# ЕС-2020

# МЕТОДИКА ВЫПОЛНЕНИЯ ОПЕРАЦИЙ

ТЕХНИЧЕСКОЕ ОПИСАНИЕ
Е13.055.001 ТО4

СОДЕРЖАНИЕ

# I. ВВЕДЕНИЕ

I.I. Данный документ содержит описание микропрограммы выборки команд (микропрограмма ВЫБОР), описание микропрограммы, обеспечивающей выход на прерывание по некорректной младшей тетраде кода операции (микропрограмма ПРНКО), а также описание микропрограмм, обеспечивающих выполнение:

- стандартной системы команд. Исключение составляют команды ЗАГРУЗКА ССП (см. EI3.055.00I T0I) и команды управления каналами (см. EI3.055.00I T02);
- команд десятичной арифметики;
- команд арифметики с плавающей запятой;
- команд средств защиты памяти;
- команд прямого управления.

С этого документа следует начинать изучение указанных выше микропрограмм.

I.2. Необходимые при изучении данного ТО диаграммы алгоритмов микропрограмм содержатся в документе EI3.055.00I ДI.

I.3. Микропрограммы, записанные на символическом языке, содержатся в документе EI3.055.00I ДСI.

I.4. Микропрограммы в машинных кодах содержатся в документе EI3.055.00I ДМI.

I.5. В приложении к данному ТО приведена сводная таблица времен выполнения команд, указанных в п. I.I.

# 2. КОМАНДЫ

2.I. Форматы команд. Имеются пять основных форматов команд, которые обозначаются форматными кодами: RR, RX, RS, SI и SS , выражающими местонахождение операндов и результата.

Форматный код РР обозначает операцию типа регистр-регистр; код RX - операцию типа регистр -память, при этом адрес памяти индексируется; код RS - операцию типа регистр-память (без индексации); код SI - операцию, когда один операнд находится в памяти, а другой - в самой команде; код SS - операцию типа память-память.

Структура основных форматов команд приведена на рис. I.

