если  $z = 0$ , то  $P_Z := P_Z - 1$  и перейти к п. 3.

10.  $s := 1$  и выдать из БОП окончательный порядок результата  $P_Z$ .
11. Выдать из БОМ очередную цифру результата  $z$  и  $j := j + 1$ .
12. Если  $j \neq n$ , то перейти к п. 3, иначе конец операции.

В алгоритме цифры операндов представлены без индексов, поскольку порядковые номера циклов вычисления и цифр в коде операндов не совпадают.

Порядок результата выдается из ВМ вместе с первым ненулевым разрядом мантиссы результата. За счет этого обеспечивается нормализация мантиссы результата.

### Оценка временных характеристик и аппаратных затрат

Следует заметить, что операции в БОП и БОМ практически совмещаются во времени. В связи с этим для оценки времени вычислений достаточно рассмотреть обработку мантисс чисел.

Процесс выполнения операций иллюстрируется диаграммой на рис. 3.

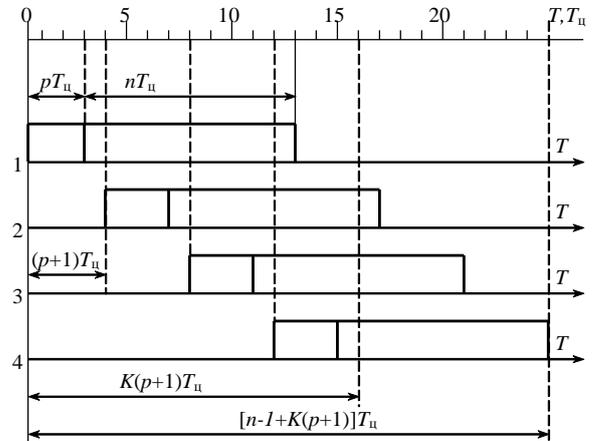


Рис. 3. Временная диаграмма выполнения последовательности зависимых по данным операций

Операции выполняются в неавтономном режиме с помощью цепочки ВМ, которые позволяют совмещать выполнение зависимых операций на уровне обработки разрядов операндов.

Из диаграммы на рис. 3 следует, что время вычислений, необходимое для получения окончательного результата, составляет

$$T = \left[ n - 1 + \sum_{j=1}^K (p_j + 1) \right] t_{ц}, \quad (3)$$

где  $n$  – разрядность операндов;  $K$  – число операций в цепочке;  $p_j$  – задержка формирования разрядов результата  $j$ -й операции;  $t_{ц}$  – длительность цикла формирования одного разряда результата в ВМ.

В рассматриваемом случае для каждой операции задержка составляет  $p = 3$ . Например, для вычисления полинома 5-ой степени при 64-разрядных операндах понадобится выполнить 83 цикла. Длительность одного цикла определяется в основном вычислением переменной  $N_i$  в приведенном выше алгоритме обработки мантисс (см. п. 3). Указанная длительность определяется суммированием пяти слагаемых с помощью дерева комбинационных сумматоров и составляет единицы наносекунд при реализации на ПЛИС.

Первый разряд окончательного результата формируется через  $K(p+1)$  циклов после начала вычислений. В дальнейшем в каждом такте формируется очередная цифра результата. Заметим, что в системах автоматического управления во многих случаях управляющее воздействие может начинать формировать при поступлении первого старшего разряда результата, а затем уточняться при поступлении каждого следующего разряда.

При использовании для выполнения последовательности зависимых по данным операций параллельных ВМ, работающих в канонической двоичной системе счисления, время выполнения цепочки операций будет определяться формулой

$$T = T_0 + \sum_{i=1}^K t_i, \quad (4)$$

где  $t_i$  – время выполнения  $i$ -й операции;  $T_0$  – дополнительные затраты времени, например, на обмен данными между ВМ (возможно через общую память), прием операндов в устройство по очереди или последовательным кодом для уменьшения требуемого числа выводов микросхем и т.д.

Для количественной сравнительной оценки времени вычислений по формулам (3) и (4) требуется более детальное рассмотрение конкретной реализации ВМ разного типа.

Для сравнительной оценки затрат ресурсов ПЛИС при реализации устройств с параллельной и последовательной передачей данных проведено моделирование двух вариантов реализации БОМ на базе микросхемы EP2C35F672C6 семейства Cyclone II фирмы Altera в среде проектирования Quartus II Version 9.1 Build 304. Результаты моде-

лирования в соответствии с отчетами о компиляции приведены в табл. 1.

**Табл. 1. Используемые ресурсы ПЛИС**

Ресурс ПЛИС	Блок обработки мантисс	
	Параллельный	Квазипараллельный
Логические элементы	502/33216 (1,5%)	695/33216 (2%)
Регистры	0/33216 (0%)	271/33216 (0,8%)
Блоки умножения	32/70 (46%)	0/70 (0%)
Выводы	256/475 (54%)	10/475 (2%)

В табл. 1 через разделитель показан соответственно ресурс для построения вычислителей и общий ресурс ПЛИС определенного вида.

