

ПРОБЛЕМЫ НАПРАВЛЕННОГО ПОИСКА МИНИМАЛЬНЫХ СТРУКТУР РЕЛЕЙНЫХ УСТРОЙСТВ*)

М.А. Гаврилов
(Москва).

Широкое развитие вычислительных и управляющих машин и устройств дискретного действия поставило, как одну из актуальных проблем сегодняшнего дня проблему синтеза их структуры.

Основной задачей здесь является построение оптимальной в определенном смысле структуры устройства, удовлетворяющей заданному для него алгоритму функционирования и заданным требованиям к аппаратуре, на которой это устройство будет реализовано.

Проблема синтеза структуры релейных устройств состоит в общем из взаимосвязанных этапов, при осуществлении которых происходит все большая конкретизация и учет заданных параметров элементов (логических свойств, энергетических и других ограничений и т.п.), на которых происходит реализация структуры.

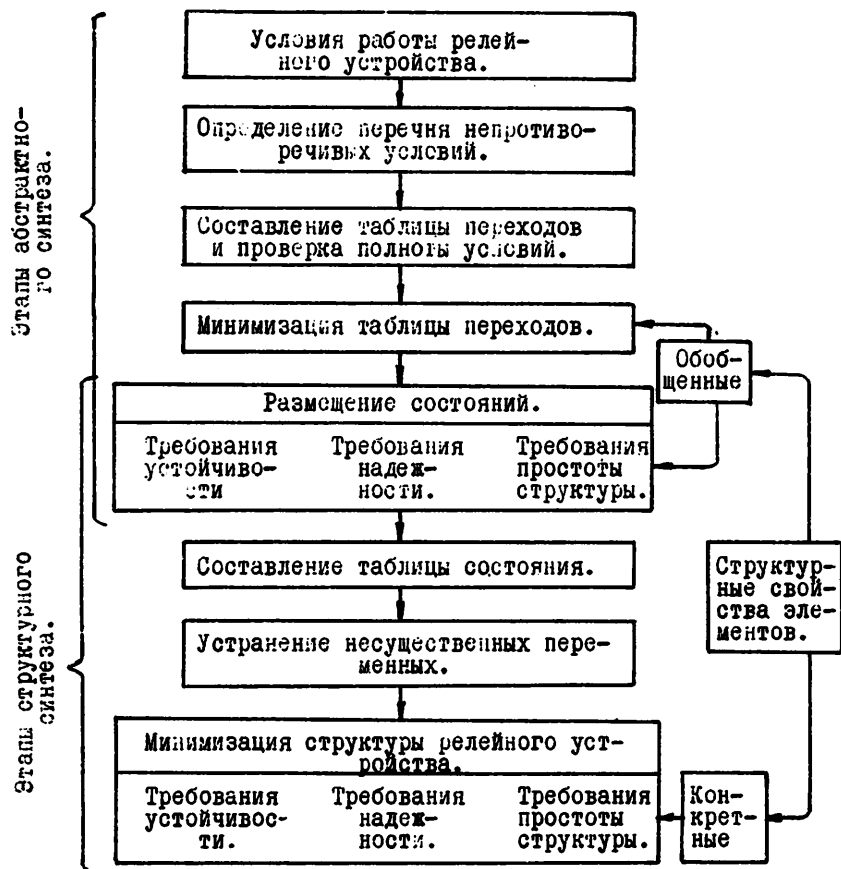
Процесс синтеза релейных устройств принято в настоящее время разделять на абстрактный и структурный (табл. I).^{xx)}

При абстрактном синтезе конкретные структурные свойства и ограничения элементов, на которых реализуется релейное устройство, не учитываются. Принимаются во внимание лишь некоторые

x) В настоящем докладе излагаются результаты работ, проведенных под руководством и при непосредственном участии автора по развитию идей, сформулированных в докладе автора: "Проблема поиска минимальных решений при синтезе структуры релейных устройств" на III-м Всесоюзном совещании по автоматическому управлению в г. Одесса в 1965 г.

xx) На рис. I для иллюстрации этапов синтеза представлен процесс синтеза и анализа структуры релейного устройства на базе одного из применяемых формализованных языков т.н. "таблиц переходов".

Таблица I



общие свойства, такие как: неодинаковость задержек, возможность отказа, характер воздействий (импульсное или потенциальное) и т.п. Элементами структуры здесь являются состояния входов, выходов и внутренних элементов. На этапе абстрактного синтеза в структуру релейного устройства в общем виде закладываются все требования к быстродействию, устойчивости и надежности функционирования устройства^{х)} и т.п.

В процессе структурного синтеза строится конкретная структу-

х) Под "устойчивостью" здесь понимается способность устройства в точности выполнять алгоритмы функционирования при изменении временных параметров элементов, а под "надежностью" свойство в точности выполнять алгоритмы функционирования при отказах элементов.

ра устройства с учетом всех свойств элементов, на которых она реализуется.

Современные задачи синтеза релейных устройств требуют значительного пересмотра ранее разработанных методов синтеза, т.к. имеют ряд следующих особенностей.

а) Существенное усложнение в ряде случаев структурных свойств элементов, на которых выполняются релейные устройства. С одной стороны это связано с реализацией структуры релейных устройств на т.н. "микромодулях", представляющих собой конструктивно целые части отдельных блоков устройств и имеющих существенно более сложные структурные свойства чем элементы, выполняющие элементарные логические функции, а с другой - с появлением новых технологических процессов изготовления релейных устройств в виде т.н. "сплошных сред", в которых должны рассматриваться совокупности элементов со связями их друг с другом.

б) Усложнение условий заданий релейных устройств как в отношении числа переменных, от которых зависит их функционирование, так и в отношении выполняемых ими функций. В ряде случаев эти условия задаются лишь по отдельным блокам и при синтезе требуется, помимо оптимальной реализации отдельных блоков, найти также оптимальную блочную структуру релейного устройства.

в) Иерархичность условий работы релейных устройств, когда устройство разбивается на ряд подустройств, обладающих функциональной иерархией, т.е. находящихся в функциональном подчинении друг другу. В этих условиях возникает задача оптимального распределения функций между иерархическими ступенями устройства.

Легко показать, что существующие методы обладают рядом существенных недостатков, которые делают их мало применимыми для решения указанных задач.

Прежде всего существенные ограничения возникают в связи с тем, что существующие методы требуют детального описания условий работы релейного устройства: при структурном синтезе вплоть до отдельных переменных, а при абстрактном - вплоть до отдельных состояний: входов, выходов и внутренних элементов.

Такое описание уже при небольшом числе переменных становится настолько громоздким, что почти полностью исключает применение существующих методов синтеза. Так, например, при использовании для записи условий работы релейных устройств языка т.н. "таблиц переходов" каждое из состояний входов отражается в виде отдельного столбца таблицы. Уже при числе входов, превыша-

чем 6-7 (64-128 столбцов таблицы), метод таблиц переходов в его обычном виде становится неприменимым. То же имеет место и при использовании т.н. "таблиц состояний" при структурном синтезе. Эти таблицы в их обычном виде требуют детального перечисления всех состояний входов и внутренних элементов и поэтому применение их для числа состояний порядка более 500 также становится затруднительным. При использовании машины такая детальная запись перегружает память машины и также ограничивает возможности существующих методов. Очевидно, что одним из выходов из указанного положения является переход к языкам и методам синтеза, допускающим более общее описание условий работы релейных устройств и их структур.

Вторым существенным ограничением является необходимость полного перебора решений для получения минимальных реализаций структур релейных устройств.

В области структурного синтеза в настоящее время наиболее развиты и теоретически обоснованы методы, дающие реализации структур в нормальной дизъюнктивной или конъюнктивной форме, что соответствует реализации структур на элементах: "И", "ИЛИ", и "НЕ". Эти методы основаны в большинстве случаев на определении из таблицы состояний т.н. "минимальных членов" - таких произведений букв, из которых нельзя исключить ни одну букву без того, чтобы была нарушена непротиворечивость реализации таблицы состояний.^{х)} Из множества всех минимальных членов затем отбираются путем перебора подмножества их, образующие избыточные реализации^{хх)}, и из всех избыточных реализаций выбираются также путем перебора все минимальные реализации (будем называть их "абсолютно минимальными"). Такой перебор весьма быстро возрастает в связи с резким ростом числа минимальных членов.

Так, например, при пяти переменных максимальное число минимальных членов составляет 32, при 10-ти - 4.300 и при 15-ти

х) Реализация таблицы состояния противоречива тогда, когда какое-либо произведение букв, принимаемое за минимальный член, содержится как в состоянии, для которого на каком-либо выходе должен быть ноль, так и в состоянии, для которого на том же выходе должна быть единица.

хх) Избыточность здесь понимается в том смысле, что в полученной в результате синтеза структуре ни один из элементов не может быть удален и ни одна из переменных, подаваемых на входы какого-либо элемента, не может быть заменена константой без того, чтобы не была нарушена непротиворечивость реализации структуры. Минимальность структуры понимается в том смысле, что полученная структура соответствует минимальному значению заранее заданного функционала минимизации.