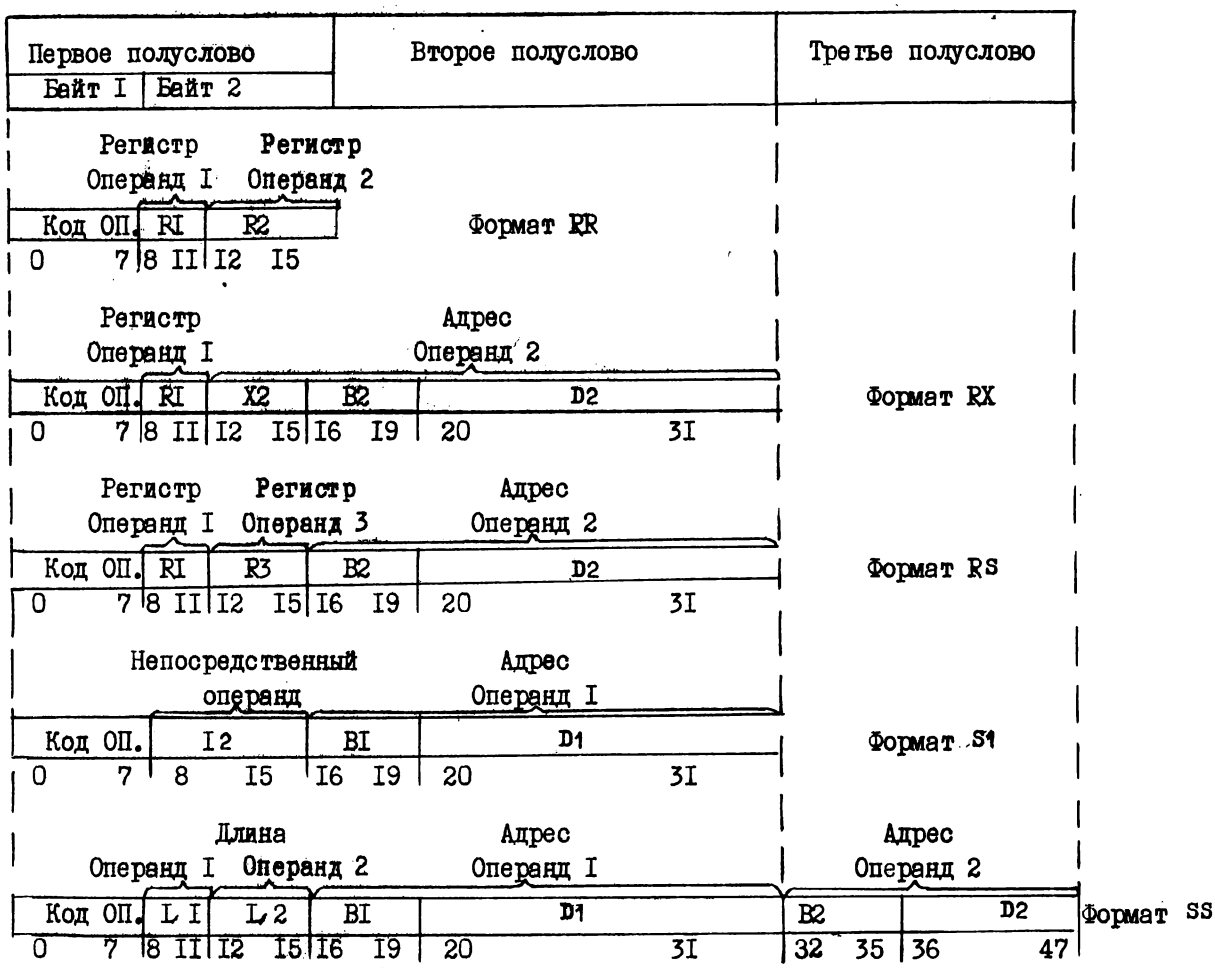



Рис. I. Пять основных форматов команд

В каждом формате первое полуслово состоит из двух частей. Первый байт содержит код о ции. Длина и формат команды определяются значением первых двух разрядов кода операции (КО соответствии с табл. I.

Таблиц

| КО разряды 0-I | Длина команды | Формат команды |
|---|---|---|
| 00 | Одно полуслово | RR |
| 0I | Два полуслова | RX |
| I0 | Два полуслова | RS или SI |
| II | Три полуслова | SS |

Второй байт используется либо в виде двух четырехразрядных полей, либо как одно восьмиразрядное поле. Этот байт может содержать следующую информацию:

- четырехразрядный номер регистра операндов ($R_1$, $R_2$ или $R_3$);
- четырехразрядный номер регистра индекса ($X_2$);
- четырехразрядную маску ($M_1$);
- четырехразрядный код длины операнда ($L_1$ или $L_2$);
- восьмиразрядный код длины операнда (L);
- восьмиразрядный байт, представляющий собой сам операнд ($I_2$), т.е. непосредственно данные.

В некоторых командах четырехразрядное поле или весь второй байт в первой половине слова игнорируется. Например, в команде ПРОВЕРИТЬ И УСТАНОВИТЬ.

Второе и третье полуслово имеют всегда один и тот же формат: четырехразрядный номер регистра базы ($B_1$ или $B_2$) и следующее за ним 12-разрядное смещение ($D_1$ или $D_2$).

Каждый формат в соответствии с разрядами 2 и 3 кода операции делится на четыре класса. Деление форматов на классы указано в разделе 7. Здесь же приведены времена выборки в микросекундах для всех классов команд.

2.2. Перечень команд. В табл. 2 приводится перечень команд, реализуемых микропрограммами, описанными в данном ТО. Названия команд перечисляются в алфавитном порядке. В графе "Идентификатор" указано имя (условное мнемоническое обозначение) микропрограммы, реализующей соответствующую команду.

