ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ

Сборник трудов
Института математики СО АН СССР

1966 г. Выпуск 25

АЛГОРИТМИЧЕСКИЙ СИНТЕЗ ДИСКРЕТНЫХ ВЫЧИСЛИТЕЛЬНЫХ УСТРОЙСТВ

И.В. Иловайский, Б.А. Сидристый, Я.И. Фет.

I. Введение

Задача синтеза дискретного вычислительного устройства состоит в том, чтобы по заданному описанию функционирования устройства получить его схему. При решении этой задачи особое внимание уделяется следующим требованиям:

I. Описание функционирования должно задаваться в как можно более общей форме (т.е. на самых ранних этапах проектирования).

2. Язык описания функционирования должен быть удобным для практического применения и достаточно формализованным, чтобы

Основное содержание настоящей работы было доложено в сентябре 1966 г. на Всесоюзном коллоквиуме по автоматизации синтеза дискретных вычислительных устройств.

обеспечить возможность алгоритмической переработки исходного описания.

3. Полученная в результате синтеза схема должна быть оптимальной в некотором смысле (например, по количеству оборудования).

Для дискретных вычислительных устройств наиболее общей формой описания является описание алгоритма выполнения команд. Идея использования такого описания в качестве исходной информации для проектирования дискретных вычислительных устройств была предложена в работе [I].

Вся работа дискретного вычислительного устройства складывается из множества пересылок информации (с преобразованиями или без них) между запоминающими элементами (регистрами). Именно в такой форме инженер обычно обдумывает проектируемое устройство. Поэтому удобным для практического применения языком описания функционирования дискретных вычислительных устройств следует считать язык, отражающий пересылки информации между регистрами. Таким является, например, язык регистровых передач([2] , [3]) или адресный язык [4] .

В статье [5] была предпринята попытка осуществить алгоритмический переход от описания системы команд, составленного на адресном языке, к схеме дискретного вычислительного устройства, однако при этом не гарантировалось получение оптимальных структур.

Рассматриваемый метод синтеза может стать эффективным, если обеспечить возможность перебора структур дискретного вычислительного устройства (с целью оптимизации) еще в записи на формальном языке. Полученную в результате этих преобразований оптимизированную запись можно рассматривать как описание функциональной схемы и использовать на дальнейших этапах синтеза.

В настоящей статье предлагается система машинного синтеза дискретных вычислительных устройств, основу которой составляет описанная выше постановка задачи. Работа системы делится на несколько этапов (рис.I):

Первый этап. Исходная запись алгоритма работы устройства на формальном языке преобразуется в описание функциональной схемы устройства;

Второй этап. Из описания функциональной схемы получается подробное описание логической схемы проектируемого устрой-