Для параллельного блока рассматривались только аппаратные средства выполнения операции  $XU + A$  без учета средств выравнивания порядков операндов и нормализации результата. Умножитель  $64 \times 64$  реализован на базе встроенных в ПЛИС быстродействующих блоков умножения.

На основании табл. 1 можно сделать вывод, что затраты ресурсов ПЛИС больше для устройства параллельного типа. Учитывая, что при данной организации вычислений практически половина выводов ПЛИС используется только для одного каскада вычислителя полиномов, применение режима параллельного ввода операндов вряд ли целесообразно. Более реальным является режим предварительного ввода в определенном порядке 64-разрядных операндов с последующей их обработкой в параллельном устройстве. Время вычисления полиномов будет во многом определяться режимом ввода операндов.

## Выводы

В работе исследована возможность повышения эффективности систем с непосредственными связями за счет реализации неавтономных вычислений с плавающей запятой при поразрядной передаче данных между модулями системы.

Показано, что по сравнению с вычислениями в формате фиксированной запятой выполнение операций с плавающей запятой позволяет расширить диапазон представления чисел, повысить точность получения результатов и ускорить вычисления за счет параллельной обработки мантисс и порядков в асинхронном режиме.

По сравнению с системами, в которых передача данных между ВМ производится параллельным кодом, для реализации систем с последовательной передачей данных на базе квазипараллельных ВМ требуется меньший ресурс ПЛИС. Экономятся как элементы внутреннего ресурса (функциональные и коммуникационные модули, встроенные устройства), так и выводы ПЛИС. Благодаря этому увеличивается вероятность размещения системы на одной микросхеме или уменьшения их числа. Реализация системы на одной микросхеме обеспечивает повышение ее

надежности, уменьшение энергопотребления и габаритов. Обмен данными между ВМ внутри микросхемы выполняется быстрее, чем между компонентами системы, реализованными на разных микросхемах. Это дает потенциальную возможность повысить частоту тактирования, что, в свою очередь, ускоряет обработку информации.

Таким образом, полученные результаты подтверждают эффективность применения неавтономных методов поразрядной обработки информации в вычислительных системах на кристалле.

### Список литературы

1. Байков В.Д. Решение траекторных задач в микропроцессорных системах ЧПУ / В.Д.Байков, С.Н.Вашкевич. – Л.: Машиностроение, 1986, 105 с.
2. Жабин В.И. Архитектура вычислительных систем реального времени / В.И.Жабин. – К.: ВЕК+, 2003. – 176 с.
3. Жабин В.И. Эффективность потоковых вычислений в системах с непосредственными связями, реализованных на ПЛИС / В.И.Жабин, В.В.Жабина, М.А. Безгинский // Вісник НТУУ "КПІ". Інформатика, управління та обчислювальна техніка: Зб. наук. праць. – К.: ВЕК+. – 2012. – №55. – С. 149-156.
4. Жабин В.И. Выполнение последовательностей зависимых операций в режиме совмещения / В.И.Жабин. // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка: Зб. Наук. Пр. – К.: Век+. – 2007. – №46. – С. 226-233.
5. Палагин А.В. Реконфигурируемые вычислительные системы: Основы и приложения / А.В.Палагин, В.Н.Опанасенко. – К.: Просвіта, 2006, 280 с.
6. Каляев И.А. Архитектура семейства реконфигурируемых вычислительных систем на основе ПЛИС / И.А. Каляев, И.И. Левин, Е.А. Семерников // Искусственный интеллект. – 2008. – № 3. – С. 663-674.
7. Максфилд К. Проектирование на ПЛИС. Архитектура, средства и методы / К.Максфилд. – М.: Издательский дом «Додэка-XXI», 2007, 408 с.
8. Жабин В.И. Некоторые машинные методы вычисления рациональных функций многих аргументов / В.И. Жабин, В.И.Корнейчук, В.П.Тарасенко // Автоматика и телемеханика. – 1977. – №12. – С. 145-154.
9. Жабин В.И. Исследование методов построения вычислительных устройств на основе FPGA фирмы XILINX / В.И.Жабин, Н.А.Ковалев // Технология и конструирование в электронной аппаратуре. – 2002. – №1. – С. 35-39.
10. Самофалов К.Г. Основы теории многоуровневых конвейерных вычислительных систем / К.Г.Самофалов, Г.М.Луцкий. – М.: Радио и связь, 1989. – 272 с.
11. Жабин В.И. Построение быстродействующих специализированных вычислителей для реализации многоместных выражений / В.И.Жабин, В.И.Корнейчук, В.П.Тарасенко // Автоматика и вычислительная техника. – 1981. – №6. – с. 18-22.
12. Жабин В.И. Повышение эффективности параллельных вычислений в потоковых системах / В.И.Жабин, В.В.Жабина // Вісник НТУУ "КПІ". Інформатика, управління та обчислювальна техніка: Зб. наук. праць. – К.: ВЕК+. – 2013. - № 59. – С. 122-128.