УДК 681.142.1.01

ЭКВИВАЛЕНТНЫЙ ПЕРЕХОД ОТ ОПИСАНИЯ ЦВМ НА Ф-ЯЗЫКЕ К МИКРОПРОГРАММНОМУ ОПИСАНИЮ

Б.А. Сидристый

В последнее время делаются попытки создания системы программ, предназначенных для синтеза ЦВМ с помощью ЦВМ на самых ранних этапах процесса проектирования, начиная с описания их функционирования на алгоритмических языках [1,3,4,6]. При этом широко используется принцип микропрограммного управления работой вычислительных устройств [4], который позволяет достаточно четко ставить задачи автоматического синтеза ЦВМ.

Схема вычислительной машины может быть разбита на две части: а) вычислительное устройство, предназначенное для переработки информации, поступающей на ее входные каналы, и для выдачи результатов на выходные каналы; и б) внешние устройства (связанные с вычислительным устройством информационными и управляющими каналами), как-то: оперативная память машины, памяти на барабанах, лентах и т.п., устройства ввода-вывода и т.д. В данной работе мы будем интересоваться функционированием первой части вычислительной машины. Вычислительное устройство, в свою очередь, может быть разбито на несколько узлов, связанных друг с другом и с внешними устройствами каналами связи. Каждый узел представляется блок-схемой, изображенной на рис. 1. Управление работой операционной части узла осуществляется генератором управления по микропрограммному принципу [4]. В частном случае, когда операционная часть представляет собой комбинационную схему, генератор управления может отсутствовать.

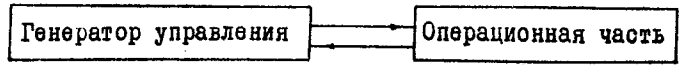

Рис. 1.

Описание работы дискретных вычислительных устройств на алгоритмических языках широко применяется для их моделирования и автоматического проектирования [1,2,3,5]. Это возможно потому, что между элементами схемы устройства и элементами алгоритма его функционирования наблюдается соответствие. Если мы говорим о схеме, то употребляем понятие регистра или шины, понятие внешних каналов для связи данного устройства с другими устройствами, понятие функциональных преобразователей, служащих для преобразования информации во время работы устройства. Если мы говорим об алгоритме функционирования, то мы соответственно употребляем понятие переменной алгоритма, понятие внешней переменной алгоритма, понятие функции.

Известно [1], что процесс автоматического проектирования ЦВМ разбивается на ряд этапов, на которых описание работы вычислительного устройства преобразуется с одного языка на другой, в результате чего все более детализируется это описание. Одним из таких этапов является этап преобразования описания алгоритма функционирования устройства, представленного на алгоритмическом языке, в микропрограммное описание. Под микропрограммным описанием здесь, грубо говоря, понимается информация следующего вида:

а) деление множества операторов исходного алгоритма на подмножества, называемые микропрограммами;

б) информация о соответствии элементам алгоритма (переменным и функциям) конкретных схем, выполняющих функции этих элементов;

в) распределение выполнения операторов алгоритма во времени.

В данной работе приводятся определения некоторых бинарных отношений в множестве операторов алгоритма функционирования, заданного на Ф-языке [7]. На основе этих отношений определяется другое средство описания алгоритма, называемое R-языком. R-язык нам представляется удобным языком для различных преобразований исходного алгоритма с целью оптимизации проекта по времени и оборудованию. Поэтому в этап получения микропрограммного описания можно включить преобразование исходного описания алгоритма функционирования на Ф-языке в описание на R-языке и все преобразования исходного алгоритма проводить в R-языке. Кроме того, в работе даются строгое определение микропрограммного описания узла вычислительного устройства и постановка задачи его получения.

Описание функционирования вычислительного устройства машины можно представить в виде системы алгоритмов функционирования его узлов, а задачу его проектирования как ряд самостоятельных задач проектирования узлов. Алгоритм функционирования узла вычислительного устройства будем рассматривать разбитым на блоки, которые определяются следующим образом.

Блоком алгоритма может быть совокупность $M_i \subset M$ операторов из множества операторов M алгоритма, одновременно удовлетворяющая условиям I.1 – I.6:

I.1. $[(a \in M_i) \wedge (b \in M_i)] \rightarrow \neg \exists c \{[[(a \beta c) \wedge (c \beta b)] \vee [(b \beta c) \wedge (c \beta a)]] \wedge \neg (c \in M_i)\}$

(определение отношений β и β^j, которые будут использоваться в дальнейшем, см. в [7]).

I.2. Передача управления на выполнение блока в алгоритме должна происходить только к одному оператору блока, называемому входным, этому условию подчиняется и передача управления по внешней метке, следовательно операторы, помеченные внешними метками, должны быть входными операторами каких-либо блоков алгоритма; таким образом, выполняется $\neg \exists a [(a \in M_i) \wedge \neg (b \beta a)]$ где b – входной оператор блока.

I.3. Выход из блока может происходить только от одного оператора, называемого выходным, которым, в частности, может быть оператор перехода и выполнение которого обеспечивает передачу управления на выполнение того или иного блока.

I.4. Внутри блока не должно быть циклов, хотя сам блок в целом может представлять собою цикл.

I.5. Два различных блока алгоритма имеют различные входные операторы, причем, если оператор является входным оператором одного блока, то он не принадлежит никакому другому блоку алгоритма.

I.6. Если b – выходной оператор какого-либо блока, то оператор a, для которого выполняется $b \beta^j a$, является входным оператором какого-либо блока. Первый оператор записи алгоритма функционирования также должен быть входным оператором какого-либо блока.

