

Э. И. ВАТУТИН, И. В. ЗОТОВ, В. С. ТИТОВ, М. М. АЛЬ-АШВАЛ

РЕАЛИЗАЦИЯ ОПЕРАЦИИ ВСТАВКИ ПОДДЕРЕВА ПРИ АППАРАТНО-ОРИЕНТИРОВАННОЙ ОБРАБОТКЕ R -ВЫРАЖЕНИЙ

Рассматривается устройство, позволяющее реализовать операцию вставки поддерева в дерево при аппаратно-ориентированной обработке R -выражений, возникающей при выполнении эквивалентных преобразований параллельных алгоритмов.

Ключевые слова: логический мультиконтроллер, параллельный алгоритм логического управления, разбиение, оптимизация, R -выражение, ориентированное дерево, специпроцессор.

Построение систем логического управления, ориентированных на реализацию параллельных управляющих алгоритмов в базе логических мультиконтроллеров (ЛМК), требует декомпозиции комплексных параллельных алгоритмов теоретически неограниченной сложности на множество частных алгоритмов ограниченной сложности [1]. Получение оптимального набора частных алгоритмов (разбиения) представляет собой сложную комбинаторную задачу, решение которой возможно лишь с помощью эвристических алгоритмов. Качество решения этой задачи существенно влияет на аппаратную сложность ЛМК и определяет, в конечном счете, время выполнения алгоритма. Эффективным путем решения данной задачи является параллельно-последовательный метод декомпозиции [2—5].

Один из ключевых этапов параллельно-последовательной декомпозиции — построение множества сечений, покрывающего все вершины исходного алгоритма. Формирование сечений осуществляется посредством выполнения трудоемких операций подстановки над множеством так называемых R -выражений, описывающих алгоритм управления. Как показывают исследования, упрощение и ускорение этих операций возможно путем их сведения к действиям над деревьями. Такие действия, в свою очередь, допускают разбиение на ряд более простых (элементарных) операций [6]. Схематично операции подстановки могут быть представлены следующим образом:

$$\begin{cases} i: X^R \rightarrow B^R \\ j: A^R \rightarrow C^R \end{cases} \Rightarrow i: X^R \rightarrow B^{R'} \text{ (} u\text{-подстановка);}$$
$$\begin{cases} i: C^R \rightarrow A^R \\ j: B^R \rightarrow X^R \end{cases} \Rightarrow j: B^{R'} \rightarrow X^R \text{ (} d\text{-подстановка),}$$

где X^R — некоторое R -выражение, не участвующее в подстановке; $B^{R'}$ — R -выражение, получаемое из R -выражения B^R после удаления поддерева A^R и вставки вместо него дерева C^R .

Методика практической реализации операций над R -выражениями, а также представление их в виде деревьев, допускающее преобразование в табличный вид, приведены в работах [7, с. 38; 8]. Здесь напомним следующее: каждый элемент a дерева X , представленного совокупностью наборов листьев $L_1^X, L_2^X, \dots, L_{N_L(X)}^X$, узлов $T_1^X, T_2^X, \dots, T_{N_T(X)}^X$ и связей между ними, кодируется набором полей. С учетом ряда особенностей обработки, наборы листьев и узлы дерева кодируются отдельно. Узлы дерева представлены следующими полями: тип узла (ТУ) — параллельный или альтернативный ($t(T_i^X)$); ссылка на предка (СП) — номер узла-предка ($u(T_i^X)$); номер соответствия (НС) — номер изоморфного эквивалента в соседнем дереве ($n_r(T_i^X)$); тип соответствия (ТС) — отсутствующее, полное или частичное соответствие ($\Delta(T_i^X)$); наборам листьев дерева при этом соответствуют поля множества вершин (МВ) — двоичный вектор с единичными битами в позициях, соответствующих номерам присутствующих в наборе вершин, а также перечисленные выше поля СП, НС и ТС.

Табличное представление R -выражений подчиняется ряду следующих требований [8].

