5

НЕКОТОРЫЕ ПРИЕМЫ ПРОГРАММИРОВАНИЯ

В настоящей главе с целью облегчения изложения принято следующее обозначение: $'a$ — содержимое ячейки с номером a.

Таким образом, запись $'a = A$ означает, что в ячейке с номером a находится число A. Приведенные ниже приемы рекомендуется использовать при программировании задач на машинах типа М-20; эти приемы используют некоторые специфические особенности машин типа М-20, отличающие их от других машин, и позволяют полнее использовать их возможности.

5.1. НЕКОТОРЫЕ ПРИЕМЫ ПРОГРАММИРОВАНИЯ ДЕЙСТВИЙ НАД ЧИСЛАМИ

Вычисление модуля числа *A*. Для вычисления модуля числа A используется операция вычитания модулей. Пусть $'a = A$. Тогда, выполнив операцию

0 63 *a* 0000 *c*,

получим $'c = |A|$. Вычитание модулей без нормализации используется для того, чтобы не нормализовать результат тогда, когда число A не нормализовано.

Присвоение числу *A* знака минус. Для присвоения числу A знака минус используется операция вычитания модулей.

Пусть $'a = A$; тогда, выполнив операцию

0 63 0000 *a*, *c*,

получим $'c = -|A|$. Вычитание модулей без нормализации применяется по причине, указанной в пункте «Вычисление модуля числа A».

Присвоение числу *A* знака числа *B*. Пусть $'a = A$; $'b = B$; $'d =$ 2 00 0000 0000 0000. После выполнения команд

Адрес	Команда			Пояснение
k+1	0 55 *d*	*b*	*c*	Выделение sign (*B*)
k+2	0 63 *a*	0000	*c*+1	|*A*|
k+3	0 15 *c*	*c*+1	*c*	sign (*B*) |*A*|

получим $'c = \text{sign}\,(B)\,|A|$.

Нормализация числа. Нормализация числа, находящегося в ячейке a, может быть выполнена одной из следующих команд:

0 01 *a* 0000 *c*

0 02 *a* 0000 *c*.

Выделение целой части положительного числа. Пусть $'a = A$; требуется в ячейку c послать $[A]$. Выделение целой части основано на сложении со специально подобранным числом $'d =$ 1 45 0000 0000 0000. При сложении число A сдвигается для выравнивания порядков так, что разряд единицы оказывается в младшем дополнительном разряде (разряды дробной части выталкиваются за разрядную сетку вправо), затем происходит нормализация; результатом будет $[A]$. Выделение целой части производится путем выполнения команды

0 21 *a d c*,

тогда $'c = [A]$.

Выделение дробной части положительного числа. Пусть $'a = A$; $'d =$ 1 45 0000 0000 0000; в ячейку c требуется послать $\{A\}$. Это делается так:

k + 1 0 21 *a d c*	$'c = [A]$
k + 2 0 02 *a c c*	$'c = \{A\}$.

Изменение знака порядка на противоположный. Пусть $'a = A$ и требуется изменить знак порядка числа A на противоположный.

В ячейке d должна быть запасена константа 1 77 0000 0000 0000. Для изменения знака порядка достаточно выполнить две команды

k + 1	0 15 *a*	*d c*
k + 2	0 06 0101	*c c*.

В ячейке c будет находиться число с той же мантиссой, что и у A, но с порядком противоположного знака.

Если число в ячейке a нормализовано, имеет не равную нулю мантиссу и нужно только изменить знак порядка на противоположный, а мантисса результата не используется, то это можно сделать, выполнив только одну команду:

0 04 *f a c*.

В ячейке f находится число 0 77 7777 7777 7777. Заметим, что такое число находится в ячейке 7710 МОЗУ при использовании ИС-2 или ИС-22.

Рассматриваемый способ основан на том, что в результате деления мантисс получится всегда единица или число, меньшее двух. Произойдет сдвиг на разряд вправо и к порядку результата прибавится единица. Следовательно, порядок результата будет вычисляться так:

$$77 - (100 + П_A) + 100 + 1 = 100 - П_A.$$

Выделение порядка, знака и признака числа без использования констант. Пусть $'a = A$; тогда, выполнив одну из команд:

$$0\ 33\ a\ \ a\ c$$
$$0\ 53\ 0000\ a\ c,$$

в ячейке c получим порядок, знак и признак числа A с мантиссой нуль.

Выделение мантиссы числа без использования констант. Пусть $'a = A$, тогда, выполнив одну из команд:

$$0\ 13\ 0000\ a\ c$$
$$0\ 73\ a\ \ a\ c,$$

в ячейке c получим мантиссу числа A.

Увеличение (уменьшение) числа в 2^n раз. Для этого достаточно по команде 06 или 46 увеличить (уменьшить) порядок на n единиц. Пусть $'a = A$. Тогда, выполнив одну из команд:

$$0\ 06\ 0100 \pm n\ a\ c$$
$$0\ 46\ 0100 \mp n\ a\ c,$$

получим $'c = 'a \cdot 2^{\pm n}$. Поэтому запасать двойку, как это иногда делают программисты, для вычисления $\frac{A}{2}$ не следует.

Отыскание ближайшего целого. Если $'a = A$ и нужно отыскать ближайшее целое, то достаточно выполнить команды $k+1$ и $k+2$:

$$k+1\ \ 0\ 41\ a\ d\ c$$
$$k+2\ \ 0\ 02\ c\ d\ c,$$

где $'d = 1\ 44\ 4000\ 0000\ 0000$.

