Г л а в а 18

Логические устройства для последовательностного вычисления булевых функций

Последовательностные устройства (ПУ), предназначенные для логического управления, называемые логическими машинами в работе [1], программными логическими устройствами в [2], программируемыми логическими контроллерами в [3] или программируемыми контроллерами в [4, 5], могут строиться на базе микропроцессоров [6], микроконтроллеров [7] или быть специализированными [8, 9].

Одной из задач, решаемых устройствами этого класса, является последовательностное вычисление булевых функций (БФУ).

Несмотря на наличие большого числа работ [10—44], связанных с вычислением БФУ, в литературе отсутствуют работы, посвященные анализу и сравнению различных принципов организации специализированных устройств для таких вычислений.

В настоящей главе сделана попытка хотя бы частично устранить этот пробел.

Принципы обработки информации, используемые в специализированных устройствах, могут применяться при программировании микропроцессоров и микроконтроллеров, что кратко изложено в разд. 18.5.

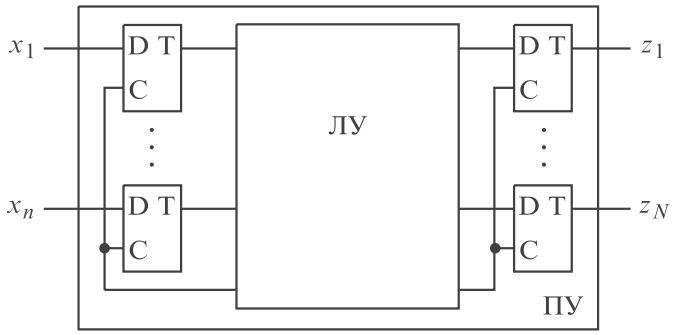

Рис. 18.1

Будем рассматривать специализированные ПУ, которые имеют обобщенную структурную схему, приведенную на рис. 18.1. В этой схеме логическое устройство (ЛУ) может быть построено на основе различных принципов, которые и излагаются в настоящей главе.

Входные *D*-триггеры в этой схеме не позволяют в течение программного цикла изменяться значениям переменных на входах ЛУ, что решает проблему риска для устройств последовательностного действия [45]. При этом ниже будет показано, что существуют последовательностные устройства, в которых входные *D*-триггеры не используются, а проблема риска не возникает.

Выходные *D*-триггеры применяются в этой схеме с целью записи окончательных результатов вычислений в конце программного цикла, что является необходимым при реализации систем булевых функций.

Запись информации в *D*-триггеры, которые в дальнейшем рассматриваться не будут, выполняется по сигналам, формируемым в постоянном запоминающем устройстве (ПЗУ) логического устройства.

В настоящей главе с целью упрощения получения оценок сложности ЛУ основное внимание уделяется вопросу реализации одной булевой функции.

18.1. Операторные устройства

Операторным будем называть устройство (ОУ), последовательностно вычисляющее булеву функцию, используя в качестве команд только логические операции.

При этом если в программе применяются команды пересылки, то они реализуются как логические операции повторения.

Будем называть программы, реализуемые такими устройствами, операторными (ОП). Такие программы имеют структуру, соответствующую последовательному выполнению операторов, так как последовательность их выполнения не зависит от результатов промежуточных вычислений (значений промежуточных или внутренних переменных).

Принципиальная особенность устройств этого класса состоит в необходимости использования одноразрядных ячеек (Я) памяти для запоминания промежуточных результатов. Совокупность этих ячеек образует оперативное запоминающее устройство (ОЗУ).

Другая особенность операторных устройств состоит в применении в общем случае процессорного элемента (ПЭ), который выполняет логические операции. Однако, как будет показано ниже, если ячейки памяти используются не только для запоминания, но и для выполнения логических операций, то при подаче на вход логиче-

ского устройства прямых и инверсных переменных процессорный элемент может быть исключен.

При наличии ПЭ в зависимости от числа его информационных входов он может параллельно обрабатывать с помощью одной команды k переменных (входных и внутренних), где $2 \leqslant \leqslant k < n$.

18.1.1. (k + 1)-Адресные операторные устройства

Наиболее простым (k + 1)-адресным операторным устройством является устройство, схема которого представлена на рис. 18.2.

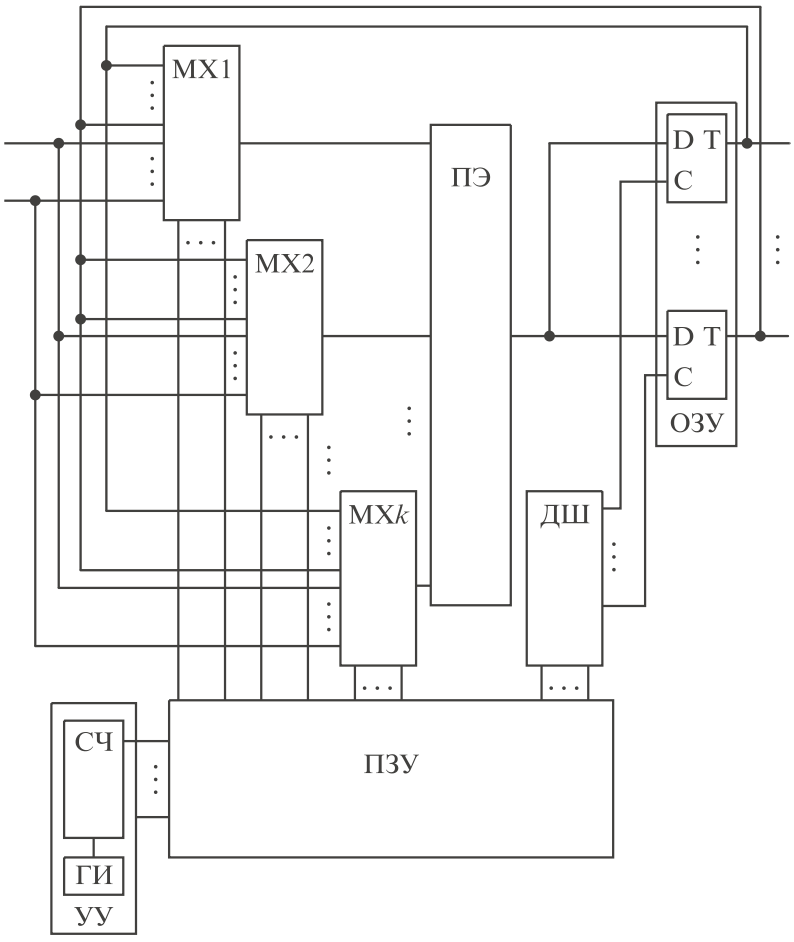

Рис. 18.2

Это устройство названо (k + 1)-адресным, так как в формате его команд указываются k адресов операндов и один адрес результата операции. Будем обозначать символом i число адресов в формате команд операторного устройства.

При этом i-адресное операторное устройство будем обозначать символом ОУ (i), i-адресную программу — символом ОП (i) , а число команд — символом K_i .

В рассмотренном устройстве используются:

— k мультиплексоров (МХ), применяемых в режиме коммутаторов каналов;

— процессорный элемент (ПЭ) с k информационными входами;

— D-триггеры для запоминания промежуточных и окончательных результатов;

— дешифратор (ДШ), выбирающий D-триггер, в который записывается результат операции;

— постоянное запоминающее устройство (ПЗУ) команд;

— устройство управления (УУ), состоящее из двоичного счетчика (СЧ) и генератора импульсов (ГИ).

Обратим внимание на то, что в этом устройстве используется дешифратор, который позволяет записывать полученный результат только в одну ячейку ОЗУ, что, однако, не снижает степени универсальности устройства.

В этом устройстве обычно в качестве ПЭ применяется многофункциональный логический модуль (МЛМ), настраиваемый только константами. Этот МЛМ должен быть универсален либо в классе произвольных булевых функций k переменных, либо в классе произвольных булевых формул в базисах {&, $\vee$, !} или {&, $\vee$, !, $\oplus$} из k букв. Естественно, что МЛМ, универсальный в классе произвольных булевых функций k переменных, универсален также и в указанных классах формул. Поэтому обычно в качестве ПЭ используется мультиплексор «2^k в 1».

Так, например, в [46] в операторном устройстве в качестве процессорного элемента применяется мультиплексор «16 в 1», универсальный в классе произвольных булевых функций четырех и менее переменных.

Программирование операторных устройств наиболее удобно проводить по функциональной схеме, построенной из логических элементов, число входов которых не превышает величины k. При этом число команд в операторной программе совпадает с числом элементов в схеме. Эта схема строится по заданной булевой функции одним из известных или предлагаемых в настоящей работе методов, например мультиплексорным.

Естественно, что операторная программа может строиться также и по схеме, построенной по БФ, или непосредственно по БФ.

Так, например, при применении формульного метода булева формула из h букв, заданная в одном из указанных базисов, реали-

зуется операторной программой из k-универсальных «модулей» с числом команд, определяемым соотношением

$$\left]\frac{h-1}{k-1}\right[\leqslant K_{k+1} \leqslant \left]\frac{2(h-1)}{k}\right[.$$

При $k = 2$

$$K_3 = h - 1.$$

Эта идея, в частности, была использована при $k \geqslant 3$ при написании операторных программ для микропроцессоров [47].

Пример 18.1. При $k = 3$ построить программу типа ОП4, реализующую булеву функцию, которой соответствует формула

$$z = x_1 (x_2 \vee x_3 \vee x_4) \vee x_2 x_3 x_4.$$

При применении формульного метода и так как $h = 7$, то

$$3 \leqslant K_4 \leqslant 4.$$

При этом программа имеет следующий вид:

$$x_2 \vee x_3 \vee x_4 \to Я_1;\ x_2 x_3 x_4 \to Я_2;\ x_1 Я_1 \vee Я_2 \to Я_1.$$

С целью упрощения программы используем мультиплексорный метод и реализуем заданную БФУ формулой вида:

$$z = x_1 \# (x_1 \# x_2 \# x_3) \# x_4,$$

где $\#$ — символ операции мажорирования для функции трех переменных.

При этом программа упрощается и приобретает следующий вид:

$$x_1 \# x_2 \# x_3 \to Я_1;\ x_2 \# Я_1 \# x_4 \to Я_1.$$

Пример 18.2. При $k = 3$ построить программу типа ОП4, реализующую параллельный двоичный одноразрядный сумматор.

Одно из описаний сумматора имеет вид:

$$P = x_1 \# x_2 \# x_3;\ S = !P (x_1 \vee x_2 \vee x_3) \vee x_1 x_2 x_3.$$

Из предыдущего примера следует, что

$$P = x_1 \# x_2 \# x_3;\ S = !P \# (!P \# x_1 \# x_2) \# x_3.$$

Эта система реализуется следующей программой:

$$1.\ x_1 \barwedge x_2 \barwedge x_3 \to Я_1;\quad 3.\ Я_2 \barwedge x_1 \barwedge x_2 \to Я_3;$$
$$2.\ !\, Я_1 \to Я_2;\qquad\qquad 4.\ Я_2 \barwedge Я_3 \barwedge x_3 \to Я_2.$$

Сумматор может быть реализован также программой из двух команд, если для ее написания применить следующую систему булевых формул:

$$P = x_1 \barwedge x_2 \barwedge x_3;\ S = x_1 \oplus x_2 \oplus x_3.$$

Из изложенного следует, что для рассматриваемого класса устройств число команд и время вычисления БФУ зависят от значения k. От этого значения зависит и длина команд этих устройств:

$$L_{k+1} = k\,]\log\,(n + Я)[\,+\,]\log Я[\,+2^k,$$

где $Я$ — требуемое число ячеек ОЗУ.

С целью упрощения устройства и ограничения длины команд на практике в большинстве случаев используются операторные устройства с $k = 2$. Перейдем к рассмотрению таких устройств.

18.1.2. Трехадресное операторное устройство

Пусть заданная булева функция n переменных представлена булевой формулой из h букв в одном из рассматриваемых базисов. Эта формула может быть реализована формульным методом схемой из $h - 1$ двухвходовых элементов.

Определим оценки сложности устройства ОУ3, моделирующего эту схему. Если при написании программы число ячеек ОЗУ минимизируется, то

$$2 \leqslant Я_3 \leqslant [\log h].$$

Выберем для этого устройства

$$Я_3 = [\log h],$$

при этом число обрабатываемых переменных

$$P_3 = n + Я_3 = n + [\log h].$$

Число команд в устройстве

$$K_3 = h - 1.$$

Эти команды имеют следующий формат: адрес первого операнда, адрес второго операнда, код логической операции, адрес результата.

Применяя в качестве процессорного элемента мультиплексор «4 в 1» в режиме модуля, универсального в классе произвольных БФУ двух переменных, получим соотношение для определения длины команд рассматриваемого устройства:

$$L_3 = 2]\log(n + [\log h])[+]\log[\log h][+4.$$

Будем использовать в качестве характеристики, по которой сравниваются логические устройства, такой показатель, как площадь ПЗУ команд.