1. Корень дерева хранится в позиции с номером 0. Значение поля СП корня указывает на заведомо несуществующий элемент дерева.
2. Для каждого узла дерева его потомки хранятся в позициях с номерами, превосходящими номер позиции самого узла (при этом порядок хранения потомков не важен).
3. Все узлы и наборы листьев дерева хранятся в смежных позициях (без „пропусков“).
4. Каждому узлу дерева соответствует не более одного дочернего набора листьев, в дереве не может быть „пустых“ наборов листьев, не содержащих вершин.
5. Если дерево представлено единственным набором листьев (без узлов), то в составе этого набора может быть всего один элемент (одна вершина алгоритма управления).
6. В составе корректного дерева не может быть совпадения типа узлов у любой пары смежных узлов.
7. Дерево содержит, по меньшей мере, один набор листьев. Число узлов дерева может быть нулевым.

В настоящей статье предложена аппаратная реализация операции вставки поддеревя. Она выполняется непосредственно после операции удаления поддеревя, r -изоморфного [7, 8] дереву A^R , в результате которого может быть нарушено требование № 3. Рассматриваемая операция состоит из двух стадий: на первой производится копирование элементов дерева, на второй — настройка связей для скопированных элементов. В результате выполнения операции копирования элементов дерева C^R должны быть учтены требования № 2 и 4. При этом в общем случае требования № 3 и 6 в результате выполнения операции вставки поддеревя не выполняются, что обуславливает необходимость удаления „пропусков“ непосредственно после рассматриваемой операции.

Общие принципы выполнения первой стадии продемонстрированы на рис. 1 (белые квадраты — свободные позиции, темные квадраты — занятые позиции, крестами помечены удаленные элементы, стрелками обозначены операции копирования элементов дерева).

На первой стадии при копировании узлов дерева (см. рис. 1, а) необходимо соблюдение требования № 2 корректности дерева. Наиболее простой способ обеспечить выполнение этого требования — последовательная вставка „новых“ узлов, начиная с позиции с номером, превосходящим номера уже имеющихся узлов дерева B^R , с сохранением порядка их следования. Номер такой позиции может быть определен как $N_{\chi}^T = g_0^{\uparrow}(\mathbf{X}_T) + 1$, где

$\mathbf{X}_T(B) = (\chi(T_0^B), \chi(T_1^B), \dots, \chi(T_{n-1}^B))$ — вектор двоичных признаков $\chi(T_i^X)$ свободных позиций среди узлов дерева B^R , $g_0^\uparrow(\mathbf{X})$ — функция выделения позиции старшего нуля в двоичном векторе \mathbf{X} [9]. Копирование наборов листьев (см. рис. 1, б) не нарушает требований корректности дерева и поэтому может быть осуществлено непосредственно в первую свободную ячейку с младшим номером, который определяется как $g_1^\downarrow(\mathbf{X}_L)$, где $\mathbf{X}_L(B) = (\chi(L_0^B), \chi(L_1^B), \dots, \chi(L_{m-1}^B))$ — вектор двоичных признаков $\chi(L_i^X)$ свободных позиций среди наборов листьев дерева B^R , $g_1^\downarrow(\mathbf{X})$ — функция выделения позиции младшей единицы в двоичном векторе \mathbf{X} [9]. Подобный способ копирования наборов листьев уменьшает число действий, затрачиваемых впоследствии на ликвидацию пустых позиций.

При добавлении каждого нового элемента в дерево B^R также необходимо инкрементировать значение $N_T(B^R)$ (при добавлении узла) или $N_L(B^R)$ (при добавлении набора листьев).