В множестве $\mathfrak{M}$ блоков алгоритма определяются отношения $\mathfrak{B}^j$. Для блоков A и B алгоритма выполняется $A \mathfrak{B}^j B$, если для выходного оператора a блока A и входного оператора b блока B выполняется соотношение:

2.1. $a \beta^j b$.

Систему $\{\mathfrak{M}, \mathfrak{B}^0, \mathfrak{B}^1, \ldots\}$ назовем блок-схемой алгоритма.

Пусть M_i – множество всех операторов i-го блока исходного

алгоритма функционирования. При этом будем считать, что в M_i входят операторы q_i и s_i – соответственно начала и конца блока. Если множеству M_i принадлежит оператор СТОП алгоритма, то он и выполняет роль s_i. В противном случае s_i – это фиктивный оператор, не требующий для своего выполнения времени. Входными переменными s_i являются выходные переменные блоки. q_i – тоже фиктивный оператор, выходными переменными которого являются входные переменные блока. Определим на множестве M_i следующие бинарные отношения.

О т н о ш е н и я $\beta^j(i)$. Для двух операторов $a, b \in M_i$ выполняется $a\beta^j(i)b (j = 0, 1, \dots)$, если выполняется 3.1, и, кроме того, выполняется $a\beta^0(i)b$, если имеет место одно из условий 3.2 – 3.3.

3.1 $a\beta^j b$;

3.2 a есть q_i, а b есть входной оператор блока;

3.3 a – выходной оператор блока, а b есть s_i.

На рис. 2 изображен граф, иллюстрирующий отношения $\beta^j(i)$, определенные на множестве операторов блока некоторого алгоритма. Запись этого блока на Ф-языке приведена на рис. 3.

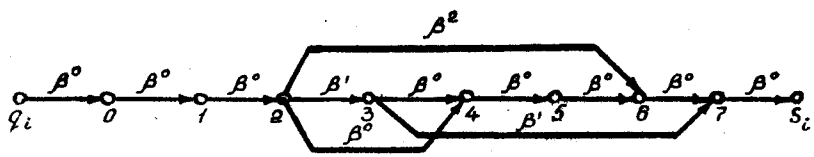

Рис. 2.

№ операторов	Операторы
0	ФI [АI(0 – I), Б(0 – I)] —→ С(0 – I);
I	Ф2-I [С(0 – I), Б(0 – I)] —→ Д5(0 – 2);
2	Если Л5 [Д5(0 – 2)] то 4 или 3 или 6;
3:	Если Н-6 [ЛI(0 – 3), Л2(4 – 7)] то 4 или 7;
4:	ФI-I [С(0 – I), Д(I – I)] —→ К(2);
5	Ф3 [Л5(I)] —→ Д7(2);
6:	К4 [Д5(0 – 4), Д6(0 – 4)] —→ ЛI(0 – 3);
7:	Ж [Д5(0 – 4)] —→ РI(0 – 4);

Рис 3.

Вершинам этого графа соответствуют операторы блока, номера которых записаны около вершин, а дуги определяются так: от вершины n_a проведена дуга к вершине n_b с надписью β^j, если выполняется $a\beta^j(i)b$ (n_a и n_b – номера операторов a и b, соответственно).

3.4. Выполняется $a\beta_1(i)b$ для $a, b \in M_i$, если выполняется $\exists j(a\beta^j(i)b)$.

3.5 Отношение $\beta(i)$ следования операторов в записи алгоритма. Выполняется $a\beta(i)b$ для операторов $a, b \in M_i$, если выполняется:

3.6 $(a\beta_1(i)b) \vee \exists x_1, \dots, x_k \left[(a\beta_1(i)x_1) \wedge (x_1\beta_1(i)x_2) \wedge \dots \wedge (x_{k-1}\beta_1(i)x_k)\right]$.

О т н о ш е н и е $\gamma(i)$ з а в и с и м о с т и в ы п о л н е н и я о п е р а т о р о в о т о п е р а т о р о в п е р е х о д а. Выполнение оператора b считается зависимым от оператора перехода a, если при выполнении алгоритма оператор b выполняется или нет в зависимости от реализации перехода по той или иной метке оператора a. Для операторов $a, b \in M_i$ где a – оператор перехода, выполняется $a\gamma(i)b$, если выполняется:

4.1 $(a\beta(i)b) \wedge \exists x_1, \dots, x_k \left[(a\beta_1(i)x_1) \wedge (x_1\beta_1(i)x_2) \wedge \dots \wedge (x_k\beta_1(i)s_i)\right]$

где $\{x_1, \dots, x_k\}$ – множество операторов из M_i, в которое не входит оператор b. Граф, иллюстрирующий отношение $\gamma(i)$ изображен на рис. 4.

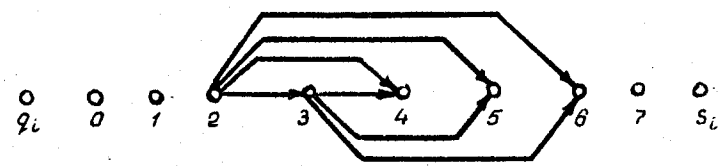

Рис. 4.

О т н о ш е н и я $\gamma^j(i)$ – н е п о с р е д с т в е н н о й з а в и с и м о с т и в ы п о л н е н и я о п е р а т о р о в о т о п е р а т о р о в у с л о в н о г о п е р е х о д а п о j-й м е т к е. Для операторов $a, b \in M_i$ выполняется $a\gamma^j(i)b$, если выполняется

5.1 $(a\gamma(i)b) \wedge \exists d \left\{\left[(a\beta^j(i)d) \wedge (d\beta(i)b)\right] \wedge \neg\exists c \left[(d\beta(i)c) \wedge (\gamma(i)b)\right]\right\}$.