Для рассматриваемого устройства

$$S_3 = K_3 L_3.$$

При $h = n \to \infty$

$$K_3 \to h; \quad L_3 \to 2\log h; \quad S_3 \to 2h\log h.$$

В [48] показано, что для булевой функции n переменных может быть построена нормальная булева формула в базисе {&, ∨, !}, число букв h в которой при $n \to \infty$ определяется соотношением

$$h \to \frac{2^n}{\log n}.$$

При этом

$$K_3 \to \frac{2^n}{\log n}; \quad L_3 \to 3\log n; \quad S_3 \to 3 \cdot 2^n.$$

Структурная схема устройства ОУ3 получается из схемы, приведенной на рис. 18.2, при $k = 2$.

Пример 18.3. Реализовать программой типа ОП3 булеву формулу

$$z = (x_1 \vee x_2)(x_3 \vee x_4) \vee (x_5 \vee x_6)(x_7 \vee x_8).$$

В этом случае $K_3 = 7$:

$$\begin{array}{ll} 1.\ x_1 \vee x_2 \to Я_1; & 5.\ x_7 \vee x_8 \to Я_3; \\ 2.\ x_3 \vee x_4 \to Я_2; & 6.\ Я_2 \,\&\, Я_3 \to Я_2; \\ 3.\ Я_1 \,\&\, Я_2 \to Я_1; & 7.\ Я_1 \vee Я_2 \to Я_1. \\ 4.\ x_5 \vee x_6 \to Я_2; & \end{array}$$

18.1.3. 2.5-Адресное операторное устройство

Устройство ОУ2.5 получается из устройства ОУ3 за счет исключения связей входных или внутренних переменных со входами второго мультиплексора.

Так как в общем случае число входных переменных больше, то исключим связи этого мультиплексора с такими переменными (рис. 18.3).

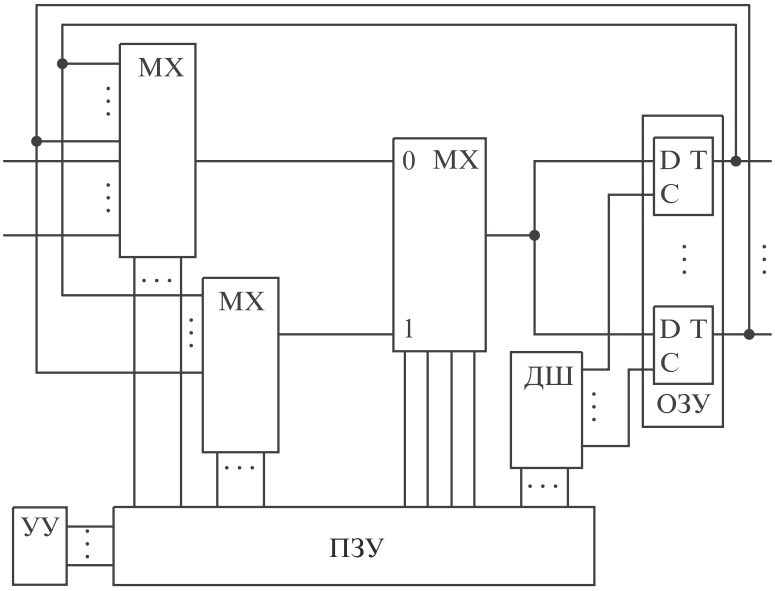

Рис. 18.3

При этом длина команд

$$L_{2.5} = 2]\log\,(n + [\log h])[+2]\log\,[\log h][+4$$

уменьшается за счет увеличения их числа

$$h \leqslant K_{2.5} \leqslant h + [h/2] - 1 = [(3h/2) - 1].$$

Составляющими этой верхней оценки являются: $[h/2]$ — число команд ввода входных переменных; $h - 1$ — число команд логической обработки.

Таким образом, в отличие от устройства ОУ3 уже в устройстве ОУ2.5 появляются команды ввода, «пустые» от логической обработки.

При $h = n \to \infty$

$$h \lesssim K_{2.5} \lesssim 3h/2;\ L_{2.5} \to \log h;\ \ h\log h \lesssim S_{2.5} \lesssim (3h/2)\log h.$$

При $n \to \infty$

$$\frac{2^n}{\log n} \lesssim K_{2.5} \lesssim 3\frac{2^{n-1}}{\log n};\ \ L_{2.5} \to 3\log n;\ \ 3\cdot 2^n \lesssim S_{2.5} \lesssim 4.5\cdot 2^n.$$

Пример 18.4. Реализовать программой типа ОП 2.5 булеву формулу, рассмотренную в предыдущем примере.

В этом случае $8 \leqslant K_{2.5} \leqslant 11$:

$1.\ x_1 \to Я_1;$	$5.\ Я_1 \,\&\, Я_2 \to Я_1;$	$9.\ x_8 \vee Я_3 \to Я_3;$
$2.\ x_2 \vee Я_1 \to Я_1;$	$6.\ x_5 \to Я_2;$	$10.\ Я_2 \vee Я_3 \to Я_2;$
$3.\ x_3 \to Я_2;$	$7.\ x_6 \vee Я_2 \to Я_2;$	$11.\ Я_1 \vee Я_2 \to Я_1.$
$4.\ x_4 \vee Я_2 \to Я_2;$	$8.\ x_7 \to Я_3;$	

18.1.4. Двухадресное операторное устройство

В схеме на рис. 18.3 исключим второй входной мультиплексор, а освободившийся вход процессорного мультиплексора соединим с выходом одного из *D*-триггеров, который выделим из ОЗУ и назовем аккумулятором (АК). При этом получается устройство ОУ2 (рис. 18.4).

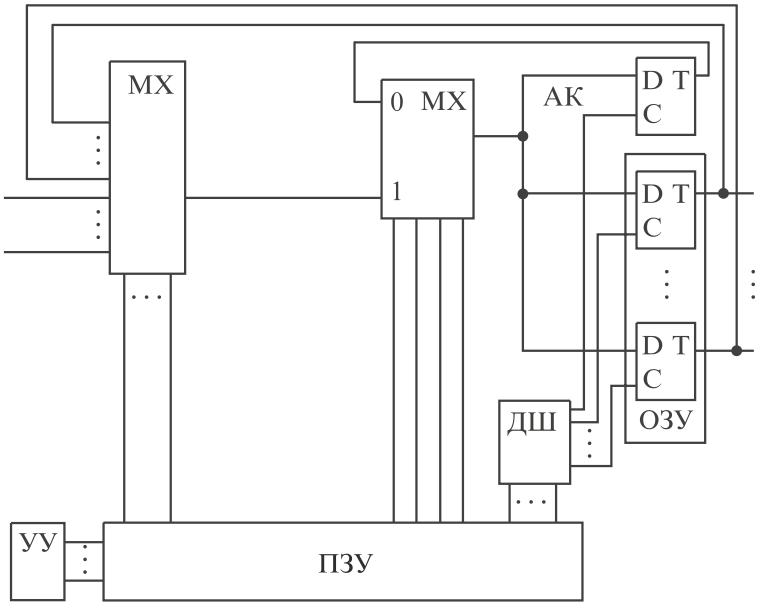

Рис. 18.4

В этом устройстве число команд по сравнению с устройством ОУ2.5 не изменяется:

$$h \leqslant K_2 \leqslant h + [h/2] - 1 = [(3h/2) - 1].$$

Длина команд в этом случае определяется соотношением

$$L_2 =]\log (n + [\log (h-1)])[+]\log [\log h][+ 4.$$

При $h = n \to \infty$

$$h \lesssim K_2 \lesssim 3h/2;\ L_2 \to \log h;\ h \log h \lesssim S_2 \lesssim (3h/2) \log h.$$

При $n \to \infty$

$$\frac{2^n}{\log n} \lesssim K_2 \lesssim \frac{3}{2}\frac{2^n}{\log n};\ L_2 \to 2 \log n;\ 2^{n+1} \lesssim S_2 \lesssim 3 \cdot 2^n.$$

Пример 18.5. Реализовать программой типа ОП2 булеву формулу, рассмотренную в примере *18.3*.

В этом случае $8 \leqslant K_2 \leqslant 11$:

$1.\ x_1 \to A;$	$5.\ Я_1 \,\&\, A \to Я_1;$	$9.\ x_8 \vee A \to A;$
$2.\ x_2 \vee A \to Я_1;$	$6.\ x_5 \to A;$	$10.\ Я_2 \,\&\, A \to A;$
$3.\ x_3 \to A;$	$7.\ x_6 \vee A \to Я_2;$	$11.\ Я_1 \vee A \to Я_1.$
$4.\ x_4 \vee A \to A;$	$8.\ x_7 \to A;$	

18.1.5. Одноадресное операторное устройство

Совместим в устройстве ОУ2 поля адресов. Введем в ПЗУ дополнительный разряд, обеспечивающий различие аккумулятора и ОЗУ, а также двухвходовые элементы И, соединенные со входами дешифратора. В результате получается устройство ОУ1 (рис. 18.5).

Число команд в этом случае определяется соотношением

$$h + 1 \leqslant K_1 \leqslant 2h - 1.$$

Составляющими верхней оценки являются: h — число команд пересылки; $h - 1$ — число команд логической обработки.

Длина команд в устройстве этого типа определяется соотношением

$$L_1 =]\log \left(n + [\log (h-1)] \right)[+ 5.$$

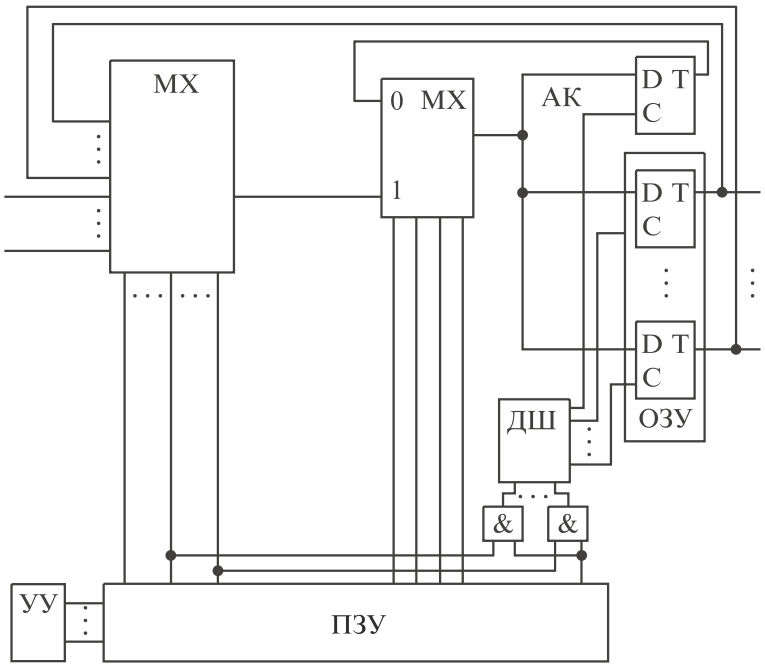

Рис. 18.5

При $h = n \to \infty$

$$h \lesssim K_1 \lesssim 2h; \quad L_1 \to \log h; \quad h \log h \lesssim S_1 \lesssim 2h \log h.$$

При $n \to \infty$

$$\frac{2^n}{\log n} \lesssim K_1 \lesssim \frac{2^{n+1}}{\log n}; \quad L_1 \to \log n; \quad 2^n \lesssim S_1 \lesssim 2^{n+1}.$$

Пример 18.6. Реализовать программой типа ОП1 булеву формулу, рассмотренную в примере *18.3*.

В этом случае $9 \leqslant K_1 \leqslant 15$:

$1.\ x_1 \to A;$	$6.\ Я_1 \& A \to A;$	$11.\ x_7 \to A;$
$2.\ x_2 \vee A \to A;$	$7.\ A \to Я_1;$	$12.\ x_8 \vee A \to A;$
$3.\ A \to Я_1;$	$8.\ x_5 \to A;$	$13.\ Я_2 \& A \to A;$
$4.\ x_3 \to A;$	$9.\ x_6 \vee A \to A;$	$14.\ Я_1 \vee A \to A;$
$5.\ x_4 \vee A \to A;$	$10.\ A \to Я_2;$	$15.\ A \to Я_1.$

Многие ПЛК могут реализовывать программы типа ОП1. Так, для ПЛК «Autolog» фирмы «FF-Automation OY» [49] рассмотренная

программа может быть реализована на языке инструкций и записана для экономии места в два столбца:

```
STR  I  x1    OR   I  x6
OR   I  x2    EQ   M  Я2
EQ   M  Я1    STR  I  x7
STR  I  x3    OR   I  x8
OR   I  x4    AND  M  Я2
AND  M  Я1    OR   M  Я1
EQ   M  Я1    EQ   O  z
STR  I  x2    STOP
```

18.1.6. Операторные устройства, универсальные в классе формул в базисе И, ИЛИ, НЕ

В рассмотренных выше устройствах при $k = 2$ в качестве процессорного элемента использовался мультиплексор «4 в 1» с четырьмя настроечными входами. Если ограничить класс реализуемых формул базисом {&, ∨, !}, то число настроечных входов может быть уменьшено до трех.