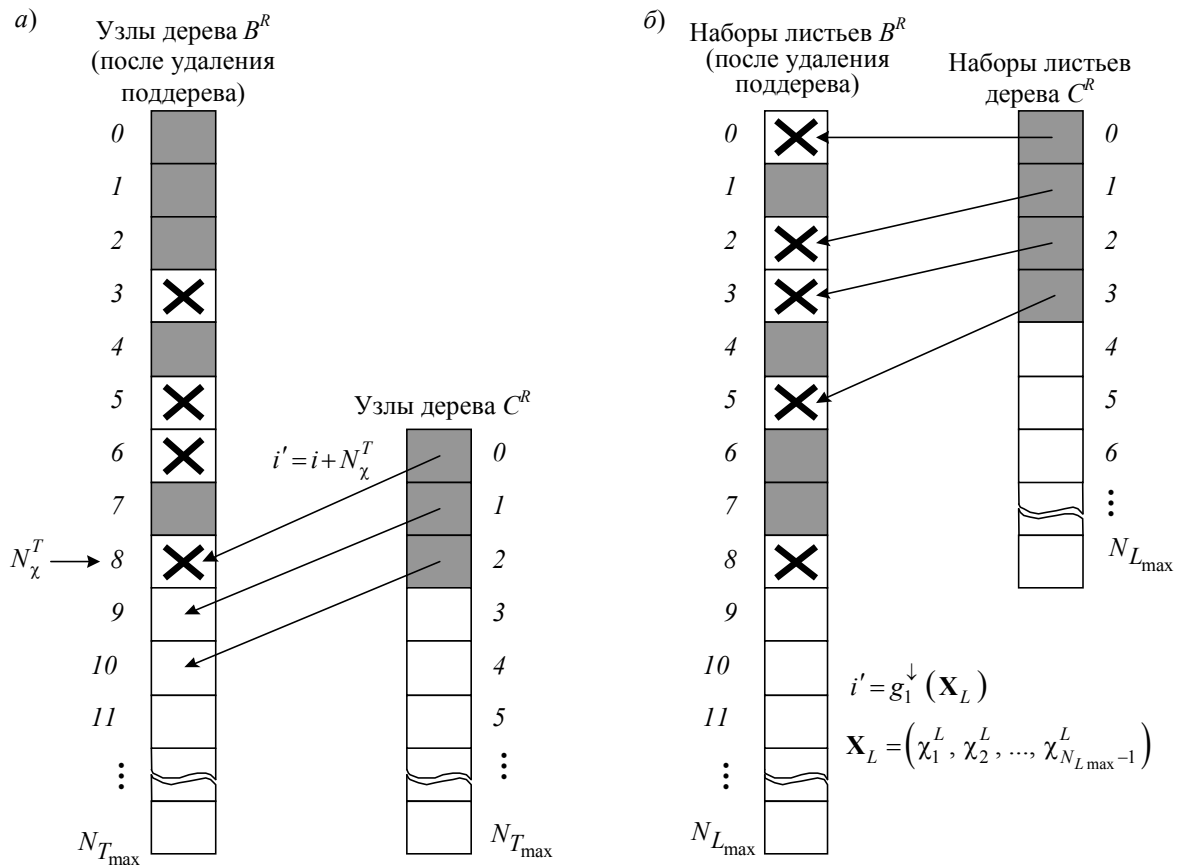


Рис. 1

Вторая стадия начинается с корректировки значений полей СП „новых“ элементов дерева B^R , скопированных в него из дерева C^R . Прежде всего корректировке подвергаются значения полей СП, обозначаемые как $u(\cdot)$, „новых“ элементов: $u(L_i^B) := u(L_i^C) + N_\chi^T$, $u(T_j^B) := u(T_j^C) + N_\chi^T$. Подобные формулы корректировки объясняются достаточно просто: нулевой узел (корень) дерева C^R попадает в дереве B^R в позицию с номером N_χ^T , первый — в позицию $N_\chi^T + 1$ и т.д.

После выполнения указанных корректировок необходимо настроить связь скопированного поддерева из „новых“ элементов с уже присутствующими элементами дерева B^R (точнее, с узлом, r -изоморфным корню дерева A^R , при частичном соответствии, или его предком при полном соответствии). Детали реализации настройки связи между деревьями представлены в табл. 1 и проиллюстрированы на рис. 2, где a , b , c соответствуют трем различным случаям выполнения операции подстановки, приведенным в таблице.

