

## О ПРОГРАМИРУЮЩЕЙ ПРОГРАММЕ ДЛЯ ПОСТРОЕНИЯ ИСПЫТАТЕЛЬНЫХ ПРОГРАММ

*Г. А. Миронов, В. П. Битюцкий, Н. А. Комиссаров, В. П. Пинаев*  
(Москва)

Выполнение испытательной программы является реализацией некоторых актов контроля. Каждый такой акт состоит из двух последовательных действий:

- 1) работа контролируемого оборудования,
- 2) сопоставление результата работы контролируемого оборудования с эталоном.

Работа контролируемого оборудования в простейшем случае заключается в том, что некоторый блок машины (в смысле [1], [2]) выполняет элементарный тест.

Элементарный тест может являться:

- 1) частью контролирующего теста, осуществляющего проверку блока с логической полнотой охвата, равной единице, 2) специальным тестом для проверки такого тяжелого режима работы блока, для создания которого достаточно выполнить изолированный элементарный тест.

Имеют место также и более сложные случаи работы контролируемого оборудования. Их можно разделить на два класса:

- 1) выполнение последовательности элементарных тестов, 2) выполнение одного или группы элементарных тестов в совокупности с элементарными операциями [1], обеспечивающими условия работы оборудования.

К первому классу относятся такие случаи проверки пра-

вильности работы оборудования, когда режим его использования определяется последовательностью элементарных тестов. Примером может служить последовательность элементарных тестов, создающая тяжелый частотный режим работы элементов. При выполнении элементарных тестов последовательности не обязательно работает одна и та же элементарная машинная операция.

Заметим, что разрывать последовательность элементарных машинных операций элементарными машинными операциями контроля можно лишь в том случае, когда это не облегчает режим работы оборудования.

Ко второму классу более сложных случаев работы контролируемого оборудования, чем при выполнении одного элементарного теста, относятся случаи, когда проверяемую элементарную операцию (или последовательность таких операций) сопровождают другие элементарные операции, результаты которых непосредственно не контролируются, а их работа создает тяжелые условия выполнения контролируемых элементарных машинных операций.

Ко второму классу относятся все случаи контроля работы оперативных и других запоминающих устройств в условиях, способствующих разрушению информации при хранении [3], а также некоторые специальные режимы работы арифметического устройства и устройства управления.

Действительно, элементарной операцией, служащей для организации хранения некоторого кода в ячейке, является операция, осуществляющая запись в эту ячейку. Далее, например, при проверке на хранение в условиях возбуждения полуточками в ОЗУ с прямым выбором, выполняются некоторые операции, результаты выполнения которых непосредственно не контролируются — не сопоставляются с эталонами. Если какая-либо из этих операций выполнится неверно, то это может никак не сказаться на работе испытательной программы. Эти операции создают условия хранения — производят возбуждение проверяемой ячейки со стороны ячеек, связанных с ней по координатам.

При проверке запоминающего устройства на многократные обращения выполняется предварительный "разогрев" контролируемой ячейки специальными операциями, а затем производится контроль правильности записи и чтения для этой ячейки. Заметим, что в отличие от предыдущего случая, тяжелые условия при проверке на многократные обращения создаются не в процессе выполнения, а перед выполнением проверяемого действия. Подобный случай также имеет место, когда в ОЗУ матричного типа контролирует-

ся воздействие помехи при считывании. Для создания условий возникновения максимальной помехи во все ячейки ОЗУ предварительно записывается тяжелый код, а затем выполняются элементарные тесты, контролирующие запись и чтение из всех ячеек ОЗУ. При этом последовательности операций, осуществляющих контроль, предшествует последовательность операций, осуществляющих запись тяжелого кода.

Для арифметического устройства и устройства управления имеет место использование последовательностей элементарных машинных операций для создания тяжелых режимов в цепях питания. Такая проверка производится, в частности, на машине М-20. Усовершенствование такой проверки было сделано инженерами Жаровым Р.Д. и Кравчуком В.Г. Ими определена резонансная частота источников питания. Последовательность операций, создающая тяжелый режим, вводит источники питания в резонанс, а затем, после достижения установившегося режима, в момент выдачи наименьшей мощности, выполняется такой элементарный тест, что для получения правильного результата необходимы импульсы большой мощности от источников питания.

Последний важный пример выполнения операций, служащих для создания условий работы контролируемых операций, имеет место в машинах, снабженных устройством автоматизации выполнения профилактических работ [4], [5]. При проверке с таким устройством перед операциями, обеспечивающими работу контролируемого оборудования, выполняются операции установления условий контроля. Отличительной особенностью таких операций является сравнительно большое время их выполнения.