5.2. НЕКОТОРЫЕ ПРИЕМЫ ПРОГРАММИРОВАНИЯ, ИСПОЛЬЗУЕМЫЕ ПРИ ПЕРЕДАЧАХ УПРАВЛЕНИЯ

Передача управления с одновременным изменением содержимого РА. Часто требуется перед передачей управления в некоторое место программы (в ячейку $k+1$) послать в регистр адреса некоторое число s. В зависимости от того, какой является передача управления (безусловной, условной по $\omega = 0$, условной по $\omega = 1$), используется одна из команд:

$$0\ 32\ 0000\ k+1\ s$$
$$0\ 71\ 0000\ k+1\ s$$
$$0\ 31\ 0000\ k+1\ s.$$

Заметим, что в случае выполнения этих операций происходит засылка в РА числа s вне зависимости от направления передачи управления (по $A2_{исп}$ или следующей команде).

Если нужно не посылать в РА число s, а увеличить содержимое РА на s, то в написанных выше командах нужно добавить признак третьего адреса.

Безусловная передача управления с запоминанием содержимого РА в некоторой ячейке. Безусловная передача управления с фиксацией в разрядах 24—13 некоторой ячейки содержимого РА может быть осуществлена по команде

$$4\ 16\ 0000\ k+1\ k+p.$$

После выполнения этой команды в ячейке $k+p$ будет находиться команда 0 16 0000 РА 0000; управление передается на $k+1$. В нужный момент содержимое РА может быть восстановлено по команде

$$0\ 72\ 0000\ k+p\ 0000.$$

Передачи управления по разным направлениям. Пусть требуется поочередно передавать управление в $k+s_1$, $k+s_2$, $k+s_3$, $k+s_4$ из одного и того же места программы. Для этого достаточно в программе написать одну команду $k+1$

$$k+1\ \ 0\ 56\ \ p+2\ \ k+s_1\ \ k+1$$

и запасти следующие четыре команды в ячейках $p+1$, $p+2$, $p+3$, $p+4$:

$$p+1\ \ 0\ \ 56\ \ p+2\ \ k+s_1\ \ k+1$$
$$p+2\ \ 0\ \ 56\ \ p+3\ \ k+s_2\ \ k+1$$
$$p+3\ \ 0\ \ 56\ \ p+4\ \ k+s_3\ \ k+1$$
$$p+4\ \ 0\ \ 56\ \ p+1\ \ k+s_4\ \ k+1.$$

Выполняя команду $k+1$, получим нужную последовательность передач управления.

Программа работает так: команда из ячейки $k+1$ передает управление на $k+s_1$ и заменяет себя командой из $p+2$, которая передает управление на $k+s_2$, и т. д.

Команда 0 56 $p+1$ $k+s_4$ $k+1$ передает управление на $k+s_4$ и восстанавливает в $k+1$ команду

$$0\ 56\ p+2\ k+s_1\ k+1.$$

Обход участка программы. Пусть требуется некоторый участок программы (с ячейки $k+1$ по ячейку $k+s$) обойти при первом просчете, а при всех последующих выполнять. Для этого достаточно в $k+0$ записать одну из команд:

а) 0 56 0000 $k+s+1$ $k+0$,

б) 0 16 $k+1$ $k+s+1$ $k+0$.

Если нужно выполнить участок программы (от $k+1$ до $k+s$) только один раз, а при последующих просчетах пропускать, то можно в $k+0$ записать команду

$$0\ 16\ \ k+s+1\ \ k+1\ \ k+0.$$

Эта команда при первом просчете передает управление на $k+1$ и заменяет себя командой

$$0\ 16\ 0000\ \ k+s+1\ \ 0000,$$

обеспечивающей обход участка от $k+1$ до $k+s$.

Передача управления с одновременной засылкой кода или очисткой ячейки. Не следует писать такую группу команд:

$k+1$ 0 00 0000 0000 $p+1$

$k+2$ 0 00 $a+1$ 0000 $p+3$

$k+3$ 0 56 0000 $k+s$ 0000.

Целесообразнее писать

$k+1$ 0 00 0000 0000 $p+1$

$k+2$ 0 56 $a+1$ $k+s$ $p+3$

или

$k+1$ 0 00 $a+1$ 0000 $p+3$

$k+2$ 0 56 0000 $k+s$ $p+1$.

5.3. ИЗМЕНЕНИЕ И ЗАПОМИНАНИЕ СОДЕРЖИМОГО РЕГИСТРА АДРЕСА

Запоминание содержимого РА без его изменения. Запомнить содержимое РА в ячейке $c+1$ можно одним из следующих способов:

а) 6 52 0000 0000 $c+1$,

б) 4 16 0000 $k+1$ $c+1$,

где $k+1$ — адрес следующей команды.

Увеличение (уменьшение) содержимого регистра адреса на m без передачи управления. Это можно сделать следующими путями:

а) 2 52 0000 m 0000,

б) 1 32 0000 $k+1$ m,

где $k+1$ — адрес следующей команды.

Для уменьшения содержимого РА на m нужно m заменить числом 10 000 — m (в восьмеричной системе); так, для уменьшения РА на 0002 следует писать 10 000 — 0002 = 7776.

Изменение содержимого РА с запоминанием его прежнего состояния. Если нужно в РА послать m и запомнить его прежнее состояние в $c+1$, то это можно сделать так:

а) 4 52 0000 m $c+1$,

б) 4 72 0000 $k+2$ $c+1$,

где $'(k+2) = 0\ 00\ 0000\ m\ 0000$.