Граф, иллюстрирующий отношения $\gamma^j(i)$, изображен на рис. 5.

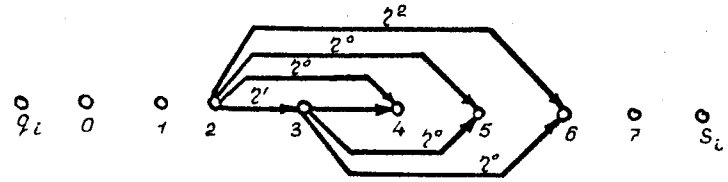

Рис. 5.

Для $a, b \in M_i$ выполняется $a\,\gamma_1(i)\,b$, если имеет место

5.2 $\exists j (a\,\gamma^j(i)\,b)$.

Отношение $\varepsilon_\rho(i)$ информационной зависимости по переменной ρ. Для операторов $a, b \in M_i$ выполняется $a\,\varepsilon_\rho(i)\,b$, если a – оператор пересылки с выходной переменной ρ, b – оператор с входной переменной ρ, и при выполнении алгоритма b использует результат a, то есть выполняется:

6.1 $(a\,\beta_1(i)\,b) \vee \exists x_1,\dots,x_\kappa \left[(a\,\beta_1(i)\,x_1) \wedge (x_1\,\beta_1(i)\,x_2) \wedge \dots \wedge (x_\kappa\,\beta_1(i)\,b)\right]$,

где $x_1, x_2, \dots, x_\kappa$ – операторы, не имеющие в качестве выходной переменной переменную ρ.

Отношение $\varepsilon_1(i)$ информационной зависимости. Для $a, b \in M_i$ выполняется $a\,\varepsilon_1(i)\,b$, если выполняется:

7.1. $\exists_\rho (a\,\varepsilon_\rho(i)\,b)$.

Отношение $\gamma(i)$ – соответствия операторам условных переменных. Каждому оператору перехода i-го блока поставим в соответствие условную переменную с разрядностью, равной числу меток оператора. Рассмотрим множество P_i всех разрядов всех условных переменных блока вместе с особой одноразрядной переменной ρ_i – сигналом запуска i-го блока. Пусть $a \in M_i$ и $\rho \in P_i$, тогда имеет место $\rho\,\gamma(i)\,a$, если:

8.1 выполняется $b\,\gamma^j(i)\,a$ и ρ есть $\rho_{b,j}$ – j-й разряд условной переменной, соответствующей оператору перехода b, или

8.2 a есть s_i или выполняется $\neg\exists c \left[(c\,\gamma(i)\,a)\right]$ и ρ есть ρ_i – сигнал запуска i-го блока. На рис. 6 изображен граф, иллюстрирующий отношение $\rho_i\,\gamma(i)\,a$ для примера блока из рис. 2.

Отношение $\alpha_1(i)$ – непосредственной зависимости операторов. Для $a, b \in M_i$ выполняется $a\,\alpha_1(i)\,b$, если выполняется хотя бы одно из условий:

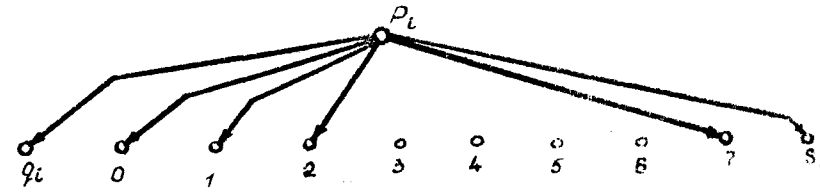

Рис. 6.

9.1 выполняется $a\,\varepsilon_1(i)\,b$,

9.2 выполняется $a\,\gamma_1(i)\,b$,

9.3 b – оператор s_i;

9.4 оператор a имеет хотя бы одну такую входную или выходную переменную, которая является выходной переменной b, и выполняется $a\,\beta(i)\,b$.

Отношение $\sigma(i)$. Для $a, b \in M_i$ выполняется $a\,\sigma(i)\,b$, если выполняется одно из следующих условий:

10.1 b – оператор условного перехода с непрерывной проверкой и выполняется $a\,\beta(i)\,b$;

10.2 b – оператор конца непрерывной проверки и выполняется $a\,\beta(i)\,b$;

10.3 a – оператор конца непрерывной проверки и выполняется $a\,\beta(i)\,b$;

10.4 b – оператор, помеченный внешней меткой, и выполняется $a\,\beta(i)\,b$;

10.5 a – оператор, помеченный внешней м[illegible] и выполняется $a\,\beta(i)\,b$.

Отношение $\alpha(i)$ – зависимости операторов. Для $a, b \in M_i$ выполняется $a\,\alpha(i)\,b$, если

11.1 $(a\,\alpha_1(i)\,b) \vee \exists x_1,\dots,x_\kappa \left[(a\,\alpha_1(i)\,x_1) \wedge (x_1\,\alpha_1(i)\,x_2) \wedge \dots \wedge (x_\kappa\,\alpha_1(i)\,b)\right]$

Отношение $\delta(i)$ – логической зависимости операторов. Для $a, b \in M_i$ выполняется $(a\,\delta(i)\,b) \wedge (b\,\delta(i)\,a)$, если выполняется

12.1 $(a\,\beta(i)\,b) \vee (b\,\beta(i)\,a)$.