Таблица 2

| Команда | | | | Микрокоманда |
|---|---|---|---|---|
| Название | Мнем. | Формат | КО | Идентификатор |
| ВЫПОЛНИТЬ | EX | RX | 44 | ИСИСЧ |
| ВЫЧИТАНИЕ | SR | RR | 1B | ARSR |
| ВЫЧИТАНИЕ | S | RX | 5B | ASRX |
| ВЫЧИТАНИЕ БЕЗ НОРМАЛИЗАЦИИ (ДЛИННОЕ) | SWR | RR | 2F | ASCFP |
| ВЫЧИТАНИЕ БЕЗ НОРМАЛИЗАЦИИ (ДЛИННОЕ) | SW | RX | 6F | ASCFP |
| ВЫЧИТАНИЕ БЕЗ НОРМАЛИЗАЦИИ (КОРОТКОЕ) | SUR | RR | 3F | ASCFP |
| ВЫЧИТАНИЕ БЕЗ НОРМАЛИЗАЦИИ (КОРОТКОЕ) | SU | RX | 7F | ASCFP |
| ВЫЧИТАНИЕ ДЕСЯТИЧНОЕ | SP | SS | FB | СЛОЖД |
| ВЫЧИТАНИЕ КОДОВ | SLR | RR | 1F | ARSR |
| ВЫЧИТАНИЕ КОДОВ | SL | RX | 5F | ASRX |
| ВЫЧИТАНИЕ ПОЛУСЛОВА | SH | RX | 4B | AHSH |
| ВЫЧИТАНИЕ С НОРМАЛИЗАЦИЕЙ (ДЛИННОЕ) | SDR | RR | 2B | ASCFP |
| ВЫЧИТАНИЕ С НОРМАЛИЗАЦИЕЙ (ДЛИННОЕ) | SD | RX | 6B | ASCFP |
| ВЫЧИТАНИЕ С НОРМАЛИЗАЦИЕЙ (КОРОТКОЕ) | SFR | RR | 3B | ASCFP |
| ВЫЧИТАНИЕ С НОРМАЛИЗАЦИЕЙ (КОРОТКОЕ) | SE | RX | 7B | ASCFP |
| ДЕЛЕНИЕ | DR | RR | 1D | ФДЕЛ |
| ДЕЛЕНИЕ | D | RX | 5D | ФДЕЛ |
| ДЕЛЕНИЕ (ДЛИННОЕ) | DDR | RR | 2D | DINOR, DFP |
| ДЕЛЕНИЕ (ДЛИННОЕ) | DD | RX | 6D | DINOR, DFP |
| ДЕЛЕНИЕ (КОРОТКОЕ) | DER | RR | 3D | DINOR, DFP |
| ДЕЛЕНИЕ (КОРОТКОЕ) | DE | RX | 7D | DINOR, DFP |
| ДЕЛЕНИЕ ДЕСЯТИЧНОЕ | DP | SS | FD | УМДЕД |
| ЗАГРУЗКА | LR | RR | 18 | RRI8 |
| ЗАГРУЗКА | L | RX | 58 | ≠5898 |
| ЗАГРУЗКА (ДЛИННАЯ) | LDR | RR | 28 | LFP |
| ЗАГРУЗКА (ДЛИННАЯ) | LD | RX | 68 | LFP |