Изменение содержимого РА. Изменение содержимого РА может быть выполнено так:

а) 0 52 0000 m 0000,

б) 0 32 0000 $k+1$ m,

где $k+1$ — адрес следующей команды,

в) 0 72 0000 $k+2$ 0000,

где $'(k+2) = 0\ 00\ 0000\ m\ 0000$.

Для изменения содержимого РА могут быть также использованы команды с кодами 11, 12, 31, 51, 71, например:

$$4\ 12\ 0000\ k+1\ m,$$

где $k+1$ — произвольный адрес (например, нуль).

В этом случае, так как условие $A1_{\text{исп}} \geqslant$ РА всегда выполняется, то управление передается следующей команде. При использовании команд с кодами 11, 31, 51, 71 следу-

ет учитывать значение ω, вырабатываемое в предыдущей операции.

Посылка в РА числа, полученного путем вычислений. Пусть число, которое нужно послать в РА, получается путем вычислений и записано в обычной форме (признак числа, порядок и мантисса). Для посылки его в РА нужно сдвинуть целую часть в разряды 24 — 13, а затем послать в РА.

Пусть $'a = A$, где A — число, целую часть которого нужно послать в РА. Засылку целой части A в РА можно осуществить в две команды:

0 61 a d c ($'d$ = 1 30 0000 0000 0000)
0 72 0000 c 0000
(ячейка c рабочая).

5.4. ПЕРЕХОД ОТ МАШИННЫХ ЧИСЕЛ К КОНСТАНТАМ ПЕРЕАДРЕСАЦИИ

Преобразование числа в константу переадресации и наоборот. Часто требуется осуществлять переход от обычных машинных целых чисел, участвующих в арифметических операциях, к числам, используемым при переадресациях команд.

Пример. Требуется переадресовать первый адрес команды, из ячейки 0021

0021	0	05	0036	0127	5000

на число 7, находящееся в ячейке 0300,

0300	1	03	7000	0000	0000

В ячейке 1000 запасаем константу·

1000	1	14	0000	0000	0000

Требуемая переадресация может быть выполнена двумя командами:

0 61 1000 0300 5001
0 13 0021 5001 0021

(ячейка 5001 рабочая).

Вообще, для образования вспомогательных кодов, используемых при переадресации в первом, втором, третьем

адресах, требуется соответственно сложение (по операции 61) обычного машинного числа с числами:

1 14 0000 0000 0000
1 30 0000 0000 0000
1 44 0000 0000 0000.

Иногда требуется переходить от чисел, используемых при переадресациях команд, к обычным машинным числам.

Пример. В ячейках 100 и 200 находятся коды:

0100	0	00	0000	0024	0000

0200	0	00	0000	0000	0012

Требуется в ячейку 0300 послать число

	* 0	00	0024:0012	0000	0000

Это может быть выполнено следующими командами:

k + 1 0 75 0006 0100 5000
k + 2 0 21 5000 0000 5000
k + 3 0 75 0007 0200 5001
k + 4 0 21 5001 0000 5001
k + 5 0 04 5000 5001 5000
k + 6 0 61 5000 0010 0300
k + 7 0 13 0000 0300 0300

В ячейках 0006, 0007, 0010 находятся следующие коды:

0006 1 30 0000 0000 0000
0007 1 44 0000 0000 0000
0010 1 14 0000 0000 0000.

Программа работает так: приформировывается порядок 1 30 к коду 0 00 0000 0024 0000, получается обычное целое ненормализованное машинное число 24. Это число нормализуется, а затем выполняется аналогичное действие над кодом 0 00 0000 0000 0012. Вычисляется частное полученных чисел и денормализуется, что превращает его в константу переадресации: эта константа имеет порядок 1 14; поэтому выделяется мантисса.

Вычисление произведения констант переадресации. При программировании иногда требуется находить произведение констант переадресации. Произведение этих констант может быть вычислено путем использования описанных выше приемов перехода от константы переадресации к числу и наоборот. Но, как правило, произведение констант перead-

ресации меньше 7777, а в этом случае произведение может быть найдено проще.

Пример. Найти произведение $m \times n (m \times n < 7777)$ и поместить его в третий адрес; константы переадресации находятся в ячейках $a + 1$ и $a + 2$:

$a+1$	0	00	0000	m	0000
$a+2$	0	00	0000	n	0000

Тогда произведение может быть найдено так:

$k + 1$ 0 13 d $a + 1$ $c + 1$
$k + 2$ 0 65 $c + 1$ $a + 2$ 0000
$k + 3$ 0 47 0000 0000 $c + 1$
$k + 4$ 0 54 0050 $c + 1$ $c + 1$
($'d$ = 1 00 0000 0000 0000).

В ячейке $c + 1$ будет находиться код

0 00 0000 0000 $m \times n$.

Программа работает так: одной из констант переадресации присваивается порядок 100, затем выполняется умножение; так как обе константы переадресации содержат значащие цифры только во вторых адресах, произведение $m \times n$ окажется в первом адресе младших 36 разрядов, откуда оно извлекается по команде с кодом 47.

Так как сумма порядков сомножителей равна 100, то порядок произведения (машинный) равен 00 и перед сдвигом в третий адрес не нужно выделять разряды мантиссы.

Если сумма номеров адресов, в которых находятся константы переадресации (в нашем примере $2 + 2 = 4$), больше трех, то результат находится в младших 36 разрядах, откуда его нужно извлекать по команде с кодом 47.

