Л. Е. БАБАЯН

ИСПЫТАТЕЛЬНАЯ ПРОГРАММА ДЛЯ ПРОВЕРКИ АРИФМЕТИЧЕСКОГО УЗЛА И УСТРОЙСТВА УПРАВЛЕНИЯ МАШИНЫ «АРАГАЦ»

В настоящей статье описывается испытательная программа (мы будем ее кратко называть тестовой программой—ТП), используемая в ВЦ АН АрмССР и ЕрГУ для проверки правильности работы арифметического узла и устройства управления универсальной цифровой машины „Арагац". При этом под проверкой правильности работы АУ и УУ для арифметических операций мы понимаем обнаружение факта неправильной работы данной операции, но отнюдь не причины. Вместе с тем для логических операций по большей части удается установить также и причины сбоя. Данная ТП не носит диагностически характер, а преследует цель создать тяжелый режим работы машины.

В § 1 описывается код команд этой машины, но только в том объеме и в той последовательности, которые достаточны и более удобны для целей данной статьи. Такие же упрощения сделаны и при описании метода распределения разрядов при изображении команды. Здесь же описываются некоторые параметры машины.

§ 2 посвящен общему описанию ТП, а в § 3 приводится более подробное описание отдельных ее частей. Хотя описываемая ТП составлена для конкретной машины, автору представляется, что использованные здесь методы контроля могут быть с успехом применены при составлении ТП для других машин.

§ 1. Код команд машины „Арагац"

Машина „Арагац" является универсальной цифровой вычислительной машиной с плавающей запятой. Ячейка запоминающего устройства содержит 32 разряда, распределение которых для представления числа изображено на рис. 1.

Как видно из этого рисунка, разряды 41, 42 для представления числа не используются. Мантисса числа изображается в прямом, а порядок в обратном кодах. Число, хранящееся в ячейке z, мы будем обозначать через $\langle z \rangle$.

В некоторых операциях содержимое ячейки z рассматривается не как число с плавающей запятой, а как целое положительное число, у которого запятая фиксирована справа от крайнего правого (справа от младшего) разряда. Чтобы отличить этот случай от предыдущего,

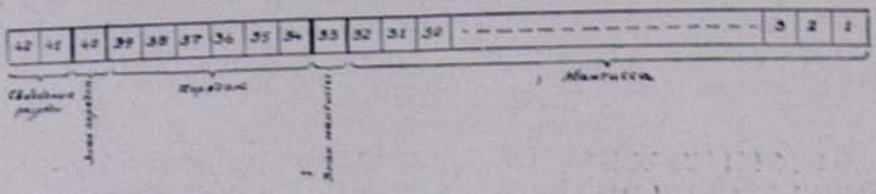

Рис. 1.

мы будем говорить о „наборе", хранящемся в ячейке z, и обозначать его символом $z_н$.

Распределение разрядов для представления команды изображено на рис. 2. Как видно из этого рисунка, на каждые из трех адре-

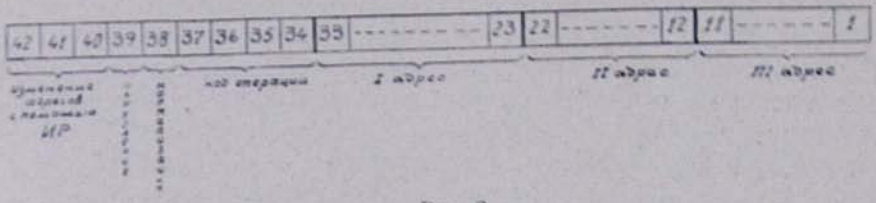

Рис. 2

сов отводится по 11 разрядов, а на код операции отводится четыре разряда. Остальные пять разрядов используются под различные модификации. При этом, если в коде арифметической операции в 38-м разряде стоит нуль, то после завершения операции к дополнительному разряду сумматора (см. ниже) прибавляется единица. Однако при операции сложения (вычитания) эта единица прибавляется только в том случае, если участвующие в операции числа имеют одинаковые (противоположные) знаки. Если в 38-м разряде стоит единица, то округление не производится. После выполнения округления *УУ* анализирует 39-й разряд и, если в нем стоит нуль, то результат нормализуется, а если единица, то не нормализуется. При выполнении неарифметических операций разряды 38, 39 используются для изображения кода операции вместе с разрядами 34—37.