Таблица 1

Наличие в дереве A^R узлов (σ_A)	ТС корня дерева A^R ($\Delta(T_0^A)$)	Формула обновленного значения поля СП корневого элемента скопированного дерева
1	Полное: $\Delta=11$	$x = u \left(T_{n_r(T_0^A)}^B \right)$
	Частичное: $\Delta=10$	$x = n_r \left(T_0^A \right)$
0	Полное, частичное: $\Delta=1^*$	$x = u \left(L_{n_r(L_0^A)}^B \right)$

Особым случаем, не указанным в табл. 1, является ситуация, когда дерево C^R не содержит узлов, что обозначим нулевым значением двоичного признака σ_C . В этом случае при выполнении описанных выше действий возможно нарушение требования № 4, если у элемента дерева B^R , на который производится настройка поля СП добавляемого набора листьев L_0^C дерева C^R , уже есть дочерний набор листьев L_k^B . Правильным действием в подобной ситуации является добавление элементов вектора L_0^C к элементам вектора L_k^B : $L_k^B := L_k^B \cup L_0^C$ вместо добавления еще одного набора листьев.

Алгоритм преобразования выполняется следующим образом.

1. Если дерево C^R не содержит узлов ($\sigma_C = 0$) и узел дерева B^R с номером x , определяемым согласно табл. 1, имеет дочерний набор листьев L_k^B (что в дальнейшем будем обозначать с использованием двоичного признака $\kappa(T_i^X)$, равного в данном случае единице), перейти к п. 7.

2. Определить значение $N_\chi^T := g_0^\uparrow(\mathbf{X}_T) + 1$. Если $\mathbf{X}_T = 00\dots 0$ (все элементы в табличном представлении заняты), установить признак ошибки $\varepsilon_1 := 1$ и перейти к п. 8, в противном случае установить признак $\varepsilon_1 := 0$.

3. Копирование узлов: скопировать узлы T_i^C дерева C^R , $i = 0, \overline{N_T(C^R) - 1}$, в смежные позиции дерева B^R , начиная с позиции N_χ^T . При недостаточном количестве свободных позиций сформировать признак ошибки $\varepsilon_T := 1$ и перейти к п. 8, в противном случае положить $\varepsilon_T := 0$.

4. Копирование наборов листьев: скопировать наборы листьев L_i^C дерева C^R , $i = 0, \overline{N_L(C^R) - 1}$, в дерево B^R , причем i -й набор листьев дерева C^R копируется в позицию $k = g_1^\downarrow(\mathbf{X}_L)$, а сама позиция помечается как используемая: $\chi(L_k^B) := 0$. В случае если на ка-

ком-либо из шагов $X_L = 00\dots 0$ (свободной позиции нет), установить признак ошибки $\varepsilon_L := 1$ и перейти к п. 8, в противном случае положить $\varepsilon_L := 0$.

5. Увеличить значения текущего количества узлов и текущего количества листьев дерева B^R : $N_T(B^R) := N_T(B^R) + N_T(C^R)$, $N_L(B^R) := N_L(B^R) + N_L(C^R)$.

6. Модифицировать значения полей СП „новых“ элементов дерева B^R , скопированных из дерева C^R , за исключением корневого, как $u(L_i^B) := u(L_i^C) + N_\chi^T$, $u(T_j^B) := u(T_j^C) + N_\chi^T$.

Для „нового“ элемента дерева B^R , соответствующего корневному элементу дерева C^R , установить значение поля СП согласно табл. 1. Перейти к п. 8.

7. Осуществить объединение наборов листьев: $L_k^B := L_k^C \cup L_0^C$.

8. Конец алгоритма.