Продолжение

| Команда | | | | Микрокоманда |
|---|---|---|---|---|
| Название | Мнем. | Формат | КО | Идентификатор |
| ЗАГРУЗКА (КОРОТКАЯ) | LER | RR | 38 | LFP |
| ЗАГРУЗКА (КОРОТКАЯ) | LE | RX | 78 | LFP |
| ЗАГРУЗКА АДРЕСА | LA | RX | 4I | RX4I |
| ЗАГРУЗКА ГРУППОВАЯ | LM | RS | 98 | ↓5898 |
| ЗАГРУЗКА ДОПОЛНЕНИЯ (ДЛИННАЯ) | LCDR | RR | 23 | LFP |
| ЗАГРУЗКА ДОПОЛНЕНИЯ (КОРОТКАЯ) | LCER | RR | 33 | LFP |
| ЗАГРУЗКА ДОПОЛНЕНИЯ | LCR | RR | I3 | RRI8 |
| ЗАГРУЗКА И ПРОВЕРКА | LTR | RR | I2 | RRI8 |
| ЗАГРУЗКА И ПРОВЕРКА (ДЛИННАЯ) | LTDR | RR | 22 | LFP |
| ЗАГРУЗКА И ПРОВЕРКА (КОРОТКАЯ) | LTER | RR | 32 | LFP |
| ЗАГРУЗКА ОТРИЦАТЕЛЬНАЯ (ДЛИННАЯ) | LNDR | RR | 2I | LFP |
| ЗАГРУЗКА ОТРИЦАТЕЛЬНАЯ | LNR | RR | II | RRI8 |
| ЗАГРУЗКА ОТРИЦАТЕЛЬНАЯ (КОРОТКАЯ) | LNER | RR | 3I | LFP |
| ЗАГРУЗКА ПОЛОЖИТЕЛЬНАЯ | LPR | RR | I0 | RRI8 |
| ЗАГРУЗКА ПОЛОЖИТЕЛЬНАЯ (ДЛИННАЯ) | LPDR | RR | 20 | LFP |
| ЗАГРУЗКА ПОЛОЖИТЕЛЬНАЯ (КОРОТКАЯ) | LPER | RR | 30 | LFP |
| ЗАГРУЗКА ПОЛУСЛОВА | LH | RX | 48 | RX |
| ЗАПИСЬ В ПАМЯТЬ | ST | RX | 50 | ST |
| ЗАПИСЬ В ПАМЯТЬ (ДЛИННАЯ) | STD | RX | 60 | SFP |
| ЗАПИСЬ В ПАМЯТЬ (КОРОТКАЯ) | STE | RX | 70 | SFP |
| ЗАПИСЬ В ПАМЯТЬ ГРУППОВАЯ | STM | RX | 90 | STMRS |
| ЗАПИСЬ В ПАМЯТЬ ПОЛУСЛОВА | STH | RX | 40 | STM |
| ЗАПИСЬ В ПАМЯТЬ СИМВОЛА | STC | RX | 42 | STCRX |
| И | NR | RR | I4 | NRORR |
| И | N | RX | 54 | NOX |
| И | NI | SI | 94 | NIOIX |
| И | NC | SS | D4 | SSД46 |
| ИЛИ | OR | RR | I6 | NRORR |
| ИЛИ | O | RX | 56 | NOX |
| ИЛИ | OI | SI | 96 | NIOIX |
| ИЛИ | OC | SS | D6 | SSД46 |
| ИСКЛЮЧАЮЩЕЕ ИЛИ | XR | RR | I7 | NRORR |
| ИСКЛЮЧАЮЩЕЕ ИЛИ | X | RX | 57 | NOX |
| ИСКЛЮЧАЮЩЕЕ ИЛИ | XI | SI | 97 | NIOIX |
| ИСКЛЮЧАЮЩЕЕ ИЛИ | XC | SS | D7 | SSД46 |
| ОБРАЩЕНИЕ К СУПЕРВИЗОРУ | SVC | RR | 0A | СУПЕР |
| ОТРЕДАКТИРОВАТЬ | E D | SS | DE | DEDF |
| ОТРЕДАКТИРОВАТЬ И ОТМЕТИТЬ | EDMK | SS | FE | DEDF |
| ПЕРЕКОДИРОВАТЬ | TR | SS | DC | TRTRT |
| ПЕРЕКОДИРОВАТЬ И ПРОВЕРИТЬ | TRT | SS | DD | TRTRT |
| ПЕРЕСЫЛКА | MVI | SI | 92 | SI92 |
| ПЕРЕСЫЛКА | MVC | SS | D2 | SS |
| ПЕРЕСЫЛКА ЗОН | MVZ | SS | D3 | D3 DI |
| ПЕРЕСЫЛКА СО СДВИГОМ | MVO | SS | FI | RINGO |
| ПЕРЕСЫЛКА ЦИФР | MVN | SS | DI | D3 DI |
| ПЕРЕХОД ПО ИНДЕКСУ БОЛЬШЕ | BXH | RS | 86 | ИНДЕК |