Для устройств ОУ3 и ОУ2.5 процессорный элемент приведен на рис. 18.6, а для устройств ОУ2 и ОУ1 — на рис. 18.7.

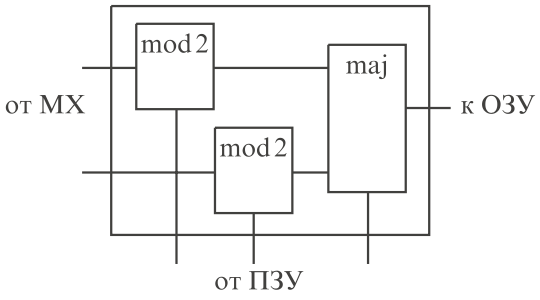

Рис. 18.6

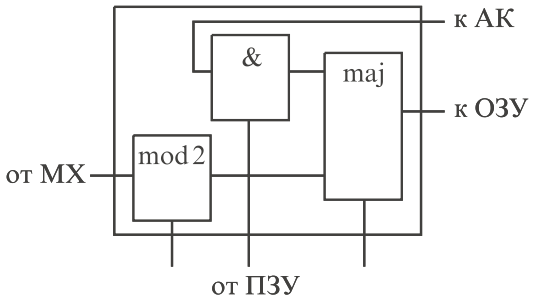

Рис. 18.7

В схеме на рис. 18.7 элемент И установлен для обеспечения возможности применения мажоритарного элемента в режиме повторителя переменной, поступающей с выхода мультиплексора.

При использовании таких процессорных элементов для рассматриваемого класса формул могут применяться оценки сложности устройств, приведенные выше, с учетом того факта, что число настроечных входов процессорного элемента уменьшается на один.

18.1.7. Операторные устройства с микропрограммной реализацией операторов

В рассмотренных устройствах процессорный элемент реализовывался аппаратно. Если его реализовать в виде последовательностной схемы, то получится двухуровневое устройство, верхний уровень которого называется программным, а нижний — микропрограммным [39].

При этом оператор в каждой команде программы интерпретируется микропрограммой.

На каждом из указанных уровней может использоваться свой принцип логической обработки информации, например на верхнем уровне — операторная обработка, а на нижнем — бинарная.

18.1.8. Операторные устройства без кода операций

В качестве процессорного элемента в устройствах ОУ3 и ОУ2.5 можно применять двухвходовой элемент, обладающий свойством функциональной полноты, например элемент И—НЕ. При этом в формате команд исключается поле «код операции». При программировании заданная булева формула реализуется схемой в базисе этих элементов, а затем моделируется рассматриваемым устройством. Отметим, что для моделирования инверторов на соответствующие входы мультиплексоров должна быть подана константа «единица».

Устройство ОУ2.5 этого типа приведено на рис.18.8.

18.1.9. Операторные устройства с *RS*-триггерами

Если в рассмотренных устройствах *D*-триггеры заменить на триггерные ячейки, каждая из которых состоит из *RS*-триггера и двухвходовых элементов И и ЗАПРЕТ, то при использовании этих

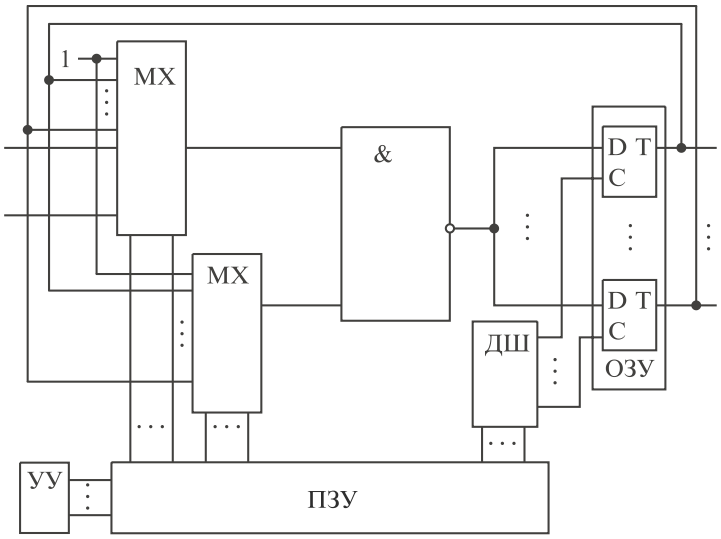

Рис. 18.8

ячеек только для запоминания информации все приведенные выше результаты сохраняются (рис. 18.9).

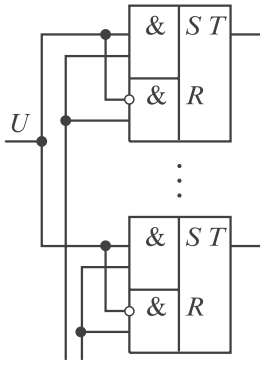

Рис. 18.9

При этом отметим, что в этих ячейках элемент ЗАПРЕТ может быть заменен на элемент И, а для всех ячеек может применяться один инвертор. Однако эти ячейки кроме функции запоминания могут выполнять также и логические операции. Это позволяет сократить число команд в программе при увеличении их длины, так как в этом случае приходится отказаться от дешифратора, ввиду того что в ряде случаев активными могут быть оба входа ячейки, обозначенные символами & и $\vee$ (рис. 18.10).

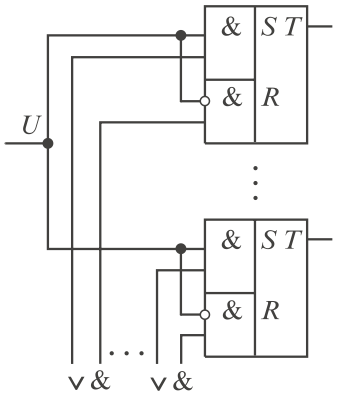

Рис. 18.10

Функционирование одной ячейки в этом случае описывается табл. 18.1.

Т а б л и ц а 18.1

∨	&	*UQ*			
		00	01	10	11
0	0	0	1	0	1
0	1	0	0	0	1
1	0	0	1	1	1
1	1	0	0	1	1

Эта таблица может быть записана и иначе (табл. 18.2).

Т а б л и ц а 18.2

∨	&	*Q*
0	0	*Q*
0	1	*U* & *Q*
1	0	*U* ∨ *Q*
1	1	*U*

Из этой таблицы следует, что описанная триггерная ячейка позволяет реализовать функции & и ∨.

1. Рассмотрим трехадресное операторное устройство этого типа, которое обозначим символом ОУ31.

Пример 18.7. Реализовать БФ из примера *18.3* с помощью рассмотренного устройства.

При $\vee = 1, \& = 1$ $\quad x_1 \vee x_2 \to Я_1$;
при $\vee = 0, \& = 1$ $\quad (x_3 \vee x_4) \& Я_1 \to Я_1$;
при $\vee = 1, \& = 1$ $\quad x_5 \vee x_6 \to Я_2$;
при $\vee = 0, \& = 1$ $\quad (x_7 \vee x_8) \& Я_2 \to Я_2$;
при $\vee = 1, \& = 1$ $\quad Я_1 \vee Я_2 \to z$.

Таким образом, получили наиболее компактную программу из рассмотренных.

Оценим число команд в этом устройстве. Из приведенного примера следует, что в данном случае используются двух- и трехместные команды. При $h = 2k$ существуют булевы формулы, при реализации которых первая команда является двухместной, а остальные — трехместными. Поэтому

$$K_{31\,н} = \frac{(h-1)-1}{3-1} + 1 = \frac{h}{2}.$$

При $h = 2k + 1$ существуют булевы формулы, при реализации которых только первая и последняя команды являются двухместными, а остальные — трехместными. Поэтому

$$K_{31\,н} = \frac{(h-2)-1}{3-1} + 2 = \frac{h+1}{2}.$$

Таким образом,

$$K_{31\,н} =]h/2[.$$

При любых h существуют булевы формулы, при реализации которых все команды являются двухместными, как например это имеет место для формулы

$$z = (x_1 \vee x_2)x_3 \vee (x_4 \vee x_5)x_6.$$

При этом

$$K_{31в} = h - 1.$$

Следовательно,

$$]h/2[\leqslant K_{31} \leqslant h - 1.$$

Оценим число ячеек в ОЗУ.

При $h = 3, \ldots, 5$ требуется одна ячейка; при $h = 6, \ldots, 11$ — две; при $h = 12, \ldots, 23$ — три; при $h = 24, \ldots, 47$ — четыре; при $h = 48, \ldots, 95$ — пять ячеек и т. д.

При $h_н = 3, 6, 12, 24, 48, \ldots = 3 \cdot 2^{Я-1}$; $Я_н = 1 + \log(h_н/3) = \log(2h_н/3)$.

При $h_н = 5, 11, 23, 47, 95, \ldots = 2h_н - 1 = 3 \cdot 2^{Я} - 1$; $Я_в = \log(h_в + 1)/3$.

Следовательно,

$$Я_{max} = [\log(2h/3)] =]\log\big((h+1)/3\big)[;\ 3 \cdot 2^{Я-1} \leqslant h \leqslant 3 \cdot 2^{Я} - 1.$$

При этом

$$L_{31} = 2]\log\big(n + [\log(2h/3)]\big)[+2\big([\log(2h/3)]+1\big) + 4.$$

При $h = n \to \infty$

$$h/2 \lesssim K_{31} \lesssim h;\ \ L_{31} \to 2\big(\log h + \log(2h/3)\big);$$
$$h\big(\log h + \log(2h/3)\big) \lesssim S_{31} \lesssim 2h\big(\log h + \log(2h/3)\big).$$

При $n \to \infty$

$$\frac{2^{n-1}}{\log n} \lesssim K_{31} \lesssim \frac{2^n}{\log n};\ \ L_{31} \to 2n;\ \ n\frac{2^n}{\log n} \lesssim S_{31} \lesssim n\frac{2^{n+1}}{\log n}.$$

2. Рассмотрим двухадресное операторное устройство этого типа, которое обозначим символом ОУ21.

Так как приведенные выше триггерные ячейки реализуют операции & и $\vee$, то для реализации инверсий будем использовать двухвходовой элемент НЕРАВНОЗНАЧНОСТЬ, работающий в режиме «повторитель — инвертор» (рис. 18.11).

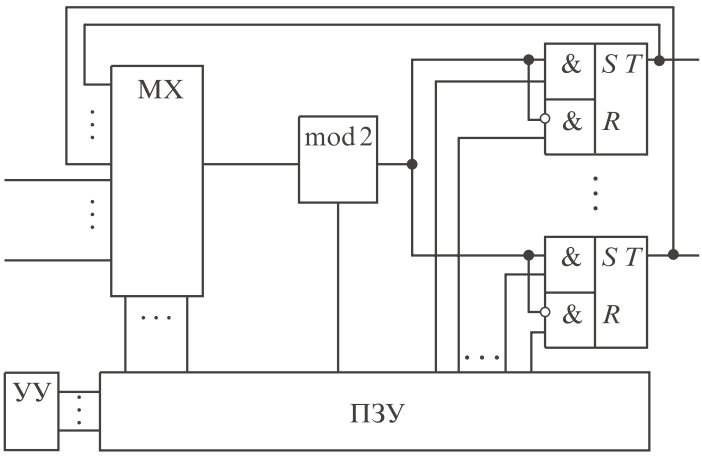

Рис. 18.11

Пример 18.8. Реализовать БФ из примера *18.3* с помощью рассмотренного устройства.