Если сумма номеров адресов меньше или равна трем, то результат находится в старших 36 разрядах и команды с кодом 47 писать не нужно.

Пример. Найти произведение $m \times n$ в третьем адресе:

$'a$ = 0 00 n 0000 0000
$'b$ = 0 00 0000 m 0000
$'d$ = 1 00 0000 0000 0000.

Это произведение может быть найдено двумя командами:

0 13 d a c.
0 65 b c c.

В ячейке c будет находиться код

0 00 0000 0000 $m \times n$.

Можно сдвигать сомножители до выполнения умножения так, чтобы их произведение получилось в нужном адресе старших 36 разрядов. Так, программа первого примера может быть такой:

$k + 1$ 0 54 0114 $a + 1$ $c + 1$
$k + 2$ 0 13 d $c + 1$ $c + 1$
$k + 3$ 0 65 $a + 2$ $c + 1$ $c + 1$.

5.5. ИСПОЛЬЗОВАНИЕ ИЗМЕНЕНИЯ СОДЕРЖИМОГО РЕГИСТРА АДРЕСА ДЛЯ ПЕРЕАДРЕСАЦИИ КОМАНД

Формирование исполнительных адресов (с помощью регистра адреса и признаков) позволяет во многих случаях обходиться без переадресации команд. Рассмотрим следующие приемы:

1. Использование изменения регистра адреса для формирования команд.

Пусть требуется образовать в ячейке c команду

0 15 a b b,

имея такие «заготовки»:

$'(d + 1)$ = 0 15 a 0000 0000
$'(d + 2)$ = 0 00 0000 b 0000.

Для формирования команды в ячейке c нужно было бы выполнить следующие операции:

0 54 0064 $d + 2$ c
0 13 c $d + 2$ c
0 13 $d+1$ c c

При использовании изменения РА команда может быть записана в ячейке c в следующем виде:

3 15 a 0000 0000.

В ячейке c — 1 нужно поместить команду

0 72 0000 $d + 2$ 0000.

2. Использование изменения регистра адреса при выборке чисел из последовательностей с различным шагом.

Пример. Пусть требуется вычислить p значений функции $z = f(x, y)$ для пар x_i и y_i, если значения x_i и y_i расположены в последовательностях ячеек с шагами n и m. Последовательности начинаются соответственно в ячейках a и b.

Рассмотрим несколько способов осуществления выборки аргументов для вычисления $f(x, y)$:

а)

Адрес	Команда
$k-1$	0 00 $d+4$ 0000 $k+3$
$k+0$	0 00 $d+3$ 0000 $k+2$
$k+1$	0 52 0000 0000 0000
$k+2$	0 00 a 0000 $c+1$
$k+3$	0 00 b 0000 $c+2$
$k+4$	0 13 $k+2$ $d+1$ $k+2$
$k+5$	0 13 $k+3$ $d+2$ $k+3$
.....	. .. .
$k+s+1$	1 12 $p-1$ $k+2$ 0001

$'(d+1) = 0\ 00\ n\ 0000\ 0000$
$'(d+2) = 0\ 00\ m\ 0000\ 0000$
$'(d+3) = 0\ 00\ a\ 0000\ c+1$
$'(d+4) = 0\ 00\ b\ 0000\ c+2.$

Программа построена так: организован цикл с использованием РА; при выполнении команд $k+2$ и $k+3$ осуществляются засылки x_i и y_i в стандартные ячейки $c+1$ и $c+2$; затем происходит переадресация команд засылки. Функция $z = f(x, y)$ вычисляется командами от $k+6$ до $k+s$; при этом предполагаем, что $'(c+1) = x_i$ и $'(c+2) = y_i$. Перед началом цикла вычислений команды $k+2$ и $k+3$ восстанавливаются.

Всего для обеспечения работы программы, вычисляющей z, требуется 8 команд, 4 константы и 2 рабочие ячейки.

б

Адрес	Команда
$k+2$	0 00 $d+2$ 0000 $k+4$
$k+3$	0 52 0000 0000 0000
$k+4$	0 00 a 0000 $c+1$
$k+5$	0 13 $k+4$ $d+1$ $k+4$
.....	. .. .
$k+s+1$	1 12 $(p-1)\,m$ $k+4$ m

$'(d+1) = 0\ 00\ n\ 0000\ 0000$
$'(d+2) = 0\ 00\ a\ 0000\ c+1$

Программа составлена так, что при вычислении $z = f(x, y)$ для выборки нужного значения y используется изменение содержимого регистра адреса. При выполнении команды $k+4$ осуществляется засылка очередного значения x, в ячейку $c+1$, а затем происходит переадресация этой засылки; z вычисляется командами от $k+6$ до $k+s$; адреса, относящиеся к y, должны быть снабжены признаками. Команда $k+s+1$ изменяет РА на m и используется как счетчик; команда $k+2$ служит для восстановления переадресуемой команды. Всего для обеспечения работы программы, вычисляющей z, требуется 5 команд, 2 константы и одна рабочая ячейка.