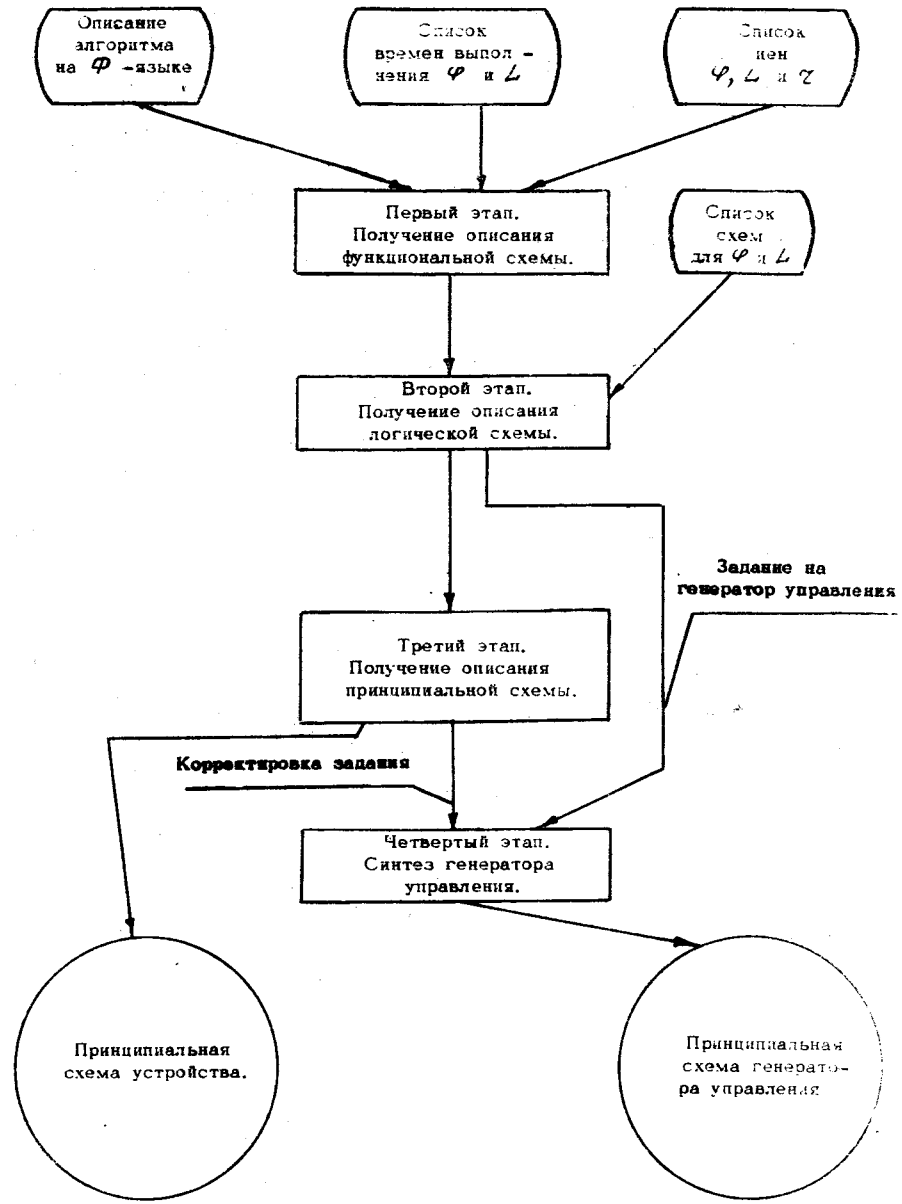


Рис.1. Общая блок-схема системы синтеза.

ства на языке булевых уравнений;

Третий этап. Описание схемы устройства на языке булевых уравнений используется для получения описания принципиальной схемы;

Четвертый этап. На основе информации о временных соотношениях в схеме устройства, полученной на предыдущих этапах, синтезируется генератор управления.

Все перечисленные действия могут быть формализованы и запрограммированы. Предполагается, что система машинного синтеза должна представлять собой набор программ, выполняющих эти действия. Эта система предназначена для выполнения работы, заключенной между ранними этапами проектирования, на которых составляется описание работы дискретного устройства на формальном языке, и более поздними этапами технического проектирования (разработки конструкторской и технологической документации).

2. Язык описания

В качестве формального языка для описания исходной информации в рассматриваемой системе предлагается язык пересылок (Ф-язык). По своей структуре Ф-язык близок к адресному языку [4]. В Ф-языке действуют два типа операторов: операторы пересылок и операторы перехода. Оператор пересылки имеет вид:

$$('a[n_a],'b[n_b],\ldots)\overset{\varphi}{\Rightarrow}c[m],d[m],\ldots,$$

где $a,b,\ldots,-n_i$ - разрядные переменные, к значениям которых применяется функциональное преобразование φ. Результат действия оператора φ становится значением m - разрядных переменных $c,d,\ldots$. Оператор условного перехода имеет вид:

$$P(L_i)(\alpha_i)\quad i=1,2,\ldots\ ,$$

где P - символ условного перехода, L_i - логическое условие, α_i - метка, по которой происходит переход, если логическое условие L_i = I. Оператор безусловного перехода имеет вид отдельной метки. Количество и типы функций и логических условий в языке не фиксируются, а выбираются в каждом конкретном случае.