Программа в данном случае имеет следующий вид:

$1.\ x_1 \to Я_1;$	$5.\ Я_1 \,\&\, Я_2 \to Я_1;$	$9.\ x_8 \vee Я_3 \to Я_3;$
$2.\ x_2 \vee Я_1 \to Я_1;$	$6.\ x_5 \to Я_2;$	$10.\ Я_2 \,\&\, Я_3 \to Я_2;$
$3.\ x_3 \to Я_2;$	$7.\ x_6 \vee Я_2 \to Я_2;$	$11.\ Я_1 \vee Я_2 \to Я_1.$
$4.\ x_4 \vee Я_2 \to Я_2;$	$8.\ x_7 \to Я_3;$	

Для рассмотренного устройства

$$h+1 \leqslant K_{21} \leqslant h+[h/2]=[3h/2]; \quad Я_{1\max}=[\log h];$$

$$L_{21}=]\log(n+[\log h])[+2[\log h]+1.$$

При $h=n\to\infty$

$$h \lesssim K_{21} \lesssim 3h/2; \quad L_{21}\to 3\log h; \quad 3h\log h \lesssim S_{21} \lesssim 4.5\log h.$$

При $n\to\infty$

$$\frac{2^n}{\log n} \lesssim K_{21} \lesssim 3\frac{2^{n-1}}{\log n}; \quad L_{21}\to 2n; \quad n\frac{2^{n+1}}{\log n} \lesssim S_{21} \lesssim 3n\frac{2^n}{\log n}.$$

Из изложенного следует, что если на входы мультиплексора подать прямые и инверсные переменные, то элемент НЕРАВНОЗНАЧНОСТЬ может быть исключен, а получающееся операторное устройство не будет содержать процессорного элемента.

18.1.10. Стековое операторное устройство

Еще одним подклассом операторных устройств являются устройства, в которых триггеры, образующие ОЗУ, заменены стеком. Будем называть такие устройства стековыми операторными устройствами (СОУ).

Одноадресное стековое операторное устройство (СОУ1) приведено на рис. 18.12.

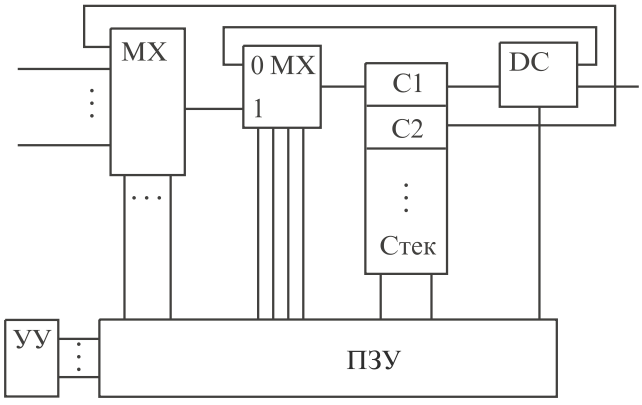

Рис. 18.12

Это устройство может реализовывать стековые операторные программы, написанные либо на основе обратной польской записи, либо непосредственно по формуле или лестничной схеме [50].

1. При реализации на основе обратной польской записи справедливы следующие соотношения:

$$K_{\text{сп}} = 2h; \quad L_{\text{сп}} =]\log(n+1)[+ 7; \quad Я_{\text{сп}} = [1 + \log h].$$

При $h = n \to \infty$

$$K_{\text{сп}} = 2h; \quad L_{\text{сп}} \to \log h; \quad S_{\text{сп}} \to 2h \log h.$$

При $n \to \infty$

$$K_{\text{сп}} \to \frac{2^n}{\log n}; \quad L_{\text{сп}} \to \log n; \quad S_{\text{сп}} \to 2^{n+1}.$$

Пример 18.9. Реализовать формулу из примера *18.3* с помощью рассмотренного устройства.

Эта формула в обратной польской записи имеет вид:

$$x_1 x_2 \vee x_3 x_4 \vee \& \; x_5 x_6 \vee x_7 x_8 \vee \& \; \vee.$$

Поэтому в данном случае стековая программа может быть записана следующим образом:

$1. x_1 \to c_1;$	$9. x_6 \to c_1 c_2;$
$2. x_2 \to c_1 c_2;$	$10. c_1 \vee c_2 \to c_1 c_2;$
$3. c_1 \vee c_2 \to c_1;$	$11. x_7 \to c_1 c_2 c_3;$
$4. x_3 \to c_1 c_2;$	$12. x_8 \to c_1 c_2 c_3 c_4;$
$5. x_4 \to c_1 c_2 c_3;$	$13. c_1 \vee c_2 \to c_1 c_2 c_3;$
$6. c_1 \vee c_2 \to c_1 c_2;$	$14. c_1 \& \; c_2 \to c_1 c_2;$
$7. c_1 \& \; c_2 \to c_1;$	$15. c_1 \vee c_2 \to c_1;$
$8. x_5 \to c_1 c_2;$	$16. c_1 \to z;$

Используя язык инструкций ПЛК фирмы «Omron» (Япония) [50], эта программа, записанная в два столбца, приобретает следующий вид:

```
LD   x1        LD   x6
LD   x2        OR   LD
OR   LD        LD   x7
LD   x3        LD   x8
LD   x4        OR   LD
OR   LD        AND  LD
AND  LD        OR   LD
LD   x5        OUT
```

2. При реализации непосредственно по формуле (лестничной схеме) справедливы следующие соотношения:

$$h+1 \leqslant K_{\text{сф}} \leqslant [3h/2]; \quad L_{\text{сф}} =]\log(n+1)[+7; \quad Я_{\text{сф}} = [\log h].$$

При $h = n \to \infty$

$$h+1 \leqslant K_{\text{сф}} \leqslant [3h/2]; \quad L_{\text{сф}} \to \log h; \quad h\log h \lesssim S_{\text{сф}} \lesssim 1.5h\log h.$$

При $n \to \infty$

$$\frac{2^n}{\log n} \lesssim K_{\text{сф}} \lesssim 1.5\frac{2^n}{\log n}; \quad L_{\text{сф}} \to \log n; \quad 2^n \lesssim S_{\text{сф}} \lesssim 1.5 \cdot 2^n.$$

Именно эти оценки и будем применять в качестве оценок одноадресного стекового операторного устройства.

Пример 18.10. Реализовать формулу из примера *18.3*.
Программа в этом случае имеет следующий вид:

$1. x_1 \to c_1;$ $7. x_6 \vee c_1 \to c_1 c_2;$
$2. x_2 \vee c_1 \to c_1;$ $8. x_7 \to c_1 c_2 c_3;$
$3. x_3 \to c_1 c_2;$ $9. x_8 \vee c_1 \to c_1 c_2 c_3;$
$4. x_4 \vee c_1 \to c_1 c_2;$ $10. c_1 \& c_2 \to c_1 c_2;$
$5. c_1 \& c_2 \to c_1;$ $11. c_1 \vee c_2 \to c_1;$
$6. x_5 \to c_1 c_2;$ $12. c_1 \to z.$

Используя язык инструкций ПЛК фирмы «Omron», эта программа, записанная в два столбца, приобретает следующий вид:

```
LD   x1        OR   x6
OR   x2        LD   x7
LD   x3        OR   x8
OR   x4        AND  LD
AND  LD        OR   LD
LD   x5        OUT
```

18.1.11. Буквенное операторное устройство

Все рассмотренные выше операторные устройства обладали тем недостатком, что они не позволяли с помощью к а ж д о й команды обрабатывать одну букву заданной булевой формулы.

Ниже предлагается п р и н ц и п и а л ь н о н о в о е операторное устройство, которое позволяет побуквенно, справа налево вычислять с помощью h команд без предварительного преобразования произвольную (в том числе скобочную произвольной глубины) нормальную булеву формулу в базисе {&, ∨, !} из h букв. Таким обра-

зом, каждая команда рассматриваемого устройства обрабатывает одну букву заданной формулы.

При этом после выполнения *i*-й команды вычисленным является фрагмент булевой формулы из *i* букв, который может не быть подформулой вычисляемой формулы.

Рассматриваемое устройство является наиболее «последовательным», а применяемый принцип обработки логической информации является более «красивым» по сравнению с операторными устройствами, описанными выше. Назовем это устройство буквенным операторным устройством (БОУ).

Предлагаемое устройство программируется совершенно необычным для операторных устройств способом — начиная с построения линейного бинарного графа (ЛБГ), используемого традиционно для написания программ бинарных устройств.

Пример 18.11. Реализовать булеву формулу $z = (x_1 \vee x_2)x_3 \vee x_4$ с помощью предлагаемого подхода.

Построим ЛБГ (рис. 18.13).

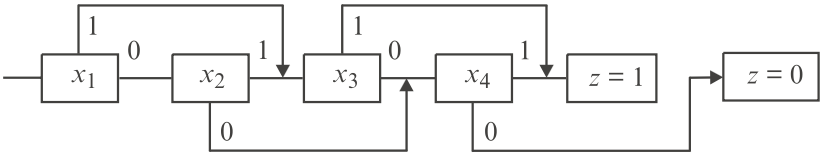

Рис. 18.13

Изменим в ЛБГ пометки так, чтобы все дуги остова были помечены единицами (рис. 18.14).

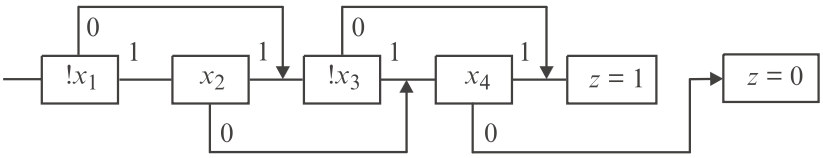

Рис. 18.14

Построим по этой ЛБГ мультиплексорный каскад (рис. 18.15).

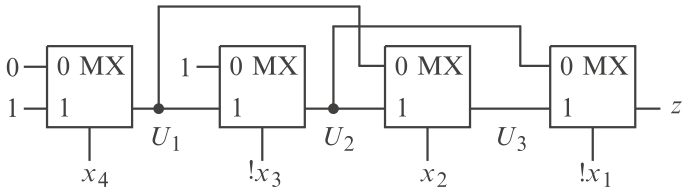

Рис. 18.15

Выполним верификацию построенного каскада:

$$U_1 = 0 \,\&\, !x_4 \vee 1 \,\&\, x_4 = x_4;$$
$$U_2 = 1 \,\&\, !(!x_3) \vee U_1 !x_3 = x_3 \vee x_4;$$
$$U_3 = U_1 !x_2 \vee U_2 x_2 = x_2 x_3 \vee x_4;$$
$$U_4 = U_2 !(!x_1) \vee U_3 !x_1 = (x_1 \vee x_2) x_3 \vee x_4.$$

Обратим внимание на тот факт, что в заданной булевой формуле подформула U_3, совпадающая с ее фрагментом, отсутствует.

Для моделирования этого каскада, содержащего две длинные связи, достаточно иметь предлагаемое устройство с тремя *D*-триггерами для запоминания промежуточных результатов (рис. 18.16).

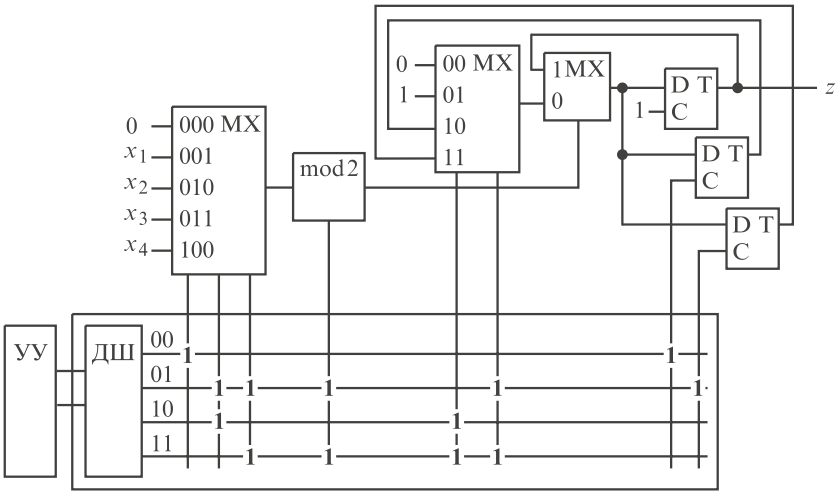

Рис. 18.16

Для обеспечения правильной работы устройства в начальном состоянии в первом *D*-триггере ОЗУ должна находиться единица, которая записывается в этот триггер последней (служебной) командой программы.

При этом программа может быть записана следующим образом:

$$\text{MX}\,(0, 1; x_4) \to Я_1 Я_3; \qquad /^* U_1 = x_4$$
$$\text{MX}\,(1, Я_1; !x_3) \to Я_1 Я_3; \qquad /^* U_2 = x_3 \vee x_4$$
$$\text{MX}\,(Я_2, Я_1; x_2) \to Я_1; \qquad /^* U_3 = x_2 x_3 \vee x_4$$
$$\text{MX}\,(Я_3, Я_1; !x_1) \to Я_1; \qquad /^* z = (x_1 \vee x_2) x_3 \vee x_4.$$

Так как в этом случае

$$1 \leqslant Я_б \leqslant h - 1,$$

то устройство должно содержать число *D*-триггеров в ОЗУ, определяемое выражением

$$Я_б = h - 1.$$

Длина команд в таком устройстве определяется соотношением

$$L_б =]\log(n+1)[+]\log(h-1)[+ 3.$$

При $h = n \to \infty$

$$K_б = h; \quad L_б \to 2\log h; \quad S_б \to 2h\log h.$$

При $n \to \infty$

$$K_б \to \frac{2^n}{\log n}; \quad L_б \to 2n; \quad S_б \to n\frac{2^{n+1}}{\log n}.$$

Описанное устройство предложено автором в [51], а класс операторных устройств рассмотрен в [52].

