1969 г. ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ Выпуск 34

УДК 681.142.001.2

ПОСТРОЕНИЕ ФУНКЦИОНАЛЬНОЙ СХЕМЫ ЦВМ ПО ОПИСАНИЮ ПРОЦЕССА ЕЕ РАБОТЫ

И.В. Иловайский

В работе рассматривается метод построения описания функциональной схемы ЦВМ по записи алгоритмического процесса работы ЦВМ.

Мы исходим из того положения, что ЦВМ представляется разным образом в зависимости от того, чем интересуются при ее описании - способом, которым выполняются преобразования, требуемые в командах, либо тем, из каких устройств (блоков) состоит ЦВМ, как они соединены и в какие моменты времени работают.

При проектировании ЦВМ исходными данными служит описание системы команд машины. Этап блочного (структурного) проектирования ЦВМ завершается, когда получена ее функциональная схема и временная диаграмма. На их основе определяются технические характеристики машины, в том числе и такие, как трудоемкость остающихся этапов проектирования.

В этой работе мы опишем ту часть этапа блочного (структурного) проектирования ЦВМ, которая связана с получением схемы ЦВМ (или ее блоков) по описанию процессов их работы.

В ряде работ рассматривается этап блочного синтеза ЦВМ: так, в [1],[2] по алгоритму выполнения системы команд строится (на примере) функциональная схема ЦВМ; в [3] описан общий подход к построению блок-схемы цифрового устройства (блок-схема в этой работе понимается как объединение достаточно крупных блоков ЦВМ [4]). Очевидно, что подход [3] может быть применен и к построению более подробных схем ЦВМ.

В [5] введены понятия "блок-схема" ЦВМ и "алгоритм работы" ЦВМ и даны правила перехода от одного объекта к другому, не зависящие от способа записи алгоритма работы и блок-схемы. В [6] описан путь перехода от алгоритма работы устройства к функциональной схеме, причем подробно рассматривается построение алгоритмического процесса работы ЦВМ, но в построении собственно функциональной схемы эта работа повторяет работу [2].

Цель настоящей статьи - дать метод построения функциональной схемы и временной диаграммы по алгоритмическому процессу работы ЦВМ, не зависящий от способа записи последнего, поскольку задача получения алгоритмического процесса работы (микропрограммного описания) ЦВМ по алгоритму работы ЦВМ уже описана [6],[9].

Для этого нам понадобится ввести определения понятий "алгоритмический процесс работы ЦВМ", "функциональная схема ЦВМ", "временная диаграмма".

Понятие алгоритмического процесса работы (микропрограммного описания) будет строиться на основе понятия "алгоритм работы устройства" (ЦВМ) [5]. Понятие функциональной схемы будет эксплицировано из инженерного представления об этом объекте, а понятие "временная диаграмма" будет близко к тем исходным объектам, которые используются для построения устройства управления в работе [8].

I. Пусть задан алгоритм работы ЦВМ [5]. Будем составлять микропрограммное описание (описание алгоритмического процесса работы) ЦВМ следующим образом. Введем такты времени.

Пусть для каждого такта составлен список, порядок элементов в котором несуществен, состоящий из тех и только тех вхождений операторов алгоритма работы ЦВМ, которые выполняются в этом такте. (Допускается, чтобы некоторые операторы выполнялись несколько тактов подряд).

Список операторов, выполняющихся в некотором такте, будем называть микрокомандой.

Порядок следования тактов во времени (для каждой реализации алгоритмического процесса) определяется порядком следования микрокоманд в ходе реализации алгоритмического процесса.

Если за микрокомандой μ_1 всегда следует микрокоманда μ_2, а за μ_2 также следует μ_3 и т.д. до μ_n, то последовательность $\mu_1,\dots,\mu_n$ назовем линейной микропрограммой.

Если после какой-либо микрокоманды μ_i может в различных реализациях алгоритмического процесса осуществиться одна из m микрокоманд $\mu_j,\dots,\mu_{j+m-1}$, то μ_i мы назовем концевой микрокомандой линейной микропрограммы, а $\mu_j,\dots,\mu_{j+m-1}$ - на-

чальными микрокомандами линейных микропрограмм.