Разряды 40—42 используются для изменения адресной части команды. Если в 42-м разряде стоит единица, то к первому адресу (1А) прибавляется содержимое индексного регистра (*ИР*), а если в 42-м разряде стоит нуль, то этого прибавления не происходит. Аналогично

Таблица

Код команд машины „Арагац"

Группа операций		Команда				Примечания
№ группы	наименование					
1	2	3				4
I	Групповые операции	→*ИР*	—	*n*	—	На *ИР* передается число *n*, записанное в двоичной системе в разрядах 22—12 кода команды
		→ *ИР* →	—	*n*	γ	В ячейку γ вписывается команда $\overline{\mid \rightarrow ИР \mid - \mid P \mid - \mid}$ (*P*— содержимое *ИР* до выполнения описываемой команды → *ИР* →), после чего на *ИР* передается число *n*, записанное в разрядах 22—12
		Гр	—	β	*m*	Число *m* записано в двоичной системе в разрядах 11—1 кода команды. Если содержимое *ИР* равно *m*, то управление передается к команде, следующей за данной, а в противном случае—к команде β. В обоих случаях после сравнений к содержимому *ИР* прибавляется единица
II	Операции над наборами	∨	α	β	γ	Логическое сложение $\alpha_н$ с $\beta_н$ и запись результата в γ
		∧	α	β	γ	Логическое умножение $\alpha_н$ на $\beta_н$ и запись результата в γ
		ц+	α	β	γ	Циклическое сложение. К $\alpha_н$ прибавляется $\beta_н$ и если в 42-м разряде возникает единица переноса, то она прибавляется к 1-му разряду. Результат записывается в ячейку γ
		ц—	α	β	γ	Циклическое вычитание. К $\alpha_н$ прибавляется обратный код $\beta_н$ (с передачей единицы переноса из 42-го разряда в 1-й) и результат записывается в γ. Если $\alpha_н > \beta_н$, то образуется их разность, в противном случае—их разность, но в обратном коде
		→	α	*n*	γ	Сдвиг $\alpha_н$ на *n* разрядов направо и запись получившегося набора в ячейку γ. Старшие *n* разрядов <γ> заполнены нулями. Число *n* записано в двоичной системе в разрядах 22—12 кода команды
		←	α	*n*	γ	Аналогично предыдущему, но $\alpha_н$ сдвигается налево и нулями заполняются младшие *n* разрядов ячейки γ

1	2	3				4
III	Условная передача управления (сравнение наборов)	$=$	α	β	γ	Если $\begin{cases} \alpha_н \neq \beta_н, \text{ то управление передается ячейке } \gamma \\ \alpha_н = \beta_н, \text{ то управление передается ячейке, следующей за данной} \end{cases}$
IV	Передача управления и останов	$пу$	—	β	—	Управление передается ячейке β
		$пу_в$	—	β	γ	Передача управления с возвратом. Управление передается ячейке β, а в ячейку γ вписывается команда $\boxed{пу \mid - \mid k+1 \mid -}$, где k номер ячейки, в которой хранится данная команда $пу_в$
		стоп	α	β	—	Машина останавливается. На регистры $P1$ и $P2$ выдаются числа, хранящиеся в ячейках α и β соответственно
V	Арифметические операции над числами	$+$	α	β	γ	$<\alpha>+<\beta> \to \gamma$
		$-$	α	β	γ	$<\alpha>-<\beta> \to \gamma$
		$\times$	α	β	γ	$<\alpha>\cdot<\beta> \to \gamma$
		$:$	α	β	γ	$<\alpha>:<\beta> \to \gamma$

используются разряды 40 и 41 для модификации второго (IIA) и третьего (IIIA) адресов соответственно.

Все рассматриваемые в данной работе операции приведены в таблице. Для большей ясности мы дадим к этой таблице некоторые пояснения, но предварительно кратко опишем устройство арифметического узла—АУ.

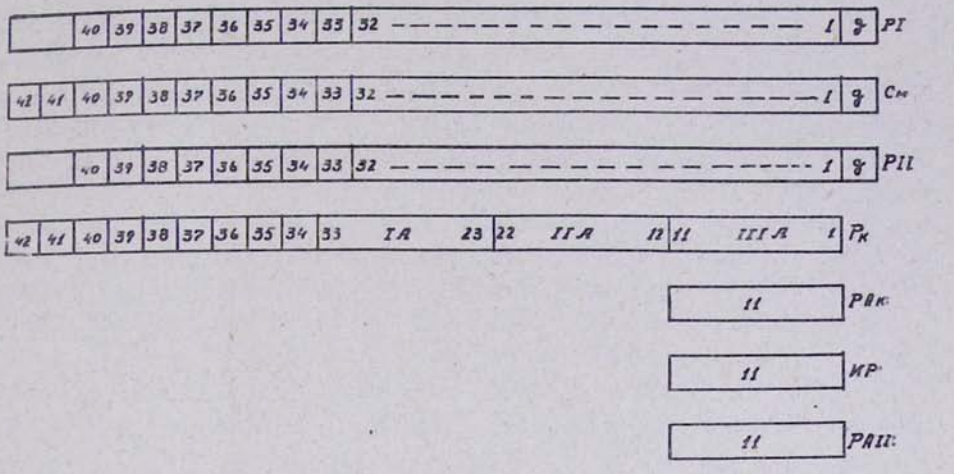

Рис. 3.