Схема, реализующая предложенный алгоритм, приведена на рис. 3 (сокращения, использованные в схеме, аналогичны принятым в работе [7]). Регистр 1 хранит текущее количество узлов дерева B^R и инкрементируется по мере добавления в дерево B^R новых узлов. Коммутатор 2 используется для выбора значения вектора, подаваемого на вход элемента В.УЭ(у) [7, 10] в качестве адреса записи. Элементы 3 и 4 используются при инициализации значений вектора В.УЭ(у). Шифратор 5 предназначен для преобразования значения номера свободной позиции с выхода схемы выделения старшего нуля (СВСН) из унитарного в двоичное представление, сохраняющегося в регистре 6. Элемент задержки 7 обеспечивает запись значения в регистр 6 только по окончании его формирования на выходе схемы СВСН. Элемент ИЛИ—НЕ 8 используется для формирования сигнала ошибки ε_1 . Регистры 9 и 27 предназначены для хранения числа узлов и наборов листьев дерева C^R соответственно. Коммутатор 10 в совокупности с дешифратором 11 обеспечивают запись в сдвиговый регистр 12 значения текущего числа элементов (наборов листьев или узлов) в унитарном коде в зависимости от этапа обработки. Элемент И 13 запрещает прохождение синхросигнала c_1 при инициализации значения В.УЭ(нл) на входы элемента В.УЭ(у) и регистра 6. Регистр 14 хранит текущее количество наборов листьев дерева B^R , инкрементируемое по мере добавления новых наборов листьев. Коммутатор 15 в совокупности с элементами 16 и 17 обеспечивает инициализацию значения вектора УЭ наборов листьев дерева B^R (в ходе дальнейшей обработки значения вектора УЭ изменяются согласно алгоритму обработки с использованием перечисленных элементов). Шифратор 18 используется для преобразования значения текущего рассматриваемого элемента из унитарного кода в двоичный. Сумматор 19 обеспечивает формирование на его выходе значения $N_T^\chi + i$ очередной свободной позиции и признака ошибки ε_T . Дешифратор 20 преобразует значение $N_T^\chi + i$ в унитарный код для его дальнейшего использования в качестве адреса записи. Коммутаторы 21, 23, 28, 31, 46 и 47 используются для выбора значения элемента дерева B^R модифицируемого элемента в зависимости от этапа алгоритма и особенностей обработки. Сумматор 22 обеспечивает формирование обновленного значения поля СП копируемого элемента. Элемент ИЛИ 24 служит для прохождения синхросигнала записи на вход элемента В.СП(у). Элементы 25 и 26 обеспечивают прохождение синхросигналов c_2 и c_3 на вход регистра 12. Элемент ИЛИ—НЕ 29 служит для формирования признака ошибки ε_L . Сумматор 30 используется для корректировки значения поля СП копируемого набора листьев. Элемент ИЛИ 32 обеспечивает прохождение синхросигнала на вход элемента В.МВ, активирующего модификацию поля МВ в зависимости от ситуации.