759.488 [1]. Число операций, необходимых для определения минимальных неизбыточных форм, растет еще быстрее и синтез минимальных структур для 12-15 переменных становится в общем случае практически неосуществимым даже на универсальных вычислительных машинах. Еще более громоздкой является реализация структуры в скобочной форме и при наличии многих выходов в виде "связанных структур".^{х)}

Попытки разработки методов, в которых возможен не полный перебор, не привели к существенному уменьшению числа операций. В связи с этим при решении задач структурного синтеза часто ограничиваются получением какой-либо одной реализации без учета ее неизбыточности и минимальности. Нужно отметить, что речь здесь идет о получении только "минимальных" решений, а не "оптимальных", когда необходимо учитывать не только число элементов, а также и их стоимость, вес или другие показатели, которые в некоторых случаях могут характеризоваться некоторым заданным функционалом оптимизации. Методов, позволяющих получить структуры, удовлетворяющие минимальному значению заданного функционала, до последнего времени вообще не было.

Еще более осложняется задача получения оптимальных в определенном смысле реализаций структур, если начать заботиться об этом на этапах абстрактного синтеза. С этой точки зрения существующие методы синтеза обладают определенной противоречивостью. Действительно, на заключительных этапах абстрактного синтеза обычно стремятся к получению минимального числа элементов памяти (этап минимизации таблицы переходов в табл. I), что в общем не гарантирует получения наиболее простой структуры, а тем более оптимальной в заданном смысле, так как уменьшение числа элементов памяти приводит, как правило, к усложнению логического блока. Кроме того, минимизация числа состояний не соответствует требованиям к устойчивости и надежности структуры и удовлетворение этим требованиям сопровождается, как правило, увеличением числа элементов памяти.

Выход из указанного положения может быть найден, если будет получена возможность оценок сложности структур еще на абстрактных этапах синтеза и разработаны методы, позволяющие из полного дерева перебора отыскивать пути, ведущие к получению абсолютно минимальных структур или близких к ним, начиная отыскание таких путей с этапов абстрактного синтеза.

х) "Связанными" или структурами типа "связанного дерева" называют структуры, в которых имеются связи между цепями отдельных выходов.

Существенные осложнения возникают при использовании существующих методов при синтезе структур на элементах со сложными структурными свойствами. Существующие методы обладают в этом отношении определенной методологической противоречивостью. Действительно, как уже говорилось, они основаны на том, что по мере перехода от одного этапа к другому происходит все большая детализация структуры с все более подробным учетом структурных свойств элементов. В частности, этапы структурного синтеза требуют особой детализации, причем в существующих методах решение получается обычно в нормальной форме булевых функций, которые затем с помощью алгоритмических (а большей частью полунититивных) методов преобразуются в форму, соответствующую заданным структурным свойствам элементов, что само по себе представляет собой весьма трудную и громоздкую задачу, не имеющую во многих случаях алгоритмического решения.

Между тем, если рассматривать элемент со сложными структурными свойствами как некоторое элементарное подустройство (под-автомат), то задачу синтеза структуры сложного релейного устройства (автомата) можно свести к задаче оптимальной его декомпозиции на заданный набор элементарных подавтоматов. При этом условия работы как автомата в целом, так и элементарных подавтоматов должны быть записаны на каком-либо удобном формализованном языке. Если, например, условия работы релейного устройства и условия работы элементов заданы в виде таблиц переходов и, кроме того, задан функционал оптимизации, то задача синтеза будет заключаться в оптимальной с точки зрения заданного функционала декомпозиции общей таблицы переходов на подтаблицы, соответствующие таблицам переходов элементов.

Такая постановка задачи имеет существенные преимущества в том смысле, что она ограничивает необходимым минимумом детализацию описания как самой структуры релейного устройства, так и составляющих ее элементов, и, кроме того, устраняет ненужные этапы синтеза. Например, в такой постановке задачи могут быть полностью устранены этапы структурного синтеза, обладающие, как правило, наибольшей громоздкостью. Естественно, что для получения оптимальных структур в этом случае для избежания перебора необходимо, как и при структурном синтезе, применение направленного поиска минимальных реализаций.

Рассмотрим основные проблемы, которые возникают на основе сформулированных выше задач, и пути их решения.

В области абстрактного синтеза возникают три основные проблемы:

а) проблема создания более общих языков для описания условий работы релейных устройств и разработка методов синтеза с использованием этих языков;

б) проблема оценки сложности структур на этапах абстрактного синтеза и

в) проблема направленного поиска минимальных реализаций блочных структур.

Необходимость более общего языка для описания условий работы релейных устройств ощущается уже довольно сильно в связи с указывавшейся выше громоздкостью описания в существующих языках. К числу языков, рассматривавших блочное построение структур и более общее описание их, нужно отнести алгоритмический язык для описания структур вычислительных машин (АЛОС), язык логических схем алгоритмов (ЛСА) и язык блочного синтеза.

Первые два языка ориентированы на синтез структур вычислительных машин, а третий имеет более общий характер.

Язык АЛОС [2] разрабатывается в Институте кибернетики АН Укр. ССР. Он рассматривает релейное устройство, как состоящее из двух взаимодействующих между собой частей (рис. 1): управляющей и операционной. Управляющая часть реализует набор функций

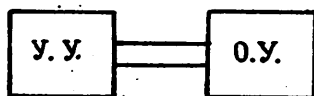


Рис.1. У.У. - управляющее устройство, О.У. - операционное устройство.

(подпрограмм), а операционная, состоящая из операционных блоков (регистров) и функциональных блоков (комбинационных устройств), является заданной. Язык предназначен для построения блочной структуры устройства на

основании заданного набора подпрограмм.

На выполненных этапах работы вопросы преобразования блочных структур и их оптимизации в языке АЛОС еще не рассматривались.

Методы синтеза релейных устройств на основе ЛСА [3], разрабатываемые в Институте проблем передачи информации АН СССР и в Институте кибернетики АН Укр. ССР, рассматривают аналогичную блочную схему. В ней различают (рис.2а): управляющую часть, оперативные блоки (операторы) и логические функциональные блоки, проверяющие выполнение некоторых условий (логические условия). Структура реализуется в виде т.н. схемы Уилкса (рис. 2б). Разработан ряд операций по объединению схем алгоритмов, и по минимизации числа логических условий. Последние основаны на анализе отдельных частей функций возбуждения, т.е. осуществляются по существу на структурной основе. Вопросы получения оптимальных структур не рассматривались.

Рис. 2а. М - микропрограммный автомат, А - выходы на операторные функциональные блоки, z_p - выходы на логические функциональные блоки, R - внешние входы, Ω - внешние условия, О - цепи обратной связи.

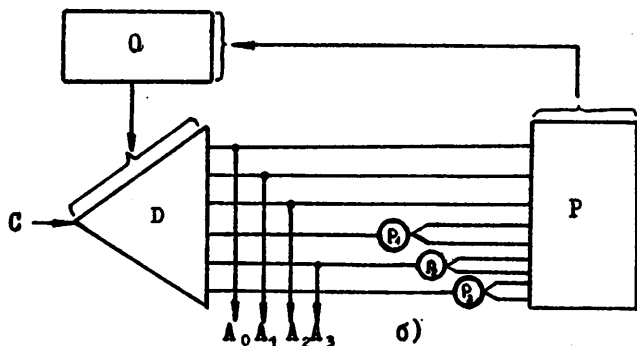
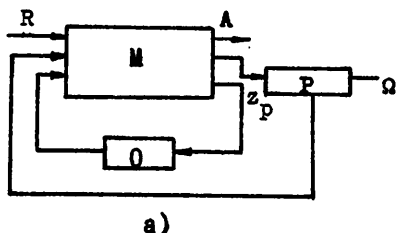


Рис.2б. О - элементы памяти в цепях обратной связи, D - дешифратор, формирующий воздействия на выходы, P - шифратор, формирующий воздействия на элементы памяти, С - синхронизирующий вход, p_1, p_2, p_3 - логические функциональные блоки, A_0, A_1, A_2, A_3 - выходы на операторные функциональные блоки.

Язык блочного синтеза разработан в Институте автоматики и телемеханики (технической кибернетики) [4,5] и рассматривает задачу преобразования структур в более общем виде, когда блоки, входящие в структуру, могут быть любыми и связи между ними не ограничены, а общая форма реализации не задана.

Рассмотрим этот язык и вытекающие из него возможности несколько более подробно.

Как уже говорилось, во многих практических случаях условия работы релейных устройств задаются по блокам.

С одной стороны это связано с тем, что, как правило, технологи формулируют условия работы управляющих релейных устройств, как совокупность отдельных алгоритмов функционирования, соответствующих выполняемым функциям. С другой стороны в ряде случаев алгоритм функционирования содержит только некоторые общие признаки функций, выполняемых устройством, без точного пе-

речисления их. Это, даже для очень простых устройств, делает практически невозможным запись их условий работы в целом.

Пусть, например, имеется устройство с двумя входами А и В и одним выходом z (рис. 3а). На выходе должна появиться единица, если после четной единицы на входе А появляется четная единица на входе В.

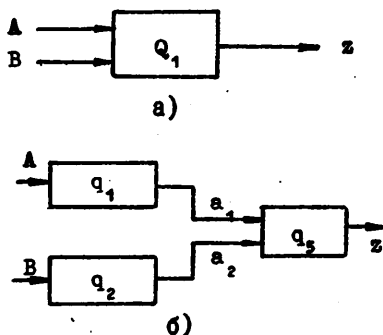


Рис. 3.