Ф-язык ооладает той особенностью, что если переменным, функциям и логическим условиям сопоставить схемы, то получится один из вариантов проекта. Однако этот вариант может ока-

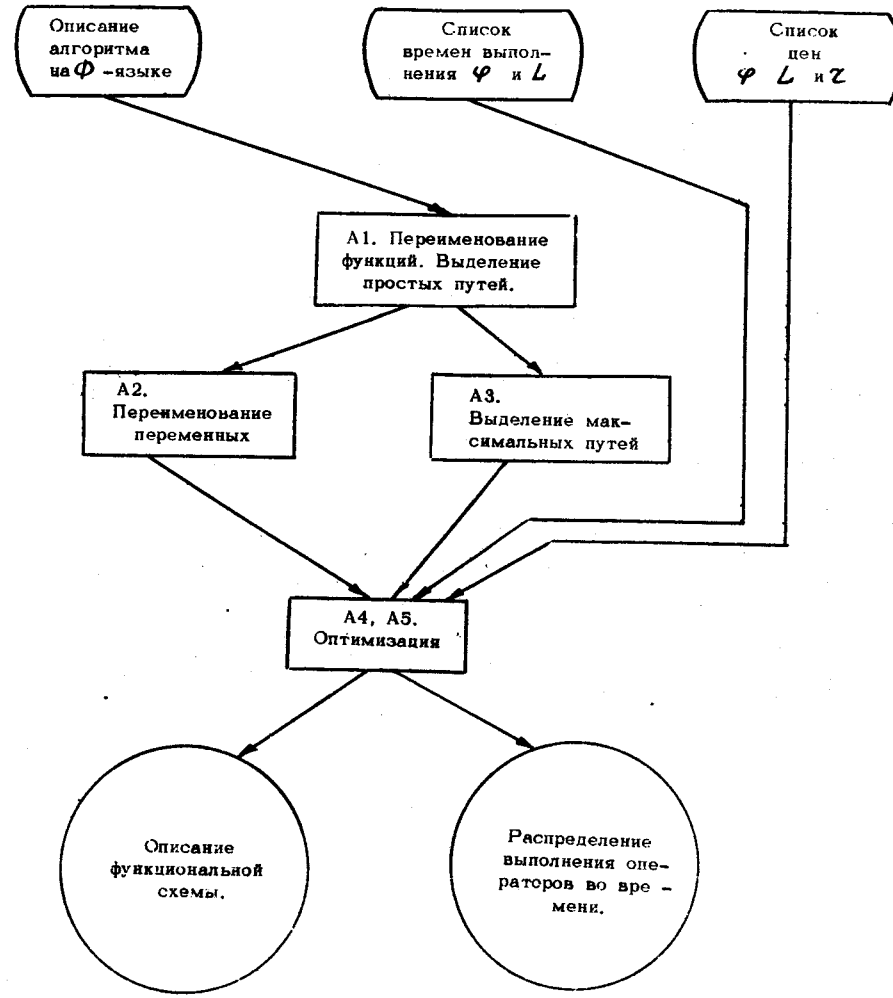


Рис.2. Блок-схема первого этапа.

заться неудовлетворительным по оборудованию. Возникает задача преобразования исходной записи с целью получения более оптимального проекта. Сущность этого преобразования заключается в совмещении некоторых функций одного типа в одну и некоторых переменных – в одну переменную. Составляя исходную запись алгоритма функционирования, человек сознательно или бессознательно производит также совмещения . Но это делается без учета всей информации о функционировании проектируемого устройства, поэтому результаты таких совмещений могут помешать нахождению оптимального варианта совмещений. Для того, чтобы результат оптимизации в этом смысле не зависел от человека, записывающего исходную информацию, необходимо первоначальную запись преобразовать так, чтобы каждая переменная и каждая функция получили свое собственное имя. В рассматриваемой системе сначала проводятся преобразования, затрагивающие только функции, а затем – переменные.

3. Первый этап

Исходной информацией для первого этапа (см.рис.2) является описание алгоритма функционирования проектируемого устройства на Ф-языке, список цен, характеризующих оборудование, необходимое для реализации функций φ , логических условий $\angle$ и переменных γ , которые используются в записи исходного алгоритма, и список времен выполнения функций φ и логических условий $\angle$. Выходной информацией первого этапа является описание функциональной схемы устройства на Ф-языке и информация о распределении выполнения операторов во времени.