Продолжение

| Команда | | | | Микрокоманда |
|---|---|---|---|---|
| Название | Мнем. | Формат | КО | Идентификатор |
| ПЕРЕХОД ПО ИНДЕКСУ МЕНЬШЕ ИЛИ РАВНО | BXLE | RS | 87 | ИНДЕК |
| ПЕРЕХОД ПО СЧЕТЧИКУ | BCTR | RR | 06 | ИСИСЧ |
| ПЕРЕХОД ПО СЧЕТЧИКУ | BCT | RX | 46 | ИСИСЧ |
| ПЕРЕХОД С ВОЗВРАТОМ | BALR | RR | 05 | ВОЗВР |
| ПЕРЕХОД С ВОЗВРАТОМ | BAL | RX | 45 | ВОЗВР |
| ПОПОЛАМ (ДЛИННОЕ) | HDR | RR | 24 | HELP |
| ПОПОЛАМ (КОРОТКОЕ) | HER | RR | 34 | HELP |
| ПРЕОБРАЗОВАНИЕ В ДВОИЧНУЮ | CVB | RX | 4F | RX4F |
| ПРЕОБРАЗОВАНИЕ В ДЕСЯТИЧНУЮ | CVD | RX | 4E | RX4E |
| ПРОВЕРИТЬ И УСТАНОВИТЬ | TS | SI | 93 | ПРИУС |
| ПРОВЕРИТЬ ПО МАСКЕ | TM | SI | 9I | TMSI |
| ПРОЧИТАТЬ КЛЮЧ ПАМЯТИ | ISK | RR | 09 | КЛЮЧП |
| ПРОЧИТАТЬ СИМВОЛ | IC | RX | 43 | ICRX |
| ПРЯМАЯ ЗАПИСЬ | WRD | SI | 84 | ПРУПР |
| ПРЯМОЕ ЧТЕНИЕ | RDD | SI | 85 | ПРУПР |
| РАСПАКОВАТЬ | UNPK | SS | F3 | РАСК |
| СДВИГ ВЛЕВО | SLA | RS | 8B | SHIFT |
| СДВИГ ВЛЕВО ДВОЙНОЙ | SLDA | RS | 8F | SHIFT |
| СДВИГ ВЛЕВО ДВОЙНОЙ КОДОВ | SLDL | RS | 8D | SHIFT |
| СДВИГ ВЛЕВО КОДОВ | SLL | RS | 89 | SHIFT |
| СДВИГ ВПРАВО | SRA | RS | 8A | SHIFT |
| СДВИГ ВПРАВО ДВОЙНОЙ | SRDA | RS | 8E | SHIFT |
| СДВИГ ВПРАВО ДВОЙНОЙ КОДОВ | SRDL | RS | 8C | SHIFT |
| СДВИГ ВПРАВО КОДОВ | SRL | RS | 88 | SHIFT |
| СЛОЖЕНИЕ | AR | RR | IA | ARSR |
| СЛОЖЕНИЕ | A | RX | 5A | ASRX |
| СЛОЖЕНИЕ БЕЗ НОРМАЛИЗАЦИИ (ДЛИННОЕ) | AWR | RR | 2E | ASCFP |
| СЛОЖЕНИЕ БЕЗ НОРМАЛИЗАЦИИ (ДЛИННОЕ) | AW | RX | 6E | ASCFP |
| СЛОЖЕНИЕ БЕЗ НОРМАЛИЗАЦИИ (КОРОТКОЕ) | AUR | RR | 3E | ASCFP |
| СЛОЖЕНИЕ БЕЗ НОРМАЛИЗАЦИИ (КОРОТКОЕ) | AU | RX | 7E | ASCFP |
| СЛОЖЕНИЕ ДЕСЯТИЧНОЕ | AP | SS | FA | СЛОЖД |
| СЛОЖЕНИЕ КОДОВ | ALR | RR | IE | ARSR |
| СЛОЖЕНИЕ КОДОВ | AL | RX | 5E | ASRX |
| СЛОЖЕНИЕ ПОЛУСЛОВА | AH | RX | 4A | AHSH |
| СЛОЖЕНИЕ С НОРМАЛИЗАЦИЕЙ (ДЛИННОЕ) | ADR | RR | 2A | ASCFP |
| СЛОЖЕНИЕ С НОРМАЛИЗАЦИЕЙ (ДЛИННОЕ) | AD | RR | 6A | ASCFP |
| СЛОЖЕНИЕ С НОРМАЛИЗАЦИЕЙ (КОРОТКОЕ) | AER | RR | 3A | ASCFP |
| СЛОЖЕНИЕ С НОРМАЛИЗАЦИЕЙ (КОРОТКОЕ) | AE | RX | 7A | ASCFP |
| СЛОЖЕНИЕ С ОЧИСТКОЙ | ZAP | SS | F8 | F8SS |
| СРАВНЕНИЕ | CR | RR | I9 | CRRR |
| СРАВНЕНИЕ | C | RX | 59 | CRX |
| СРАВНЕНИЕ (ДЛИННОЕ) | CDR | RR | 29 | ASCFP |
| СРАВНЕНИЕ (ДЛИННОЕ) | CD | RX | 69 | ASCFP |
| СРАВНЕНИЕ (КОРОТКОЕ) | CER | RR | 39 | ASCFP |
| СРАВНЕНИЕ (КОРОТКОЕ) | CE | RX | 79 | ASCFP |
| СРАВНЕНИЕ ДЕСЯТИЧНОЕ | CP | SS | F9 | F9SS |