Предполагается, что условный переход от μ_i к $\mu_j, \ldots, \mu_{j+m-1}$ осуществляется в силу того, что в μ_i есть точно один P -оператор такой, что при единичном значении τ -й метки из вектора его выходных переменных осуществляется передача управления к одному из операторов, находящихся в микрокоманде $\mu_{j+\tau}$. При этом, естественно, предполагается, что в микрокоманде μ_i выполнение P -оператора заканчивается, т.е. если P -оператор занял микрокоманды $\mu_{i-s}, \ldots, \mu_{i-1}, \mu_i$, то переход по нему происходит только после окончания микрокоманды μ_i. Аналогично передача управления идет на ту микрокоманду, где соответствующий оператор начинает осуществляться.

Итак, мы придем к такой записи алгоритма работы ЦВМ,где некоторые вхождения операторов объединены в списки.Очевидно, согласно [7] мы получим запись алгоритма, сильно эквивалентную исходной, а значит, задающую ту же блок-схему ЦВМ [5].

Поставим в соответствие каждой микрокоманде вершину графа,а дуги графа определим следующим образом. Если микрокоманда μ_j может осуществиться (условно или безусловно) за микрокомандой μ_i, то в графе проведем дугу $\mu_i \rightarrow \mu_j$. Каждой вершине поставим в соответствие τ - длительность выполнения соответствующей микрокоманды.

Полученный граф G назовем граф-схемой алгоритмического процесса работы ЦВМ по аналогии с граф-схемой алгоритма [7].

2. Функциональная схема понимается как графическое изображение устройства в виде запоминающих элементов (запоминающих устройств, регистров, триггеров), элементов преобразования информации и связей между ними, охватывающих все информационные и управляющие сигналы.

Функциональная схема состоит из элементов, вообще,- логических сетей, объединенных друг с другом так, что совокупная логическая сеть является правильной [4].

Если на один и тот же вход элемента схемы надо подать N сигналов, существующих неодновременно, каждая из N входных шин, несущих сигналы, снабжается клапаном, а далее ставится "собирательная схема", выполняющая дизъюнкцию всех N выходных сигналов клапанов. Будем считать, что "собирательная схема" и клапаны, находящиеся на входе элемента схемы, входят в него и являются управлением элемента, а собственно элемент назовем функциональным ядром.

Каждый элемент A_i (рис. I) обладает множеством M_i выходных

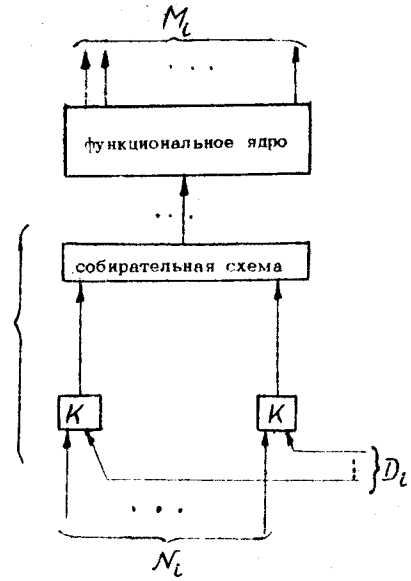

Рис. I

полюсов, множеством N_i входных полюсов, множеством D_i управляющих полюсов. Функциональная схема описывает машину с точностью до описания передачи одного разряда информационного слова между элементами.

Разрядам слов, выдаваемых и принимаемых элементами схемы,мы и ставим в соответствие входные и выходные полюса. Принимается нумерация полюсов по нумерации разрядов слов.

Поскольку между D_i и N_i существует (по построению) взаимно однозначное соответствие, всегда можно указать,какому $d \in D_i$ соответствует какой $n \in N_i$ и какой клапан соответствует этой паре (n, d).

Выделим множество L_i выходных полюсов собирательной схемы и множество Λ_i входных полюсов ядра. Будем говорить, что элемент простой, если между L_i и Λ_i существует взаимно однозначное соответствие. Это тот случай, когда на входы элемента может поступать не более одного информационного слова одновременно,тогда на входе ядра "собирательная схема" осуществляет поразрядную дизъюнкцию слов, поступающих на этот элемент в разные такты, и между каждым $J_j \subseteq N_i$ и Λ_i существует взаимно однозначное соответствие (J_j - группа полюсов, через которую j-е слово поступает на элемент).