18.2. 2^k-нарные устройства

2^k-нарным будем называть устройство, последовательностно вычисляющее булеву функцию и использующее при этом только условные переходы по значениям входных переменных с 2^k исходами и присваивания выходным переменным констант.

Принципиальная особенность этого класса устройств состоит в том, что в них нет необходимости применять оперативное запоминающее устройство для запоминания промежуточных результатов и процессорный элемент для выполнения логических операций. При этом отметим, что ОЗУ для запоминания окончательных результатов вычислений и в этом случае является необходимым.

Устройства рассматриваемого класса описаны в [53]. Программирование таких устройств будем проводить по граф-схеме без обратных связей, состоящей только из двух типов вершин: условных с 2^k исходами и операторных вершин, в которых выходной переменной присваиваются константы.

Такие граф-схемы могут быть разбиты на два подкласса [54]:

— простые, в которых операторные вершины располагаются только в конце;

— обобщенные, в которых операторные вершины могут располагаться в любом месте.

Рассмотрим простые граф-схемы с двумя операторными вершинами: $z = 0$ и $z = 1$.

Будем называть 2^k-графами простые граф-схемы с двумя операторными вершинами. Такие графы строятся по БФУ и БФ с помо-

щью разложения Шеннона по k переменным и имеют в общем случае «плоскостную» структуру.

Среди таких графов важное значение имеют бинарные графы и их подкласс, представители которого строятся формульным методом [23, 55] и обладают линейной структурой. Эти графы называются линейными бинарными графами (ЛБГ). Ниже рассматриваются два типа устройств — тетрадные и бинарные.

18.2.1. Тетрадные устройства

Рассмотрим шестиадресное устройство, программируемое по тетрадным графам, которые строятся с помощью разложения Шеннона одновременно по двум входным переменным — x_i и x_j.

Каждая команда в этом устройстве содержит восемь полей:

1) адрес переменной x_i;
2) адрес переменной x_j;
3) адрес перехода при $x_i = 0, x_j = 0$;
4) адрес перехода при $x_i = 0, x_j = 1$;
5) адрес перехода при $x_i = 1, x_j = 0$;
6) адрес перехода при $x_i = 1, x_j = 1$;
7) управление (сохранение — запись) D-триггером;
8) запись информации в D-триггер.

Отметим, что для удобства третье, четвертое, пятое и шестое поля команды изображаются в виде] log B [полей, каждое из которых образовано четырьмя битами, где B — число вершин в графе. При этом каждое такое поле соответствует одному и тому же разряду всех четырех адресов переходов.

Пример 18.12. Реализовать тетрадный граф (рис. 18.17), построенный по булевой формуле

$$z = x_1 (x_2 \vee x_3 \vee x_4) \vee x_2 x_3 x_4.$$

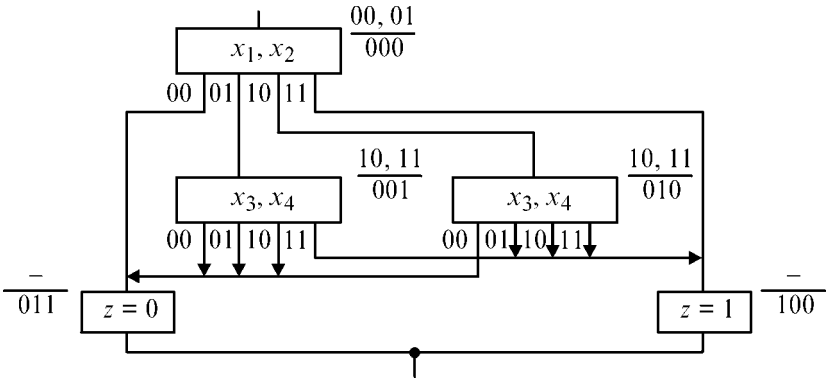

Рис. 18.17

На этом рисунке рядом с каждой вершиной расположена дробь, в числителе которой указаны адреса вводимых в этой вершине входных переменных, а в знаменателе — адрес этой вершины.

Этот тетрадный граф реализуется устройством на рис. 18.18, где РГ — параллельный двоичный регистр.

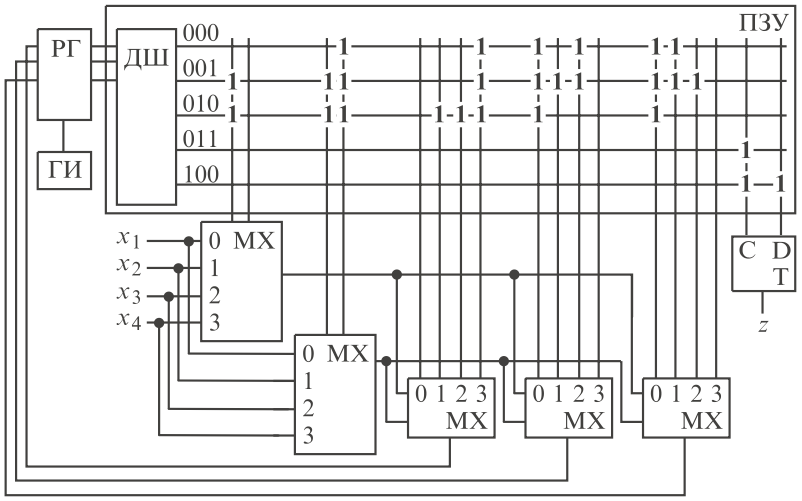

Рис. 18.18

Число разрядов в командах этого устройства определяется соотношением

$$L_{\text{т}} = 2]\log n[+ 4]\log B[+ 2,$$

а число команд в программе, построенной указанным образом, равно B.

18.2.2. Бинарные устройства

Устройства этого класса давно разработаны [2] и используются в течение многих лет при автоматизации, в частности судовых технических средств [2, 8].

Рассмотрим два подкласса устройств этого класса — трехадресные бинарные устройства (БУ3) и двухадресные бинарные устройства (БУ2).

Первый подкласс устройств программируется на основе бинарных графов, а второй — на основе линейных бинарных графов.

1. Рассмотрим первоначально трехадресные бинарные устройства. В них каждая команда должна содержать пять полей:

1) адрес входной переменной x_i;
2) адрес перехода при $x_i = 0$;
3) адрес перехода при $x_i = 1$;
4) управление D-триггером;
5) запись информации в D-триггер.

При этом отметим, что для удобства изображения второе и третье поля совмещены: четные разряды соответствуют второму адресу, а нечетные — третьему.

Пример 18.13. Реализовать бинарный граф (рис. 18.19), построенный методом каскадов по булевой формуле $z = !x_1 x_2 \vee x_1 !x_3$.

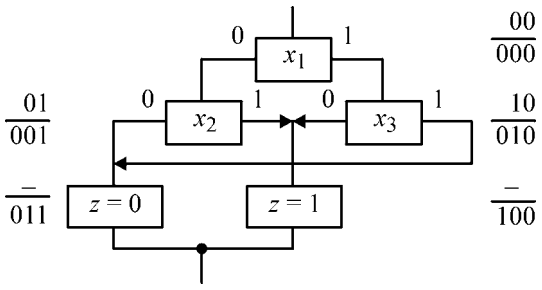

Рис. 18.19

Трехадресное устройство, реализующее этот бинарный граф, приведено на рис. 18.20.

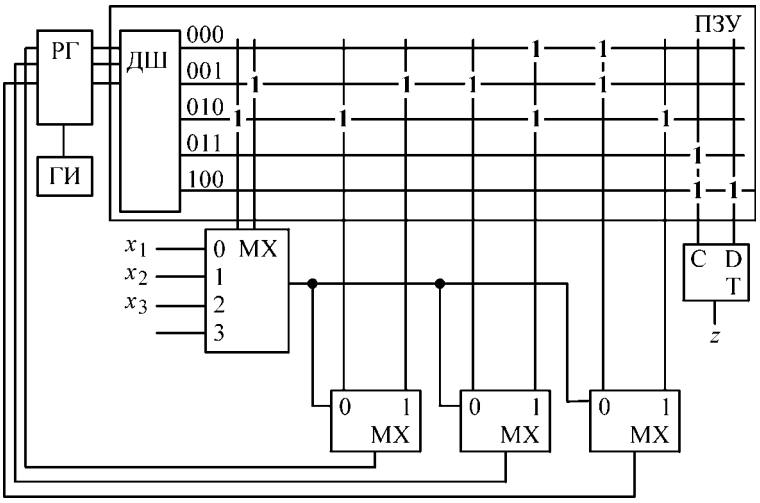

Рис. 18.20

Число разрядов в командах этого устройства определяется соотношением

$$L_{3б} =]\log n[+ 2]\log B[+ 2,$$

а число команд в программе, построенной указанным образом, равно B.

В [22] показано, что для бинарных графов при $n \to \infty$ справедливо соотношение

$$B \to \frac{2^n}{n}.$$

При этом

$$K_{3б} \to \frac{2^n}{n};\ \ L_{3б} \to 2n;\ \ S_{3б} \to 2^{n+1}.$$

При применении этого устройства в схеме на рис. 18.1 проблема риска не возникает даже при отсутствии входных D-триггеров, так как в построенном указанным образом бинарном графе на каждом пути от входа к выходу каждая переменная встречается только один раз.

Предложенное устройство является модификацией микропрограммного автомата Уилкса [56, 57].

2. Перейдем к рассмотрению двухадресных бинарных устройств. В них каждая команда также должна содержать пять полей:

1) адрес входной переменной x_i;

2) сравнение значения входной переменной x_i с константой;

3) адрес «далекого» перехода, осуществляемого в случае, когда имеет место равенство при сравнении;

4) управление D-триггером;

5) запись информации в D-триггер.

Пример 18.14. Реализовать ЛБГ (рис. 18.21), который построен формульным методом по булевой формуле, рассмотренной в предыдущем примере.

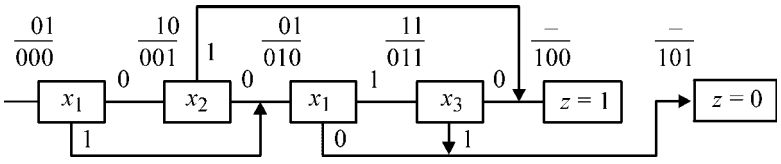

Рис. 18.21

Двухадресное бинарное устройство, реализующее этот ЛБГ, приведено на рис. 18.22.

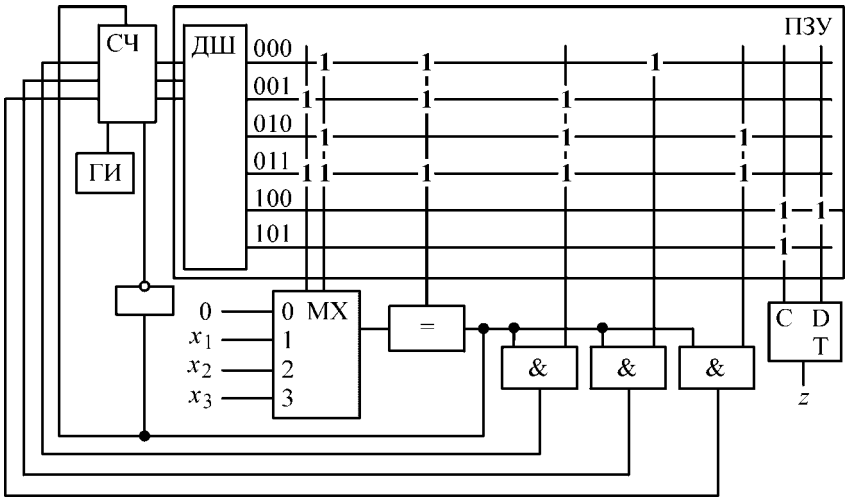

Рис. 18.22

В этой схеме используется счетчик, работающий в двух режимах — прибавление единицы и параллельная запись. Число команд, требующихся в данном случае для реализации произвольной булевой формулы в базисе {&, ∨, !} из h букв, удовлетворяет соотношению

$$K_{2б} = h + 2,$$

а число разрядов в командах — соотношению

$$L_{2б} =]\log(n+1)[+]\log(h+2)[+ 3.$$

При $h = n \to \infty$

$$K_{2б} = h;\ L_{2б} \to 2\log h;\ S_{2б} \to 2h\log h.$$

При $n \to \infty$

$$K_{2б} \to \frac{2^n}{\log n};\ \ L_{2б} \to n;\ \ S_{2б} \to n\frac{2^n}{\log n}.$$

При использовании этого устройства в схеме на рис. 18.1 входные D-триггеры можно не применять при вычислении бесповторных формул в базисе {&, ∨, !}, а также повторных формул в этом же базисе, в которых значения повторных переменных за время программного цикла не изменяются.