Продолжение

| Команда | | | | Микрокоманда |
|---|---|---|---|---|
| Название | Мнем. | Формат | КО | Идентификатор |
| СРАВНЕНИЕ КОДОВ | CLR | RR | I5 | CLRR |
| СРАВНЕНИЕ КОДОВ | CL | RX | 55 | CLRX |
| СРАВНЕНИЕ КОДОВ | CLI | SI | 95 | CLSI |
| СРАВНЕНИЕ КОДОВ | CLC | SS | D5 | CLCSS |
| СРАВНЕНИЕ ПОЛУСЛОВА | CH | RX | 49 | CHRX |
| УМНОЖЕНИЕ | MR | RR | IC | MRM |
| УМНОЖЕНИЕ | M | RX | 5C | MRM |
| УМНОЖЕНИЕ (ДЛИННОЕ) | MDR | RR | 2C | DINOR, MFP |
| УМНОЖЕНИЕ (ДЛИННОЕ) | MD | RX | 6C | DINOR, MFP |
| УМНОЖЕНИЕ (КОРОТКОЕ) | MER | RR | 3C | DINOR, MFP |
| УМНОЖЕНИЕ (КОРОТКОЕ) | ME | RX | 7C | DINOR, MFP |
| УМНОЖЕНИЕ ДЕСЯТИЧНОЕ | MP | SS | FC | УМДЕД |
| УМНОЖЕНИЕ ПОЛУСЛОВА | MH | RX | 4C | MRM |
| УПАКОВАТЬ | PACK | SS | F2 | PACK |
| УСЛОВНЫЙ ПЕРЕХОД | BCR | RR | 07 | УСЛОВ |
| УСЛОВНЫЙ ПЕРЕХОД | BC | RX | 47 | УСЛОВ |
| УСТАНОВИТЬ КЛЮЧ ПАМЯТИ | SSK | RR | 08 | КЛЮЧП |
| УСТАНОВИТЬ МАСКУ ПРОГРАММЫ | SPM | RR | 04 | МПРОГ |
| УСТАНОВИТЬ МАСКУ СИСТЕМЫ | SSM | SI | 80 | МСИСТ |

## 3. ПРЕДСТАВЛЕНИЕ ЧИСЕЛ

3.I. Число с фиксированной запятой. Числа с фиксированной запятой имеют формат фиксированной длины, в котором один разряд отводится под знак, а последующие разряды образуют поле целой части числа. Таким образом, только целые числа имеют представление в машине. Когда число находится в одном из общих регистров, его целая часть имеет 3I разряд, а само число занимает все 32 разряда регистра (рис. 2). Некоторые операции (умножение, деление, сдвиг) могут выполняться над 64-разрядными числами, целая часть которых имеет 63 разряда. Такие числа помещаются в два смежных общих регистра, а при адресации указывается общий регистр с четным адресом, который является левым регистром пары. В этом случае знаковый разряд регистра, имеющего нечетный адрес, используется в целой части числа (рис. 3).

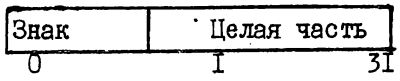



Рис. 2. Число с фиксированной запятой длиной в слово

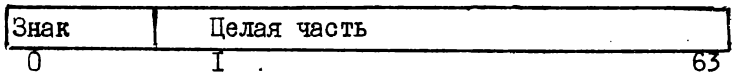



Рис. 3. Число с фиксированной запятой длиной в двойное слово

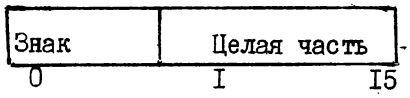



Рис. 4. Число с фиксированной запятой длиной в полуслово

Числа с фиксированной запятой, находящиеся в основной памяти, представляют собой 32-разрядные слова или 16-разрядные полуслова (рис. 4), поле целой части которых соответственно равно 31 или 15 разрядам. Данные различной длины должны помещаться в память в соответствии с границами, установленными для каждого случая, т.е. операнды, являющиеся обычными словами или полусловами, должны иметь адреса, в которых соответственно два или один младших разряда равны нулю.