Многие элементы машины имеют несколько "равноправных" входов (групп входов). Примером может служить сумматор на n слагаемых. Обычно такие схемы имеют итеративную структуру, и каждый компонент такой схемы воспринимает один разряд (не обязательно один и тот же) всех слагаемых. В этом случае вводим (непересекающиеся) множества входных полюсов $J_{1i}, \ldots, J_{ki}, \ldots, J_{ni}$ $(\bigcup_k J_{ki} = N_i)$. Каждое такое множество полюсов разбивается на множества J_j, служащие для подачи различных информационных слов на один вход элемента в разные такты времени. Каждой такой группе J_{ki} соответствует множество $W_{ki} \subseteq D_i$ управляющих полюсов. В свою очередь W_{ki} разбивается на непересекающиеся множества полюсов $V_j \subseteq W_{ki}$, каждое из которых однозначно соответствует $J_{jki} \subseteq J_{ki}$.

Очевидно, существует взаимно однозначное соответствие между Λ_i и $\bigcup_k L_{ki}$.

Полная схема элемента (элементарной логической сети) функциональной схемы приведена на рис. 2.

Из описанного с очевидностью следует запись оператора управления элементом:

$$\begin{cases} \ell_t = \bigcup_j i_{jt} \wedge \vartheta_{jt}\,; \forall \ell_t \in L_{1i}, \forall i_t \in J_j; \forall \vartheta_t \in V_j\,, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \ell_t = \bigcup_j i_{jt} \wedge \vartheta_{jt}; \forall \ell_t \in L_{Ni}\,; \forall i_t \in J_j; \forall \vartheta_t \in V_j\,. \end{cases}$$

Здесь отождествлены имена сигналов (разрядов информационных слов) и соответствующие им полюса.

Описание функциональной схемы есть описание связей ее элементов между собой. Мы будем говорить, что задано отношение связи R элемента i с прочими элементами функциональной схемы, если известно множество M_i выходных полюсов этого элемента, известно множество всех входных полюсов всех элементов (обозначим его через $NN = \bigcup N_i$) и задано множество $R \subseteq M_i \times NN$ ($M_i \times NN$ есть множество всех возможных связей между выходами элемента A_i функциональной схемы и входами всех других элементов).

Функциональная схема задана, если заданы

$$MM = \bigcup_i M_i\,,\ NN = \bigcup_i N_i\,,\ DD = \bigcup_i D_i\,,\ M_0 \subseteq MM$$

и заданы

$$RR = \bigcup_i R_i \subseteq (MM \times NN)\,,\ PP = \bigcup_i P_i \subseteq M_0 \times (DD)$$

($MM \times NN$ есть множество всех возможных связей между элементами функциональной схемы, за исключением связи управления). Заметим, что вход устройства управления описывается как входы