Арифметический узел машины (см. рис. 3) содержит три регистра $P1$, $P2$ и $C_м$, каждый из которых содержит по 42 разряда. Кроме того, сумматор имеет один дополнительный разряд, расположенный справа от крайнего правого разряда. Этот дополнительный разряд используется при округлении (см. § 3, V, 5). При выполнении операций групп II, III и V содержимое первого адреса выдается на P1, а второго—на P2. При этом, если операция арифметическая, то разряды 42 и 41 регистров P1 и P2 не используются. В сумматоре разряды 1—32 служат для изображения мантиссы, разряды 33, 34—знака мантиссы, разряды 35—40—порядка и разряды 41, 42—знака порядка. При операциях над наборами разряды $P1$, $P2$ и $C_м$ используются естественным образом.

Как хорошо известно, выполнение всех арифметических операций может быть сведено к сдвигам и сложениям. В машине „Арагац" все сложения выполняются на сумматоре. При этом сами сложения образуются из операции циклического сложения с помощью блокировки двух переносов: из T_{42} в T_1, из T_{33} в T_{34}. Таким образом, эта операция является как бы тем „кирпичом", из которого строятся многие операции.

Реализация циклов в программах может быть достигнута двумя способами. При одном способе выполняются обычные передачи исходного вида команды на ее рабочее место и последующая переадресация с помощью операций циклического сложения и вычитания. При втором методе в начале циклического куска ставится команда

$k+1$	$\rightarrow ИР$	—	n	—

(обычно число n берется равным нулю). Все команды в цикле, подлежащие переадресации, пишутся в их первоначальном виде (если $n=0$), а в конце цикла ставится команда

$k+s$	Γ_p	—	$k+2$	m

Здесь $m-n+1$ равно числу повторений цикла. Адреса тех команд, заключенных между командами $k+1$ и $k+s$, которые должны измениться при повторном прохождении этого куска, преобразуются с помощью *ИР*, как это описано немного выше.

Скажем еще несколько слов о характере работы операции $\rightarrow ИР \rightarrow$, которая в большинстве случаев используется для реализации „внутренних" циклов. Пусть в ячейке $k+1$ стоит команда

$k+1$	$\rightarrow ИР \rightarrow$	—	n	γ

и пусть к моменту выполнения этой команды в *ИР* хранится число d. Тогда при выполнении этой команды сначала в ячейку γ будет передана команда

γ	$\rightarrow ИР$	—	p	—

и только потом на *ИР* будет передано число n.

Остановимся кратко на устройстве управления—*УУ*. Оно содержит регистр команд *РК*, на котором хранится адрес выполняемой команды, и регистр адреса памяти РАП, на котором хранится адрес извлекаемого из памяти слова (числа, набора или команды). Все узлы *АУ* и *УУ* выведены на пульт управления в виде неоновых лампочек.

§ 2. Общая структура ТП

Тестовая программа рассчитана на обнаружение „не очень грубых" неисправностей арифметического узла и устройства управления машины „Арагац". Это значит, что до того как ставится эта программа, отдельные узлы машины уже проверены с помощью простых тестовых программ. В частности, предполагается, что уже проверены и исправно работают запоминающее устройство и ввод. Никаких точных критериев здесь, конечно, привести нельзя и целесообразность применения ТП решается дежурным инженером, исходя из опыта его работы с этими программами. Очень важно при этом, чтобы предварительно была проверена операция сравнения (группа III), которая особенно часто используется в ТП.

Последовательность проверки отдельных операций определялась исходя из двух принципов:

а) в порядке возрастания вероятности неисправности (сначала проверяются более надежные операции, а потом менее надежные);

б) в порядке необходимости использования одних операций при проверке других.

Само собой очевидно, что эти два принципа могут противоречить друг другу и поэтому окончательный порядок проверки операций был выработан экспериментально.

Вся ТП состоит из отдельных частей, проверяющих отдельные операции или отдельные группы операций. Каждая из этих частей оперирует двумя числами, хранящимися в ячейках, обозначаемых α и β. После того как прошла вся программа для одной пары значений этих чисел, в конце программы эти числа меняются, и вся программа повторяется. Само изменение чисел производится следующим образом. Сначала это число $<\alpha>$ образуется самой программой с помощью операции циклического сложения содержимого рабочих ячеек с содержимым одной ячейки, которое набирается произвольным образом от руки (с пульта). Тем самым достигается изменение чисел, с которым работает ТП при каждом новом применении ее (машина не может „привыкнуть" к одним и тем же числам). Кроме того, в формировании числа $<\alpha>$ участвуют некоторые специальные константы, такие как единица крайних разрядов мантиссы и порядка, единицы знаковых разрядов и т. п. При первом прохождении ТП числа $<\alpha>$ и $<\beta>$ равны между собой, затем $<\alpha>=-<\beta>$, затем $<\alpha>$ равно обратному коду $<\beta>$, после чего оба они уже совершенно произвольны. Через некоторое время опять повторяется то же самое, но только с новыми значеннями $<\alpha>$ и $<\beta>$.

Для простоты, при описании отдельного куска программы будем считать, что изменение чисел происходит в конце каждого куска. Каждый кусок будем помещать в ячейку $k+1$, $k+2, \cdots$

Код команд машины „Арагац" содержит несколько операций сравнения. Из них для проверки правильности работы операций используется операция условной передачи управления, приведенная в группе III таблицы.