Рассмотрим исходную запись некоторого алгоритма, выполненную на Ф-языке (см. рис. 3а). Сопоставим этой записи граф-схему (рис. 3б). Вершинам граф-схемы соответствуют операторы записи.(В дальнейшем мы будем отождествлять вершины граф-схемы с соответствующими операторами). Дуги граф-схемы определяются следующим образом. Дуга, исходящая из вершины оператора пересылки, входит в вершину того оператора, который следует в записи за данным оператором. Дуга, исходящая из вершины оператора безусловного перехода, входит в вершину того оператора, к которому происходит безусловный переход. Из вершины условного перехода исходит столько дуг, сколько логических условий содержит данный оператор. Эти дуги входят в вер-

Ⓘ $'a \overset{\varphi_0}{\Rightarrow} b$

② $\alpha 1:\ ('a,'b) \overset{\varphi_1}{\Rightarrow} c$

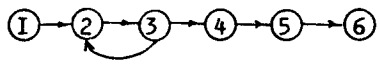
③ $P('a='c, 'a\neq'c)(\alpha 1,\alpha 2)$

④ $\alpha 2:\ ('p,'d) \overset{\varphi_2}{\Rightarrow} c$

⑤ $('a,'b) \overset{\varphi_3}{\Rightarrow} g$

⑥ $('g,'c) \overset{\varphi_4}{\Rightarrow} d$

3 а) Запись исходного алгоритма.



3 б) Граф-схема алгоритма.

I Ⓘ $a' \overset{\varphi_0}{\Rightarrow} b$

② $('a,'b) \overset{\varphi_1}{\Rightarrow} c$

③ $P('a='c, 'a\neq'c)(\alpha 1,\alpha 2)$

II ⊗ $('a,'b) \overset{\varphi_6}{\Rightarrow} c$ [⑦ $('a,'b) \overset{\varphi_5}{\Rightarrow} c$]

③ $P('a='c, 'a\neq'c)(\alpha 1,\alpha 2)$

III ④ $\alpha 2:\ ('p,'d) \overset{\varphi_2}{\Rightarrow} c$

⑤ $('a,'b) \overset{\varphi_3}{\Rightarrow} g$

⑥ $('g,'c) \overset{\varphi_4}{\Rightarrow} d$

3 в) Список простых путей (I,II и III).

Ⓘ $'a \overset{\varphi_0}{\Rightarrow} b$

② $('a,'b) \overset{\varphi_1}{\Rightarrow} c$

③ $\alpha 3:\ P('a='c, 'a\neq'c)(\alpha 1,\alpha 2)$

⑦ $\alpha 1:\ ('a,'b) \overset{\varphi_5}{\Rightarrow} c$

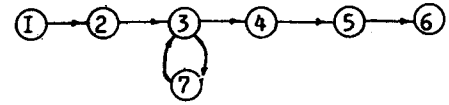
$\alpha 3:$

④ $\alpha 2:\ ('p,'d) \overset{\varphi_2}{\Rightarrow} c$

⑤ $('a,'b) \overset{\varphi_3}{\Rightarrow} g$

⑥ $('g,'c) \overset{\varphi_4}{\Rightarrow} d$

3 г) Новая запись алгоритма.



3 д) Новая граф-схема (вершина, соответствующая оператору безусловного перехода $\alpha 3:$, опущена).

Ⓘ $'a \overset{\varphi_0}{\Rightarrow} b$

② $('a,'b) \overset{\varphi_1}{\Rightarrow} c$

③ $\alpha 3:\ P('a='c, 'a\neq'c)(\alpha 1,\alpha 2)$

⑦ $\alpha 1:\ ('a,'b) \overset{\varphi_5}{\Rightarrow} c$

$\alpha 3:$

④ $\alpha 2:\ ('p,'d) \overset{\varphi_2}{\Rightarrow} e$

⑤ $('a,'b) \overset{\varphi_3}{\Rightarrow} g$

⑥ $('g,'e) \overset{\varphi_4}{\Rightarrow} f$

3 е) Запись алгоритма после переименования переменных.