Это устройство рассмотрено в [51], а аналогичное устройство для реализации систем булевых формул, использующее вместо счетчика сумматор, — в [58].

18.3. Операторно-бинарные устройства

В силу того что операторно-2^k-нарные устройства являются весьма громоздкими, а метод построения бинарных графов для их программирования не известен, то рассмотрим только операторно-бинарные устройства.

Среди устройств этого класса рассмотрим два подкласса, для которых отсутствует необходимость в применении ОЗУ для запоминания промежуточных результатов.

18.3.1. Четырехадресное операторно-бинарное устройство

Если булева формула реализуется операторно-бинарным графом (ОБГ) на основе разложения Шеннона по подформулам, то этот граф, в свою очередь, может быть реализован четырехадресным операторно-бинарным устройством (ОБУ4).

Пример 18.15. Реализовать ОБГ (рис. 18.23), который построен на основе разложения Шеннона по подформулам из двух букв булевой формулы из примера *18.13*.

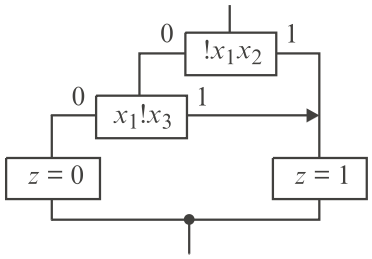

Рис. 18.23

На рис. 18.24 приведено устройство, позволяющее реализовать этот операторно-бинарный граф, а также другие ОБГ, в условных вершинах которых могут размещаться произвольные БФУ двух переменных.

Длина команд в этом устройстве определяется соотношением

$$L_{4\,\text{об}} = 2]\log n[+ 2]\log B[+ 6,$$

а число команд — соотношением

$$K_{4\,\text{об}} = B.$$

В [21] показано, что для ОБГ при $n \to \infty$

$$B \to \frac{2^{n-1}}{n},$$

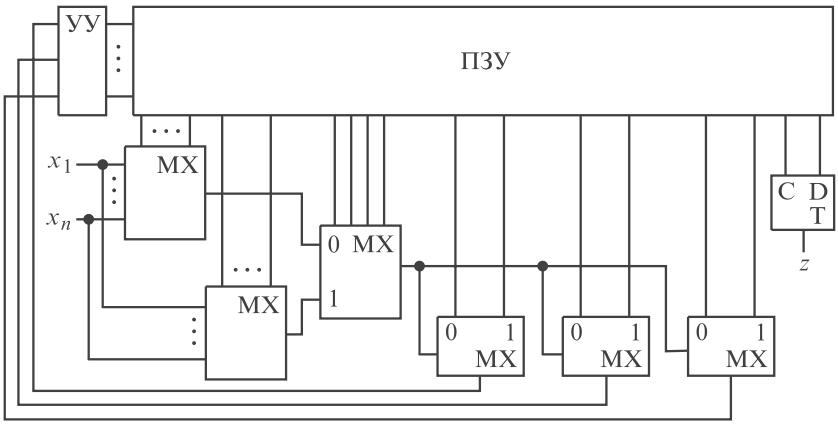

Рис. 18.24

и поэтому

$$K_{4\,\text{об}} \to \frac{2^{n-1}}{n};\ L_{4\,\text{об}} \to 2n;\ S_{4\,\text{об}} \to 2^n.$$

Таким образом, построено последовательностное устройство, площадь ПЗУ которого совпадает с площадью ПЗУ, предназначенного для однотактной реализации произвольной булевой функции n переменных.

Усложнение предлагаемого устройства по сравнению с ПЗУ той же площади позволяет упростить наращивание устройства по входам.

18.3.2. Трехадресное операторно-бинарное устройство

Если булева формула реализуется линейным ОБГ (ЛОБГ), то этот граф, в свою очередь, может быть реализован трехадресным операторно-бинарным устройством (ОБУ3).

Пример 18.16. Реализовать ЛОБГ (рис. 18.25), который построен с помощью обобщенного формульного метода [59] по булевой формуле, рассмотренной в примере *18.13*.

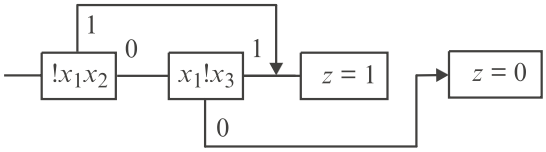

Рис. 18.25

На рис. 18.26 приведено устройство, позволяющее реализовать этот линейный ОБГ, а также другие ЛОБГ, в условных вершинах которых могут размещаться произвольные БФУ двух переменных.

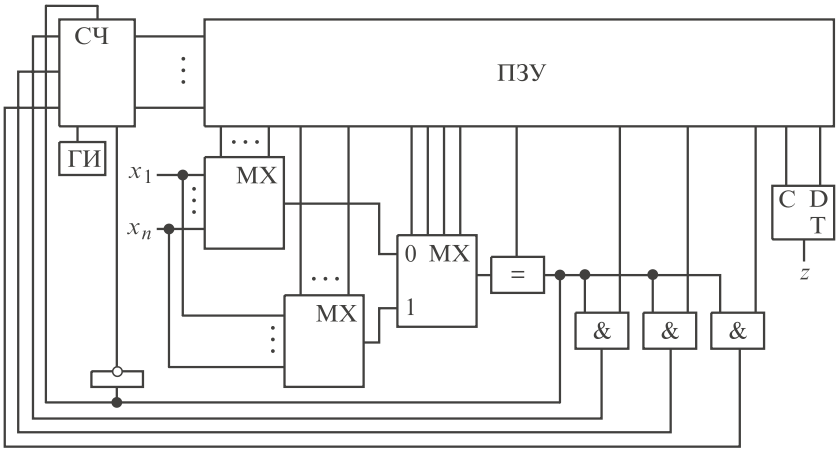

Рис. 18.26

Число команд в данном случае определяется соотношением

$$K_{3\,об} =]h/2[+ 2,$$

а их длина — соотношением

$$L_{3\,об} = 2]\log n[+ 2]\log(]h/2[+ 2)[+ 7.$$

При $h = n \to \infty$

$$K_{3\,об} = h/2;\ L_{3\,об} \to 3\log h;\ S_{3\,об} \to 1.5h\log h.$$

При $n \to \infty$

$$K_{3\,об} \to \frac{2^{n-1}}{\log n};\ L_{3\,об} \to 2n;\ S_{3\,об} \to n\frac{2^{n-1}}{\log n}.$$

18.4. Ассоциативные устройства

18.4.1. Последовательностный аналог постоянного запоминающего устройства

Известно, что постоянные запоминающие устройства (ПЗУ) хорошо наращиваются по выходам и плохо — по входам [60]. Поэтому если при реализации булевой функции n переменных их подавать на входы ПЗУ, то такое устройство будет «гибким» до тех пор, пока $n \leqslant b$, где b — число входов ПЗУ.

Предложим устройство, построенное на базе ПЗУ, которое за счет подачи в х о д н ы х переменных на его в ы х о д ы и перехода к многотактной обработке информации позволяет снять указанное ограничение.

Программирование таких устройств выполняется по совершенной дизъюнктивной нормальной форме (СДНФ). Такая форма используется при аппаратной (однотактной) реализации булевых функций в ПЗУ, и поэтому рассматриваемое устройство можно считать последовательностным аналогом ПЗУ.

В конъюнкцию СДНФ переменная x_i может входить как в прямой, так и в инверсной форме. Занесем для каждой конъюнкции СДНФ в ПЗУ маску *M*, *i*-й разряд которой (M_i) задается табл. 18.3.

Т а б л и ц а 18.3

Переменная	M_i
$!x_i$	0
x_i	1

Определим по этой таблице функцию $\Phi_i = f(M_i, x_i)$ (табл. 18.4).

Т а б л и ц а 18.4

M_i	x_i	Φ_i
0	0	1
0	1	0
1	0	0
1	1	1

Из табл. 18.4 следует, что

$$\Phi_i = (x_i = M_i).$$

Сформируем для каждой конъюнкции СДНФ из констант M_i маску *M*. В этой маске единицы соответствуют прямым переменным, а нули — инверсным.

Пример 18.17. Построить устройство рассматриваемого типа (рис. 18.27), вычисляющее СДНФ $z = !x_1 x_2 \vee x_1 !x_2$.

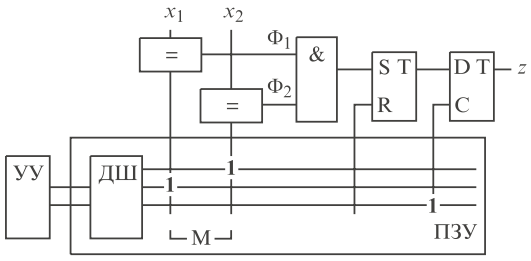

Рис. 18.27

Для корректного функционирования устройства *RS*-триггер должен иметь нулевую начальную установку.

Число команд в этом случае определяется соотношением

$$K_{\text{A1}} = q_1 + 1 \leqslant 2^{n-1} + 1,$$

где q_1 — число конъюнкций в СДНФ.

При этом длина команд (с учетом выходных *D*-триггеров) определяется соотношением

$$L_{\text{A1}} = n + 2.$$

Таким образом,

$$S_{\text{A1}} \leqslant (n+2)\,(2^{n-1}+1).$$

Компенсацией за большую площадь ПЗУ являются:

— простота наращивания устройства по входам;

— возможность одновременной реализации одинаковых конъюнкций, принадлежащих различным СДНФ.

Приведем пример реализации системы булевых формул с помощью рассматриваемого устройства.

Пример 18.18. Построить устройство рассматриваемого типа (рис. 18.28), реализующее систему булевых формул вида: $z_1 = x_1 \oplus x_2$; $z_2 = x_1 \vee x_2$.

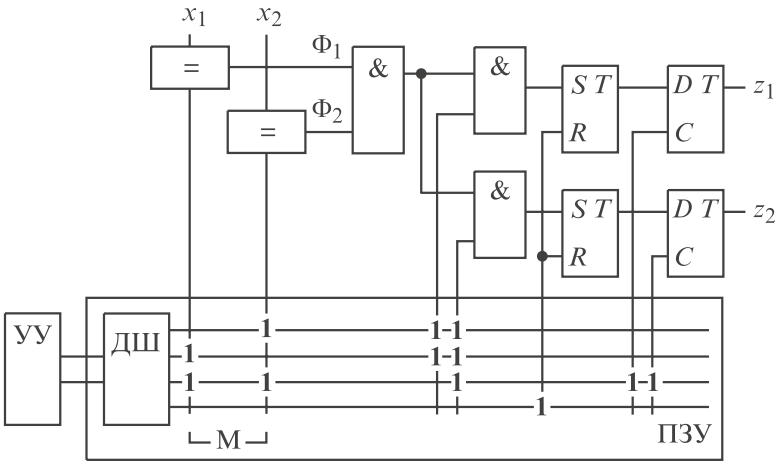

Рис. 18.28

Представим эти булевы формулы в СДНФ:

$$z_1 = !x_1 x_2 \vee x_1 !x_2;\ z_2 = !x_1 x_2 \vee x_1 !x_2 \vee x_1 x_2.$$

Модификация рассмотренного устройства описана в [25].

18.4.2. Последовательностный аналог программируемой логической матрицы

Известно, что программируемые логические матрицы (ПЛМ) хорошо наращиваются по термам и выходам, но плохо — по входам [60].

Поэтому расширим подход, рассмотренный в предыдущем разделе, применительно к булевым функциям, заданным в дизъюнктивной нормальной форме (ДНФ).

Дизъюнктивные нормальные формы используются при аппаратной реализации булевых функций в ПЛМ, и поэтому рассматриваемое устройство можно считать последовательностным аналогом ПЛМ.

В конъюнкцию ДНФ переменная x_i может не входить или входить в прямой или инверсной форме. Занесем для каждой конъюнкции ДНФ в постоянное запоминающее устройство (ПЗУ) две маски $M1$ и $M2$, i-е разряды которых ($M1_i$ и $M2_i$) задаются табл. 18.5.

Т а б л и ц а 18.5

Переменная	$M1_i$	$M2_i$
—	0	0
$!x_i$	1	0
x_i	1	1

Используя эту таблицу, определим функцию $\Phi_i = f(M1_i, M2_i, x_i)$, описанную табл. 18.6.