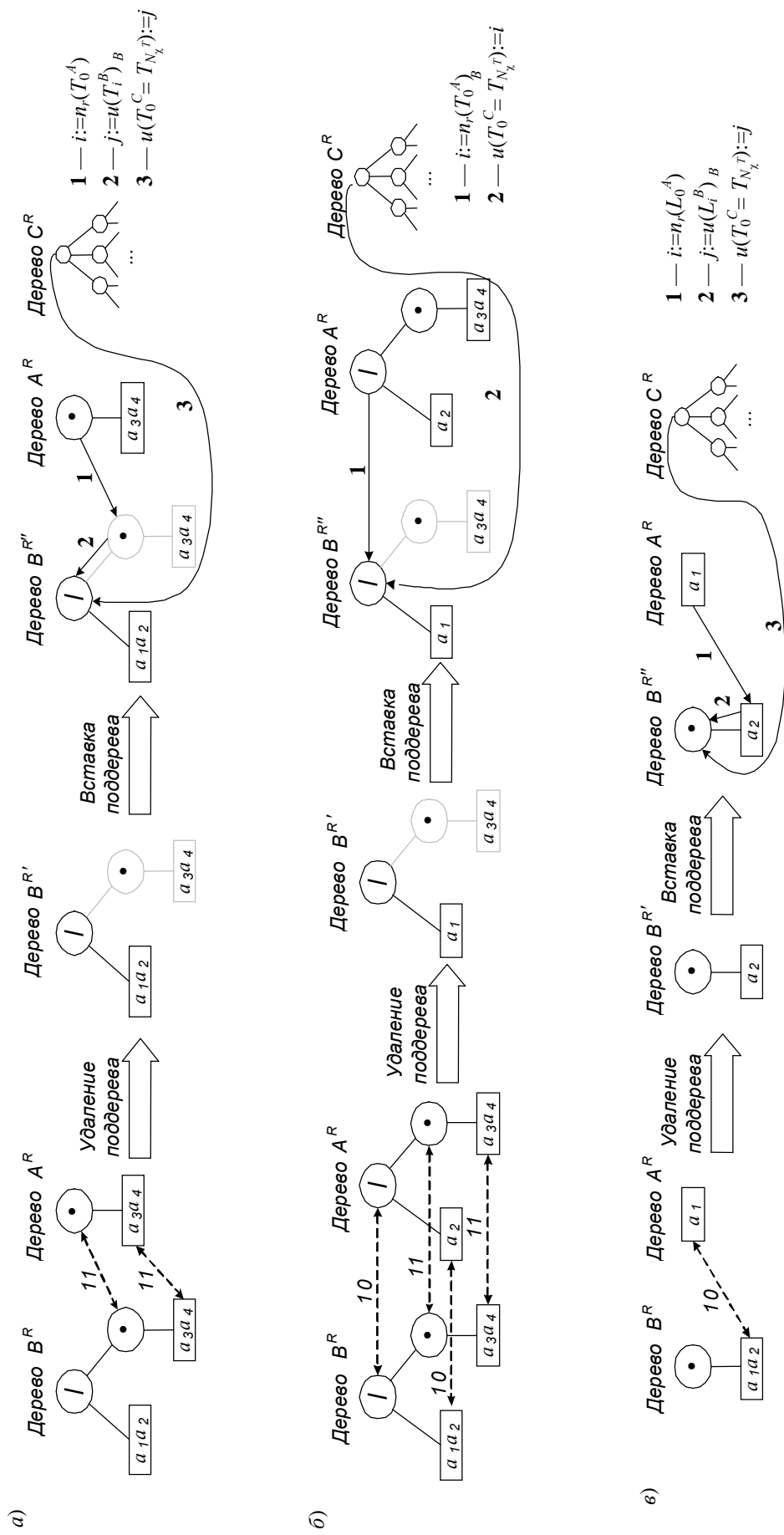


Рис. 2

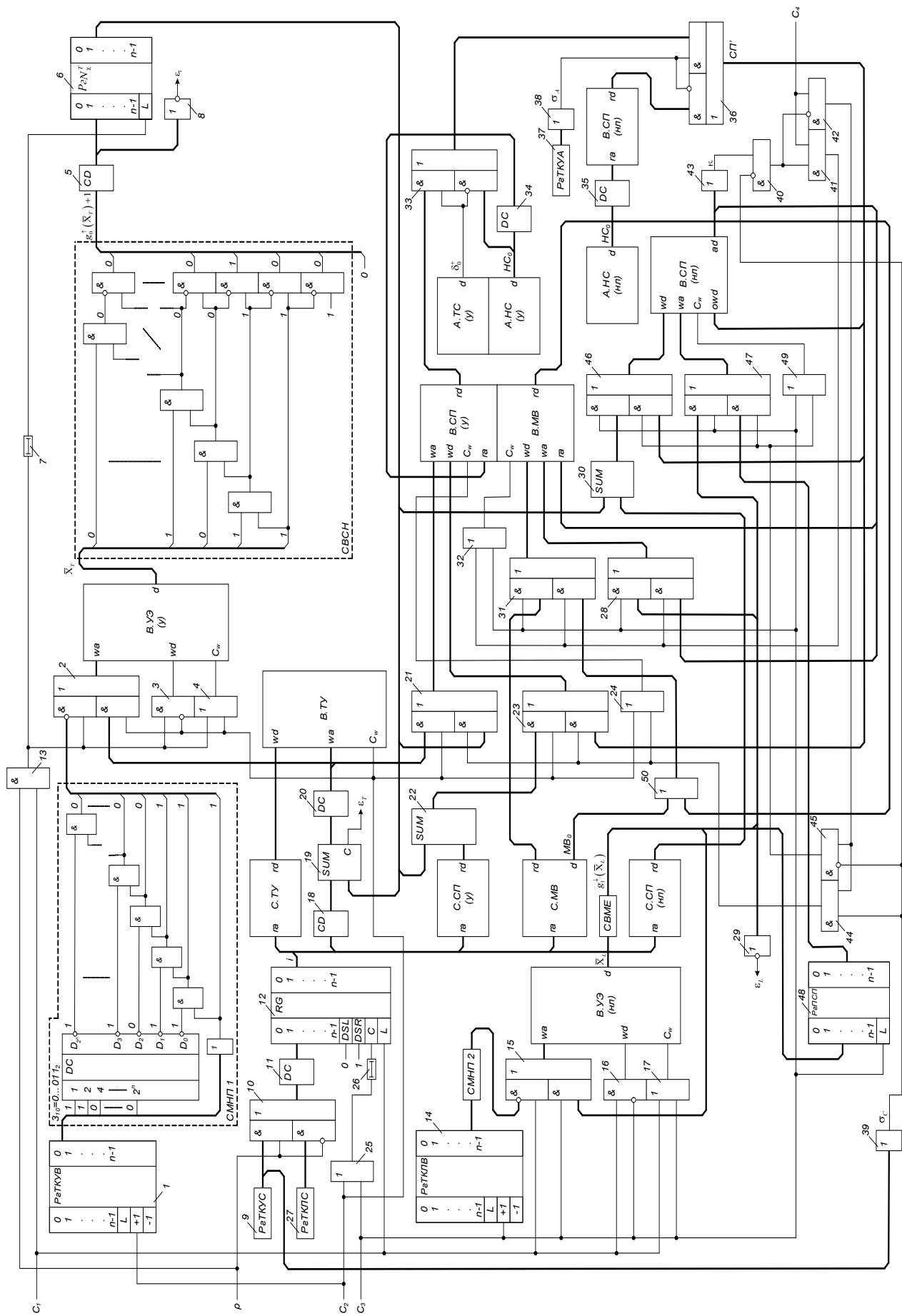


Fig. 3

Коммутатор 33 обеспечивает формирование на своем выходе значения предка корня подставляемого дерева в зависимости от типа соответствия. Дешифратор 34 обеспечивает преобразование номера узла, r -изоморфного корневному узлу дерева A^R , из двоичного кода в унитарный. Дешифратор 35 обеспечивает преобразование номера набора листьев, r -изоморфного нулевому набору листьев дерева A^R , из двоичного кода в унитарный. Коммутатор 36 обеспечивает формирование на своем выходе значения номера узла-предка для корня подставляемого дерева. Регистр 37 хранит значение текущего числа узлов дерева A^R , используемое для формирования значения признака σ_A на выходе элемента ИЛИ 38, подаваемого на вход коммутатора 36. Элемент ИЛИ 39 используется для формирования признака σ_C . Элемент ИЛИ 43 обеспечивает формирование признака κ , используемого в совокупности с элементами 40—42, 44 и 45 для управления прохождением синхросигнала c_4 . Регистр-зашелка 48 используется для хранения значения первой свободной позиции, формируемого на выходе схемы выделения младшей единицы (СВМЕ). Элемент ИЛИ 49 обеспечивает прохождение синхросигнала записи к элементу В.СП(нл) в зависимости от обрабатываемой ситуации. Блок элементов ИЛИ 50 используется при объединении уже имеющегося набора листьев с единственным набором листьев, входящим в состав подставляемого дерева C^R .

Вычислительные затраты на выполнение отдельных элементарных операций рассмотренного выше алгоритма приведены в табл. 2.