В дальнейшем считаем, что все элементарные тесты, нужные для осуществления проверки с полнотой охвата или эффективностью контроля, равными единице [3], найдены заранее с помощью каких-либо способов [7]; [8]; [9]; [10]. Считаем, также, что количество и характер элементарных операций, создающих требуемые условия работы, также известны. Таким образом, остается проблема рационального применения тестов и операций, обеспечивающих условия работы, то есть задача построения "программы испытаний" - испытательной программы. Испытательные программы обладают некоторыми характеристиками качества, важнейшими из которых являются быстроедействие и надежность [3]. Настоящая работа описывает первые попытки такого построения испытательных программ формальными методами, чтобы обеспечивалось

требуемое их качество.

В свете изложенного уточним некоторые определения, данные ранее в [3] .

Основная элементарная машинная операция - такая элементарная машинная операция, результат выполнения которой подлежит контролю.

Основная команда испытательной программы - такая команда, в состав которой входит основная элементарная машинная операция.

Вспомогательные элементарные машинные операции по условиям работы - такие элементарные машинные операции, которые создают условия выполнения основных элементарных машинных операций.

Вспомогательные команды по условиям работы - такие команды, которые содержат вспомогательные элементарные машинные операции по условиям работы.

Вспомогательные элементарные машинные операции по контролю - такие элементарные машинные операции, которые производят сопоставление результата работы контролируемого оборудования с эталоном.

Вспомогательные команды по контролю - команды, в состав которых входят элементарные машинные операции по контролю.

Приведенным перечнем в основном исчерпываются типы элементарных машинных операций и команд, имеющие место в испытательных программах, не содержащих операторов формирования.

Для простоты мы ограничивались элементарными операциями и командами, имеющими место в контролируемых программах. В диагностических программах, основанных, например, на методе словарей [6], могут иметь место команды других типов.

Если испытательная программа содержит операторы формирования, то имеют место также вспомогательные элементарные машинные операции формирования, осуществляющие формирование основных или любых вспомогательных команд. Соответственно имеют место вспомогательные команды формирования.

Приведенный анализ (в действительности значительно более детализированный) позволяет поставить задачу построения программирующей программы, строящей испытательные программы.

Обычно программирующая программа трактуется как синоним понятия "транслятор" и представляет собой программу (на некотором машинном языке  $M_1$ ), построения программы на ма-

шинном языке  $M_j$ , равносильной записи на некотором алгоритмическом языке, отличном от языка  $M_j$ . При этом не обязательно  $i = j$ .

Обычно алгоритмический язык, с которого осуществляется транслирование, охватывает некоторый класс задач.

В нашем случае класс задач исчерпывается испытательными программами и характеризуется четко определенными типами используемых элементарных машинных операций. Поэтому "входной язык" такой программирующей программы может охватывать лишь те типовые операторы, которые применяются при построении испытательных программ.

Эти типовые операторы могут быть заложены в программирующую программу однажды и "входной язык", служащий для их записи, может быть неизвестен пользователю программирующей программы. В дальнейшем список типовых операторов может пополняться при появлении новых типов элементарных машинных операций, новых запоминающих устройств.

Предлагаемая программирующая программа должна строить типовые операторы, не для одной машины (язык  $M_j$ ), а для некоторого класса машин  $\{M_j\}$ . При этом программирующей программой задается формальное описание соответствующего машинного языка  $M_j$ , [1].

Имея описание машинного языка и описание типовых операторов, программирующая программа производит программирование этих операторов на языке заданной машины. Следует заметить, что такое программирование обычно может быть выполнено несколькими способами. Это дает возможность произвести оценки получаемых равносильных программы по времени или надежности их выполнения. Для таких оценок дополнительно к описанию языка задаются характеристики по времени выполнения элементарных операций и команд языка  $M_j$ ; и надежность их выполнения, например, в виде вероятностей безотказной работы элементов, составляющих блоки, реализующие элементарные машинные операции.

В качестве примеров рассмотрим некоторые типовые операторы и возможности их программирования.

Одним из простейших типовых операторов является оператор, состоящий из основной элементарной машинной операции, осуществляющей запись в ячейку оперативного запоминающего устройства. Очевидно, что в трехадресных машинах таких элементарных операций несколько. Каждая из них характеризуется ко-

мандами, в которые она входит, а через них быстродействием и надежностью выполнения. Составленные с использованием всех таких команд операторы могут сравниваться между собой по соответствующим критериям.