Т а б л и ц а 18.6

$M1_i$	$M2_i$	x_i	Φ_i
0	0	0	1
0	0	1	1
0	1	0	—
0	1	1	—
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Доопределим функцию Φ_i так, чтобы она могла быть реализована с помощью минимального числа элементов. При доопределении ее нулями получим:

$$\Phi_i = \big((x_i \,\&\, M1_i) = M2_i \big).$$

Сформируем для каждой конъюнкции ДНФ из констант $M1_i$ и $M2_i$ маски $M1$ и $M2$ соответственно. Тогда в маске $M1$ единицы будут соответствовать применяемым в конъюнкции переменным, а нули — переменным, отсутствующим в ней. В маске $M2$ единицы соответствуют переменным без инверсии, а нули — переменным с инверсией или переменным, отсутствующим в конъюнкции.

Число команд в этом устройстве определяется соотношением

$$K_{\text{A}2} = q_2 + 1,$$

где q_2 — число конъюнкций в ДНФ.

При этом длина команд (с учетом выходных D-триггеров) определяется соотношением

$$L_{\text{A}2} = 2(n+1).$$

Таким образом,

$$S_{\text{A}2} = 2(n+1)\,(q_2+1).$$

В разд. 12.2 показано, что после раскрытия скобок в произвольной БФ в базисе {&, ∨, !} из h букв выполняется неравенство [61]:

$$q_2 \leqslant 3^{h/3}.$$

Таким образом,

$$S_{A2} \leqslant 2(3^{h/3}+1)\,(n+1).$$

Пример 18.19. Построить устройство рассматриваемого типа (рис. 18.29), вычисляющее ДНФ $z = !x_1 !x_2 \vee x_1 x_2 \vee x_3$.

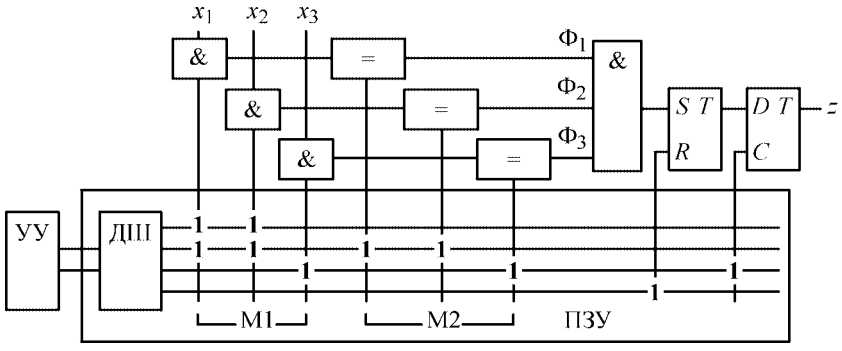

Рис. 18.29

Для корректного функционирования устройства *RS*-триггер должен иметь нулевую начальную установку.

Приведем пример реализации системы ДНФ с помощью рассматриваемого устройства.

Пример 18.20. Построить устройство рассматриваемого типа (рис. 18.30), вычисляющее систему ДНФ $z_1 = !x_1 !x_2 \vee x_1 x_2 \vee x_3$, $z_2 = !x_1 !x_2 \vee !x_4$.

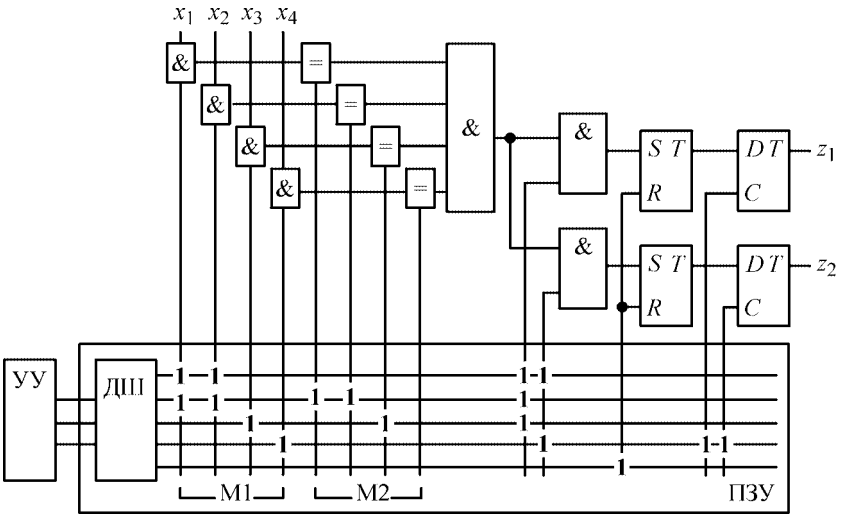

Рис. 18.30

Идея построения устройств рассмотренного типа изложена в [62, 63].

Рассмотренные выше устройства названы ассоциативными (АУ), так как в них поиск информации осуществляется не по адресу (входам ПЗУ), а по содержимому ПЗУ за счет сравнения на его выходах информации, записанной в нем, со значениями входных переменных. Такой подход назван в [64] поиском по ассоциации.

Устройства, рассмотренные в предыдущем разделе, обозначим символом АУ1, а устройства, описанные в настоящем разделе, — символом АУ2.

Менее удачным, по мнению автора, является другое название аналогичных устройств — «параллельные логические контроллеры» [42, 43].

18.5. Сравнительные характеристики рассмотренных устройств

Оценки числа команд и площади ПЗУ для рассмотренных устройств приведены в табл. 18.7.

Таблица 18.7

Обозначение устройства	Число команд	Площадь ПЗУ ($h = n \to \infty$)	Площадь ПЗУ ($n \to \infty$)
ОУ3	$h-1$	$2h\log h$	$3\cdot 2^n$
ОУ2.5	$h \leqslant k \leqslant [(3h/2)-1]$	$h\log h \lesssim S \lesssim (3h/2)\log h$	$3\cdot 2^n \lesssim S \lesssim 4.5\cdot 2^n$
ОУ2	$h \leqslant k \leqslant [(3h/2)-1]$	$h\log h \lesssim S \lesssim (3h/2)\log h$	$2^{n+1} \lesssim S \lesssim 3\cdot 2^n$
ОУ1	$h+1 \leqslant k \leqslant 2h-1$	$h\log h \lesssim S \lesssim 2h\log h$	$2^n \lesssim S \lesssim 2^{n+1}$
ОУ31	$h/2 \leqslant k \leqslant h-1$	$h\log(h+f(h)) \lesssim S \lesssim$ $\lesssim 4h\log(h+f(h))$	$n\cdot 2^n/\log n \lesssim S \lesssim$ $\lesssim n\cdot 2^{n+1}/\log n$
ОУ21	$h+1 \leqslant k \leqslant [3h/2]$	$3h\log h \lesssim S \lesssim 4.5h\log h$	$n\cdot 2^{n+1}/\log n \lesssim S \lesssim$ $\lesssim 3n\cdot 2^n/\log n$
СОУ1	$h+1 \leqslant k \leqslant [3h/2]$	$h\log h \lesssim S \lesssim 1.5h\log h$	$2^n \lesssim S \lesssim 1.5\cdot 2^n$
БОУ	h	$2h\log h$	$n\cdot 2^{n+1}/\log n$
БУ3	$2^n/n$	—	2^{n+1}
БУ2	$h+2$	$2h\log h$	$n\cdot 2^n/\log n$
ОБУ4	$2^{n-1}/n$	—	2^n
ОБУ3	$]h/2[+2$	$1.5h\log h$	$n\cdot 2^{n-1}/\log n$
АУ1	$\leqslant 2^{n-1}+1$	—	$n\cdot 2^{n-1}$
АУ2	$\leqslant 3^{h/3}+1$	$2h\cdot 3^{h/3}$	—

В этих устройствах не ограничивалась длина команд, так как предполагалось, что каждое устройство реализовано в виде одной компоненты. В случае, если оно реализуется в виде двух компонент (например, микросхем), одна из которых ПЗУ, указанное ограничение может накладываться [9]. При этом команды, записанные в ПЗУ, кодируются «плотно», а во вторую компоненту вводится дешифратор, преобразующий «узкую» команду в ту, которая требуется в соответствующем из рассмотренных выше устройств [65].

18.6. Микропроцессорные устройства

Структура приведенных выше логических устройств существенно отличается от структуры универсальных микропроцессоров, одной из особенностей которых является многоразрядность.

Несмотря на то что некоторые из рассмотренных устройств могут одновременно обрабатывать несколько входных переменных, для них более характерна битовая обработка, так как логические переменные по своей природе являются битовыми (одноразрядными): один бит отражает всю требуемую информацию о каждом объекте или процессе (например, открыт или не открыт клапан). Поэтому такие устройства, может быть и не всегда точно, называют однобитовыми [20], или одноразрядными.

При необходимости выполнения в одном устройстве как логической, так и арифметической обработки информации возможны различные подходы к его построению и использованию:

— в устройстве применяются два процессорных элемента — однобитовый и многоразрядный [4];

— в устройстве используется один многоразрядный процессорный элемент, а логические функции заменяются арифметическими, например арифметическими полиномами [66];

— в устройстве применяется один многоразрядный процессорный элемент, в котором логические операции выполняются с помощью логических (битовых) команд и (или) условных переходов, что позволяет в этом случае моделировать работу однобитовых процессоров;

— в устройстве используется один многоразрядный процессорный элемент, в котором логические команды выполняются с помощью многоразрядных битовых команд, что позволяет, в частности, моделировать работу ассоциативных логических устройств.

Еще одна возможность имеется в микроконтроллерах, в которых применяется один многоразрядный процессорный элемент: логические операции могут выполняться в том числе и с помощью логических команд над битами одного слова, что также позволяет моделировать работу однобитовых процессоров [7].

Пример 18.21. Реализовать булеву формулу $z = (x_1 x_2 \vee x_3 ! x_4) x_5$ с помощью бинарной программы, написанной на языке макроассемблера ЭВМ PDP-11 или VAX, при условии, что переменные x_1, x_2, x_3 соответственно размещены в 14-, 6- и 3-м разрядах первого поля, а переменные x_4, x_5 — соответственно в 5- и 2-м разрядах второго поля.

Построим по заданной формуле линейный бинарный граф (рис. 18.31), введя в него операторную вершину, помеченную символом *w*, которая соответствует команде безусловного перехода в программе.

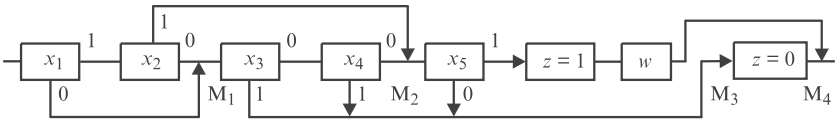

Рис. 18.31

Этому ЛБГ соответствует следующая программа:

```
        BIT #040000, POLE1
        BEQ M1
        BIT #000100, POLE1
        BNE M2
M1:     BIT #000010, POLE1
        BEQ M3
        BIT #000040, POLE2
        BNE M3
M2:     BIT #000004, POLE2
        BEQ M3
        MOV #1, z
        BR  M4
M3:     CLR z
M4:
```

Константы в этой программе записаны в восьмеричной системе счисления и являются масками для выделения соответствующих упакованных входных переменных.

В работах [52, 67] показано, что вне зависимости от расположения входных переменных в одном или разных полях число команд K и число слов V в программе, реализующей произвольную формулу в базисе $\{\&, \vee, !\}$ из h букв, определяются соотношениями

$$K = 2h + 3, \quad V = 2K.$$

В [45] рассмотрено более 60 программ на языке СИ, построенных различными методами, и каждая из них реализует логическую функцию «голосование два и более из трех» при размещении каждой входной переменной в крайнем правом разряде соответствующего слова. Выполнено сравнение этих программ по объему памяти (без учета оформления в виде функций) и быстродействию при их реализации на процессоре 80286. Показано, что наилучшей по этим показателям является программа, построенная по арифметическому полиному с маскированием [68]:

```
 laps (int x1, int x2, int x3)
{int z;
 z = (x1 + x2 + x3) >> 1;
 return (z);}
```

Объем этой программы 14 байт, в то время как программа, использующая битовые операции и компилируемая как операторная,

```
 sf2 (int x1, int x2, int x3)
{int z;
 z = ((x1 | x2) & x3) | (x1 & x2);
 return (z);}
```

требует 20 байт памяти, а программа, в которой применяются логические операции, компилируемая как бинарная,

```
 sfb (int x1, int x2, int x3)
{int z;
 z = (x3 && (x1 || x2)) || (x1 && x2);
 return (z);}
```

требует 40 байт памяти.

В заключение главы отметим, что вопрос о реализации булевых формул на машинах Тьюринга рассмотрен в [71].

Выводы

1. Рассмотрены четыре класса специализированных логических устройств для последовательностного вычисления булевых функций: операторные, 2^k-нарные, операторно-бинарные и ассоциативные. Эти классы устройств, использующие традиционные принципы логической обработки информации, не исчерпывают всех классов логических устройств, которые, в частности, могут быть построены на основе вычисления арифметических полиномов.