I Ⓘ ② ③ ④ ⑤ ⑥

II Ⓘ ② ③ ⑦ ③ ④ ⑤ ⑥

3 ж) Максимальные пути (I и II).

Пример преобразования исходной записи.

вины тех операторов, к которым происходят переходы.

А л г о р и т м А1. На граф-схеме исходного алгоритма алгоритм А1 выделяет простые пути. П р о с т ы м п у т е м назовем путь, который начинается начальным оператором записи или оператором, непосредственно следующим за оператором условного перехода, и кончается оператором условного перехода или оператором конца. Если некоторый оператор записи встречается в нескольких путях, как это имеет место с оператором ② (рис.3а), то в каждом таком пути функция, входящая в этот оператор, получает новое имя : φ_1 в первом пути и φ_2 во втором (см. рис. 3в). В запись исходного алгоритма должным образом вводится новый оператор (⑦ на рис.3в,г). Таким образом устраняются те совмещения функций, которые сделал человек при записи исходного алгоритма. После работы алгоритма А1 граф-схема исходного алгоритма изменяется за счет введения новых операторов (рис. 3д).

А л г о р и т м А2. Работает на новой граф-схеме исходного алгоритма и осуществляет переименования переменных, в результате чего устраняются сделанные человеком совмещения (рис. 3е). Вариант алгоритма А2, работающего на операторных схемах, изложен в статье [6].

А л г о р и т м А3. Для работы алгоритмов оптимизации необходимо выделить на граф-схеме максимальные пути. М а к с и м а л ь н ы м п у т е м назовем такой путь, что в граф-схеме алгоритма не найдется ни одного другого пути, включающего данный. В граф-схеме алгоритма, приведенного в примере, имеется два максимальных пути (см.рис. 3ж). Максимальные пути выделяются алгоритмом А3, исходной информацией для которого является новая запись алгоритма, полученная после введения новых операторов при выделении простых путей (рис.3г).

Далее приступают к работе алгоритмы оптимизации А4 и А5. Алгоритм А4 работает на массиве простых путей и осуществляет временное сжатие. Алгоритм А5 работает на массиве максимальных путей и осуществляет переименование переменных и операторов с целью сокращения оборудования. Результатом работы этих алгоритмов является: 1) информация о распределении выполнения операторов во времени для каждого простого пути; 2) описание функциональной схемы проекта на Ф-языке, в котором учтены все совмещения функций и переменных, дающие оптимальное или близкое к оптимальному значение оборудования.

Алгоритм А4. Рассмотрим множество простых путей исходного алгоритма. Запись алгоритма на Ф- языке не дает представления о времени выполнения путей. Например, в третьем простом пути (рис. 3г.) операторы ④ и ⑤ могут выполняться одновременно, так как переменные и функция первого оператора отличаются от переменных и функции второго. Это означает, что пути, в которых встречаются подобные участки, можно сжимать во времени, т.е. объединять в один временной набор те операторы, которые можно выполнять одновременно. Каждому временному набору соответствует один временной такт при работе схемы. Те операторы, которые требуют для своего выполнения r тактов, должны входить в r следующих друг за другом наборов. Исходя из требований всей системы проектирования, при объединении операторов в наборы должно выполняться условие минимума времени выполнения каждого пути. После того, как определено конкретное объединение операторов во временные наборы во всех путях, можно переходить к этапу поиска такого переименования операторов и переменных, которое для данного варианта объединения операторов во временные наборы дает минимум оборудования.

Алгоритм А5. Рассмотрим множество максимальных путей граф-схемы исходного алгоритма. В каждом максимальном пути может быть заключено несколько простых путей. Перенумеруем единой нумерацией все переменные и операторы алгоритма и перенумеруем единой нумерацией все временные наборы простых путей, входящих в данный максимальный путь в порядке их следования в максимальном пути. Каждому K -му максимальному пути сопоставим матрицу занятости $\|A_K\|$, в которой $a_{ij} = 1$, если в i -м наборе данного максимального пути j -ый элемент (переменная или оператор) занят, $a_{ij} = 0$, если j -ый элемент или совсем не встречается в данном максимальном пути или не занят в i -м временном наборе. Оператор считается занятым в данном наборе, если он в него входит. Переменная считается занятой в данном наборе, если ее текущее значение используется в данном наборе или в следующих. Далее из матриц $\|A_K\|$ строится матрица $\|C\|$ следующим образом:

$$\|C\| = \left\| \begin{matrix} A_1 \\ A_2 \\ \cdots \\ A_m \end{matrix} \right\| ,$$