Определенные выше отношения на множестве операторов блока алгоритма функционирования с различных сторон характеризуют структуру блока. Более того, множества P_i и M_i вместе с отношениями $\sigma(i)$, $\alpha_1(i)$ и $\gamma(i)$ и определенной системой правил выполнения операторов блока могут служить представлением

блока, а совокупность таких представлений для всех блоков – новым представлением всего алгоритма функционирования, отличающимся от представления в виде записи на Ф-языке. Система правил R выполнения блоков алгоритма и операторов в блоках представляет собой следующее:

I3.I Выполнение алгоритма – это последовательное выполнение его блоков. Вначале, при отсутствии переходов по внешним меткам, выполняется блок, которому принадлежит первый оператор в записи алгоритма (в алгоритме может быть только один такой блок – см. I.5). После выполнения операторов i -го блока выполняется блок, с входным оператором α , для которого выполняется $\beta\rho^{\circ}\alpha$, если выходной оператор i -го блока не является оператором перехода. В противном случае выполняется блок с входным оператором, помеченным меткой, по которой реализовался переход после выполнения оператора β . В начальный момент выполнения i -го блока начинается отсчет с нуля внутреннего времени $\tau(i)$ i -го блока, а сигнал запуска – ρ_i блока принимает значение I. Если во время выполнения операторов какого-либо блока (например, i -го) произошел переход к выполнению оператора α с номером n_α , то выполнение операторов этого блока прекращается, ρ_i принимает значение 0, отсчет времени $\tau(i)$ прекращается и начинает выполняться блок, входным оператором которого является оператор α .

I3.2 После выполнения в некоторый момент $\tau_c(i)$ оператора перехода c и реализации перехода по j -й метке, стоящей в списке меток c на j -м месте, $\rho_{c,j}$ принимает значение I.

I3.3 Каждому оператору $\alpha\in M_i$ соответствует момент $\tau_\alpha(i)$ внутреннего времени i -го блока, в который оператор α начинает выполняться, причем на значение $\tau_\alpha(i)$ накладываются условия:

а) $\neg\exists\beta\left\{\left[(\beta\alpha_1(i)\alpha)\vee(\beta\sigma(i)\alpha)\right]\wedge(\tau_\beta(i)\geqslant\tau_\alpha(i))\right\}$

б) оператор α можно выполнять в момент времени $\tau_\alpha(i)$ согласно временным ограничениям, наложенным на выполнение операторов.

Оператор α выполняется в момент времени $\tau_\alpha(i)$, если выполняется условие:

в) $\left\{\exists_{c,j}\left[(\rho_{c,j}\gamma(i)\alpha)\wedge(\rho_{c,j}(\tau_\alpha(i)=1)\right]\vee(\rho_i\gamma(i)\alpha)\right\}\wedge(\rho_i=1)$.

I3.4 После выполнения s_i,ρ_i и $\rho_{c,j}$ для всех j и c принимают значение 0.

I3.5 Выполняются пп. I4.8 – I4.II системы правил R_0 (см. [7]). (Под состоянием алгоритма здесь понимается совокупность

имен блока, операторов, выполняющихся в данный момент, и значений всех переменных алгоритма, кроме входных.

Систему правил R совместно с системой множеств и отношений $\{\mathfrak{M},\mathfrak{B}^J,M_i,P_i,\sigma(i),\alpha_1(i),\gamma(i)\}$ будем называть R – описанием алгоритма функционирования. R – описание нам представляется удобным для различного рода эквивалентных преобразований алгоритма и для получения микропрограммного описания работы узлов вычислительного устройства. Ниже показывается, что в определенном смысле Ф-описание и R -описание алгоритма функционирования являются эквивалентными.

При выполнении алгоритма функционирования его операторы используют в разные моменты времени ряд значений $X_i=(x_{i,1},x_{i,2},\ldots,x_{i,\kappa_i})$ каждой входной переменной и вырабатывает ряд значений $X_j=(x_{j,1},x_{j,2},\ldots,x_{j,\kappa_j})$ каждой выходной переменной, где i и j – номера переменных, а $x_{i,l}$ – l -е значение i -й переменной. Алгоритмы A_1 и A_2 будем считать логически эквивалентными, если во время выполнения алгоритмов внешние воздействия отсутствуют, т.е. при замене в алгоритмах операторов условного перехода с непрерывной проверкой на обычные операторы условного перехода с той же записью, при отсутствии (разве лишь только исключая начальный момент времени) переходов по внешним меткам и постоянных значениях сигналов запуска алгоритмов, равных единице, выполняются следующие условия:

I2.I. Каждой входной переменной i_1 алгоритма A_1 можно поставить во взаимно однозначное соответствие входную переменную $i_2=\varphi_1(i_1)$ алгоритма A_2 с такой же областью значений, и наоборот.

I2.2. Каждой выходной переменной j_1 алгоритма A_1 можно поставить во взаимно однозначное соответствие выходную переменную $j_2=\varphi_2(j_1)$ алгоритма A_2 с той же областью значений и наоборот.

I2.3. Можно выбрать такие φ_1 и φ_2 , что для всех выходных переменных алгоритмов всегда выполняется $X_j=X_{\varphi_2(j)}$, если возможно каким-либо способом (например, с помощью временных ограничений) обеспечить равенство $X_i=X_{\varphi_1(i)}$ для всех входных переменных i и $\varphi_1(i)$ алгоритмов.

ЛЕММА I. Если при выполнении алгоритма, представленного на R -языке, отсутствуют внешние воздействия на алгоритм и временные ограничения непротиворечивы, то для опе-

ераторов i-го блока из выполнения (*) $\neg\exists c(c\eta(i)\alpha)\vee\exists c,j[(c\eta^{j}(i)\alpha)\wedge(c\multimap j)]$ следует выполнение

(**) $\exists\tau\, H(\alpha,\tau)$

и наоборот,

где $c\multimap j$ означает, что оператор перехода c выполняется при выполнении блока, и после его выполнения реализуется переход по метке, стоящей в списке его меток на j-м месте, а предикат $H(\alpha,\tau)=1$, если оператор α начинает выполняться в момент времени τ. В противном случае $H(\alpha,\tau)=0$.