2. В каждом классе предложены одно или несколько устройств, каждое из которых обладает особенностями, позволяющими выделить его в отдельный подкласс.

3. Предложен принципиально новый подкласс операторных устройств, обеспечивающих побуквенное вычисление произвольных нормальных булевых формул в базисе {&, ∨, !} из h букв, несмотря на наличие в них скобок произвольной глубины.

4. Получены оценки числа команд и площади ПЗУ для рассмотренных устройств.

5. Предложено операторно-бинарное устройство, у которого площадь ПЗУ асимптотически равна 2^n, что совпадает с площадью ПЗУ при однотактной реализации булевой функции n переменных. Преимущество такого устройства по сравнению с ПЗУ состоит в упрощении расширения по входам.

6. Предложены ассоциативные логические устройства, обеспечивающие наиболее простое расширение по входам.

7. Рассмотрен вопрос о применении различных методов последовательностного вычисления булевых функций при программировании универсальных микропроцессоров.

8. Все устройства, рассмотренные в настоящей главе, являются синхронными. Реализация простых бинарных графов асинхронными автоматами и метод противогоночного кодирования их состояний описаны в работах [65, 69].

Литература

1. *Павлов В. В.* Управляющие логические машины — новое средство автоматического управления производственными процессами // Приборы и средства автоматизации. 1969. № 3.

2. *Бутин Ю. Н., Маковеев О. Л., Харинский С. В. и др.* Программные логические устройства. Л.: Судостроение, ЦНИИ «Румб», 1979.

3. *Амбарцумян А. А., Потехин А. И., Запольский Е. Н.* Программируемые логические контроллеры и их применение // Измерения. Контроль. Автоматизация. 1979. Вып. 4.

4. *Мишель Ж.* Программируемые контроллеры. Архитектура и применение. М.: Машиностроение, 1992.

5. *International* Standard IEC 1131. Programming languages // International Electrotechnical Commission. 1993.

6. *Новик Г. Х., Першеев В. Г., Шамров М. И.* Реализация комбинационных схем микропроцессорным автоматом // Приборы и системы управления. 1980. № 3.

7. *Сташин В. В., Урусов А. В., Мологонцева О. Ф.* Проектирование цифровых устройств на однокристальных микроконтроллерах. М.: Энергоатомиздат. 1990.

8. *Бутин Ю. Н., Золотаревская М. Я., Кириллов А. П., Юнг В. Н.* О реализации алгоритмов логического управления в специализированных программируемых логических устройствах // Автоматика и телемеханика. 1983. № 6.

9. *MC 14500B.* Industrial control unit. Handbook Motorola. S. Products. 1977.

10. *Ли С.* Представление переключательных схем с помощью программ двоичного решения // Вопросы теории математических машин. М.: Машиностроение, 1964.

11. *Хазацкий В. Е.* Некоторые вопросы построения управляющих логических устройств с общим функциональным логическим блоком // Труды ЦНИИКА. 1966. Вып. 14.

12. *Альтшуль С. Д., Гильман Г. И., Рог Г. В. и др.* Логическое устройство. А. с. СССР № 189630 // Бюл. изобр. 1966. № 24.

13. *Гильман Г. И.* Исследование и разработка универсальной логической машины для автоматизации судовых систем // Труды ЦНИИ им. А. Н.Крылова. 1966. Вып. 231.

14. *Альтшуль С. Д., Буянов Б. Б., Волков А. Ф. и др.* Универсальная логическая машина для управления производственными процессами // Приборы и системы управления. 1967. № 6.

15. *Нефедов Г. С.* О последовательном вычислении логических функций // Автоматика и телемеханика. 1973. № 2.

16. *Савченко Ю. Г., Хмелевая А. В.* Некоторые задачи синтеза дискретных устройств управления медленными процессами // Дискретные системы. Т. 1. Симпозиум IFAC. Рига: Зинатне, 1974.

17. *Балашов Е. П., Пузанков Д. В.* Логические процессоры для реализации разветвленных алгоритмов // Управляющие системы и машины. 1974. № 6.

18. *Блох А. Ш.* Граф-схемы и их применение. Минск: Вышэйшая школа, 1975.

19. *Загарий Г. И.* Принципы построения устройств управления последовательными операциями (обзор) // Управляющие системы и машины. 1975. № 1.

20. *Свит А.* Новый подход к программируемой логике на основе использования однобитного процессора // Экспресс-информация. Вычислительная техника. 1976. № 8.

21. *Волков А. Ф., Лукашенко Г. А., Луненко К. М. и др.* Управляющая логическая машина. А. с. СССР № 539301 // Бюл. изобр. 1976. № 46.

22. *Кузьмин В. А.* Оценки сложности реализации функций алгебры логики простейшими видами бинарных программ // Методы дискретного анализа в теории кодов и схем. Новосибирск, 1976. Вып. 29.

23. *Кузнецов О. П.* О программной реализации логических функций и автоматов // Автоматика и телемеханика. 1977. № 7, 9.

24. *Глушков В. М., Деркач В. П., Корсунский В. М. и др.* Микропроцессоры на ПЗУ // Управляющие системы и машины. 1977. № 5.

25. *Румянцев И. А.* Введение в судовую последовательную автоматику и принципы агрегатирования // Теоретические основы автоматизации процессов управления судном. Труды Ленингр. высш. инженер. морск. училища. 1975. Вып. 3.

26. *Гильман Г. И., Лазарев В. Г., Альтшуль С. Д.* Устройство для решения логических задач. А. с. СССР № 631160 // Бюл. изобр. 1977. № 12.

27. *Камынин Ю. Н., Матвиенко Н. П., Жигульцев А. Ю.* Управляющие логические устройства с перестраиваемой структурой // Приборы и системы управления. 1978. № 11.

28. *Pokoski J. L.* Software analyses for combinatoriallogic // Compt. Design. 1978. № 6.

29. *Plavic V., Danielsson P.* Sequantial evaluation of Boolean functions // IEEE Trans. Computers. 1979. № 12.

30. *Абугов Ю. О., Диденко К. И., Загарий Г. И. и др.* Микроэлектронные устройства программного и логического управления. М.: Машиностроение, 1979.

31. *Гаранин Н. А.* Программирование релейно-контактных схем управления в системе команд УЛП // Труды Всесоюз. науч.-исслед., проектно-конструктор. и технол. ин-та релестроения. Чебоксары, 1979. Вып. 10.

32. *Капитонова Ю. В., Алексеенко В. Г., Мищенко А. Т.* Реализация устройств управления на основе больших стандартных схем // Кибернетика. 1979. № 3, 4.

33. *Балашов Е. П., Негода В. Н., Пузанков Д. В.* Проектирование логических процессоров на ПЗУ // Управляющие системы и машины. 1980. № 5.

34. *Золотаревская М. Я.* О реализации систем булевых функций в программном логическом устройстве // Проблемы управления в технике, экономике, биологии. М.: Наука, 1981.

35. *Срибнер Л. А.* Программируемые устройства автоматики. Киев: Техника, 1982.

36. *Иванов А. В.* Минимизация длины стека при вычислении выражений // Программирование. 1982. № 1.

37. *Бойет Г., Кац Р.* Программная реализация комбинационных логических схем // Электроника. 1983. № 6.

38. *Лапкин Л. Я.* О векторной программной реализации логических функций // Автоматика и телемеханика. 1983. № 3.

39. *Лазарев В. Г., Пийль Е. И., Турута Е. Н.* Построение программируемых управляющих устройств. М.: Энергоатомиздат, 1984.

40. *Пупырев Е. И.* Перестраиваемые автоматы и микропроцессорные системы. М.: Наука, 1984.

41. *Девятков В. В.* Стековая программная реализация алгоритмов логического управления // Проектирование устройств логического управления. М.: Наука, 1984.

42. *Вышинский В. А., Фурман И. А.* Об использовании булевых матриц и операции над ними для построения быстродействующих программируемых устройств логического управления // Управляющие системы и машины. 1986. № 2.

43. *Фурман И. А., Никонов А. И.* Математическая модель и принципы структурной организации параллельных логических контроллеров // Управляющие системы и машины. 1986. № 4.

44. *Лапкин Л. Я.* Оценки сложности программной реализации при различных формах задания булевых функций // Автоматика и телемеханика. 1987. № 1.

45. *Шалыто А. А.* SWITCH-технология. Алгоритмизация и программирование задач логического управления. СПб.: Наука, 1998.

46. *Евтодьев А. И., Клунт Б. Я., Шалыто А. А. и др.* Устройство для реализации логических функций. А. с. СССР № 1001080 // Бюл. изобр. 1983. № 8.

47. *Артюхов В. Л., Кузнецов Б. П., Шалыто А. А.* Настраиваемые бинарные процедуры и циклические программы // Автоматика и телемеханика. 1984. № 11.

48. *Лупанов О. Б.* Асимптотические оценки сложности управляющих систем. М.: Изд-во МГУ, 1984.

49. *Autolog* 32. Руководство пользователя. FF-Automation OY.

50. *SYSMAC.* Programmable controllers. CV500/CV100. Operation Manual. Omron. 1993.

51. *Артюхов В. Л., Кузнецов Б. П., Шалыто А. А.* Настраиваемые логические устройства для судовых управляющих систем. Л.: ИПК СП, 1986.

52. *Шалыто А. А.* Программная реализация алгоритмов логического управления судовыми системами. Л.: ИПК СП, 1989.

53. *Артюхов В. Л., Шалыто А. А.* Судовые управляющие логические системы. Л.: ИПК СП, 1983.

54. *Рубинов В. И., Шалыто А. А.* Метод построения граф-схем простых бинарных программ для систем булевых функций // Автоматика и вычисл. техника. 1986. № 4.

55. *Артюхов В. Л., Кузнецов Б. П., Шалыто А. А.* Линейные бинарные графы: свойства, характеристики, алгоритмы // Методы и программы решения оптимизационных задач на графах и сетях. Ч. 2. Тезисы докл. III Всесоюз. совещ. Новосибирск, 1984.

56. *Wilkes M. V., Stringer J. B.* Microprogramming and the design of the control circuits in an electronic digital computer // Proc. Cambrige Philos. Soc. 1953. № 4.

57. *Лазарев В. Г., Пийль Е. И.* Синтез управляющих автоматов. М.: Энергия, 1978.

58. *Вавилов В. Н., Вальшонок Е. С., Шалыто А. А. и др.* Устройство для вычисления булевых функций. А. с. СССР № 1501033 // Бюл. изобр. 1989. № 8.

59. *Артюхов В. Л., Кузнецов Б. П., Шалыто А. А. и др.* Особенности записи булевых формул при их программной реализации // Вопросы судостроения. Сер. Судовая автоматика. 1984. Вып. 31.

60. *Баранов С. И., Скляров В. А.* Цифровые устройства на программируемых БИС с матричной структурой. М.: Радио и связь, 1986.

61. *Артюхов В. Л., Кузнецова О. С., Шалыто А. А.* Оценка функциональных возможностей программируемых логических матриц // Автоматика и вычисл. техника. 1985. № 2.

62. *Артюхов В. Л., Шалыто А. А.* Программируемое логическое устройство. А. с. СССР № 1587487 // Бюл. изобр. 1990. № 31.

63. *Артюхов В. Л., Шалыто А. А.* Программируемое логическое устройство. А. с. СССР № 1587488 // Бюл. изобр. 1990. № 31.

64. *Коханен А.* Ассоциативные устройства. М.: Мир, 1986.

65. *Артюхов В. Л., Сагалович Ю. Л., Шалыто А. А. и др.* Исследование целесообразности и возможности создания малогабаритных перепрограммируемых логических устройств для систем с децентрализованной структурой. Отчет по НИР. Л.: НПО «Аврора», 1982.

66. *Артюхов В. Л., Кондратьев В. Н., Шалыто А. А.* Реализация булевых функций арифметическими полиномами // Автоматика и телемеханика. 1988. № 4.

67. *Кузнецов Б. П., Шалыто А. А.* Реализация булевых формул линейными бинарными графами. I. Синтез и анализ // Известия РАН. Техн. кибернетика. 1994. № 5.

68. *Кондратьев В. Н., Шалыто А. А.* Реализация булевых функций одним линейным арифметическим полиномом с маскированием // Автоматика и телемеханика. 1996. № 1.

69. *Сагалович Ю. Л., Шалыто А. А.* Бинарные программы и их реализация асинхронными автоматами // Проблемы передачи информации. 1987. № 1.

70. *Проблемы* выбора и перспективы систем автоматизации. Отдел управления и информации компании «Rockwell Automation» // Мир компьютерной автоматизации. 1998. № 2.

71. *Сэвидж Д. Э.* Сложность вычислений. М.: Факториал, 1998.