В целом запись этих условий требует анализа всех возможных последовательностей изменения состояний входов и соответствующих им последовательностей изменения состояний выхода. Легко видеть, что уже в таком простом случае такая запись представляется исключительно громоздкой, если не полностью невозможной.

Однако, если разделить устройство на блоки (рис. 3б), то указанные выше условия записываются весьма просто, а именно: блок q_1 должен выдавать единицу на выходе при подаче на вход А четной единицы, блок q_2 должен выполнять те же условия в отношении входа В, а блок q_3 должен выдавать на выходе единицу, если на его вход a_1 единица поступит раньше, чем на входе a_2 . Соответствующая запись этих условий работы на языке таблиц переходов весьма проста (табл. 2).

Очевидно, что запись условий работы каждого из блоков должна быть осуществлена на достаточно детальном языке. Однако громоздкость такой записи находится в руках проектировщика, поскольку разделение на блоки может быть сделано достаточно детальным. Нужно сказать, кроме того, что существуют приемы, которые позволяют существенно уменьшить громоздкость записи и при отсутствии разбивки на блоки. Одним из таких приемов является введение обобщенных состояний входов [6], который состоит в том, что в таблице переходов вместо записи в столбцах отдельных состояний входов записываются некоторые совокупности их, соответствующие отдельным переходам, а в таблицах состояний записываются интервалы, соответствующие этим совокупностям.

Одним из важных вопросов является выбор языка для первоначальной записи условий работы блоков.

Таблица 2

Σ_{13}^2

	0	1
1	$\langle 1 \rangle^0$	2
2	3	$\langle 2 \rangle^0$
3	$\langle 3 \rangle^0$	4
4	5	$\langle 4 \rangle^1$
5	$\langle 5 \rangle^1$	2

а)

Σ_{23}^2

	0	1
6	$\langle 6 \rangle^0$	7
7	8	$\langle 7 \rangle^0$
8	$\langle 8 \rangle^0$	9
9	10	$\langle 9 \rangle^1$
10	$\langle 10 \rangle^1$	7

б)

$\Sigma_{13}^2 \Sigma_{23}^2$

	00	10	11	01
11	$\langle 11 \rangle^0$	$\langle 11 \rangle^0$	12	13
12	-	13	$\langle 12 \rangle^1$	13
13	11	$\langle 13 \rangle^0$	$\langle 13 \rangle^0$	$\langle 13 \rangle^0$

в)

$\Delta \Sigma^2$

	00	10	11	01
1	$\langle 1 \rangle^0$	6	-	2
2	3	-	7	$\langle 2 \rangle^0$
3	$\langle 3 \rangle^0$	8	-	4
4	5	-	9	$\langle 4 \rangle^0$
5	$\langle 5 \rangle^0$	10	-	2
6	11	$\langle 6 \rangle^0$	7	-
7	-	8	$\langle 7 \rangle^0$	12
8	13	$\langle 8 \rangle^0$	9	-
9	-	10	$\langle 9 \rangle^0$	14
10	15	$\langle 10 \rangle^0$	7	-
11	$\langle 11 \rangle^0$	16	-	12
12	13	-	18	$\langle 12 \rangle^0$
13	$\langle 13 \rangle^0$	20	-	14
14	15	-	23	$\langle 14 \rangle^0$
15	$\langle 15 \rangle^0$	25	-	12
16	26	$\langle 16 \rangle^0$	18	-
17	26	$\langle 17 \rangle^0$	19	-
18	-	20	$\langle 18 \rangle^0$	28
19	-	21	$\langle 19 \rangle^0$	29
20	30	$\langle 20 \rangle^0$	22	-
21	31	$\langle 21 \rangle^0$	23	-
22	-	24	$\langle 22 \rangle^1$	32
23	-	25	$\langle 23 \rangle^0$	33
24	34	$\langle 24 \rangle^1$	19	-
25	35	$\langle 25 \rangle^0$	19	-
26	$\langle 26 \rangle^0$	6	-	28
27	$\langle 27 \rangle^0$	6	-	29
28	30	-	7	$\langle 28 \rangle^0$
29	31	-	7	$\langle 29 \rangle^0$
30	$\langle 30 \rangle^0$	8	-	32
31	$\langle 31 \rangle^0$	8	-	33
32	34	-	9	$\langle 32 \rangle^1$
33	35	-	9	$\langle 33 \rangle^0$
34	$\langle 34 \rangle^1$	10	-	29
35	$\langle 35 \rangle^0$	10	-	29

г)

В настоящее время предложено достаточно большое количество таких языков. С точки зрения объема информации об условиях работы релейного устройства их можно разделить на две группы. В первой группе условия работы записываются в виде соответствия между входными и выходными последовательностями без описания в явном виде переходов устройств из одного состояния внутренних элементов в другое и без указания зависимости выходов от состояний внутренних элементов. Такие языки получили название "начальных". К начальным языкам относятся, например, языки регулярных событий, одноместных предикатов, таблиц воздействий и др. Другие языки описывают условия работы в виде явного задания функции переходов и функции выходов релейного устройства. Такие языки условно будем называть "стандартными". К стандартным относятся, например, языки: таблиц переходов, диаграмм переходов, матриц переходов, таблиц включений и др.

Наиболее существенными требованиями к формализованным языкам с точки зрения их практического использования являются требования к определению полноты и непротиворечивости первоначальных условий, выраженных в словесном виде.

Условия полноты заключаются в том, что для любой входной последовательности можно либо установить соответствующую выходную, либо выяснить, что соответствующая выходная последовательность не определена (запрещена, либо безразлична), причем существует конечное число операций, позволяющих установить, что все возможные последовательности перечислены.

Условия работы называются непротиворечивыми, если любой входной последовательности соответствует не более чем одна выходная последовательность (в противном случае условия работы физически нереализуемы). Требуется, чтобы алгоритм проверки полноты и непротиворечивости условий работы был проще алгоритма синтеза, т.е. чтобы правильность формулировки условий работы можно было проверить, не проводя (быть может напрасно), синтеза структуры устройства до конца.

При блочном синтезе существенным является также удобство выбранного языка для осуществления преобразований одной блочной схемы устройства в другую, равносильную ей.

Высказанные выше требования к полноте и непротиворечивости выполняются лишь на стандартных языках. В полной мере их выполняет язык таблиц переходов. На других стандартных языках они выполняются только частично. Язык таблиц переходов оказался также удобным и для осуществления операций, связанных с преобразованием блочных структур.

В алфавит языка блочного синтеза входят следующие символы.

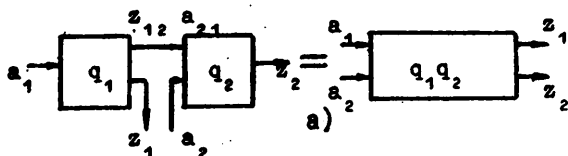
1) Блоки: q_1, q_2, \dots, q_n . Условия работы каждого из блоков характеризуются таблицей переходов. Поэтому операции над символами блоков представляют собой по существу операции над таблицами переходов, определяющими их.

2) Внешние входы и внешние выходы, к которым относятся входы и выходы блоков, не связанные с входами и выходами других блоков, входящих в рассматриваемую блочную структуру. Будем обозначать их соответственно через: a_i и z_i , где индекс означает номер блока, которому они принадлежат.

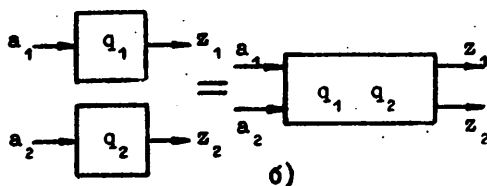
3) Внутренние входы и выходы, к которым относятся входы и выходы блоков, связанные с входами и выходами других блоков, входящих в рассматриваемую блочную структуру. Будем обозначать их соответственно через: a_{ij} и z_{ij} , где первый индекс означает номер блока, к которому вход или выход принадлежит, а второй индекс - номер блока, с которым он связан.

В языке блочного синтеза определены следующие операции.

а) Операция замены последовательно действующих блоков одним эквивалентным (рис. 4а). Будем называть эту операцию умножением блоков и обозначать точкой.



б) Операция замены параллельно действующих блоков одним эквивалентным (рис. 4б). Будем называть эту операцию объединением блоков и обозначать кружком.



в) Операция разделения одного блока на несколько по выходам. Будем называть эту операцию разделением. Блоки, полученные в результате

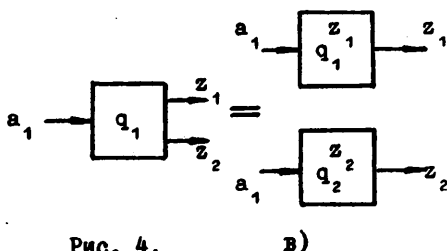


Рис. 4.

Будем обозначать через q_i^j , где нижний индекс означает номер блока, а верхний - выход, по которому он разделен.

Легко показать, что с помощью операций разделения и умножения любая блочная структура может быть преобразована к эквивалентному блоку или к совокупности параллельно действующих блоков, которые затем с помощью операции объединения могут быть сведены опять-таки к одному эквивалентному блоку. Общее доказательство этого дано в [5]. Для иллюстрации характера преобразования рассмотрим пример.