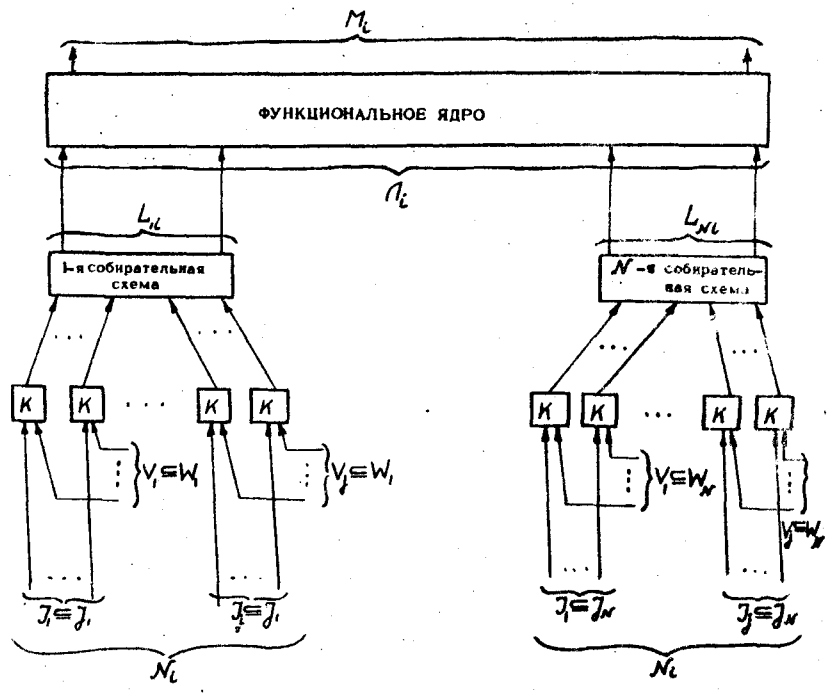

Рис. 2.

прочих элементов, и связи прочих элементов со входами A_0 даны в RR. Связи же выходов устройства управления с элементами функциональной схемы выделены особо. Это – множество PP. В нем указано, какие выходные полюса устройства управления из M_0 соединены с какими управляющими полюсами каких элементов. Устройство управления обозначим A_0.

3. Поставим в соответствие каждой переменной, входящей в запись алгоритмического процесса (или в запись алгоритма, что все равно), элемент функциональной схемы, ядро которого есть регистр (или шины). Разрядность ядра равна максимальной разрядности этой переменной из всех, употребленных в микропрограммном описании, причем M_i взаимно однозначно соответствует Λ_i. Соответственно "разрядность" на выходе, т.е. количество элементов множества M_i, и "разрядность" на входах, т.е. количество

элементов в каждом из множеств J_j, равны максимальной разрядности этой переменной. Нумерация полюсов в каждом множестве J_j и M_i начинается с единицы. Очевидно, что регистр (шины) – всегда простой элемент.

Поставим в соответствие каждой функции оператора [9], входящего в запись алгоритмического процесса, элемент функциональной схемы, ядро которого есть функциональный преобразователь (возможно, с памятью), выполняющий заданное этой функцией преобразование.

(Заметим, что если некоторый оператор должен дать один раз сумматор, а другой раз вычитатель и мы хотим эти схемы совместить, то мы должны в записи назвать функцию оператора одним именем и строить уже обобщенную схему, предусмотрев входную переменную, изменение значения которой "перестраивает" эту схему).

Разрядность на выходе подобного элемента определяется следующим образом. Отметим все те переменные, которые присутствовали хотя бы однажды в векторе выходных переменных операторов с данной функцией [5]. Сравним их максимальные разрядности. Возьмем наибольшую и будем считать, что она определяет количество выходных полюсов M_i. Нумерацию полюсов будем вести с единицы.

Для определения входной разрядности элемента введем ряд предварительных понятий.

Рассмотрим отмеченную граф-схему алгоритма работы проектируемого устройства [5]. Заменим у всех вершин граф-схемы векторы входных переменных на матрицы входных переменных, столбцы которых соответствуют входным переменным, а номер строки указывает позицию переменной в списке аргументов данного оператора.

Действительно, если оператор определен над значениями нескольких переменных, их порядок в записи (а значит, и способ обработки этим оператором) существен.

Кроме того, если мы хотим указать, что в i-м операторе надо обработать переменную a_t так же, как в j-м операторе переменную a_3, мы должны как-то упорядочить все входные переменные при данном операторе.

Фактически строки матрицы есть векторы входных переменных соответствующих позиций в списке аргументов. Очевидно, в каждой строке такой матрицы (и в столбце) может быть не более одного непустого элемента.

Если функции оператора соответствует элемент с одной группой входных полюсов $J=N_i$ (т.е. список аргументов одноэлементный), то разрядность по входу (количество полюсов) каждой из групп $J_j \in N_i$ будет равной и определится по максимальной разрядности данной переменной (т.е., как и в случае выходных переменных).

В случае нескольких групп J_k поступим аналогично для каждой группы (для каждой переменной из списка аргументов), поставив в соответствие k-му множеству входных полюсов k-ю переменную из списка аргументов.