Таблица 2

Действие	Программная реализация, шаг, не более	Аппаратная реализация, шаг, не более
Определение значений вектора \mathbf{X}_T	N_T	1
Определения значения N_χ^T	N_T	1
Копирование узлов	N_T	N_T
Определение значений вектора \mathbf{X}_L	N_L	1
Определение младшей свободной позиции в \mathbf{X}_L	N_L	1
Копирование наборов листьев	N_L	N_L
Корректировка СП	N_L	1
Итого	$3N_T + 2N_L + N_L^2$	$N_T + N_L + 4$

Анализ показывает, что при использовании предлагаемого устройства, по сравнению с программной реализацией, преимущество по времени обработки составляет

$$\frac{3N_T + 2N_L + N_L^2}{N_T + N_L + 4} \approx N_R \text{ раз,}$$

где $N_R = N_L + N_T$, что достигается благодаря использованию схем маскировки неиспользуемых позиций (СМНП 1 и 2) и схемы выделения старшего нуля (СВСН), а также возможности ассоциативного поиска информации в памяти акселератора [7, 10].

СПИСОК ЛИТЕРАТУРЫ

1. Организация и синтез микропрограммных мультимикроконтроллеров / И. В. Зотов, В. А. Колосков, В. С. Титов и др. Курск: КурскГТУ, 1999. 368 с.
2. Зотов И. В., Колосков В. А., Титов В. С. Выбор оптимальных разбиений алгоритмов при проектировании микроконтроллерных сетей // Автоматика и вычислительная техника. 1997. № 5. С. 51—62.

3. Ватутин Э. И., Зотов И. В. Метод формирования субоптимальных разбиений параллельных управляющих алгоритмов // Параллельные вычисления и задачи управления (РАСО'04). М.: ИПУ РАН, 2004. С. 884—917.
4. Ватутин Э. И., Зотов И. В. Параллельно-последовательный метод формирования субоптимальных разбиений параллельных управляющих алгоритмов // Свид. об официальной регистрации программы для ЭВМ. № 2005613091 от 28.11.05.
5. Ватутин Э. И., Волобуев С. В., Зотов И. В. Комплексный сравнительный анализ качества разбиений при синтезе логических мультиконтроллеров в условиях присутствия технологических ограничений // Параллельные вычисления и задачи управления (РАСО'08). М.: ИПУ РАН, 2008. С. 643—685.
6. Ватутин Э. И., Зотов И. В. Поиск базового сечения в задаче разбиения параллельных алгоритмов / Курск. гос. техн. ун-т. Курск, 2003. 30 с. Деп. в ВИНТИ. 24.11.03, № 2036-B2003.
7. Ватутин Э. И., Зотов И. В., Титов В. С. Выявление изоморфных вхождений R -выражений при построении параллельных алгоритмов логического управления // Изв. вузов. Приборостроение. 2009. Т. 52, № 2. С. 37—45.
8. Ватутин Э. И., Зотов И. В., Титов В. С. Выявление изоморфных вхождений R -выражений при построении множества сечений параллельных алгоритмов логического управления // Информационно-измерительные и управляющие системы. 2009. Т. 7, № 11. С. 49—56.
9. Ватутин Э. И., Зотов И. В., Титов В. С. Использование схемных формирователей и преобразователей двоичных последовательностей при построении комбинаторно-логических акселераторов // Изв. КурскГТУ. 2008. № 4 (25). С. 32—39.
10. Ватутин Э. И. Однородная среда электронной модели дерева для аппаратно-ориентированной обработки R -выражений // Оптико-электронные приборы и устройства в системах распознавания образов, обработки изображений и символической информации (Расознавание — 2008). Курск: КурскГТУ, 2008. Ч. 1. С. 90—92.

Сведения об авторах

- | | |
|---------------------------------------|--|
| Эдуард Игоревич Ватутин | — канд. техн. наук; Курский государственный технический университет, кафедра вычислительной техники;
E-mail: evatutin@rambler.ru |
| Игорь Валерьевич Зотов | — д-р техн. наук, профессор; Курский государственный технический университет, кафедра вычислительной техники;
E-mail: zotovigor@yandex.ru |
| Виталий Семенович Титов | — д-р техн. наук, профессор; Курский государственный технический университет, кафедра вычислительной техники;
E-mail: titov-kstu@rambler.ru |
| Муджиб Мохаммед Ахья Аль-Ашвал | — аспирант; Курский государственный технический университет, кафедра вычислительной техники; E-mail: mogibm2007@yandex.ru |

Рекомендована кафедрой
вычислительной техники

Поступила в редакцию
14.04.10 г.