где m - количество максимальных путей в граф-схеме алгоритма. Матрица $\|C\|$ имеет столько же столбцов, сколько каждая матрица $\|A_K\|$. Число строк матрицы $\|C\|$ равно сумме числа строк всех матриц $\|A_K\|$.

Переименование операторов и переменных с целью уменьшения оборудования в схеме, которая синтезируется, эквивалентно замене нескольких столбцов в $\|C\|$ одним столбцом, равным поразрядной дизъюнкции исходных заменяемых столбцов. Естественно, таким образом можно совмещать только такие два столбца матрицы $\|C\|$, у которых поэлементная конъюнкция равна 0, потому что в этом случае соответствующие рассматриваемым столбцам элементы заняты в непересекающихся временных наборах и, следовательно, их можно объединить и назвать одним именем. Столбцы, соответствующие переменным, нельзя совмещать со столбцами, соответствующими операторам. Совмещение столбцов матрицы $\|C\|$ должно проводиться с учетом того оборудования, которое получается при реализации элементов в результате совмещения. Для этого необходимо ввести правила подсчета цены элементов, которая отражала бы оборудование, идущее на их реализацию. В каждом конкретном случае эти правила свои. Однако алгоритмы оптимизации от конкретных правил подсчета цены не зависят. Алгоритм поиска оптимальных совмещений столбцов матрицы $\|C\|$ сводится к задаче нахождения оптимальных покрытий [7].

4. Второй этап

На втором этапе из описания функциональной схемы получается подробное описание логической схемы устройства на языке булевых уравнений. (Отметим, что впервые алгоритмический переход от описания функциональной схемы на языке регистровых передач к системе логических уравнений был предложен в работе [8]).

Как уже было сказано, в описываемой системе список комбинационных схем, реализующих функции φ и условия L, должен задаваться в начале проектирования. Предполагается, что каждая из этих схем описана булевыми уравнениями, связывающими переменные, сопоставленные её входным и выходным шинам.

В результате работы алгоритмов первого этапа некоторые переменные (функции) получают одинаковые имена (символы). Кроме того, каждому оператору записи присваиваются метки, определяющие время и условия выполнения этого оператора.

Для того, чтобы реализовать схемы устройства в соответствии с полученным на первом этапе описанием, на втором этапе операторы объединяются в классы (возможно-пересекающиеся) по одинаковым именам переменных, входящих в их правые части, а также по одинаковым символам функций. Каждому такому классу сопоставляется ряд однотипных булевых уравнений (количество их определяется разрядностью переменной или функционального преобразователя).

Каждое булево уравнение имеет следующую форму :

$$z = \Phi(a_1,\ldots,a_i,\ldots,a_n,\tau_e),$$

где z - выход запоминающего элемента, соответствующего переменной в правой части оператора пересылки;

a_i - выход запоминающего элемента, соответствующего i-ой переменной в левой части оператора;

τ_e - логическая переменная, определяющая временной такт выполнения оператора.

Конкретный вид уравнения зависит от особенностей оборудования (комплекса элементов), на котором реализуется схема. В статье [5] приводились примеры построения булевых уравнений для комплекса потенциальных элементов "УРАЛ-10".

Набор уравнений для всех классов операторов функциональной схемы представляет собой полное описание логической схемы проектируемого устройства.

5. Третий этап

На третьем этапе описание логической схемы на языке булевых уравнений используется для получения описания принципиальной схемы.

При переходе от системы логических уравнений к принципиальной схеме в общем случае возникает задача комбинационного синтеза - представление логических функций системы в виде суперпозиции операторов, соответствующих выбранному комплексу элементов. В описываемой системе проектирования предполагается использовать некоторые из известных алгоритмов комбинационного синтеза.