в)

Адрес	Команда
$k+2$	0 52 0000 0000 $c+2$
$k+3$	4 72 0000 $c+2$ $c+3$
$k+4$	4 00 a 0000 $c+1$
$k+5$	4 72 n $c+3$ $c+2$
.....	. .. .
$k+s+1$	1 12 $(p-1)\,m$ $k+3$ m

При работе программы команда $k+2$ очищает РА, а в ячейку $c+2$ посылает код 0 52 0000 0000 0000; команда $k+3$ посылает второй адрес $c+2$ в РА и запоминает старое значение РА в $c+3$; $k+4$ осуществляет засылку x_i в стандартную ячейку $c+1$; $k+5$ увеличивает второй адрес $c+2$ на n и посылает в РА содержимое $c+3$. Команды от $k+6$ для $k+s$ осуществляют вычисление $z = f(x, y)$, причем адреса, относящиеся к y, должны иметь признаки. Команда $k+s+1$ изменяет РА на m (для выборки y_i) и используется как счетчик.

Всего для обеспечения работы программы, вычисляющей z, требуется 5 команд и три рабочие ячейки.

Из приведенного примера следует, что использование вместо переадресации изменения содержимого регистра адреса дает хорошие результаты.

5.6. ПОЛУЧЕНИЕ КОНТРОЛЬНОЙ СУММЫ

При получении контрольной суммы используются следующие приемы:

1. Иногда для получения контрольной суммы программы поступают так: вводят программу, на сумматоре читают

полученную сумму, записывают ее, а затем пробивают на перфокарте. Время, расходуемое на чтение, запись и проверку, тратится непроизводительно.

Следует пользоваться перфокартой получения контрольной суммы, вводимой перед основной программой. Эта карта осуществляет фиктивный ввод программы и выдает ее контрольную сумму на перфокарту с признаком конца ввода:

0001		0 00 0000 0000 0000	
0002		0 30 0000 0003 0001	
0003		0 50 0200 1777 0001	
0004		0 70 0001 0000 0000	
0005		0 16 0000 0002 0000	
	1	2 06 0201 2004 0002	1

Команда 0002 производит фиктивный ввод программы и засылает в 0001 ее контрольную сумму. Команды 0003 и 0004 производят выдачу контрольной суммы, при этом содержимое 0001 записывается в последнюю ячейку буферного регистра*).

Так как запись производится с контролем, то в ячейку 0000 буферного регистра запишется контрольная сумма содержимого ячейки 0001, т. е. контрольная сумма программы.

Перфорация осуществляется всегда с нулевой ячейки буферного регистра, следовательно, на перфокарту будет выдана только контрольная сумма программы. Команда 0005 передает управление на ввод следующего массива. Заметим, что при выдаче контрольной суммы используются только ячейки 0001—0005, поэтому такая выдача может быть произведена при находящейся в МОЗУ другой программе, в которой указанные ячейки не заняты. Полученной перфокартой с контрольной суммой заменяют перфокарту с признаком конца в основной программе.

2. Перед контрольным суммированием программы и результатов для сравнения при повторении счета ячейка, в которой накапливается контрольная сумма, очищается.

Можно обойтись без очистки этой ячейки, написав следующие команды:

$k+1$	0 52 0000		0000	$c+1$
$k+2$	4 07 $a+1$		$c+1$	$c+1$
$k+3$	1 12 $n-1$		$k+2$	0001.

*) В БЭСМ-4 и М-20 адрес 1777 из команды 0003 автоматически воспримется как 0777. (*Прим. ред.*)

В ячейке $c+1$ накапливается контрольная сумма. Вместо очистки $c+1$ мы посылаем туда число 0 52 0000 0000 0000. Добавление этого числа к контрольной сумме будет происходить каждый раз при вычислении контрольных сумм и не повлияет на результат сравнения контрольных сумм.

3. При недостатке места в МОЗУ контрольная сумма может быть вычислена с помощью двух команд:

$k+1$	0	50	УЧ	0000	$a+n$
$k+2$	0	70	$a+1$	0000	$c+1$

путем записи на барабан или ленту, при этом расходуется больше машинного времени, чем при обычном суммировании.

4. При обычном контрольном суммировании величина суммы не зависит от порядка расположения кодов в МОЗУ, так как контрольное суммирование в смысле операции 07 коммутативно. Это свойство в ряде случаев создает неудобства. Так, если переставлены коды (например, из-за того, что неправильно сложены перфокарты), то установить это при помощи контрольного суммирования не удается.

Для получения контрольной суммы, зависящей от порядка, в котором расположены коды, может быть использована следующая программа:

Адрес	Команда	Пояснение
$k+1$	0 00 0000 0000 $c+1$	Очистка ячейки для суммы
$k+2$	0 52 0000 0000 0000	Очистка РА
$k+3$	0 07 $c+1$ $c+1$ $c+1$	Контрольное суммирование
$k+4$	2 07 $c+1$ $a+1$ $c+1$	
$k+5$	1 12 $n-1$ $k+3$ 0001	

После окончания работы программы в ячейке $c+1$ находится контрольная сумма массива, расположенного начиная с ячейки $a+1$ по ячейку $a+n$. Эта сумма зависит от порядка расположения кодов в МОЗУ, так как содержимое ячейки $a+1$ входит в сумму 2^{n-1} раз, содержимое ячейки $a+2$ входит в сумму 2^{n-2} раз; содержимое ячейки $a+n-1$ входит в сумму 2 раза, содержимое ячейки $a+n$ — один раз.

Контрольная сумма, подсчитанная таким способом, может быть сравнена с полученной заранее суммой и по-

зволяет выявить нарушение порядка расположения кодов. Такое суммирование целесообразно использовать для контроля вводимых с перфокарт программ. Его не имеет смысла применять для контроля обмена МОЗУ — МЗУ.