Другим примером могут служить типовые операторы создания условий работы при контроле ячеек оперативных запоминающих устройств (например - долбление, возбуждение полутками). Каждый из этих операторов обычно может быть реализован с помощью разных элементарных операций, входящих в разные команды, с соответствующими оценками по надежности и быстродействию. Для автоматизации программирования этих операторов должен быть задан в общем случае перечень тяжелых режимов, характерных для разных типов оперативных запоминающих устройств, структуры соответствующих операторов, тип устройства, применяемого в машине  $M_3$ . Кроме типа ОЗУ должны быть заданы характеристики прошивки (пределы изменения значений адресов ячеек, в которые производятся обращения). Заметим, что для получения команд, содержащих элементарные операции, вспомогательные по условиям работы, приходится зачастую применять вспомогательные операции их формирования. Соответствующие операторы также задаются формально. Из сказанного следует, что построение типовых операторов создания тяжелых режимов (условий) работы оперативных запоминающих устройств является самостоятельной, весьма сложной задачей, в которую входят: классификация устройств, исследование эффективности режимов, формализация информации, полученной в результате соответствующих исследований. То же относится и к контролю тяжелых режимов в других запоминающих устройствах (магнитных лент, магнитных барабанов)

Остановимся несколько более подробно на структуре операторов, осуществляющих второе действие акта контроля - сопоставление результата работы контролируемого оборудования с эталоном. Структура операторов во многом определяется доступностью ячеек, в которых образуется проверяемый результат, для элементарных машинных операций, осуществляющих сопоставление. Рассмотрим некоторые типовые примеры. При этом будем считать, что сопоставление может иметь один из двух результатов: 1) останов машины, когда совпадения нет и 2) переход к последующим проверкам, когда совпадение имеет место. Операторы такого вида применяются главным образом для контролируемых программ, не имеющих автоматического перехода к диагностической программе.

а) Контролируемый результат фиксируется в одnorазрядной запоминающей ячейке  $\Phi$ , помещающей сигнал переполнения. Для простоты рассмотрим только случай, когда эталонное значение сигнала равно нулю.

Сопоставление результата с эталоном при этом обычно производится двумя элементарными операциями :

$$Я : = \Phi, \quad S : = S + 1,$$

здесь Я - ячейка останова;

$\Phi$  - ячейка переполнения;

S - счетчик команд.

Обычно приведенные элементарные операции выполняются в той же команде, в которой выполняется основная элементарная операция  $\Phi : = f(\Sigma, x, y)$ .

Тогда операторная схема, осуществляющая контроль одной элементарной операции, состоит из одного оператора А, состоящего из одной команды, в состав которой входят элементарные операции:

$$R := f_R(N), \quad \Phi := f_\Phi(\Sigma, x, y), \quad \Phi : = Я, \quad S := S + 1.$$

Здесь: R - регистр команд;

N - содержимое счетчика команд;

$\Sigma$  - обозначение сумматора;

x, y - содержимое адресной части регистра команд; в одноадресных машинах x или y обычно не является существенным аргументом функции  $f_\Phi$ , в трехадресных несущественным аргументом является  $\Sigma$ ;

б) Результат фиксируется в счетчике команд.

Сопоставление результата с эталоном обычно производится путем выполнения элементарной операции:

$$R : = N$$

при соблюдении следующего условия: " в ячейке с номером N хранится команда, не содержащая элементарную операцию Я : = I, и во всех ячейках, с номерами  $N \neq N_{\text{эт}}$  хранятся команды, содержащие элементарную операцию Я : = I. Другими словами, необходимо, чтобы во всех ячейках были останова, кроме той, из которой будет производиться выборка команды в случае правильного результата на счетчике команд.

Логическая схема программы имеет вид:

$$Я_1, Я_2, \dots, Я_{i-1}, B_i \left[ \begin{matrix} N \\ Я_{i+1} \dots Я_{N-1} \end{matrix} \right] B_N Я_{N+1} \dots Я_M. \quad (I)$$

Здесь  $Я_k$  - оператор, состоящий из команды, записанной в ячейку  $K$  и содержащей элементарную операцию

$$Я := I \quad (1 \leq k \leq M, k \neq i, k \neq N,$$

$M$  - емкость ОЗУ);

$B_i$  - основной оператор, состоящий из команды, содержащей контролируемую элементарную операцию с результатом  $B$   $S$  (в частном случае возможно  $N = i + 1$ );

$B_N$  - оператор, состоящий из команды, отличной от остановки (не содержащей  $Я := I$ ).

Можно ослабить сформулированное выше условие, размещая остановки лишь в тех ячейках, которые соответствуют наиболее вероятным ошибкам. При этом можно обеспечить автоматическое исполнение более чем одной проверки,.