Далее необходимо выполнить построение принципиальной схемы , т.е. установить, какими конструктивными единицами (из

заданного набора) будут реализоваться уравнения, описывающие логическую схему, и получить схему связей между этими конструктивными единицами.

В зависимости от конкретного исполнения комплекса элементов при построении принципиальной схемы на разных уровнях возникает следующая задача: из заданных принципиальных схем более мелких конструктивных единиц построить принципиальную схему более крупной конструктивной единицы (например, из нескольких модулей - ячейку, из нескольких ячеек - блок).Для решения этой задачи необходимо:

1. Иметь формальное описание рассматриваемых конструктивных единиц.

2. Установить правила построения сложной схемы из простых.

3. Иметь алгоритм, который, используя установленные правила и формальные описания простых схем, строит формальное описание сложной, а также соответствующую схему связей.

Для формального описания конструктивных единиц предлагается использовать специальные к а р т о т е к и .На рис. 4 приведен пример карточки , соответствующей некоторой конструктивной единице (схеме).

Правила построения сложных схем из простых на разных уровнях проектирования различны. Они зависят также от конкретного электрического и конструктивного исполнения.Иногда такие правила являются формальным описанием интуитивных приемов построения схем. В некоторых случаях эти правила имеют вид формул, связывающих переменные, сопоставленные простым схемам. Тогда алгоритм построения сложной схемы из простых заключается в следующем:

1. Рассматривается формула построения сложной схемы.

2. Для каждой переменной, входящей в эту формулу, выбирается соответствующая карточка из картотеки простых схем. В ней заполняется графа "имя переменной".

3. В полученном наборе карточек простых схем строки подвергаются сортировке по именам переменных. В результате все строки,помеченные одним и тем же именем переменной, объединяются в одну группу. Полученные группы "связанных"строк описывают соединения между входящими простыми схемами, а информация остальных строк используется для заполнения карточки сложной схемы.

Рассмотрим в качестве примера один из этапов построения принципиальной схемы по заданной системе логических уравнений.

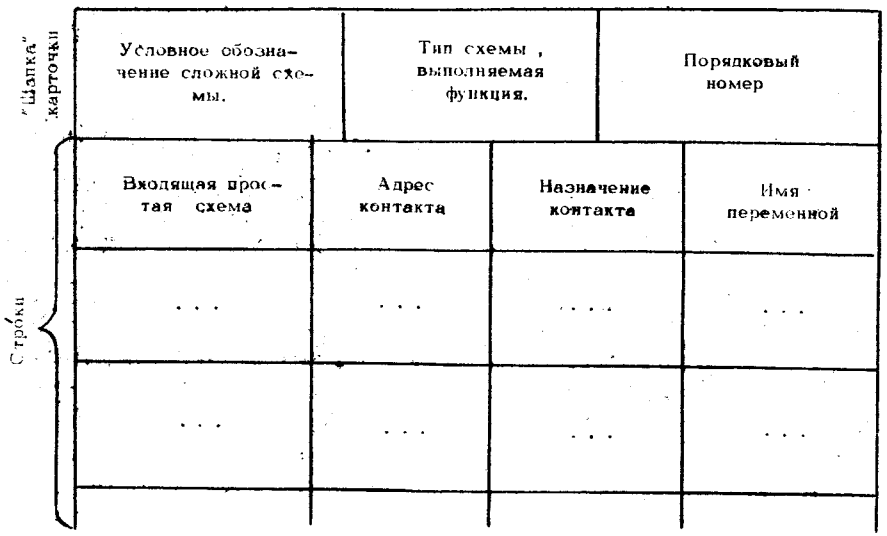


Рис.4. Карточка схемы.

Из практических соображений вся система логических уравнений делится на ограниченные с п и с к и (обычно список относится к некоторому функционально цельному блоку). Часть принципиальной схемы, соответствующую одному списку, назовем л и с т о м, а часть, соответствующую одному уравнению, - с е к т о р о м.