В конце каждого куска ТП стоит команда сравнения, после которой стоит команда „стоп". В случае обнаружения неисправности машина останавливается именно на этом „стопе", а если неисправность не обнаружена, то управление передается к следующему куску. В первом и втором адресах команды „стоп" записаны адреса интересующих нас в данном месте чисел. Следовательно, в случае останова оба эти числа могут быть тут же (без специального обращения к памяти) прочитаны на пульте управления. Третий адрес команды „стоп" в коде команды не используется. В его разряды вписан порядковый номер (в тестовой программе) данного „стопа". В случае останова машины достаточно взглянуть на третий адрес РК, чтобы понять в каком месте программы произошел останов. Кроме того, адрес данного останова можно увидеть на счетчике адреса команд

(*СчАК*). Первый способ более нагляден, и, кроме того, неоновые лампочки на пульте управления не всегда правильно работают, так что эти два способа дополняют друг друга.

Нужно отметить, что после нескольких месяцев работы с ТП достаточно было понять на каком из „стопов" остановилась машина, чтобы довольно точно сказать, в каком месте машины произошел сбой.

УУ машины работает так, что при выполнении команды „стоп" на *СчАК* стоит номер ячейки, следующей за той командой, в которой хранится данная команда „стоп". В эту следующую ячейку вписана команда, передающая управление к началу данного куска. Достаточно простого нажатия на кнопку „пуск" и весь кусок повторяется заново. Таким способом, в частности, легко отличить случайный сбой от систематического.

Нам остается сделать несколько технических замечаний.

Вся программа, включая используемые ею рабочие ячейки, занимала ячейки с 0 по 1040 (в десятичной системе) из имеющихся в машине 2048 ячеек ОЗУ на ферритовых сердечниках. Это число 1040 мы обозначим буквой M и впредь ячейки с $(M+1)$-й по 2048-ю (включительно) будем называть свободной зоной ОЗУ.

Программа составлена так, что отдельные ее куски могут быть легко исключены, благодаря чему достигается сокращение времени ее работы.

В машине нет команды передачи числа, которая осуществляется с помощью операции логического сложения. Однако для наглядности мы передачу числа из ячейки α в ячейку β будем изображать с помощью команды

пч	α	—	β

Опыт более чем годовой эксплуатации описываемой ТП показал, что случаи необнаружения ею неисправностей в работе АУ и УУ машины очень редки. Как указывалось выше, ТП для арифметических операций (группа V) лишь устанавливает факт неправильного выполнения данной операции, но не указывает на неисправное место в машине. Правда, в силу достаточной детализации программы, в большинстве случаев место, на котором программа остановилась, служит косвенным указанием на место неисправности в машине. В особенности это верно по отношению к операции группы II.

§ 3. Проверка отдельных операций

В настоящем параграфе будут описаны конкретные приемы, с помощью которых проверяются отдельные операции, или отдельные группы операций. Как уже отмечалось, изложение ведется для каждого куска ТП. Из всей ТП описаны лишь те части, которые нам представлялись заслуживающими интереса. Много мелких, хотя и существенных, по

нашему мнению, деталей пришлось опустить ради простоты изложения. Весь параграф разбит на части, пронумерованные римскими цифрами. Эти части соответствуют разбиению команд на группы в таблице. Последовательность этих частей соответствует последовательности их расположения в ТП.

I. Групповая операция

1. Проверка операции „→ *ИР*“,

Чтобы установить правильность работы этой операции, выполняется несколько различных проверок. Во-первых, при этой операции записи по IIIA не производится. Поэтому в третий адрес вписывается номер некоторой ячейки l, и выполняется команда

→ *ИР*	—	n	l

после чего проверяется, изменилось ли содержимое ячейки l или нет. Если содержимое l изменилось, то проверяется 38-й разряд, который управляет записью по IIIA.

2. Проверка операции „→ *ИР* → “

Предыдущая операция „→ *ИР*“ должна была передать на *ИР* число n. В ТП выполняется команда

→ *ИР* →	—	0	l

в результате которой в ячейке l должен образоваться код команды

l	→ *ИР*	—	n	—

Этот код хранится в отдельной ячейке l, и после операции „→ *ИР* →“ идет сравнение l и | → *ИР* | — | n | — | на совпадение.

3. Проверка операции „Гр“

Для проверки этой операции в ее второй адрес пишется номер той ячейки, в которой хранится она сама

$k+1$	Гр	—	$k+1$	n

Неправильная работа этой операции (зацикливание — т. е. повторение больше, чем n раз) означает по большей части одно из двух:

а) на *ИР* неправильно образуется число повторений цикла (отметим, что это число образуется на сумматоре с последующей пере-

дачей на *ИР*) или неверно выполнено сравнение третьего адреса команды „Гр" с содержанием *ИР*;

б) в счетчик адреса команд не прибавляется единица (это прибавление выполняется в самом *СчАК*).