Заключение леммы I следует из пп. 8.I, 8.2, I3.3в).

ЛЕММА 2. При условиях леммы I во время выполнения операторов i-го блока алгоритма, представленного на Φ-языке, для операторов этого блока имеет место заключение леммы I.

Из определения блока алгоритма и из п. I4.I2 из [7] утверждение "из (*) следует (**)" равносильно утверждению "из выполнения соотношения (I) $\exists\tau H(\ell,\tau)$ следует выполнение соотношения (**)", где ℓ - входной оператор блока, если в (*) выполняется (2) $\neg\exists c(c\eta(i)\alpha)$ и ℓ - оператор, для которого выполняется $c\beta^{j}(i)\ell$, если в (*) выполняется

(3) $\exists c,j[(c\eta(i)\alpha)\wedge(c\multimap j)]$

Определим бинарное отношение $\beta_n(i)$: для операторов α и β выполняется $\alpha\beta_n(i)\beta$ для $n>1$, если найдутся такие операторы $x_1, x_2, \ldots, x_{n-1}$, что имеет место (4) $(\alpha\beta_1(i)x_1)\wedge(x_1\beta_1(i)x_2)\wedge\ldots\wedge(x_{n-1}\beta_1(i)\beta)$ и не найдется большего, чем $n-1$ числа операторов таких, чтобы выполнялось (4).

Истинность утверждения "из (I) следует (**)" будем доказывать индукцией по n. Для $n=0$, то есть для случая, когда оператор α является оператором ℓ в (I), это утверждение вытекает из п. I4.I2 б) системы правил R_0 (см.[7]) для случая, когда выполняется (3), и, очевидно, для случая, когда выполняется (2), так как входной оператор блока всегда выполняется при выполнении блока. Предположим, что выполняется $\ell\beta_n(i)\alpha$ (если ℓ - входной оператор блока, это возможно потому, что по определению блока для любого оператора α блока выполняется $\ell\beta(i)\alpha$, если ℓ такой, что $c\eta^{j}(i)\ell$, то это возможно по определению $\eta^{j}(i)$) и утверждение "из (*) следует (**)" истинно для всех $\kappa<n$. Если ℓ не является оператором перехода, то, в

силу I4.I2 б) системы правил R_0, из (I) следует выполнение оператора α, для которого имеет место $\ell\beta_1(i)\alpha$, следовательно, по предположению, и оператора α, так как выполняется $\alpha\beta_{n-1}(i)\alpha$. Если ℓ - оператор перехода, то согласно (3) и 5.I имеет место $\neg(\ell\eta(i)\alpha)$ или, в силу 4.I, для любого α и j таких, что $\ell\beta^{j}(i)\alpha$, выполняется $\alpha\beta(i)\alpha$. Пусть $\ell\multimap j$, тогда, в силу I4.I2 б) системы правил R_0, выполняется оператор α, для которого $\ell\beta^{j}(i)\alpha$, следовательно, и оператор α, так как $\alpha\beta_{\kappa-1}(i)\alpha$, где $\kappa<n$. Истинность утверждения "из (**) следует (*)" можно доказать тоже индукцией по n, отправляясь от I4.I2 б) системы правил R_0.

ЛЕММА 3. Для обоих представлений i-го блока алгоритма при отсутствии внешних управляющих воздействий на алгоритм во время выполнения i-го блока из (*) $\exists\tau H(\alpha,\tau)\wedge\exists\tau H(\beta,\tau)$ следует (**) $\alpha\delta(i)\beta$.

Из определения отношений $\delta(i)$, $\beta(i)$ и $\eta^{j}(i)$ заключаем, что если выполняются соотношения $c\eta^{j}(i)\alpha$, $c\eta^{j}(i)\beta$, то выполняется $\alpha\delta(i)\beta$, $c\delta(i)\alpha$ и $c\delta(i)\beta$, то есть выполняются соотношения:

(I) $[(c\eta^{j}(i)\alpha)\wedge(c\eta^{j}(i)\beta)]\multimap(\alpha\delta(i)\beta)$,

(2) $(c\eta^{j}(i)\alpha)\multimap(c\delta(i)\alpha)$.

Из I.2, I.3 и 4.I заключаем, что выполняется $[\neg\exists c(c\eta(i)\alpha)\wedge(\alpha\in M_i)]\multimap[(\alpha\beta(i)\alpha)\vee(\alpha\beta(i)\alpha)]$, а, следовательно, выполняется соотношение

(3) $[\neg\exists c(c\eta(i)\alpha)\wedge\neg\exists c(c\eta(i)\beta)]\multimap(\alpha\delta(i)\beta)$.

Для всех операторов из $\{\alpha\}_0$, для которых всегда выполняется $\neg\exists c(c\eta(i)\alpha)$ (все эти операторы в силу лемм I и 2 выполняются), в силу (3), выполняется $\alpha\delta(i)\beta$ $(\alpha,\beta\in\{\alpha\}_0)$. Среди операторов из $\{\alpha\}_0$ могут быть операторы переходов. Пусть $c\in\{\alpha_0\}$ - оператор перехода и для $\alpha,\beta\notin\{\alpha\}_0$ выполняются соотношения: $c\eta^{j}(i)\beta$, $c\eta^{j}(i)\alpha$, $c\multimap j$. Операторы α и β в силу лемм I и 2 выполняются, и для них, в силу (I) и (2), имеет место (4) $c\delta(i)\alpha$, $c\delta(i)\beta$ и $\alpha\delta(i)\beta$ и, в силу транзитивности отношения $\delta(i)$, выполняется $\beta\delta(i)\alpha$ и $\alpha\delta(i)\alpha$ для любого α, принадлежащего $\{\alpha\}_0$. Если $\ell\in\{\alpha\}_0$ - другой оператор перехода и выполняется $\ell\eta^{\kappa}(i)f$, $\ell\multimap\kappa$, то выполняется $\ell\delta(i)f$ и, в силу транзитивности отношения $\delta(i)$, выполняется $f\delta(i)\alpha$, $f\delta(i)\beta$. Таким образом, выполняющиеся операторы из $\{\alpha\}_1$, выполнение которых зависит только от результата выполнения опе-