Пусть имеется блочная структура рис. 5а. С помощью разделения блока q_1 на два по выходам z_{12} и z_{13} она преобразуется в блочную структуру рис. 5б. Применим к блокам $q_1^{z_{12}}$ и q_2 операцию умножения. После этого блочная структура примет вид рис. 5в. Применяя после этого операцию объединения к блокам $q_1^{z_{13}}$ и $q_1^{z_{12}} \cdot q_2$ получим структуру рис. 5г, которая с помощью операции умножения преобразуется в один эквивалентный блок Q (рис. 5д).

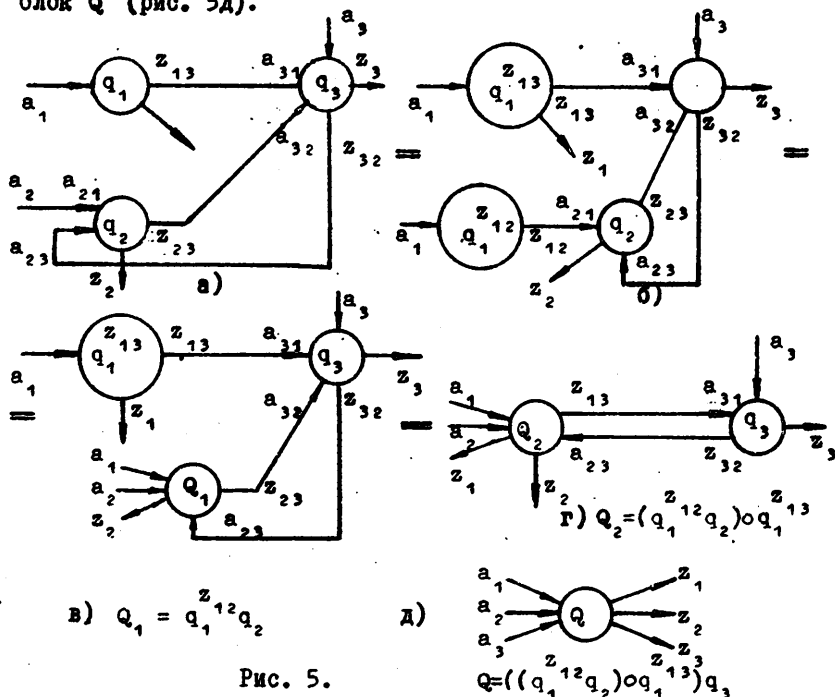


Рис. 5.

Эта последовательность операции может быть записана в виде некоторой формулы:

$$Q = ((q_1^{z_{12}} \cdot q_2) \circ q_1^{z_{13}}) q_3.$$

В [6] для каждой из перечисленных выше операций дан algo-

ритм построения таблицы переходов эквивалентного блока по заданным таблицам переходов первоначальных блоков. Этот алгоритм состоит из операции определения устойчивых состояний и места расположения их в таблице переходов эквивалентного блока и из операции определения переходов из одного устойчивого состояния в другое.

Разработан алгоритм для осуществления декомпозиции блоков, т.е. разделения блока на составляющие части, одна из которых может быть задана заранее. В результате по заданным таблицам переходов первоначального блока и одной из составляющих частей производится построение таблицы переходов другой части [7]. Описанная система равносильных преобразований позволяет перейти от одной блочной структуры к другой без изменения ее условий работы.

Для выбора наиболее оптимальной блочной структуры необходимо иметь возможность соответствующей их оценки. Существенным здесь является получение такой оценки без построения таблиц переходов для всех модификаций блоков, поскольку это является достаточно громоздким. Оказалось, что для операций композиции блоков такая оценка может быть проведена на основе формулы преобразований заданной блочной структуры в любую промежуточную вплоть до одного эквивалентного блока.

Для элементов памяти эта оценка получается из соотношения, связывающих число устойчивых состояний в таблицах переходов первоначальных блоков с числом устойчивых состояний в таблице переходов эквивалентного блока.

Для операции умножения такое соотношение имеет вид:

$$K^* = \sum_{f=1}^m [L_{M_f(z_{1j})} L_{M_1(a_{21})}]_{z_{1j}} = a_{21},$$

где $L_{M_f(z_{1j})}$ - число строк переходов первого блока (блока, имеющего внутренние выходы), имеющих подмножество устойчивых состояний, соответствующее определенной комбинации выходов z_{1j} ,

m - число различных подмножеств $M_f(z_{1j})$,

$L_{M_1(a_{21})}$ - число строк таблицы переходов второго блока (блока, имеющего внутренние входы), содержащих подмножество устойчивых состояний, соответствующее определенной комбинации входов a_{21} .

Под знаком суммы берутся все произведения, содержащие хотя бы одну пару: $z_{1j} = a_{21}$.

Для операции объединения соотношение для получения числа устойчивых состояний эквивалентного блока имеет вид:

$$K^0 = \sum_{e=1}^n I_{M_e(a_{11})} \cdot I_{M_d(a_{2j})},$$

где $I_{M_e(a_{11})}$ - число строк таблицы переходов блока, принятого условно за первый, имеющих подмножество устойчивых состояний $M_e(a_{11})$ размещающихся в подмножестве столбцов, соответствующих некоторой комбинации состояний входов a_{11} ,

n - число различных подмножеств состояний $M_e(a_{1j})$,

$I_{M_d(a_{2j})}$ - число строк таблицы переходов второго блока, содержащих устойчивые состояния в столбцах таблицы, являющихся добавлением к состояниям входов M_e до состояния входов таблицы переходов эквивалентного блока Q .

В [5] дан алгоритм определения числа внутренних элементов в преобразованной блочной структуре на основе первоначальных таблиц переходов без построения таблиц переходов эквивалентных блоков. Число внутренних элементов получается при этом без учета возможного сокращения их за счет возникновения дополнительных эквивалентных состояний в эквивалентном блоке и без учета введения дополнительных состояний для устранения недопустимых состязаний внутренних элементов, т.е. представляет собой лишь некоторую приближенную оценку числа внутренних элементов, находящуюся между нижним и верхним пределом. Практика показывает, что при минимизации числа состояний таблиц переходов первоначальных блоков дополнительная минимизация за счет появления эквивалентных состояний в эквивалентном блоке или незначительна или вовсе отсутствует. Увеличение же числа внутренних состояний для устранения недопустимых состязаний заранее пока еще учтено быть не может.

Для выбора оптимальной блочной структуры необходима также оценка сложности логических частей всех блоков. Эта оценка должна производиться так же, как и оценка числа внутренних элементов, на базе таблиц переходов первоначальных блоков без проведения операции размещения состояний, т.е. тогда, когда известно только число необходимых состояний и переходы из одного состояния в другое. Исследования этой трудной задачи, проведенные в лаборатории, руководимой автором настоящего доклада, позволили предложить некоторую оценку, основанную на частоте вхождений каждого данного состояния в таблицу переходов, характеризуемую соотношением:

$$p_1 = \frac{\omega_1}{N},$$

где N - общее число заполненных клеток в таблице переходов и
 ω_i - число вхождений в таблице переходов i -го состояния
 внутренних элементов, как в устойчивом, так и в неустойчивом
 виде.

Критерий для оценки сложности логической части структуры,
 использующий этот параметр имеет вид:^{x)}

$$R = N(-\sum_{i=1}^k p_i \cdot \log_2 p_i).$$

Предварительное опробование этого критерия на ряде приме-
 ров показало, что он достаточно правильно отражает сложность
 логической части структуры. Однако для окончательной рекомен-
 дации его требуется проведение соответствующих статистических
 испытаний.

Наличие оценок для определения числа элементов памяти и
 сложности логической части структуры создает предпосылки для
 решения задачи направленного поиска оптимальной реализации
 структур релейных устройств, начиная с этапа абстрактного син-
 теза. В настоящее время разработка соответствующих методов про-
 водится под руководством автора настоящего доклада в Институте
 автоматики и телемеханики (технической кибернетики), однако со-
 стояние работ в решении этой задачи можно характеризовать, как
 достаточно начальное.

Наряду с рассмотренной выше задачей преобразования блоч-
 ных структур в направлении нахождения оптимальных решений су-
 щественный интерес для оптимального синтеза релейных устройств
 представляют следующие две задачи.

I. Задача итеративного синтеза релейных устройств. Эта за-
 дача может быть сформулирована следующим образом. Задано неко-
 торое множество алгоритмов функционирования. Необходимо опре-
 делить некоторую совокупность модулей (итеративных блоков), на
 которых реализация этих алгоритмов будет осуществляться опти-
 мальным (в заданном смысле) путем на всей совокупности моду-
 лей (итеративных блоков) или некоторой части из них.

Решение этой задачи весьма важно с точки зрения обоснован-
 ного выбора типовых решений при построении ряда модификаций уп-
 равляющих устройств из типовых блоков или выбора серий типовых
 логических элементов или модулей. Однако исследования, направ-
 ленные к решению этой задачи, по существу еще не развиты.

К задаче итеративного построения релейных устройств может

x) Критерий предложен аспирантом А. Григорян.

быть сведена и упоминавшаяся выше задача построения их структуры на базе операции абстрактного синтеза. Эта задача может быть сформулирована следующим образом: задан алгоритм функционирования релейного устройства (в целом или по блокам) и алгоритмы функционирования модулей, на которых должно быть реализовано устройство. Нужно построить структуру устройства, содержащую наименьшее число модулей, или отвечающую каким-либо другим требованиям минимальности.