Процесс построения схемы листа из схем секторов подчиняется описанным выше общим правилам. Формулами построения сложной схемы в данном случае являются сами уравнения списка. Действительно, поскольку расположение имен логических переменных в уравнениях отражает все необходимые связи принципиальной схемы, то после заполнения этими именами карточек секторов и сортировки набора карточек вырабатывается описание схемы связей между секторами внутри листа и карточка листа. Одновременно подготовляется информация для описания схемы связей между листами.

6. Четвертый этап

На третьем этапе при построении принципиальных схем одновременно ведется учет нагрузок на отдельные элементы комплекса и в нужных местах ставятся усилители. Естественно, усилители вносят дополнительные задержки в схемы. Информация об этих задержках используется для корректирования задания на синтез генератора управления, которое выдается на втором этапе работы системы. На последнем, четвертом этапе, откорректированное задание на генератор используется для синтеза генератора управления. Синтез генератора управления проводится известными методами теории автоматов [9].

Каждому j -му простому пути граф-схемы алгоритма функционирования проектируемого устройства сопоставляется автомат A_j. Входным сигналом (параметром) этого автомата является переменная, соответствующая условию, по которому начинает выполняться j -ый путь. Входной сигнал переводит автомат A_j из начального состояния в первое. При этом на выходе A_j появляется сигнал, который управляет всеми схемами, реализующими операторы, стоящие в первом временном наборе j -го простого пути. Далее автомат A_j переходит во второе, третье и т.д. состояния, соответствующие второму, тре -

тьему и т.д. временным наборам j -го простого пути. Если j -ый путь имеет r временных наборов, то автомат A_j имеет $r+1$ состояние: начальное, первое, второе, ... r-ое. Из r -го состояния A_j переходит в начальное и остается в нем до тех пор, пока на него снова не воздействует его входной сигнал.

Все автоматы A_j объединяются в один автомат A путем объединения состояний автоматов A_j с одинаковыми номерами. Например, первым состояниям всех автоматов A_j соответствует первое состояние автомата A , вторым - второе и т.д. При этом, если объединяются такие состояния автоматов A_j, которые управляют одинаковыми схемами, то соответствующие выходы автоматов A_j сливаются в один выход автомата A . Если же объединяются состояния, которые управляют разными схемами, то на соответствующем выходе автомата A организуется распределительная схема, управляющими сигналами для которой служат параметры автоматов A_j .

Описанный процесс проектирования дает схему генератора управления, близкую к схемам, получаемым инженерными методами.

ЛИТЕРАТУРА

I. Ф. Реймон. Автоматика переработки информации. Пер. с фра М., Физматгиз, I96I.

2. G.P.Dinneen, I.L.Lebow, I.S.Reed. The logical design of CG- 24.-Proc. EJCC, Boston, Dec. I959, 9I-94.

3. T.C.Bartee, I.L.Lebow, I.S.Reed. Theory and design of digital machines. New York, McGraw-Hill, I962.

4. Е.Л. Ющенко. Адресное программирование. Киев, Гостехиздат УССР, I963.

5. И.В. Иловайский, В.С. Лозовский, Я.И. Фет. Применение адресного языка для автоматизации синтеза цифровых

вычислительных устройств.-Вычислительные системы , Новосибирск, I965, вып. I8, стр. 34-7I.

6. С.С. Лавров. Об экономии памяти в замкнутых операторных схемах. Журн.выч.матем.и матем.физ.,I96I, I, № 4, стр. 687-70I.

7. А.Д. Закревский. Оптимизация покрытий множеств.-Логический язык для представления алгоритмов синтеза релейных устройств. М., "Наука", I966 , стр. I36-I48.

8. H. Schorr. Computer aided digital system design and analysis using a register transfer language.- IEEE Trans. Electron. Comput., 1964, EC-13, N6, p.730-737.

9. С.И. Баранов, Р.С. Гольдман, О.Ф. Немолочнов. Система автоматизации проектирования логических схем ЦВМ.- Тезисы докладов к предстоящему Всесоюзному коллоквиуму по автоматизации синтеза дискретных вычислительных устройств, Новосибирск, I966, стр.84.

Институт математики
СО АН СССР

Поступила в редакцию
I.X. I966г.