Связи элементов логической сети (функциональной схемы) вводятся следующим образом. Обозревается j-й столбец матриц входных переменных всех операторов. От выходных полюсов элемента, поставленного в соответствие переменной a_j, будут проводиться связи к входным полюсам элементов, поставленных в соответствие тем функциям операторов, у которых j-й столбец матрицы входных переменных непуст. Устанавливается, какая строка содержит непустой элемент j-го столбца матрицы. Связи выходных полюсов M_j элемента a_j организуются с такими группами входных полюсов J_{ki} соответствующих функций операторов, что k есть номера выделенных строк.

Так поступаем для всех входных переменных a.

Далее находятся все вхождения операторов с функцией Q_i в запись алгоритма [5]. Обозреваются их выходные векторы. Находятся все выходные переменные b, связанные с этими операторами. Строятся связи полюсов из M_i с полюсами из N_j.

Так поступаем для всех функций операторов Q.

Если Q_i есть функция P-оператора, то считаем, что выходные полюса соответствующего ему элемента должны быть связаны с входными полюсами элемента A_0.

Тем самым мы получаем множество информационных каналов RR. При построении связей между полюсами из некоторого множества M_i и полюсами из некоторого множества N_i (или J_{ki}) действует правило, по которому каждое R составляется из пар полюсов с одинаковыми индексами.

Заметим, что при построении мы нигде не использовали того факта, что исходным является микропрограммное описание. Мы исходили из отмеченной граф-схемы алгоритма [5], заменив (из-за введения разрядности) векторы входных переменных на матрицы. Однако мы еще не достроили функциональную схему, а лишь получили "поразрядную блок-схему". Для построения связей PP нам из-

обходимы данные об истинном порядке выполнения операторов в алгоритмическом процессе.

Поскольку между N и D - множествами входных и управляющих полюсов существует взаимно однозначное соответствие, то мы знаем, какие полюса $d \in D$ должны быть возбуждены, в какой микрокоманде. Действительно, по построению мы знаем, какой паре "переменная - функция оператора" ("функция оператора - переменная"), из какой микрокоманды соответствует та или иная группа связей R, но эта группа связей взаимно однозначно связана с группой входных полюсов N, а значит, и управляющих полюсов D.

Будем считать, что все управляющие полюса $d \in D_{ij}$, соответствующие одной группе входных полюсов $J_j \subseteq J_{ki} \subseteq N_i$, связаны с одним и тем же полюсом $m_i \in M_0$.

Тогда количество элементов в M_0 равно сумме количеств групп $J_j \subseteq J_{ki} \subseteq N_i$ всех элементов функциональной схемы. Тем самым функциональная схема построена полностью.

В том случае, если запись алгоритмического процесса содержит блоки, построение ведется аналогично тому, как описано в [5], т.е. строится функциональная схема с элементами, соотнесенными этим блокам, а затем из схемы изымается элемент, соответствующий раскрываемой функции оператора (блока), и заменяется функциональной схемой этого блока. Как и в [5], связь общей схемы и схемы элемента осуществляется через входные (выходные) переменные блока (оператора).

Видно, что подобный подход позволяет, имея описания элементов функциональной схемы через логические операторы, перейти к логической схеме ЦВМ, т.е. к схеме, выраженной в логических операторах (заметим, что для описанного нами "раскрытия" блока не требуется, чтоб его алгоритм (алгоритмический процесс) был задан на том же языке, что и алгоритм (процесс) работы схемы в целом).

4. Поставим в соответствие каждой микрокоманде такой вектор-слово β_j на выходе элемента A_0, у которого равны единице все те разряды, которые соответствуют полюсам из M_0, питающим группы D_i управляющих полюсов, которые соответствуют парам (переменная - функция оператора, функция оператора - переменная), участвующим в данной микрокоманде. Тем самым мы зададим распределение открытых и закрытых клапанов в схеме.

Действительно, если известно, что в микрокоманде μ_i информация, содержащаяся в переменной a_i, должна быть переработана оператором с функцией Q_j, то соответствующая связь $R_t \subseteq$

$\subseteq M_i \times N_j$ должна осуществиться, а это значит, что клапаны, питаемые от полюсов n_t, входящих в связи из множества связей R_t, должны быть открыты и, следовательно, на полюсах $m \in M_0$, соответствующих тем $d \in D_j$, которые управляют этими клапанами, должны появиться сигналы разрешения передачи (для удобства - "единичные").