Последнее маловероятно, так как при неисправности подобного типа мы вряд ли дошли бы до этого места ТП. Какая из этих причин вызвала сбой, устанавливается инженером. При этом случай зацикливания легко устанавливается визуально: состояние всех регистров на пульте управления не меняется, машина как бы стоит.

Групповая операция может „пропустить" раньше нужного момента. Поэтому после того как она пропускает, содержимое *ИР* записывается (с помощью операции $\rightarrow ИР \rightarrow$) в заданную ячейку l_2, ее содержание сравнивается с известным числом.

4. Нам осталось проверить, как выполняется изменение адресов с помощью *ИР*, которое мы будем называть модификацией адресов. Кроме того, для наглядности, единицы, указывающие на переадресацию адресов, будем ставить не в разрядах 40—42, а в самих адресах:

	1		1		1		

Для проверки, на *ИР* передается число $M+1$. Далее выполняется команда

$k+1$	V	1	001	1	001	1	000

которая означает, что число из ячейки $M+2$ должно быть передано в ячейку $M+1$. Далее выполняется команда

$k+2$	V	0	$M+1$	0	$M+1$	0	l

т. е. число из ячейки $M+1$ передается в ячейку l.

После этого сравнивается содержимое ячеек l, $M+1$ и $M+2$. При этом адреса команды сравнения также формируются с помощью *ИР*. В случае совпадения I-й и II-й адреса команды $k+2$ переадресовываются (с помощью операции ц+, а адреса команды $k+1$ модифицируются с помощью индексного регистра). Следовательно, при вторичном прохождении этого куска выполнение команд $k+1$ и $k+2$ будет означать передачу числа из ячейки $M+3$ в ячейку $M+2$, а оттуда в ячейку l, и вновь $<l>$ будет сравниваться с $<M+2>$ и $<M+3>$. Отметим также, что само сравнение l с $<M+p>$ и $<M+p+1>$ выполняется в различных вариантах, так что удается проверить не только модификацию сразу трех адресов (как в команде $k+1$), но и в разных комбинациях. Например используются следующие команды сравнения:

$\neq$	1	000	1	001	0	P'
$\neq$	1	001	0	α	0	P

Для проверки модификации отдельно третьего адреса выполняются две команды (при которых одновременно проверяется операция циклического сложения):

ц+	0	d	0	δ	0	δ
ц+	0	γ	0	δ	1	000

где $<d>=1$.

После p-кратного повторения всей описываемой группы команд в ячейке $M+p$ должно образоваться число $<\gamma>+<\delta>$, т. е. $M+p$, что и проверяется. Здесь M — число, большее, чем число ячеек, занятых тестовой программой, включая рабочие. Таким образом проверяется ОЗУ от ячейки $M+1$ до ячейки 2047.

II. *Операция над наборами*

Операция сложения, в частности циклического сложения, входит составной частью во многие операции, как арифметические, так и операции сравнения чисел и наборов. Поэтому проверка этой операции выполняется в нескольких местах ТП. Заметим, что операция логического сложения уже многократно проверялась, так как с ее помощью осуществляется в машине передача чисел из одних ячеек в другие.

1. Для проверки циклического сложения строятся две команды

$k+1$	ц+	α	β	l_1
$k+2$	ц+	β	α	l_2

После выполнения команды $k+2$ сравниваются числа, хранящиеся в ячейках l_1 и l_2. Затем выполняются команды

$k+3$	ц−	l_1	β	l_3
$k+4$	ц−	l_2	α	l_4

и сравнивается содержимое ячеек l_3 с α и l_4 с β.

2. Для проверки логического умножения достаточно проверить равенство

$$a \wedge b = \overline{\bar{a} \wedge \bar{b}}.$$

Здесь черта над буквой a означает отрицание числа a, которое осуществляется с помощью операции циклического вычитания a из нуля. Параллельно, чтобы проверить работу каждого отдельного разряда сумматора, проводится циклическое сложение числа a с числом $\bar{a}$. Это сложение характерно тем, что не происходит ни одного пере-

носа, и, таким образом, сложение осуществляется фактически в каждом разряде отдельно. При этом в каждом разряде должна выработаться единица, что позволит проверить работу каждого разряда сумматора в отдельности.

3. Проверка операции сдвига ($\rightarrow$, $\leftarrow$).

Заметим, прежде всего, что неправильность работы этих операций встречается довольно часто и носит следующий характер. Во первых, во время сдвига осуществляется „круговой перенос" единиц из старшего разряда в младший, во вторых, число сдвигов оказывается неравным требуемому и, в третьих, правильность работы операции зависит от характера сдвигаемого набора. Проверка операций сдвига осуществляется с помощью программы

$k+1$	ц—	γ_2	γ	γ_1
$k+2$	$\wedge$	γ_1	α	l_1
$k+3$	$\rightarrow$	α	n	l_2
$k+4$	$\leftarrow$	l_2	n	l_3
$k+5$	$\neq$	l_1	l_3	p_1

Разъясним эту программу. Здесь в ячейке γ_2 хранится набор, состоящий из одних единиц, а в рабочую ячейку γ помещается число, n последних разрядов которого равны единице, и в остальных разрядах стоят нули. В результате выполнения команд $k+1$ и $k+2$ в ячейке l_1 образуется число с погашенными младшими n-разрядами. Такое же число должно получиться в ячейке l_3 в результате выполнения команд $k+3$ и $k+4$. Это проверяется командой $k+5$. Та же группа операций (но с заменой порядка операций $k+3$ и $k+4$) выполняется над числом, у которого единицы стоят уже в старших n-разрядах, а в остальных разрядах стоят нули. При повторном выполнении этого куска число n меняется (на единицу) от 1 до 42-х.