Как уже указывалось, заданные модули можно представить себе, как некоторые элементарные релейные устройства, при этом решение сформулированной выше задачи будет сводиться к оптимальной декомпозиции заданного релейного устройства на эти модули. Оптимальные выше результаты по преобразованию блочных структур и оценкам их создают определенные предпосылки для успешного решения этой задачи.

2. Задача оптимального распределения функций в иерархических структурах. Эта задача может быть сформулирована следующим образом. Задана система взаимодействующих между собой релейных устройств, территориально разобщенных между собой. Функции, выполняемые системой в целом, могут быть различным образом распределены между релейными устройствами. Требуется распределить их таким образом, чтобы структура всей системы в целом была в определенном смысле оптимальной.

К этой проблеме в её наиболее простом случае относится, например, задача распределения функций по управлению рядом объектов между местными и центральными автоматическими устройствами. Существенное значение при решении этой задачи имеет также и получение итеративных управляющих устройств.

Исследования в области оптимального распределения функций между управляющими устройствами находится еще в самой первоначальной стадии.

Перейдем теперь к рассмотрению задач направленного поиска минимальных реализаций структур релейных устройств для случая, когда условия их работы заданы в виде таблиц состояний, что имеет место, например, для одноконтурных релейных устройств.

Работы в области структурного синтеза развиты в теории релейных устройств значительно шире, чем в области абстрактного синтеза и поэтому в решении указанной задачи имеется большее число работ. Наиболее существенные результаты получены здесь в двух направлениях.

Одно из них базируется на т.н. "функциональной декомпози-

ций" структур, основная идея которой заключается в последовательном разбиении булевой функции, характеризующей заданные условия работы релейного устройства, на подфункции, зависящие от меньшего числа переменных. (Работы американских ученых Кертиса [5] и Рота и Карпа [6]). В первой работе структурный синтез основан на т.н. "декомпозиционных" картах, представляющих собой перечисление всех состояний переменных при всех возможных разбиениях их по группам. Высказывается эвристическое предположение, что наиболее простая структура получается при выборе разбиений, содержащих наименьшее число пересечений переменных. Из некоторого подмножества таких разбиений выбирается путем перебора разбиение, наиболее подходящее к структурным свойствам заданного набора элементов. Применение метода ограничивается задачами с весьма небольшим числом переменных, т.к. число декомпозиционных карт и их громоздкости резко возрастают с числом переменных.

Во второй работе декомпозиция осуществляется на основе выделения так называемых "совместимых" классов функций с помощью разбиения таблиц состояний по всем возможным комбинациям столбцов (переменных) и сравнения остатков в рабочих и нерабочих состояниях. Направленный выбор осуществляется с помощью сравнения сложности получаемых структур со сложностью заранее полученной структуры (например, в нормальной форме) или заранее установленной величиной и отбрасывания более сложных вариантов на этапе, когда эта сложность, превышающая наперед заданную, становится ясной. Метод весьма громоздок даже при использовании вычислительных машин, и по существу содержит несколько укороченный полный перебор всех возможных решений.

Оба метода не гарантируют получение неизбыточных структур и разработаны лишь для одновыходных структур.

Отметим, что описанные выше методы основаны на весьма распространенной концепции, исходящей из утверждения, что сложность структуры в основном зависит от числа переменных. Это справедливо в общем только для наиболее сложных структур. Для большинства практических случаев сложность структуры в сильной степени зависит от распределения состояний между рабочими и нерабочими, что требует при оценке структур учета этого обстоятельства.

В основу другого направления, разработанного в СССР (Институт автоматики и телемеханики) [10], положены совершенно другие идеи. Основой его является получение на каждом этапе синтеза некоторой неизбыточной части структуры и оценка сложности

ти "тающей" части. Выделение избыточной части производится с помощью определения т.н. "обязательных" букв, что осуществляется путем отыскания в противоположных частях таблицы состояний таких из них, которые отличаются друг от друга значением только одной переменной (т.н. "соседних" состояний). Если исключить эту букву, то остатки указанных состояний в обеих частях таблицы будут одинаковыми, что приведет к противоречивости в реализации, которая ничем не может быть устранена. Соответствующая буква является поэтому обязательной. Переменные, по которым нет ни одной обязательной буквы, являются "несущественными" и поэтому может быть произведено последовательное исключение их. При исключении каждой из несущественных переменных могут появляться новые обязательные буквы. Процесс исключения несущественных переменных продолжается до тех пор, пока в каждой из оставшихся переменных хотя бы в одной из строк таблицы состояний не появится обязательная буква.

Уже на этом первом этапе появляется необходимость выбора из всех возможных порядков вычеркивания несущественных переменных такого, который приводил бы к наиболее простой реализации структуры. В.М. Копыленко [10] предложил для этого критерий, основанный на оценке вероятности появления при вычеркивании данной переменной соседних состояний в противоположных частях таблицы состояний. Эта вероятность оценивается следующим выражением:

$$R = n_1^0 n_0^1 + n_1^1 n_0^0,$$

где n_1^0 и n_1^1 — число состояний в рабочей части таблицы состояний, имеющих значение данной переменной, равное соответственно нулю и единице, и n_0^0 и n_0^1 — число состояний с такими же свойствами в нерабочей части таблицы состояний.

Вычеркивается переменная, для которой значение критерия R получается наименьшим с тем, чтобы оставались наибольшие возможности для вычеркивания других переменных.

Пусть, например, имеется таблица состояний 3а. Обязательные буквы имеются в ней только для переменных x_6 и x_5 (подчеркнуты одной чертой). В нижней строке таблицы приведены значения критерия R для остальных переменных, являющихся несущественными. Наименьшее значение критерия имеет переменная x_{10} . Произведем поэтому ее вычеркивание в первую очередь. Будем определять обязательные буквы в дальнейшем с помощью записи состояний в восьмиричном счислении (табл.3б). при вычеркивании какой-либо переменной пересчет номеров состояний при этом сводится к

Таблица 3

	x_{10}	x_9	x_8	x_7	x_6	x_5	x_4	x_3	x_2	x_1	y_4	y_3	y_2	y_1	y_4	y_3	y_2	y_1
65	0	0	0	1	0	0	0	0	0	1	0	1	0	1	0	1	0	1
123	0	$\langle 0 \rangle$	0	1	1	1	1	$\langle 0 \rangle$	1	1	0	1	7	3	0	1	7	3
170	0	0	1	0	1	0	1	0	1	0	0	2	5	2	0	2	5	2
344	0	1	0	1	<u>0</u>	1	1	0	0	0	0	5	3	0	0	5	3	0
360	0	1	0	1	1	<u>0</u>	1	$\langle 0 \rangle$	0	0	0	5	5	0	0	5	5	0
437	0	1	1	$\langle 0 \rangle$	1	1	0	1	0	1	0	6	6	5	0	6	6	5
450	0	1	1	1	0	0	0	0	1	0	0	7	0	2	0	7	0	2
615	1	[0]	0	(1)	1	$\langle 0 \rangle$	0	1	1	1	1	1	4	7	0	1	4	7
680	1	<u>0</u>	1	0	1	0	1	0	0	0	1	2	5	0	0	2	5	0
704	1	0	1	1	0	0	0	0	0	0	1	3	0	0	0	3	0	0
808	1	1	<u>0</u>	0	1	0	1	0	0	0	1	4	5	0	0	4	5	0
358	0	[1]	0	1	1	0	0	$\langle 1 \rangle$	1	0	0	5	4	6	0	5	4	6
376	0	$\langle 1 \rangle$	0	1	<u>1</u>	<u>1</u>	1	0	0	0	0	5	7	0	0	5	7	0
394	0	1	1	0	0	0	0	1	1	1	0	6	0	7	0	6	0	7
424	0	<u>1</u>	<u>1</u>	0	1	0	1	0	0	0	0	6	5	0	0	6	5	0
511	0	1	1	$\langle 1 \rangle$	1	1	1	1	1	1	0	7	7	7	0	7	7	7
559	1	0	0	(0)	1	0	1	1	1	1	1	0	5	7	0	0	5	7
629	1	0	0	1	1	$\langle 1 \rangle$	1	$\langle 1 \rangle$	0	1	1	1	7	5	0	1	7	5
902	1	1	1	0	0	0	0	1	1	0	1	6	0	6	0	6	0	6
	41	46	44	44	-	-	43	58	47	44								

a)

б)