Тогда смена векторов β_i и определит смену микрокоманд, а длительность существования данного вектора β_i задает длительность выполнения микрокоманды. В свою очередь, некоторые микрокоманды (концевые микрокоманды линейных микропрограмм) вызывают смену векторов β_i, осуществляя передачу сигналов (соответствующих "меткам" P - операторов) на вход элемента A_0.

Граф G, описывающий алгоритмический процесс, нагрузим по вершинам векторами β_i, а по дугам-условиями смены микрокоманд α_j. Если микрокоманда μ_j безусловно следует за μ_i, дуга не нагружается, если $\mu_j, \ldots, \mu_{j+m-1}$ следуют за μ_i по некоторым условиям, то дугам ставятся в соответствие имена меток, вызывающих эти переходы. Далее, удаляем микрокоманды, которыми были отмечены вершины графа G.

Получаем граф G^*, вершинам которого соотнесены векторы β_i и времена их существования τ_i, а дугам - условия передачи управления (или имена сигналов передачи управления - что все равно).

Назовем G^* временной диаграммой. Действительно, развертки линейных участков графа (от одной вершины с полустепенью исхода >1 до другой) дают то, что принято у инженеров называть участком временной диаграммы.

Граф G^* может служить исходной информацией для построения известными методами [8] схемы элемента A_0, т.е. устройства управления.

5. Из построения видно, что множество операторов можно восстановить по описанию связей функциональной схемы. Порядок выполнения операторов также можно восстановить, если известен граф G^*, ибо каждому вектору β_i соответствует свое множество возбужденных групп управляющих полюсов, взаимно однозначно связанных с теми или иными вхождениями операторов в запись алгоритмического процесса. Обратная процедура может быть полезна, если исходной информацией для системы автоматического проектирования служит функциональная схема и временная диаграмма. Тогда восстановленная запись алгоритмического процесса служит целям моделирования проектируемого устройства.

6. Опишем в основных чертах ход алгоритма получения функциональной схемы и описания работы устройства управления.

Пусть исходная информация задана в виде:

А. Линейной записи алгоритма функционирования и списков переменных (с указанием максимальной их разрядности) и функций операторов, упомянутых в этой записи.

Б. Списков вхождений операторов по микрокомандам. Элементами списков служат номера вхождений операторов в запись п. А.

В. Линейной записи матрицы смежности граф-схемы алгоритмического процесса в виде списков непустых элементов строк, начинающихся именами этих строк. Элементами матрицы служат номера (имена) микрокоманд с указанием длительности их выполнения.

Примем, что результирующая информация - это:

Г. Запись отношения $RR\,UPP$ всех связей функциональной схемы, данная в виде последовательности списков, где каждый список начинается указанием полюса $m \in MM$ и элемента, которому принадлежит этот полюс, и содержит все те полюса $n \in NN$, с которыми полюс m связан (то же для M_0 и DD). Списки объединены в группы, такие, что их начала $m \in M_i$ (т.е. принадлежат какому-либо одному элементу функциональной схемы).

Д. Граф G^*, заданный матрицей смежности, строки которой соответствуют именам тех сигналов (меток), которые поступают на вход устройства управления A_0, а столбцы - полюсам $m \in M_0$, с указанием времени возбуждения сигналов на них.

Матрица записана последовательностью списков, каждый из которых начинается именем сигнала, возбуждающего A_0, и содержит элементами те $m \in M_0$, которые надо возбудить в данном случае.

Е. Массив списков - сопроводителей длительности сигналов на выходах A_0, построенный синхронно с массивом из п. Д.

Отношение п. Г строится следующим образом:

а) Для каждого разряда t каждой переменной a_i из записи алгоритма определяется, в каких микрокомандах μ_3, с какими функциями операторов Q_r и каким номером j в списке аргументов он употреблен.

Строится список $R \subseteq RR$, начинающийся именем этой переменной с указанием соответствующего выходного полюса $m_t \in M_i$. В него выписываются имена соответствующих функций операторов Q_r с указанием входного полюса $n_t \in J_3 \subseteq J_j \subseteq N_r$ и соответствующего управляющего полюса $d \in D_r$.