4. Для проверки пробега единицы переноса при выполнении операций циклического сложения и вычитания используется программа, в которой содержимое ячеек γ_1 и γ_2 такое же, как в предыдущем случае

$k'+1$	ц—	γ_2	γ	γ_1
$k'+2$	$\leftarrow$	γ	1	l_4
$k'+3$	ц—	γ_2	l_4	l_5
$k'+4$	ц+	γ	l_5	l_6
$k'+5$	$\neq$	γ_1	l_6	p_2

Заметим, что здесь при выполнении команды $k'+1$ единица переноса пробегает из n-го разряда в 42-й, затем передается в 1-й и вновь доходит до n-го разряда. Такого характера проверки, но с небольшими изменениями, осуществляются в командах $k'+2$, $k'+5$. Кроме того, числа, образующиеся в ячейках γ и $\alpha+7$, должны совпадать.

5. Проверка переноса единицы является в то же время и проверкой операций циклического сложения и вычитания. Для окончательной проверки этих операций в ТП создается режим, который, исходя из опыта, представляется наиболее тяжелым для машины. Этот режим заключается в том, что циклически складывается содержимое всех ячеек ОЗУ и полученная сумма записывается в ячейку γ_3. Затем $\langle\gamma_3\rangle$ передается в ячейку γ_2, после чего из $\langle\gamma_2\rangle$ циклически вычитается содержимое всех ячеек. В результате в ячейке γ_2 должно образоваться число, состоящее из одних единиц (по определению операции циклического вычитания; см. таблицу). Заметим, что в этом сложении и вычитании участвуют и те ячейки, в которых записана сама программа, за исключением двух последних ячеек, обозначенных нами γ_2 и γ_3. Интересно отметить, что эта процедура очень часто обнаруживает неправильность в работе машины, хотя операции ц+ и ц— уже многократно проверялись. По-видимому, объяснение этому следует искать в исключительном скородействии этих операций, в силу чего машина не выдерживает такого режима. Чтобы установить, какие из разрядов неисправны, та же программа повторяется, но в ячейке γ_2 накапливается циклическая сумма только i-х разрядов всех ячеек (при этом i последовательно пробегает все значения от 1 до 42), точно так же циклически складываются и вычитаются некоторые группы разрядов. Такая группировка разрядов позволила установить неисправный разряд. Заметим, что эта группировка разрядов занимает много времени и включается при обнаружении сбоя в предыдущем куске.

III. *Условная передача управления*

Все описанные выше проверки правильности работы отдельных операций основывались на операции сравнения наборов. Если эта операция работала неверно, то, по существу, все предыдущие проверки теряют свой смысл. Все же, исходя из опыта эксплуатации машины, нам кажется целесообразной специальная проверка этой операции именно в этом месте.

Для сравнения кодов на совпадение применяется следующий метод. Первое число подается на сумматор, затем к нему прибавляется обратный код второго числа, а затем к 33-му разряду прибавляется циклически единица. В случае совпадения сравниваемых кодов в результате этих действий должен получиться набор, у которого 33-й разряд равен единице, а все остальные разряды равны нулю. Правда, как легко видеть, точно такой же результат получится при сравнении набора, состоящего из одних единиц, с набором, со-

стоящим из одних нулей. Чтобы исключить этот случай, в машине предусмотрена специальная схема. Чтобы проверить эту схему, сначала сравниваются именно эти два набора. После этого рассматривается несколько пар совпадающих и несовпадающих наборов. Команды, идущие после команд сравнений, построены так, что, в случае неправильного срабатывания операций, машина останавливается. Заметим, что сбой машины на этих операциях встречался сравнительно редко. Чаще всего ошибка возникала при сравнении двух наборов, отличающихся только 33-м разрядом.