в)

	y_4	y_3	y_2	y_1	y_4	y_3	y_2	y_1	y_4	y_3	y_2	y_1	x_9	x_8	x_7	x_6	x_5	x_4
65	0	1	0	1	0	1	0	0	0	1	0	0	0	0	1	0	0	0
123	0	1	6	3	0	1	6	2	0	1	6	0	<u>0</u>	0	1	1	1	<u>0</u>
170	0	2	4	2	0	2	4	2	0	2	4	0	0	1	0	1	0	0
344	0	5	2	0	0	5	2	0	0	5	2	0	1	0	1	<u>0</u>	1	0
360	0	5	4	0	0	5	4	0	0	5	4	0	1	0	1	1	<u>0</u>	0
437	0	6	6	5	0	6	6	4	0	6	6	4	1	1	<u>0</u>	1	1	1
450	0	7	0	2	0	7	0	2	0	7	0	2	1	1	1	0	0	0
615	0	1	4	7	0	1	4	6	0	1	4	4	<u>0</u>	0	<u>1</u>	1	<u>0</u>	1
680	0	2	4	6	0	2	4	0	0	2	4	0	<u>0</u>	1	0	1	0	0
704	0	3	0	0	0	3	0	0	0	3	0	0	0	1	1	0	0	0
808	0	4	4	0	0	4	4	0	0	4	4	0	1	<u>0</u>	0	1	0	0
358	0	5	4	6	0	5	4	6	0	5	4	4	<u>1</u>	0	1	1	0	<u>1</u>
376	0	5	6	0	0	5	6	0	0	5	6	0	<u>1</u>	0	1	<u>1</u>	<u>1</u>	0
394	0	6	0	7	0	6	0	6	0	6	0	4	1	1	0	0	0	1
424	0	6	4	0	0	6	4	0	0	6	4	0	<u>1</u>	<u>1</u>	0	1	0	0
511	0	7	6	7	0	7	6	6	0	7	6	4	1	1	<u>1</u>	1	1	1
559	0	0	4	7	0	0	4	6	0	0	4	4	0	0	<u>0</u>	1	0	1
629	0	1	6	5	0	1	6	4	0	1	6	4	0	0	1	1	<u>1</u>	<u>1</u>
902	0	6	0	6	0	6	0	6	0	6	0	4	1	1	0	0	0	1

г)

д)

е)

ж)

 $\bar{x}_9, \bar{x}_8,$ $\bar{x}_6,$ $\bar{x}_5, \bar{x}_4,$ $\bar{x}_7,$ $\bar{x}_9, \bar{x}_7, \bar{x}_5,$ $\bar{x}_9,$ $\bar{x}_8,$ $\bar{x}_7,$ $\bar{x}_5, \bar{x}_3,$ $\bar{x}_9, \bar{x}_8,$ $\bar{x}_7,$ $\bar{x}_5, \bar{x}_3,$ $\bar{x}_7,$ $\bar{x}_5, \bar{x}_3,$

замене в соответствующей триаде всех значений переменной нулями. В соответствии с этим после вычеркивания переменной получим таблицу 3в. В ней появятся новые обязательные буквы в переменных x_9 и x_8 (в табл. 3а подчеркнуты двумя чертами). Производя дальнейшие вычеркивания переменных в порядке значений критерия R, получим в конце концов таблицу состояний 3ж (появление обязательных букв при поочередном вычеркивании переменных x_1 , x_2 и x_4 отмечено соответственно круглыми, квадратными и угловыми скобками). В ней будут содержаться произведения обязательных букв, записанные справа от таблицы.

В [II] доказано, что, если эти произведения не содержатся в противоположной части таблицы состояний (реализуют таблицу состояний непротиворечиво), то они являются минимальными членами ядра, т.е. существует хотя бы одно состояние таблицы состояний, которое реализуется только этим членом и поэтому этот член должен входить в любую неизбыточную реализацию структуры. Произведения обязательных букв, содержащиеся в противоположной части таблицы состояний (т.н. "недостаточные минимальные члены"), реализуют ее противоречиво и поэтому требуют доопределения с помощью некоторых дополнительных функций.

В таблице 3ж минимальными членами ядра по рабочим состояниям являются произведения обязательных букв: $\bar{x}_9\bar{x}_8$ и $\bar{x}_9x_7\bar{x}_5$. Для нерабочих состояний минимальные члены ядра отсутствуют. Недостаточными минимальными членами являются по рабочим состояниям \bar{x}_6 , $\bar{x}_5\bar{x}_3$, \bar{x}_7 , \bar{x}_8 и \bar{x}_9 и по нерабочим состояниям - x_9x_3 , $x_9x_6x_5$, x_9x_8 , x_7 и x_5x_3 .

Таким образом на первом же этапе синтеза выделяется некоторая часть структуры, о которой в точности известна сложность ее реализации (на рис. 6 она обозначена через P_ϵ), и некоторая дополнительная часть в виде доопределяющих функций, которая еще подлежит реализации и сложность которой может быть оценена только приблизительно (на рис. 6 она обозначена через I_ϵ).

Обычно недостаточные минимальные члены реализуют таблицу состояний с некоторой избыточностью. Поэтому для неизбыточной реализации из множества их должно быть выбрано некоторое подмножество. При этом могут быть учтены структурные особенности элементов, на которых должна быть реализована структура, ограничения по числу входов и по затуханиям и т.п.

Для этой цели, а также для определения таблиц состояний доопределяющих функций, строится т.н. "таблица реализации",

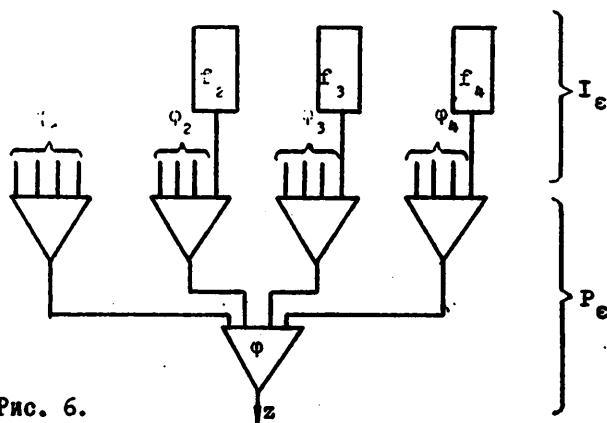


Рис. 6.

столбцам которой соответствуют состояниям таблицы состояний (рабочим и нерабочим), а строки – минимальным членам. В клетках таблицы для каждой строки отмечаются крестиком состояния, которые реализуются данным минимальным членом.

Как известно, структура релейного устройства может быть реализована как по рабочим, так и по нерабочим состояниям (в последнем случае на выходе должен быть включен элемент, имеющий в своем составе отрицание). Будем называть сторону таблицы реализации, (а вместе с тем и таблицу состояний), по которой производится реализация структуры, "реализуемой" стороной, а противоположную сторону – "нереализуемой".

Таблица реализации для рассмотренного выше примера, приведенного к таблице состояний Z_k , показана в табл. 4а.

При выборе избыточного подмножества минимальных членов существенное значение имеет т.н. "характеристическое число", равное минимальному числу входов элемента, подача на которые единицы (нуля) однозначно определяет значение выхода элемента. Обозначим его через $[w]$. Например, для элементов "И" и "ИЛИ": $[w] = 1$. Действительно, для получения в элементе "ИЛИ" на выходе единицы достаточно на один из его входов подать воздействие, равное единице, а для элемента "И" для получения на выходе нуля достаточно подать воздействие, равное нулю, также на один из его входов.

Очевидно, что для избыточной реализации заданной структуры для этих элементов нужно выбрать такой набор минимальных

Таблица 4

	65	123	344	360	437	450	615	680	704	808	358	376	391	424	511	559	629	
$\bar{x}_9 \bar{x}_3$	+	+						+	+									4-0
$\bar{x}_9 \bar{x}_6$	+		+			+			+			+						4-1
$\bar{x}_5 \bar{x}_3$	+			+		+		+	+	+			+					6-1
$\bar{x}_5 \bar{x}_7$					+			+		+			+	+				3-3
$\bar{x}_9 x_7 \bar{x}_5$	+						+		+									3-0
$\bar{x}_9 \bar{x}_9$	+	+					+	+	+							+	+	5-2
\bar{x}_8	+	+	+	+			+			+	+	+				+	+	6-4

а)

$\bar{x}_9 \bar{x}_3$	+	+						+	+									2-1
$\bar{x}_9 \bar{x}_6$	*		+			+		*		*		+						3-1
$\bar{x}_5 \bar{x}_3$	*			+		+		*	*	+			+					3-3
$\bar{x}_5 \bar{x}_7$					+			*		+			+	+		+		1-0
$\bar{x}_9 x_7 \bar{x}_5$	*						+	*	*									2-2
$\bar{x}_9 \bar{x}_9$	*	*					+	*	*	*						+	+	4-4
\bar{x}_8	*	*	+	+			+			+	+	+				+	+	

б)

$\bar{x}_9 \bar{x}_3$	+	+						+	+									
$\bar{x}_9 \bar{x}_6$	*		+			+		*		*		+						
$\bar{x}_5 \bar{x}_3$	*			+		+		*	*	+			+					
$\bar{x}_5 \bar{x}_7$					+			*		+			+	+		+		
$\bar{x}_9 x_7 \bar{x}_5$	*						+	*	*									
$\bar{x}_9 \bar{x}_9$	*	+					*	*	*							+	+	
\bar{x}_8	*	+	+	+			*			+	+	+				+	+	

в)

членов, чтобы в каждом из столбцов реализуемой части таблицы состояний был по крайней мере один крестик и ни одного из минимальных членов нельзя было бы исключить без того, чтобы хотя бы в одном столбце не осталось ни одного крестика. К элементам с $[w] = 1$ относятся также: "ИЛИ-НЕ", "И-НЕ" и условно можно отнести элементы "сумма по модулю два", "равнозначность" и "запрет" [10]. К элементам с $[w] > 1$ относятся, напри-

мер, мажоритарные элементы. Для избыточной их реализации нужно выбрать такое подмножество минимальных членов, чтобы в каждом из столбцов реализуемой части число крестиков составляло $\frac{n+1}{2}$, где n - число входов элементов, а для нереализуемой части было бы равно или меньше $\frac{n+1}{2} - 1$.