торов перехода из $\{a\}_0$, и операторы из $\{a\}_0$ находятся в отношении $\delta(i)$. Далее можно рассмотреть операторы из $\{a\}_2$, куда входят операторы a, для которых выполняются соотношения $c\,\eta^{j(i)}\,a$, где $c \in \{a\}_0 \cup \{a\}_1$, и показать, что выполняющиеся операторы из $\{a\}_2$ находятся в отношении $\delta(i)$ с операторами из $\{a\}_0$ и $\{a\}_1$ и т.д. По индукции получаем заключение леммы 3.

ЛЕММА 4. Из условий леммы 3 и одновременного выполнения $(*)\ (a\,\varepsilon_\rho(i)\,b) \wedge \exists\tau H(a,\tau) \wedge \exists\tau H(b,\tau)$ и $(**)\ \neg\exists c,\tau[(a\,\beta(i)\,c) \wedge (c\,\varepsilon_\rho(i)\,b) \wedge H(c,\tau)]$ следует, что оператор b во время выполнения использует результат выполнения оператора a по переменной ρ.

Для блока алгоритма, заданного на Φ-языке, заключение леммы 4 следует из I4.I2 б) системы правил R_0 и из определения отношения $\varepsilon_\rho(i)$. Из 9.4 и из определения $\varepsilon_\rho(i)$ заключаем, что для любого оператора c, имеющего в качестве выходной переменной переменную ρ, для которого выполняется $b\,\beta(i)\,c$ или $c\,\beta(i)\,a$ (т.е. $c\,\delta(i)\,a$ и $c\,\delta(i)\,b$), выполняется $c\,\alpha_1(i)\,a$ или $b\,\alpha_1(i)\,c$, соответственно. Из этих условий, а также в силу леммы 3, п. I3.3 а) системы правил R и соотношений и получаем заключение леммы 4 для блока алгоритма, представленного на R-языке.

ТЕОРЕМА I. Алгоритмы A и B (где A представлен на языке, в котором $\alpha_1(1)$, $\beta(1)$, $\sigma(1)$, $\mathfrak{M}$, M_1 и P_1 определены для множества операторов алгоритма A, отображения φ_1 и φ_2 суть тождественные отображения переменных в себя) без циклов и внешних меток, представленные блок-схемой из одного блока, логически эквивалентны.

При доказательстве теоремы I вместо $\beta^j(i)$, $\beta_1(i)$ и т.д. будем писать просто β^j, β_1 и т.д., так как алгоритмы A и B состоят из одного блока.

Пусть множество $\{a\}_0$ – множество операторов, для которых выполняется $\neg\exists b(b\,\alpha\,a)$. Множество операторов, для каждого оператора a, из которых выполняется $\exists a[(a \in \{a\}_0) \wedge (a\,\alpha_1\,a_1) \wedge \neg\exists a[(a \notin \{a\}_0) \wedge (\eta\,\alpha\,a_1)]$, будем обозначать через $\{a\}_1$; множество операторов, из которых для каждого оператора a_2 выполняется $\neg\exists a[(a \notin \{a\}_1 \cup \{a\}_0) \wedge (a\,\alpha\,a_2)] \wedge \exists a[(a \in \{a\}_1) \wedge (a\,\alpha_1\,a_2)]$, будем обозначать через $\{a\}_2$ и т.д. Теорема I может считаться доказанной, если будет доказано, что во время выполнения A и B

а) при одинаковых значениях входных переменных каждый оператор этих алгоритмов присваивает своим выходным переменным одинаковые значения, или, если это – оператор перехода, в обоих случаях реализуется переход по одной и той же метке;

б) если оператор выполняется в A, то он выполняется в B, и наоборот.

Доказательство теоремы проводится индукцией по номеру множеств $\{a\}_n$. Для $n=0$ (т.е. для $\{a\}_0$) утверждения а) и б) выполняются в силу следующих причин. В силу леммы I и 2 все операторы из $\{a\}_0$ выполняются в A (это операторы, для которых имеет место $\neg\exists c(c\,\eta\,a)$) и выполняются в B. Кроме того, в силу условий теоремы все операторы из $\{a\}_0$ как в A, так и в B используют одинаковые значения внешних переменных и выполняются только один раз. Отсюда заключаем, что операторы из $\{a\}_0$ в A и в B присваивают своим выходным переменным одинаковые значения.