IV. *Проверка передачи управления*

В разделе I в каждую из ячеек свободного массива вписывалась команда останова, содержащая в IA номер той ячейки, в которой хранится данная команда:

$N+i$	стоп	$N+i$	0	0

Следовательно, если использованные до сих пор команды передачи управления (а они стоят после каждой команды останова) работали неверно и передавали управление к свободной зоне, то машина должна была остановиться. Однако, если команда передачи управления работала неверно, но при этом передавала управление к какому-либо месту ТП, то такой сбой мог бы быть не обнаружен. В большинстве случаев такая ошибка становится очень скоро очевидной. Нам остается проверить тот случай, когда в этой команде вообще не происходит передачи управления (команда пропускает), и управление передается к следующей команде. Для проверки этого случая используется следующая программа:

$k+1$	ny	—	2045	—
$k+2$	$\neq$	α	R	$k+5$
$k+3$	стоп	α	R	n_1
$k+4$	ny	—	$k+1$	—
$k+5$	ny_B	—	2043	2046
$k+6$	$\neq$	γ	R	$k+11$
$k+7$	стоп	γ	R	n_1+1
$k+8$	$\neq$	2046	u	$k+11$
$k+9$	стоп	2046	u	n_1+2
$k+10$	ny	—	N_1	—

где | *и* | *пу* | — | *k*+6 | — |

Кроме того, в ячейки 2043—2047 засылается еще одна вспомогательная программа:

2043	*пч*	γ	—	*R*
2044	*пу*	—	2046	—
2045	*пч*	α	—	*R*
2046	*пу*	—	*k*+2	—
2047	стоп	2047	—	—

Из вида этих программ ясно, что сравнение $k+2$ сработает правильно только в том случае, если правильно сработают передачи управления из ячейки $k+1$ в ячейку 2045. Если идущая после этого (в ячейке 2046) передача управления пропустит, то машина остановится. Если обе эти передачи управления сработают правильно, то сравнение $k+2$ передает управление в ячейку $k+5$. По этой команде управление передается в ячейку 2043 и, кроме того, в команде 2046 должна быть сформирована команда:

пу	—	*k*+6	—

После выполнения команды 2043 в ячейке R хранится число $\langle\gamma\rangle$, машина выполняет команду 2044. Если эта команда сработала верно, то мы приходим к команде 2046. Здесь возможны два случая: команда $k+5$ выработала в ячейке 2046 команду возврата к ячейке $k+6$, с помощью которой можно убедиться в том, что в ячейке R сейчас хранится число $\langle\gamma\rangle$. В противном случае надо полагать, что в ячейке 2046 хранится команда

пу	—	*k*+2	—

в результате которой будут сравниваться числа $\langle\alpha\rangle$ и $\langle R\rangle$. Но, так как сейчас в R хранится не $\langle\alpha\rangle$, а $\langle\gamma\rangle$, то машина остановится. Если команда 2046 пропускает, то машина вновь останавливается.

V. *Проверка арифметических операций*

Для описания метода проверок операций этой группы отметим, что числа в машине хранятся в следующем виде: порядок числа изображается в обратном коде в разрядах 34—40 (в 40-м разряде стоит знак порядка), а мантисса изображается в прямом коде в разрядах 33—1 (в 33-м разряде стоит знак мантиссы).

1. Проверка операции умножения. При выполнении операции умножения порядки должны складываться, а мантиссы умножаться. Сложение порядков производится в модифицированном обратном коде и, следовательно, единица переноса из 42-го разряда подается в 35-й.

Проверка операции умножения выполняется следующим образом. Из числа $<\alpha>$ и $<\beta>$ выделяются разряды 40—34. Затем у обоих чисел проверяется 40-й разряд и, если он равен единице, то в 41-й разряд данного числа вписывается (логическим сложением) единица, а если 40-й разряд равен нулю, то состояние 41-го разряда не изменяется. Затем разряды 42 и 33—1 этого числа заполняются единицами, после чего два полученных таким образом набора циклически складываются. Из полученной суммы извлекаются разряды 40—34, в которых, как можно видеть, должен быть записан порядок произведения перемножаемых чисел (чисел $<\alpha>$ и $<\beta>$). Для получения мантиссы произведения мы выделяем из первого сомножителя разряды 32—1. Затем второй сомножитель сдвигается разряд за разрядом направо и анализируется образующийся таким образом последний разряд. В зависимости от того нуль или не нуль этот разряд, мы либо не производим, либо производим логическое сложение выделенных разрядов мантиссы первого сомножителя с содержимым сумматора. В обоих случаях после каждого сдвига (в *См—е*) второго сомножителя выполняется сдвиг сумматора направо на один разряд. Иными словами, здесь программно реализуется схема умножения (с заменой ролей множимого и множителя). Знак мантиссы произведения вырабатывается с помощью анализа знаковых разрядов мантисс сомножителей. Полученные таким образом порядок, знак мантиссы и сама мантисса объединяются с помощью операции логического сложения. Образовавшееся число сравнивается с результатом умножения тех же чисел, но полученных непосредственно с помощью команды умножения (с блокировкой округления и нормализации).

2. Для проверки операции сложения выполняются следующие действия. Как легко видеть, для любого числа a справедливо равенство:

$$a = \frac{a}{2} + \frac{a}{2^2} + \cdots + \frac{a}{2^{31}} + \frac{a}{2^{32}} + \frac{a}{2^{32}}.$$

Каждое из этих слагаемых образуется из предыдущего умножением его на 1/2 (с блокировкой округления). Чтобы не потерять разряды, выходящие при таком умножении из разрядной сетки направо, мы при образовании слагаемого $a/2^i$ выделяем у числа a последние i разрядов и вписываем их в разряды $32—(32—i+1)$ некоторой ячейки τ_i. Содержимое ячеек τ_i складывается циклически.