Одновременно строится массив G^*, синхронный с записью матрицы смежности граф-схемы алгоритмического процесса (п.В) так,

что если элемент списка R имеет полюс $d \in D_r$, то в элемент массива G^* синхронно с именем микрокоманды μ_3, которая в данный момент рассматривалась, заносится имя d этого управляющего полюса.

б) Определяется разрядность (по выходу) всех элементов схемы, соотносимых функциям операторов. Это делается просмотром векторов всех выходных переменных (см. стр. 44).

в) Для каждого разряда t каждой функции оператора Q_j из записи алгоритма определяется, в каких микрокомандах μ_3 и с какими выходными переменными b_f он употреблен.

Строится список $R \subseteq RR$, начинающийся именем этой функции оператора с указаниями соответствующего выходного полюса $m_t \in M_j$. В него выписываются имена соответствующих выходных переменных b_f с указанием входного полюса $n_t \in J_3 \subseteq J_j \subseteq N_f$ и соответствующего управляющего полюса $d \in D_f$ для каждой переменной.

Одновременно строится массив G^* методом, описанным в п. а) Если строится R такое, что именем этого отношения служит функция P -оператора, то, кроме имени d, в соответствующий элемент массива G^* заносится имя выходной переменной P -оператора, которая в данном случае должна принять единичное значение. Поскольку меток (выходных переменных) у P -оператора несколько, то и заполняется несколько элементов массива G^*.

В массиве G^* выделим списки, началами которых будут элементы, содержащие имена выходных переменных P -операторов. Эти имена вынесем из списков и объявим именами списков (началами). В каждый элемент списка (кроме имени списка) из G^* введем длительность τ из синхронного G^* массива п. В.

Всем тем $d \in DD$, которые попали в один и тот же элемент $g_t \in G^*$, поставим в соответствие полюс $m_t \in M_0$ элемента A_0 и составим PP в виде списков, начинающихся именем A_0 с указанием соответствующего полюса $m_t \in M_0$ и содержащих все $d \in DD$ связанные с ним.

Заменим в массиве G^* каждый элемент G^* на соответствующий ему $m_t \in M_0$. Мы получим граф G^*, описывающий закон функционирования устройства управления.

Литература

I. Ф.РЕЙМОН. Автоматика переработки информации. М., Физматгиз, 1961.

2. И.В. ИЛОВАЙСКИЙ, В.Ф. ЛОЗОВСКИЙ, Я.И. ФЕТ. Применение адресного языка для автоматизации синтеза цифровых вычислительных устройств. – Вычислительные системы, Новосибирск, 1965, вып. 18, стр. 34-71.

3. В.М. ГЛУШКОВ, Ю.В. КАПИТОНОВА, А.А. ЛЕТИЧЕВСКИЙ. Об автоматизации проектирования вычислительных машин. – Кибернетика, Киев, 1967, 5, стр. 2 – 14.

4. В.М. ГЛУШКОВ. Синтез цифровых автоматов. М., Физматгиз, 1962.

5. И.В. ИЛОВАЙСКИЙ, Э.Е. СЕРГЕЕВА. Построение блок-схемы ЦВМ по алгоритму ее функционирования. – Вычислительные системы, Новосибирск, 1968, вып. 31, стр. 5-23.

6. И.В. ИЛОВАЙСКИЙ, Б.А. СИДРИСТЫЙ, Я.И. ФЕТ. Алгоритмический синтез дискретных вычислительных устройств. – Вычислительные системы, Новосибирск, 1966, вып. 25, стр. 3-17.

7. Л.А. КАЛУЖНИН. Об алгоритмизации математических задач. – Проблемы кибернетики, 2, 1959, стр. 51 – 67.

8. С.И. БАРАНОВ, Л.Н. ШУРАВИНА. Автоматизация синтеза дискретного устройства по линейной микропрограмме. – Тезисы докладов к Всесоюзному коллоквиуму по автоматизации синтеза дискретных вычислительных устройств. Новосибирск, 1966.

9. Б.А. СИДРИСТЫЙ. Эквивалентный переход от описания на Ф-языке к микропрограммному описанию. – Настоящий сборник, стр. 19-37.

Поступила в редакцию
5.У.1968 г.