Из всех возможных подмножеств минимальных членов, удовлетворяющих этим условиям, нужно выбрать такой, который давал бы реализацию наиболее близкую к абсолютно минимальной форме. В [10] предложено производить этот выбор на основании оценки сложности реализации доопределяющих функций. Эта сложность, как показано в [12], существенно зависит от общего числа переменных и от распределения их между рабочими и нерабочими состояниями (рис. 7).

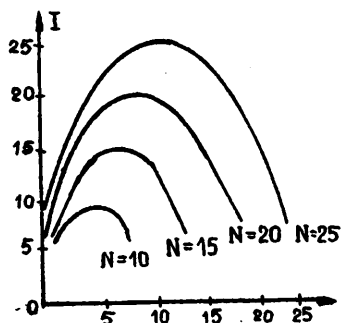


Рис. 7

Для непротиворечивой реализации заданной таблицы состояния, необходимо чтобы при доопределении недостаточных минимальных членов доопределяющая функция для элементов с $[w] = 1$ устраняла полностью крестики нереализуемой части таблицы реализации, а для $[w] > 1$ сводила их к числу меньшему или равному $[w] - 1$. В связи с этим в число нерабочих состояний для до-

определяющих функция должны включаться только те, в которых для данного недостаточного минимального члена в нереализуемой части таблицы реализации имеются крестики.

Представим совокупность доопределяющих функций в виде некоторой функции

$$F = \bigvee_{i=1}^k f_i,$$

где k - число недостаточных минимальных членов, входящих в избыточное подмножество их. Очевидно, что эта функция должна реализовать все рабочие состояния, реализуемые доопределяемыми минимальными членами. Число их для каждого из вариантов будет неизменным. Поэтому для получения наименьшей сложности реализации доопределяющих функций нужно выбирать такое подмножество минимальных членов, которое имеет наименьшее число крестиков в нереализуемой части таблицы реализации.

Для простоты изложения рассмотрим алгоритм выбора подмноже-

ства минимальных членов, удовлетворяющих этим условиям, для случая элементов с $[w] = 1$.

Алгоритм состоит из следующих операций.

а) Выбирается минимальный член, в строке которого имеется наименьшее число крестиков в нереализуемой части таблицы состояний.

б) Из всех остальных строк в реализуемой и нереализуемой частях таблицы реализации вычеркиваются крестики в столбцах, в которых крестики имеются в выбранной строке. В реализуемой части это делается для того, чтобы устранить избыточную реализацию, а в нереализуемой части - для того чтобы выбрать подмножество минимальных членов, содержащие крестики в минимальном числе столбцов.

в) Из оставшихся минимальных членов снова выбирается такой, который содержит наименьшее число крестиков, и указанные операции продолжают до тех пор, пока в остальных строчках реализуемой части таблицы реализации не окажутся только вычеркнутые крестики, что будет свидетельствовать о реализации всех состояний реализуемой части.

Сложность реализации заданной таблицы состояний может существенно зависеть также и от типа элемента, включаемого на выходе структуры, в частности от того реализуется ли таблица состояний по рабочим или нерабочим состояниям. Для выбора наименее сложной реализации таблицы реализации строятся для каждого из возможных вариантов. Для каждой из них описанным выше путем выделяется неизбыточное подмножество минимальных членов. После этого для каждого из вариантов определяется:

1) сложность реализации уже реализованной части структуры

$$P = \sum p_i q_i; \quad (2)$$

2) сложность реализации доопределяющих функций

$$I = N \log_2 N - n_1 \log_2 n_1 - n_0 \log_2 n_0. \quad (3)$$

Здесь q_i - число элементов i -го типа, p_i - "вес" элемента i -го типа, n_1 - число столбцов в реализуемой части таблицы реализации, за исключением столбцов, соответствующих состояниям, реализуемым минимальными членами ядра, n_0 - число столбцов, нереализуемой части, имеющих крестики в строках, соответствующих выбранному подмножеству минимальных членов и $N = n_1 + n_0$.

Критерий I , предложенный в [12] Л.Шеломовым, как показали статистические исследования на УВМ, достаточно правильно отра-

дает относительную сложность реализации таблицы состояний, однако не дает точного представления о числе необходимых для реализации элементов. Поэтому этот критерий может использоваться только в относительном виде.

Можно с достаточным основанием предположить, что окончательные сложности двух вариантов реализации структуры для одних и тех же условий работы релейных устройств при различной сложности уже реализованных частей этих структур будут одинаковыми, если выполняется следующее соотношение:

$$\frac{p_1}{p_2} = \frac{I_2}{I_1}, \quad (4)$$

где p_1 и p_2 - точные оценки сложности уже реализованных частей структуры в соответствии с (2), а I_1 и I_2 - относительные оценки сложности реализации доопределяющих функций для этих вариантов в соответствии с (3).

Преобразуя выражение (4), получим условия одинаковой сложности реализации в виде:

$$S = p_1 I_1 = p_2 I_2.$$

Величина $S = p_1 I_1$ может служить таким образом для оценки сложности реализации различных вариантов структур и чем она будет больше, тем сложнее будет реализация структуры.

Алгоритм направленного поиска минимальной реализации можно сформулировать в следующем виде.

1. С помощью вычеркивания несущественных переменных определяется минимальный перечень существенных переменных для заданной таблицы состояний, обеспечивающий приближение к наиболее простой реализации.

2. На основании перечня обязательных букв определяются минимальные члены ядра и недостаточные минимальные члены.

3. Строятся таблицы реализации для различных вариантов включения элементов на выходе структуры и с помощью выделения избыточных подмножеств минимальных членов производится оценка сложности их реализации. Выбирается вариант с наименьшей сложностью реализации.

4. Для выбранного варианта строятся таблицы состояний доопределяющих функций и по отношению к каждой из них операции алгоритма применяются в указанном порядке до тех пор пока все доопределяющие функции не будут реализованы (реализация доопределяющих функций производится в определенном порядке, исключающем возникновение избыточности).

Для иллюстрации применения описанного выше алгоритма рассмот-

рим пример реализации таблицы состояний Z_3 на однородной среде, состоящей из элементов "ИЛИ-НЕ", расположенных в двумерной сетке, в которой выходы каждого из элементов могут соединяться только со входами соседних элементов, а ввод и вывод информации осуществляется только с краев сетки. Соединения между элементами устанавливаются путем настройки однородной сети с помощью ключей типа "И", располагаемых на входах и выходах элементов и управляемых извне. В сетке помимо логических элементов предусмотрены соединительные элементы, также управляемые извне. Каждый из логических элементов имеет четыре входа и выхода, причем одновременно с каждой из сторон элемента в сетке может быть использован только один вход или выход. Максимальное число входов при одном выходе составляет таким образом три.

Для простоты изложения будем реализовывать структуру по рабочим состояниям. В соответствии с этим используем для выхода структуры в целом два элемента "ИЛИ-НЕ", включенные друг за другом.

В таблице реализации 4а в соответствии с алгоритмом выберем минимальный член, имеющий наименьшее число крестиков в нереализуемой части. Такими членами являются минимальные члены ядра: $\bar{x}_9\bar{x}_3$ и $\bar{x}_9x_7\bar{x}_5$, не имеющие крестиков в нереализуемой части таблицы реализации. Поскольку минимальный член $\bar{x}_9\bar{x}_3$ реализует большее число состояний в реализуемой части, то выберем его первым, вычеркнув в соответствующих столбцах крестики в других строках таблицы (в табл. 4б, вычеркнутые крестики обозначены символом *). Вторым выбираем по тем же основаниям минимальный член ядра: $\bar{x}_9x_7\bar{x}_5$. Производя дополнительное вычеркивание крестиков, получим таблицу реализации в виде табл. 4в. В ней остались нереализованными состояния: 344, 360, 437, 450 и 808. Поскольку на элементе остался неиспользованным всего один вход, а указанные выше состояния требуют для своей реализации нескольких минимальных членов, то заменим последние некоторой функцией f_1 , которая будет содержать в качестве рабочих все указанные выше нереализованные состояния, а в качестве нерабочих - все нерабочие состояния первоначальной таблицы состояний (табл. 5а). Структуру устройства получим после этого в виде рис. 8а.

Приступим к реализации функции f_1 . Для простоты будем реализовывать ее снова по рабочим состояниям, что потребует включения двух элементов "ИЛИ-НЕ". Таблица реализации для этой функции представлена в табл. 5б. Справа для каждой строки при-

f_1

	x_9	x_8	x_7	x_6	x_5	x_3
344	1	0	1	<u>0</u>	1	0
360	1	0	1	1	<u>0</u>	0
437	1	1	<u>0</u>	1	1	1
450	1	1	1	0	0	0
808	1	<u>0</u>	0	1	0	0
358	1	0	1	1	0	<u>1</u>
376	1	0	1	<u>1</u>	<u>1</u>	0
391	1	1	0	0	0	1
424	1	<u>1</u>	0	1	0	0
511	1	1	<u>1</u>	1	1	1
559	0	0	0	1	0	1
625	0	0	1	1	1	1

а)

 f'_1

	x_9	x_8	x_7	x_6	x_5	x_3
437	1	1	<u>0</u>	1	1	1
358	1	0	1	1	0	1
376	1	0	1	1	1	0
391	1	1	0	0	0	1
424	1	1	0	1	0	0
511	1	1	<u>1</u>	1	1	1
559	0	0	0	1	0	1
625	0	0	1	1	1	1

в)

Таблица 5

	344	360	437	450	808	358	376	391	424	511	559	625
x_9	+											
x_8			⊕				+					
x_7			+	(+)	+				+			
x_6										+		
x_5								+			+	
x_3											+	+

б)

 $f'_{2,1}$

	x_9	x_8	x_7	x_6	x_5	x_3
344	1	0	1	0	1	0
391	1	1	0	0	0	1

г)

 $f'_{1,1}$

	x_9	x_8	x_7	x_6	x_5	x_3
360	1	0	1	1	0	0
808	1	0	0	1	0	0
424	1	1	0	1	0	0

д)

ведено число крестиков в реализуемой и нереализуемой частях. Вы-
 бираем последовательно друг за другом минимальные члены: $\bar{x}_5\bar{x}_3$,
 и \bar{x}_6 . После этого нереализованным остается только состояние
 437. Заменяем его функцией f'_1 , после чего структура устройст-
 ва примет вид, изображенный на рис. 86.