В результате такого сложения во всех разрядах 32—1 должны образоваться нули, что и проверяется. Затем эта сумма сдвигается на 32 разряда и прибавляется (циклическим сложением) к сумме

$$\frac{a}{2}+\frac{a}{2^2}+\cdots+\frac{a}{2^{32}}+\frac{a}{2^{32}},$$

вычисленной с помощью операции обычного сложения с блокировкой округления. Образовавшееся число должно совпадать с исходным числом a. Для проверки операции сложения чисел с различными знаками выполняются те же действия, но вычисляется уже сумма

$$\frac{a}{2}+\left(-\frac{a}{2^2}\right)+\cdots+\left(-\frac{a}{2^{31}}\right)+\left(-\frac{a}{2^{32}}\right)+\left(-\frac{a}{2^{32}}\right),$$

которая должна равняться нулю.

3. Для проверки операции вычитания вычисляются и сравниваются результаты сложения произвольных чисел a и b с вычитанием числа $(-b)$ из числа a ($(-b)$ образуется умножением b на (-1)).

4. Операция деления проверяется сравнением с некоторым числом a результата, полученного умножением того же числа a на число b, и делением этого произведения на b. Однако здесь, в силу неточности операции деления и умножения, результат этого умножения и деления может отличаться от числа a на единицу последнего разряда. Поэтому сравнение числа a с результатом умножения и деления его на b выполняется с помощью операций сравнения чисел (а не сравнения наборов). Так как операция сравнения чисел еще не проверялась, то в ТП операция деления проверяется несколько позже (описание проверки операции сравнения чисел мы спускаем).

5. Проверка округления выполняется следующим образом. Пусть θ любая из операций $+$, $-$, $:$, $\times$. Рассмотрим две команды

θ	α	β	γ'
θ'	α	β	γ

в которых первая выполняется без округления, а вторая с округлением. Число β подбирается так, чтобы результаты, образовавшиеся в ячейках γ и γ', совпадали бы, если последний разряд α есть нуль, и были бы различны, если последний разряд α есть единица.

6. Для проверки нормализации составлена подпрограмма, которая выполняет нормализацию любого числа. Применяется эта подпрограмма следующим образом. Выполняются две команды

θ	α	β	γ'
θ'	α	β	γ

из которых первая работает с блокировкой нормализации, а вторая — без блокировки. Число, полученное в ячейке γ', нормализуется подпрограммой и результат сравнивается с числом, хранящимся в ячейке γ.

В заключение опишем еще две проверки, которые часто обнаруживали неисправность выполнения арифметических операций, хотя и не говорили, какие именно операции работают неверно.

Первая проверка заключается в следующем. Выполняется серия операций, результаты которых записываются в рабочие ячейки. С помощью групповых операций те же операции и с теми же числами повторяются вновь, но результаты записываются в другую группу рабочих ячеек. Затем числа, образовавшиеся в соответствующих ячейках, сравниваются между собой. Обычно здесь обнаруживается случайный сбой.

Одновременно по тождеству

$$\frac{a^3+b^3}{a+b}=a^2-ab+b^2$$

вычисляются отдельно правая и левая части и они сравниваются с точностью ε. Этот кусок программы независимо от остальной части многократно повторяется и при том каждый раз с новыми числами. Ввиду того, что описываемая проверка очень коротка, многократное повторение этого куска не сказывается на времени работы ТП.

Так же полезной оказалась проверка, использующая подпрограмму перевода чисел из двоичной системы счисления в десятичную. Эта программа была введена в ТП, однако после выработки очередной десятичной цифры проверялось, не больше ли эта цифра, чем цифра девять.

Автор приносит благодарность Т. М. Тер-Микаэляну за помощь оказанную при написании данной работы.

Լ. Բ. ԲԱԲԱՅԱՆ

«ԱՐԱԳԱԾ» ՄԵՔԵՆԱՅԻ ԹՎԱԲԱՆԱԿԱՆ և ՂԵԿԱՎԱՐՄԱՆ ԱՍԲՔԵՐԻ ՍՏՈՒԳՄԱՆ ՀԱՄԱՐ ՓՈՐՁԱՐԿՎՈՂ ԾՐԱԳԻՐ

Ա մ փ ո փ ո ւ մ

Այս աշխատանքում տրվում է «Արագած» մեքենայի թվաբանական և դեկավարման սարքերի աշխատանքի ճշտությունն ստուգող ծրագրի նկարագրությունը։

Ծրագիրը ստեղծում է մեքենայի համար բավական ծանր աշխատանքային ռեժիմ, որի դեպքում մեքենայի անկայուն էլեմենտների սխալ աշխատանքի ի հայտ գալու հավանականությունը մեծանում է։ Այս ծրագիրը հաջողությամբ շահագործվել է մոտ երկու տարի, որի ընթացքում համարյա չի պատահել, որ մեքենայի սխալ աշխատանքն ի հայտ չբերվի։