5.7. ВЫБОР ЧИСЕЛ ИЗ ПОСЛЕДОВАТЕЛЬНОСТЕЙ И ПЕРЕСЫЛКИ МАТЕРИАЛА

Выбор наибольшего из двух чисел и засылка его в некоторую ячейку. Пусть $'a = A$, $'b = B$. Требуется в ячейку c послать max (A, B). Это делается так:

$k+1$	0	02	a	b	0000
$k+2$	0	36	b	$k+4$	c
$k+3$	0	00	a	0000	c
$k+4$	.	.	.	.	.

Отыскание наибольшего (наименьшего) числа из ряда чисел. В ячейках $a+1$, $a+2$, ... , $a+n$ расположено n чисел. Требуется найти наибольшее и послать его в ячейку $c+1$.

Наибольшее из чисел может быть выбрано по следующей программе:

$k+1$	0	52	0000	0000	0000
$k+2$	4	00	$a+1$	0000	$c+1$
$k+3$	2	02	$c+1$	$a+2$	0000
$k+4$	1	11	$n-1$	$k+2$	0001
$k+5$	1	12	n	$k+3$	0000.

Команда $k+1$ очищает РА, $k+2$ засылает в $c+1$ число из ячейки $a+1$. При дальнейшей работе программы обращение к этой команде происходит только в том случае, если в $a+i$ оказалось число большее, чем то, которое находится в $c+1$.

Команда вычитания вырабатывает управляющий сигнал $\omega = 1$, если $'(a+i) > '(c+1)$, при этом управление передается на $k+2$. Если $'(a+i) \leqslant '(c+1)$, то после команды $k+4$ выполняется команда $k+5$ и управление передается на команду вычитания.

В программе число повторений цикла взято таким, что в последнем цикле происходит сравнение содержимого ячеек

$c+1$ и $a+n+1$. Последняя ячейка не принадлежит рассматриваемой последовательности, но каково бы ни было значение ω, в результате этого сравнения оно не приведет к изменению содержимого ячейки $c+1$. Лишнее выполнение цикла нужно для того, чтобы обеспечить пересылку наибольшего числа в $c+1$ в том случае, когда оно находится в $a+n$.

Для отыскания наименьшего числа, наибольшего по модулю или наименьшего по модулю, может быть использована аналогичная программа (в ней должна быть изменена только команда $k+3$).

Числа могут располагаться в ячейках, номера которых идут не подряд, а образуют арифметическую прогрессию с любым шагом; это изменит лишь второй адрес команды $k+3$ (вместо $a+2$ будет $a+n+1$, где n — шаг) и команды из $k+4$ и $k+5$.

Если требуется определить и номер ячейки, в которой находится наибольшее число, то это может быть сделано по следующей программе:

$k+1$	0	52	0000	0000	0000
$k+2$	6	52	$a+1$	0000	$c+2$
$k+3$	4	00	$a+1$	0000	$c+1$
$k+4$	2	02	$c+1$	$a+2$	0000
$k+5$	1	11	$n-1$	$k+2$	0001
$k+6$	1	12	n	$k+4$	0000.

Программа работает так же, как было описано выше. В нее включена команда $k+2$, фиксирующая во втором адресе ячейки $c+2$ номер ячейки, содержащей наибольшее число.

Пересылка n чисел, стоящих подряд. Пусть требуется переслать n чисел, стоящих в ячейках $a+1$, $a+2$, ... , $a+n$, в ячейки $b+1$, $b+2$, ... , $b+n$.

Если $b+1$ не лежит между $a+1$ и $a+n$, то пересылка выполняется так:

$k+1$	0	52	0000	0000	0000
$k+2$	5	00	$a+1$	0000	$b+1$
$k+3$	1	12	$n-1$	$k+2$	0001.

Если выполнено условие $a+1 < b+1 < a+n$, т. е. начало того массива, куда производится пересылка, лежит

между началом и концом массива, из которого происходит пересылка, то можно воспользоваться такой программой:

$k+1$ 0 52 0000 n 0000

$k+2$ 5 00 $a+0$ 0000 $b+0$

$k+3$ 1 32 0002 $k+2$ 7777.

При работе этой программы происходит вначале пересылка из ячейки $a+n$ в $b+n$, затем из $a+n-1$ в $b+n-1$ и т. д.

Упорядочение последовательности путем размещения чисел в порядке возрастания. Пусть в ячейках $a+1, a+2, \ldots, a+n$ находятся произвольные числа. Требуется расположить их в порядке возрастания. Для этого достаточно реализовать следующий алгоритм: сравниваем числа $'(a+1)$ и $'(a+2)$, если $'(a+1) \geqslant '(a+2)$, то числа меняются местами, затем делаем то же с парой $'(a+2)$ и $'(a+3)$ и продолжаем так до пары $'(a+n-1), '(a+n)$. После $n-1$ таких сравнений в ячейке $a+n$ оказывается наибольшее число. Те же сравнения применяются к последовательности чисел $'(a+1), '(a+2), \ldots, '(a+n-1)$, после чего в ячейке $a+n-1$ окажется второе по величине число из всей последовательности [может также быть $'(a+n-1) = '(a+n)$]. После обработки таким образом $n-1$-й последовательности [последней будет последовательность из двух чисел $'(a+1)$ и $'(a+2)$] все числа будут расположены в порядке возрастания. Заметим, что этот алгоритм не является самым экономным.