Предположим, что это утверждение выполняется для всех операторов, принадлежащих множествам $\{a\}_0, \{a\}_1, \ldots, \{a\}_n$. Покажем, что тогда оно выполняется для операторов из $\{a\}_{n+1}$. Операторы из $\{a\}_{n+1}$ могут использовать результаты работы операторов из $\bigcup_{i=0}^{n}\{a\}_i$ и значения внешних переменных алгоритмов. Поскольку операторы из $\bigcup_{i=0}^{n}\{a\}_i$ как в A, так и в B имеют одинаковые результаты работы, то, в силу лемм I,2 и пп.9.2,9.3, все те операторы из $\{a\}_{n+1}$, которые выполняются в A, выполняются в B, и наоборот. Отсюда, в силу леммы 4, если оператор a в алгоритме A использует результат работы оператора b, то и в алгоритме B оператор a использует результат работы b. Следовательно, каждый из выполняющихся операторов из $\{a\}_{n+1}$ как в A, так и в B использует результат работы одних и тех же операторов из $\bigcup_{i=0}^{n}\{a\}_i$. Но все эти операторы своим выходным переменным присваивают в A и в B одинаковые значения. Кроме того, операторы из $\{a\}_{n+1}$ по условиям теоремы используют одинаковые значения внешних переменных алгоритмов и, со-

гласно правилам R_0 и R, выполняются только один раз (A и B - алгоритмы без циклов). Отсюда делается вывод, что все выполняющиеся операторы из $\{a\}_{n+1}$ имеют одинаковые результаты работы при выполнении алгоритмов A и B. Теорема I доказана.

Следствие теоремы I. Системе правил R удовлетворяет множество конкретных вариантов распределения выполнения операторов алгоритма во времени. Поскольку теорема I была доказана безотносительно к какому-либо варианту распределения в алгоритме B, можно сделать заключение: все варианты распределения выполнения операторов алгоритма B во времени эквивалентны друг другу в смысле конечного результата выполнения алгоритма B.

Обладая полной информацией о внешней среде алгоритма, которая в определенные моменты времени присваивает определенные значения входным переменным алгоритма и использует значения его выходных переменных, человек может так задать временные ограничения, что алгоритмы A и B из теоремы I будут одинаково работать и во времени.

ТЕОРЕМА 2. Алгоритмы A и B, где A представлен записью на Φ-языке, а B представлен на R-языке, логически эквивалентны.

При выполнении алгоритма A согласно системе правил R_0 после выполнения оператора, который является входным оператором какого-либо (например, i-го) блока алгоритма B, начинают выполняться операторы этого блока. В силу I.I и I.3 из определения блока, п. 8.5 системы правил R_0 и того условия, что отсутствуют внешние воздействия на алгоритм, во время выполнения операторов i-го блока не выполнится ни одного оператора, не принадлежащего этому блоку, до тех пор, пока не выполнился выходной оператор i-го блока. После выполнения выходного оператора i-го блока начинает выполняться входной оператор следующего (например, j-го) блока, что следует из пп. I.5, I.6 в определении блока. В начальный момент выполнения алгоритма B выполняются операторы того блока алгоритма B, входным оператором которого является первый оператор записи алгоритма. Таким образом, выполнение алгоритма A, как и алгоритма B, представляет собой процесс последовательного выполнения блоков алгоритма A по тем же правилам, что и в B (см. I3.I системы правил R). Это имеет место и в том случае, когда в алгоритме A в начальный момент времени происходит переход по внешней метке, так как выполняется условие п. I.2 в определении блока. В силу этого и теоремы I выполняется теорема 2.

При получении микропрограммного описания работы узла вычислительного устройства будем исходить из следующих принципов построения функциональной схемы операционной части (ОП) узла и генератора управления.

Функциональная схема ОП

I. Схема ОП представляет собою совокупность элементарных схем, соединенных друг с другом согласно логике алгоритма функционирования.

2. Каждая элементарная схема выполняет одну из следующих функций:

а) функцию хранения информации (примером может служить регистр или триггер);

б) функцию передачи информации (например, шина для передачи n - разрядного двоичного кода);

в) функцию преобразования информации (например, сумматор двух n - разрядных кодов).

Элементарные схемы, выполняющие эти функции, в дальнейшем мы будем называть функциональными преобразователями.

3. Элементарные схемы в схеме ОП узла могут соединяться друг с другом или непосредственно, или посредством временных клапанов, управляемых временными сигналами из генератора управления.

4. При проектировании ОП узла каждому оператору алгоритма его функционирования ставится в соответствие совокупность элементарных схем. Переменным оператора ставятся в соответствие схемы типа а) или б), а функциям - типа в). При этом одним и тем же именам переменных и функций в разных операторах алгоритма ставятся в соответствие одни и те же элементарные схемы.

5. Операторам условного перехода и условного перехода с непрерывной проверкой ставятся в соответствие совокупность регистров или шин, соответствующих переменным оператора, функциональный преобразователь, реализующий логическое условие оператора, и регистр или шина, каждый разряд которого соответствует определенной метке оператора. Выходы этого регистра (шины) подаются в генератор управления.

6. Операторам СТОП и конца непрерывной проверки ставится в

соответствие триггер (или одноразрядная шина), выходы которого подаются в генератор управления.

7. Оператору безусловного перехода в схеме ОП ничего не ставится в соответствие, так как операторы этого вида присутствуют в алгоритме, чтобы обеспечить линейную запись операторов.

8. Каждой условной переменной для каждого оператора условного перехода и условного перехода с непрерывной проверкой в ОП ставится в соответствие регистр (или шина). Выходы этого регистра (шины) соединяются со специальными схемами, которые управляют подачей временных сигналов из генератора управления на определенные временные клапаны в ОП. В зависимости от реализации соответствующего оператора условного перехода по той или иной его метке единичное значение имеет тот или иной разряд регистра (шины), а это, в свою очередь, определяет подачу временного сигнала на соответствующий временной клапан. Таким образом осуществляется выполнение одних операторов блока и невыполнение других. Выходы регистра (шины), соответствующего условной переменной выходного оператора блока, подаются в генератор управления и служат сигналами запуска соответствующих блоков.

Таким образом, просматривая запись исходного алгоритма, можно некоторым регулярным способом построить функциональную схему операционной части узла, элементами которой являются регистры, шины, функциональные преобразователи и связи между ними.