Заметим, что логическая схема (I) требует предварительного выполнения вспомогательных операции и команд формирования.

в) Результат фиксируется в одноразрядной запоминающей ячейке  $\omega$ , помещающей сигнал управления командой условного перехода. Для простоты рассмотрим случай, когда эталонное значение сигнала равно нулю.

Сопоставление производится путем выполнения элементарных операций:

$$S := \begin{cases} A & \text{при } \omega = 1, \\ S + 1 & \text{при } \omega = 0, \end{cases} \quad R := N,$$

при условии, что в ячейке с номером  $N$  указанным в адресной части регистра команд -  $A$ , стоит команда, содержащая в своем составе элементарную операцию  $Я := I$ .

Логическая схема проверки имеет вид:

$$\begin{matrix} A & A \\ P [ & \dots & ] Я_1. \\ S+1 & & A \end{matrix}$$



Здесь : Р - оператор, в котором выполняется основная элементарная машинная операция

$$\omega := f_{\omega}(\Sigma, x, y),$$

$\begin{matrix} \Lambda \\ \vdots \\ S+1 \end{matrix}$  - знак перехода, являющийся операцией условного перехода.

$\mathcal{I}_1$  - оператор останова, выполняющийся при  $\omega = 1$ .

г) Результат фиксируется в сумматоре или в ячейке оперативного запоминающего устройства.

Сопоставление производится путем выполнения элементарных операции:

$$\begin{aligned} \Sigma &:= c, & \Sigma_1 &:= \Sigma_1 \cong c_1^{эТ}; \\ \omega &:= \Sigma_1 \vee \Sigma_2 \vee \dots \vee \Sigma_n; & (2) \\ S &:= \begin{cases} \Lambda & \text{при } \omega = 1, \\ S+1 & \text{при } \omega = 0, \end{cases} & R &:= N, \end{aligned}$$

при условии, что в ячейке с номером N, указанном в адресной части регистра команд -  $\Lambda$ , стоит команда, содержащая в своем составе элементарную операцию  $\mathcal{I} := 1$ .

В последовательности формул 2:

$\Sigma_i$  - i-й разряд сумматора  $1 \leq i \leq n$ ,

$c^{эТ}$  - содержимое ячейки ОЗУ, в которой помещается эталонный результат основной операции  $c := f(\Sigma, x, y)$ .

Последовательность 2 при программировании команд для конкретных машин подвергается преобразованиям. Рассмотрим некоторые из них:

1. Если результат в сумматоре, то операция  $\Sigma := c$  исключается.

2. Если программирование ведется для одноадресной машины, а операция поразрядного отрицания разнзначности с выработкой сигнала  $\omega$  и условного перехода выполняются по отдельным командам, то последовательность 2, может иметь вид:

$$\Sigma := f(\Sigma, x); \quad S := S + 1; \quad R := N;$$

$$\Sigma_1 := \Sigma \cong c_1^{эТ}; \quad \omega_1 := \Sigma_1 \vee \Sigma_2 \vee \dots \vee \Sigma_n; \quad S := S + 1; \quad R := N;$$

$$S := \begin{cases} A & \text{при } \omega = 1, \\ S + 1 & \text{при } \omega = 0, \end{cases} \quad R := N.$$

Здесь каждая из строк представляет собой одну команду.

3. Если программирование ведется для трехадресной машины, а операции выработки сигнала  $\omega$  по результату поразрядного сравнения, и условного перехода выполняется по отдельным командам, то последовательность 2 может иметь вид:

$$c := f(x; y); \quad S := S + 1; \quad R := N; \quad (4)$$

$$\omega := (c_1 \neq c_1^{ST}) \vee (c_2 \neq c_2^{ST}) \vee \dots \vee (c_n \neq c_n^{ST}); \quad S := S + 1; \quad R := N;$$

$$S := \begin{cases} A & \text{при } \omega = 1, \\ S + 1 & \text{при } \omega = 0, \end{cases} \quad R := N.$$

Здесь также каждая из строк представляет собой одну команду.

Последовательности команд 3 и 4 могут быть представлены логической схемой:

$$A_1 P_2 \left[ \begin{matrix} N \\ 3 \end{matrix} \right] \dots \left[ \begin{matrix} N \\ N \end{matrix} \right] Y_N. \quad (5)$$

В этой схеме

$A_1$  - оператор, состоящий из основной команды, в которой выполняется контролируемая элементарная машинная операция

$$c := f(x; y); \quad \Sigma := f(\Sigma; x);$$

$P_2$  - оператор, состоящий из вспомогательной команды, в которой выполняется операция выработки сигнала  $\omega$ ;

$\left[ \begin{matrix} N \\ 3 \end{matrix} \right]$  - знак перехода, выполняющий команду, в которой содержится элементарная операция условного перехода.