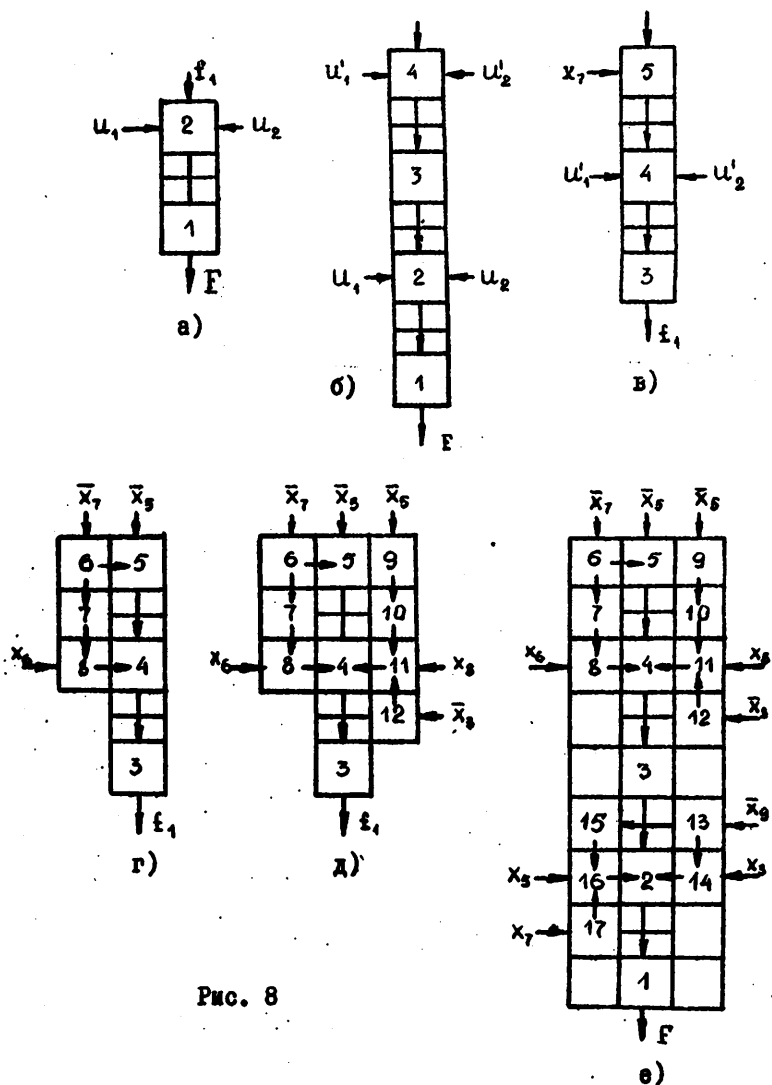


Рис. 8

Таблица состояний для функции r_1' представлена в табл. 5в. Легко видеть, что она реализуется минимальным членом \bar{x}_7, x_5 , что при реализации на входах элемента "ИЛИ-НЕ" можно представить в виде x_7, \bar{x}_5 . Реализация этого выражения приводит к структуре устройства, представленной на рис. 8в.

Перейдем теперь к реализации функций: u_2, u_1', u_2 и u_1 , начав с первой из них.

Поскольку минимальный член $u'_2 = \bar{x}_6$ реализует состояние 39I из нереализуемой части таблицы реализации 5б, то его необходимо доопределить. Доопределяющая функция $f'_{2,1}$ должна иметь в качестве рабочего состояние 344 и в качестве нерабочего - состояние 39I. Она может быть реализована, как это видно из таблицы состояний 5г с помощью букв \bar{x}_8, x_7, x_5 и \bar{x}_3 . Наиболее выгодно взять буквы x_7 или \bar{x}_3 , т.к. обе эти буквы реализуют ранее вычеркнутое состояние 450, что позволит восстановить его в строке \bar{x}_6 (в табл. 5б обозначено символом \oplus), вычеркнув соответствующий крестик для избежания избыточной реализации и для упрощения реализации доопределяющей функции минимального члена $\bar{x}_5 \bar{x}_3$ из строки последнего (в табл. 5б обозначено символом $(+)$). Из указанных двух букв выберем букву x_7 , поскольку она совпадает с одним из входов для функции f'_1 , что может уменьшить число внешних входов однородной среды. Таким образом функция $u'_1 = \bar{x}_6 x_7 = x_6 \downarrow \bar{x}_7$.

Соответствующая структура представлена на рис. 8г. Для подачи переменных на входы граничных элементов реализации буквы \bar{x}_7 в функции u'_1 произведена через дополнительный элемент 7.

Таблица состояний для доопределяющей функции $f'_{1,1}$ минимального члена u'_1 представлена в табл. 5д. Эта функция реализуется одной буквой \bar{x}_5 . Таким образом, $u'_1 = \bar{x}_8 \bar{x}_5 \bar{x}_3$. Реализация этой функции с учетом размещения входов с внешней стороны среды представлена на рис. 8д.

Функции u_1 и u'_1 , являющиеся минимальными членами ядра, не требуют доопределения. Реализация их, а вместе с тем и структура всего релейного устройства в целом, представлена на рис. 8д.

Рассмотренный метод синтеза имеет достаточно универсальный характер. На основе его разработаны алгоритмы для синтеза многовыходных структур, для структур с заданной надежностью и устойчивостью (устранение состязаний в исполнительных цепях) и т.п. Он достаточно просто позволяет синтезировать структуры на различных типовых элементах, как контактных, так и бесконтактных. Для элементов с $[w] = 1$ алгоритм обладает весьма быстрой сходимостью, что сильно сокращает трудоемкость синтеза и значительно расширяет предел числа переменных, при котором можно синтезировать структуру релейных устройств от руки. Для элементов с $[w] > 1$ сходимость алгоритма является несколько меньшей, однако и для этого случая применение описанного метода дает вполне удовлетворительные результаты.

В заключение укажем, что применение направленного поиска ми-

нимальных реализаций структур релейных устройств является в настоящее время вероятно единственным выходом из того тяжелого положения с практическим применением теории релейных устройств, которое имело место до сих пор в связи с исключительной громоздкостью существовавших "классических" методов. Особенно важным является распространение направленного поиска на этапы абстрактного синтеза и разработка методов синтеза структур без этапов структурного синтеза.

Л и т е р а т у р а

1. В.Д. Казаков. Нахождение максимального числа простых импликантов произвольной логической функции n переменных. Сб. "Автоматическое регулирование и управление. Изд. АН СССР, 1960.
2. В.М. Глушков, А.А. Лetichevский. Язык для описания алгоритмических структур вычислительных машин и устройств. Семинар "Теория автоматов", Изд. АН УССР, Киев, 1966.
3. В.Г. Лазарев. Вопросы синтеза управляющих устройств системы распределения информации. Диссертация на соискание ученой степени доктора технических наук.
4. М.А. Гаврилов. Построение релейных устройств и конечных автоматов из блоков. Изв. АН СССР, Техническая кибернетика, 1963, № 3, стр. 13.
5. М.А. Гаврилов. Оценка условий работы релейных устройств с точки зрения сложности их реализации. Сб. Абстрактная и структурная теория релейных устройств. Наука, Москва, 1966 г.
6. М.А. Гаврилов. Синтез таблиц состояний. Учеб. пособие по курсу "Структурная теория релейных устройств", часть IV, Изд. Всес. заочн. энергетического института. Москва, 1964.
7. А.К. Григорян. Метод декомпозиции конечных автоматов. Автоматика и телемеханика, 1967.
8. H.A. Curtis. A New Approach to the Design of Switching Circuit. Van Nostrand Pres. Ceton, 1962.
9. I.P. Roth, R.M. Karp. Minimisation over boolean graphs. IBM J. Res. Develop., 1962, vpl. 6, N 2.
10. М.А. Гаврилов, В.М. Копыленко. Метод минимизации структуры бесконтактных релейных устройств на функционально пол-

ных наборах элементов. Труды Института электроники и вычислительной техники АН Латв.ССР, 1966 г.

11. В.П.Диденко. Методы минимизации структур релейных устройств. Диссертация на соискание ученой степени кандидата технических наук, 1964 г.

12. И.А. Неломов. Критерий сложности булевых функций. Сб. Проблемы кибернетики, 1966, вып. 17.