Генератор управления

Так как, в силу I3.I, выполнение алгоритма функционирования узла рассматривается как последовательное выполнение его блоков, то каждому блоку алгоритма в генераторе управления узла ставится в соответствие последовательность временных сигналов, которые будут управлять работой элементарных схем в ОП. Работа генератора управления заключается в последовательной смене таких последовательностей сигналов. По окончании выполнения всех операторов очередного блока по сигналу запуска следующего блока, приходящему из ОП, генератор управления начинает выдавать последовательность временных сигналов, соответствующих следующему блоку.

Выполнение операторов блоков на соответствующих им элементарных схемах в ОП занимает определенное время. Поэтому при

распределении выполнения операторов во времени возникает задача выбора такого варианта распределения в соответствии с системой правил I3.I – I3.5, который бы удовлетворял временным ограничениям, наложенным на выполнение операторов. Такое распределение мы будем называть временной диаграммой алгоритма. Кроме того, до получения временной диаграммы алгоритма необходимо определить схемную интерпретацию элементов исходного алгоритма, то есть информацию о соответствии элементам операторов конкретных элементарных схем, так как одной элементарной схеме может соответствовать несколько элементов из разных операторов (например, функции логического сложения из двух операторов могут в ОП выполняться на одной схеме). Если учесть, что процесс выполнения алгоритма представляет собой последовательное выполнение его блоков, то понятие временной диаграммы можно определить так. Временной диаграммой алгоритма функционирования узла вычислительного устройства называется совокупность конкретных вариантов распределения во времени выполнения операторов в каждом блоке алгоритма. При этом в каждом блоке устанавливается свое время, отсчитываемое от нуля. Множество операторов i-го блока алгоритма вместе с определенным для него вариантом распределения выполнения операторов во времени, схемной интерпретацией элементов его операторов, множеством P_i и отношением $\gamma(i)$ будем называть i-й микропрограммой, а совокупность m микропрограмм узла вместе с системой отношений $\{\mathcal{B}^0, \mathcal{B}^1, \dots\}$ – микропрограммным описанием узла.

Таким образом, задача получения микропрограммного описания должна решаться при следующих дополнительных к системе A условиях.

I. Каждому имени переменной и функции ставится в соответствие элементарная схема. Каждая элементарная схема характеризуется временем работы, оборудованием и другими свойствами, связанными с ее реализацией на элементах конкретного вычислительного комплекса.

2. Если оператор распределен на выполнение в некоторый момент времени, то нельзя прерывать выполнение этого оператора.

3. Время срабатывания элементарной схемы (а следовательно, и время выполнения любого оператора) в принятых единицах измерения равно целому числу.

4. В силу леммы 3, операторы, выполняющиеся на одних и тех же элементарных схемах, нельзя распределять на выполнение в одном такте, если они находятся в отношении логической зависимости $\delta(c)$.

З а м е ч а н и я. Условия из вышеперечисленных пп. 2, 3, 4 можно выразить с помощью дополнительных временных ограничений, наложенных на начальные времена выполнения операторов, в результате чего уменьшится только число возможных вариантов распределения операторов алгоритма во времени. Следовательно, добавление перечисленных четырех пунктов к системе правил R (п. I совместно с п. 4 косвенно влияет на выполнение операторов) не нарушает логической эквивалентности алгоритмов A и B из теорем I и 2.

Задача получения микропрограммного описания работы узла может быть сформулирована так. Необходимо провести эквивалентное преобразование алгоритма функционирования, представленного на Ф-языке, в алгоритм, представленный на R-языке. Затем для каждого оператора в блоках алгоритма нужно определить такт во внутреннем времени блока, с которого он начнет выполняться, и те элементарные схемы, на которых он будет выполняться в схеме ОП, после чего заменить имена элементов записи исходного алгоритма на имена соответствующих им элементарных схем. При этом необходимо удовлетворить заданным критериям, связывающим оборудование и общее время выполнения алгоритма, для чего нужно иметь информацию о частоте выполнения блоков при выполнении алгоритма.

В заключение автор выражает благодарность Ю.В. Капитоновой за ценные замечания по этой работе.

Л и т е р а т у р а

I. В.М. ГЛУШКОВ, Ю.В. КАПИТОНОВА, А.А. ЛЕТИЧЕВСКИЙ. Об одной методике проектирования устройств вычислительных машин. - Тезисы докладов к предстоящему Всесоюзному коллоквиуму по автоматизации синтеза дискретных вычислительных устройств. Новосибирск, I966 г., стр. 3.

2. I.SHORR. Computer-aided digital system design and analysis using a register transfer language. IEEE Trans. on Electronic Computers, 1964, EC-13, N 6, pp. 730-737.

3. R.PROCTOR. A logic design translator experiment demonstrating relationship of language to system and logic design. IEEE Trans. on Electronic Computers, 1964, EC-13, N 4, pp. 422-430.

4. В.М. ГЛУШКОВ. К вопросу о минимизации микропрограмм и схем алгоритмов. - Кибернетика. Киев, I966, № 5.

5. T.BARTEE, I.LEBOW, I.REED. Theory and design of digital machines, N.Y., McGrow-Hill, 1962.

6. И.В. ИЛОВАЙСКИЙ, Б.А. СИДРИСТЫЙ, Я.И. ФЕТ. Автоматический синтез дискретных вычислительных устройств. - Вычислительные системы. Новосибирск, I966, вып. 25, стр.5-I7.

7. С.В. ПИСКУНОВ, С.Н. СЕРГЕЕВ, Б.А. СИДРИСТЫЙ. Язык для описания алгоритмов функционирования ЦВМ. - Настоящий сборник, стр. 5-I8.

Поступила в редакцию
I8.I2. I968 г.