Программа, реализующая описанный выше алгоритм, может быть такой:

Адрес	Команда
$k+1$	0 52 0000 n 0000
$k+2$	4 52 0000 0000 $k+12$
$k+3$	6 02 $a+1$ $a+2$ 0000
$k+4$	4 36 $a+2$ $k+7$ $c+1$
$k+5$	5 00 $a+1$ 0000 $a+2$
$k+6$	1 00 $c+1$ 0000 $a+1$
$k+7$	6 52 0003 0000 $c+1$
$k+10$	0 33 $k+12$ $c+1$ 0000
$k+11$	1 71 0000 $k+3$ 0001
$k+12$	0 00 0000 0000 0000
$k+13$	1 32 0003 $k+2$ 7777

Внутренний цикл осуществляет сравнения $'(a+i)$ с $'(a+i+1)$ и, если это потребуется, меняет их местами. Внешний цикл уменьшает число сравнений на единицу после каждого выполнения внутреннего цикла.

5.8. ПОЛУЧЕНИЕ ПСЕВДОСЛУЧАЙНЫХ ЧИСЕЛ

При решении ряда задач используются случайные числа. Ниже приводится ряд программ получения псевдослучайных чисел.

Программы № 1 и № 2 получения псевдослучайных чисел, распределенных равномерно на отрезке от 0 до 1, предложены А. Н. Хватовым, В. В. Ганюшкиной и В. К. Томшиным.

Программа получения псевдослучайных чисел № 1:

Адрес	Команда
$k+1$	0 54 0107 $d+1$ $c+1$
$k+2$	0 15 $d+1$ $c+1$ $d+1$
$k+3$	0 54 0045 $d+1$ $c+1$
$k+4$	0 07 $c+1$ $d+1$ $d+1$
$k+5$	0 13 $d+2$ $d+1$ $c+1$
$k+6$	0 21 $c+1$ 0000 $c+1$

Очередное псевдослучайное число получается в ячейке $c+1$. В ячейках $d+1$ и $d+2$ перед началом счета должны быть размещены следующие коды:

$d+1$ 1 65 4573 2253 2570

$d+2$ 1 00 0000 0000 0000.

Программа получения псевдослучайных чисел № 2:

Адрес	Команда
$k+1$	0 54 0107 $d+1$ $c+1$
$k+2$	0 15 $d+1$ $c+1$ $c+1$
$k+3$	0 54 0045 $d+1$ $d+1$
$k+4$	0 07 $d+1$ $c+1$ $d+1$
$k+5$	0 13 $d+2$ $d+1$ $c+1$
$k+6$	0 21 $c+1$ 0000 $c+1$

Очередное псевдослучайное число получается в ячейке $c + 1$. В ячейках $d + 1$ и $d + 2$ перед началом работы должны находиться следующие коды:

$d + 1$ 3 64 3177 7143 1074

$d + 2$ 1 00 0000 0000 0000.

Длина отрезка апериодичности для программы № 1 более 39 миллионов, для программы № 2 более 10 миллионов.

Программы № 3 и № 4 получения псевдослучайных чисел, распределенных равномерно на отрезке от 0 до 1, предложены Л. С. Гуриным и В. И. Семеновым.

Программа получения псевдослучайных чисел № 3:

Адрес	Команда
$k+1$	0 65 $d+1$ $d+2$ $c+1$
$k+2$	0 47 0000 0000 $c+2$
$k+3$	0 55 $c+1$ $d+3$ $c+1$
$k+4$	0 55 $c+2$ $d+4$ $c+2$
$k+5$	0 13 $c+1$ $c+2$ $c+1$
$k+6$	0 13 $d+5$ $c+1$ $d+1$
$k+7$	0 02 $d+1$ 0000 $d+1$

Очередное псевдослучайное число хранится в ячейке $d + 1$, ячейки $c + 1$ и $c + 2$ — рабочие, в ячейках $d + 2$, $d + 3$, $d + 4$, $d + 5$ перед началом счета должны быть размещены следующие коды:

Адрес	Команда
$d+1$	1 00 7065 7651 1332
$d+2$	1 00 7065 7651 1332
$d+3$	0 00 0000 0077 7777
$d+4$	0 00 7777 7700 0000
$d+5$	1 00 0000 0000 0000

Программа получения псевдослучайных чисел № 4:

Адрес	Команда
$k+1$	0 54 0107 $d+1$ $c+1$
$k+2$	0 07 $c+1$ $d+1$ $c+1$
$k+3$	0 13 $d+2$ $c+1$ $c+1$
$k+4$	0 23 $c+1$ 0000 $d+1$

В ячейке $d + 1$ постоянно находится очередное псевдослучайное число. Перед началом работы в ячейках $d + 1$ и $d + 2$ должны находиться следующие коды:

Адрес	Команда
$d+1$	1 00 7263 1247 5314
$d+2$	1 00 0000 0000 0000

Для программы № 3 длина периода равна 92870 и длина отрезка апериодичности равна 220329, для программы № 4 — соответственно 203137 и 321629.

Программа № 5 получения псевдослучайных чисел, равномерно распределенных в интервале [0,1], предложена В. С. Губенко и А. И. Черкуновым. Она имеет вид:

Адрес	Команда
$k+1$	0 14 0115 $d+1$ $c+1$
$k+2$	0 15 $d+1$ $c+1$ $c+2$
$k+3$	0 55 $c+2$ $d+2$ $c+2$
$k+4$	0 14 0054 $c+2$ $c+2$
$k+5$	0 75 $c+1$ $c+2$ $d+1$
$k+6$	0 21 $d+1$ 0000 $c+1$
.....	. .. .
$d+1$	1 00 5042 7732 6410
$d+2$	0 00 7777 4000 0000

Очередное псевдослучайное число получается в ячейке $c + 1$.