$Y_N$  - команда, содержащая операцию  $Y := I$ ,

Приведенные выше структуры типовых операторов не являются единственными. Например, те же действия, которые реализуются последовательностью формул (2), могут быть выполнены с помощью двукратного выполнения вычитания (сложения) с вычиткой сигнала  $\omega$  по знаку результата.

число возможных реализации типовых операторов определяется набором элементарных машинных операций в конкретной машине, а также числом возможных равносильных схем типовых операторов, имеющихся в информации к программирующей программе.

Заметим, что во многих случаях программирование основных команд не вызывает затруднений – контролирующие тесты зачастую определяют структуру основных операторов.

Задача автоматического построения типовых операторов является центральной и обуславливает успешное решение всей проблемы построения как отдельных испытательных программ, так и систем таких программ, обладающих заданными характеристиками.

Действительно, если типовые операторы построены, то их объединения в испытательные программы не вызывает принципиальных затруднений. При этом должна применяться оптимизация характеристик испытательных программ путем подбора типовых операторов, входящих в эти программы.

В ряде случаев могут быть сформулированы требования к дополнительным элементарным машинным операциям, или к операциям, выполняемым вручную с помощью наладочного оборудования, обеспечивающим высокую скорость и надежность процесса проверки при наладке машины, проведении профилактических работ или при оперативном контроле машины.

Определяющим критерием для построения систем испытательных программ, служащих для наладки, профилактических работ, или поиска неисправностей в процессе эксплуатации, являются априорные вероятности наличия неисправностей в машине при её наладке, профилактике или эксплуатации. При этом, в частности, может быть формально определена "база" – исправно работающая часть машины, необходимая для осуществления её наладки.

Системы испытательных программ могут строиться автоматически и для вычислительных систем. При этом определяющим является то обстоятельство, что вспомогательные типовые операторы могут выполняться в исправной машине системы, а основные – в неисправной. При этом, разумеется, имеют значение априорные вероятности наличия неисправностей в той или иной машине системы, что в свою очередь определяется в основном периодичностью и порядком проведения профилактических работ. Очевидно, что вероятности наличия неисправностей, заданные априорно при составлении системы испытательных программ, мо-

гут уточняться по результатам эксплуатации системы. Соответственно, могут быть перепрограммированы отдельные программы или вся их система.

В настоящее время предпринята попытка построения программирующей программы, позволяющей строить типовые операторы и из них испытательную программу для проверки элементарных машинных операций, выполняемых в арифметических устройствах универсальных машин. При этом считается, что совокупность элементарных тестов задана и каждая проверка представляет собой выполнение одного элементарного теста с результатом в ячейке ОЗУ или в сумматоре и сопоставление этого результата с эталоном. Такие типовые операторы имеют логическую схему, подобную (5).

## Л И Т Е Р А Т У Р А

1. Н.А. Криницкий, Г.А. Миронов, Г.Д. Фролов. Использование понятий элементарных машинных операций для анализа вычислительных систем. Настоящий сборник.
2. Г.А. Миронов. Основные проблемы программного контроля дискретных автоматических систем. Настоящий сборник.
3. Г.А. Миронов. Испытательные программы для контроля электронных цифровых машин. Наука, 1964.
4. Ю.В. Гаикович, Г.А. Миронов. К вопросу об автоматизации профилактических работ на ЭЦМ. Настоящий сборник.
5. Ю.В. Гаикович, Б.Г. Гусаков, А.Я. Каменир, Г.А. Миронов. Устройство для автоматизации профилактических работ. в сб. "Цифровая вычислительная техника и программирование". № 3, 1967.
6. Н.А. Криницкий, Г.А. Миронов, Г.Д. Фролов. Описание систем команд ЭЦМ с помощью элементарных машинных операций. Сб. цифровая вычислительная техника и программирование, № 4 "Советское радио", 1968.
7. Г.А. Миронов, Д.Э. Федотова. Методика построения полных контролирующих и диагностических тестов. Настоящий сборник.
8. З.А. Вадова. Полный тест АУ для машины БЭСМ-6. Настоящий сборник.
9. А.в. Горбунов. Некоторые вопросы автоматического построения систем диагностических тестов для ЭЦМ. Настоящий сборник.
10. А.К. Олефир. О логическом контроле вычислительных устройств. Настоящий сборник.