Период датчика $2^{33} - 1 = 8\ 589\ 934\ 591$, т. е. датчик имеет практически бесконечный период.

Программа № 6, составленная В. С. Губенко и А. И. Черкуновым, позволяет получать псевдослучайные числа, распределенные по нормальному закону с $m = 0$, $\sigma = 1$.

Программа получения псевдослучайных чисел № 6:

Адрес	Команда	Пояснение
$k+1$	0 52 0000 0000 0000	
$k+2$	0 00 $d+6$ 0000 $c+1$	Засылка—2,5
$k+3$	0 14 0115 $d+1$ $c+2$	Датчик чисел x_l, равномерно распределенных в интервале [0,1]
$k+4$	0 15 $d+1$ $c+2$ $c+3$	
$k+5$	0 55 $c+3$ $d+2$ $c+3$	
$k+6$	0 14 0054 $c+3$ $c+3$	
$k+7$	0 75 $c+2$ $c+3$ $d+1$	
$k+10$	0 21 $d+1$ $c+1$ $c+1$	$y' = \sum_{l=1}^{5} x_l - 2{,}5$
$k+11$	1 12 0004 $k+3$ 0001	
$k+12$	0 05 $c+1$ $d+3$ $c+1$	$y = y' \sqrt{\frac{12}{5}}$
$k+13$	0 05 $c+1$ $c+1$ $c+2$	y^2
$k+14$	0 02 $c+2$ $d+5$ $c+2$	$y^2 - 3$
$k+15$	0 05 $c+2$ $d+4$ $c+2$	$0{,}01\ (y^2 - 3)$
$k+16$	0 05 $c+2$ $c+1$ $c+2$	$0{,}01\ y\ (y^2 - 3)$
$k+17$	0 01 $c+2$ $c+1$ $c+1$	$z = y + 0{,}01\ y\ (y^2 - 3)$
.....	. .. .	
$d+1$	1 00 5042 7732 6410	Исходное число датчика равномерных чисел
$d+2$	0 00 7777 4000 0000	
$d+3$	1 01 6144 5767 5040	$\sqrt{\frac{12}{5}}$
$d+4$	0 72 5075 3412 1727	0,01
$d+5$	1 02 6000 0000 0000	3,0
$d+6$	3 02 5000 0000 0000	—2,5

Очередное псевдослучайное число получается в ячейке $c+1$.

Программа № 7, предложенная В. С. Губенко и А. И. Черкуновым, выдает независимые псевдослучайные числа x, распределенные по закону $W(x) = xe^{-\frac{x^2}{2}}$.

Умножая эти числа на параметр σ, можно получить числа $y = \sigma x$, которые имеют распределение

$$W(y) = \frac{y}{\sigma} e^{-\frac{y^2}{2\sigma^2}}.$$

Программа № 7 получения псевдослучайных чисел, распределенных по закону Рэлея:

Адрес	Команда	Пояснение
$k+1$	0 14 0115 $d+1$ $c+1$	
$k+2$	0 15 $d+1$ $c+1$ $c+2$	
$k+3$	0 55 $c+2$ $d+2$ $c+2$	
$k+4$	0 14 0054 $c+2$ $c+2$	
$k+5$	0 75 $c+1$ $c+2$ $d+1$	
$k+6$	0 16 $k+7$ 7501 7610	Нормализация x и вычисление $\ln x$
$k+7$	0 21 $d+1$ 0004 $a+1$	
$k+10$	0 06 0101 $a+1$ $c+1$	
$k+11$	0 02 0000 $c+1$ $a+1$	
$k+12$	0 44 $c+1$ 0000 $a+1$	
.....	. .. .	
$d+1$	1 00 5042 7732 6410	
$d+2$	0 00 7777 4000 0000	

Следует обратить внимание на то, что в обращении к СП-0004 в коде второй строки написано 0 21. Это обеспечивает нормализацию x перед вычислением логарифма.

5.9. ВВОД «НА СЕБЯ»

Если вводимый массив должен быть помещен на том месте, где записана вводящая его команда, можно выполнить команду

0 56 a 0000 0000.

В ячейке a находится команда

0 10 b c 0000,

где b — адрес начала массива в МОЗУ; c — адрес передачи управления при несовпадении контрольных сумм.

При этом способе для размещения команды ввода не требуется ячейки (она заносится непосредственно на регистр команд), на КРА находится 0000. При совпадении контрольных сумм при вводе управление передается в ячейку 0001.

Этот прием можно использовать для ввода произвольного числа массивов, снабженных признаками конца ввo-

да, и для последующей передачи управления на начало счета. Пусть имеется программа:

```
0001    0 56 0002 0000 0000
0002    0 10 0001 0001 0000
     1  0 66 0003 0001 0000 1.
```

Если ввести эту программу по начальному вводу, то она будет вводить массивы до тех пор, пока какой-либо массив не затрет записанную в 0001 команду 0 56 0002 0000 0000. При таком вводе последний массив заносит передачу управления на начало счета в ячейку 0001. Использование большого числа массивов удобно при отладке, исправления выносятся в отдельные массивы, для которых легко подсчитать контрольную сумму. У основной программы сохраняется неизменяемая контрольная сумма.