Когда операнд длиной в полуслово выбирается из основной памяти, он превращается в операнд длиной в слово и участвует в операции как операнд длиной в слово.

Положительные числа представляются в прямом коде со знаковым разрядом, равным нулю. Отрицательные числа представляются в дополнительном коде со знаковым разрядом, равным единице. Дополнительный код числа получается инвертированием каждого разряда числа с последующим прибавлением единицы к младшему разряду. Такой способ представления чисел позволяет рассматривать любое число как младшие разряды неопределенно длинного числа. Когда число положительно, все разряды левее старшего значащего разряда числа, в том числе и знаковый разряд, равны нулю. Когда число отрицательно, все эти разряды, включая знаковый разряд, равны единице. Поэтому, при необходимости увеличить длину операнда в сторону старших разрядов, увеличение достигается расширением поля целой части числа в сторону старших разрядов и присвоением каждому новому разряду значения, равного знаковому разряду исходного числа.

При представлении чисел в дополнительном коде отсутствует отрицательный нуль. Наибольшее положительное число состоит из целой части, все разряды которой равны единице, и знакового разряда, равного нулю. Например, наибольшее положительное 16-разрядное число равно $2^{15} - 1 =$ = 32767 (десятичное) и имеет вид:

0111111111111111

Наибольшее отрицательное число состоит из целой части, все разряды которой равны нулю, и знакового разряда, равного единице. Например, наибольшее отрицательное 16-разрядное число равно $-2^{15} = -32768$ (десятичное) и имеет вид:

1000000000000000

Можно считать, что отрицательное число представляет собой сумму целой части этого числа, рассматриваемой как положительное число, и наибольшего отрицательного числа. Поэтому, если при сдвиге вправо отрицательных чисел результат округляется, округление производится в сторону $-\infty$, а не в сторону нуля.

Дополнение наибольшего отрицательного числа не может быть представлено. Поэтому, когда в результате некоторой операции в процессоре должно получиться дополнение наибольшего отрицательного числа, число остается без изменения и формируется сигнал переполнения в операции с фиксированной запятой.

3.2. Десятичное число. Только целые десятичные числа имеют представление в машине. Числа представляются в прямом коде со знаком плюс или минус и выравнены по правым концам соответствующих полей. Цифры 0-9 имеют двоичные коды 0000-1001. Коды 1010-1111 используются только как коды знака, причем коды 1010, 1100, 1110 и 1111 рассматриваются как плюс, а коды 1011 и 1101 - как минус. Коды 0000-1001 недопустимы в качестве кодов знака.

Десятичные числа могут быть как в упакованном формате (рис. 5), так и в формате с зоной (рис. 6).

В упакованном формате две десятичные цифры расположены рядом в одном байте. Исключение составляет самый правый байт, в котором справа от цифры помещен знак.

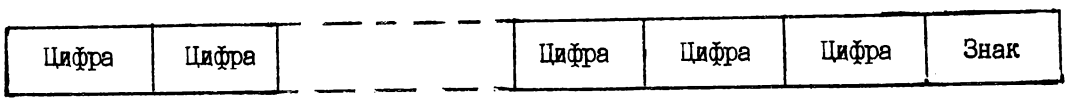



Рис. 5. Упакованное десятичное число

В формате с зоной младшие четыре разряда байта, числовые разряды, заняты десятичной цифрой. Старшие четыре разряда называются зоной; исключение составляет самый правый байт поля, в котором место зоны занято знаком.

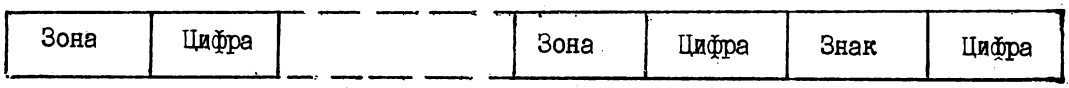



Рис. 6. Десятичное число в формате с зоной

Коды знака и зоны, получаемые в результате операции, различаются для двоичного кода (ДКОИ) и восьми-битного кода для обмена информацией (КОИ-8). Выбор одного из двух кодов определяется разрядом I2 ССП (см. раздел 5). Если в разряде I2 - нуль, вырабатываются коды, соответствующие ДКОИ: плюс - IIOO, минус - IIOI и зона - IIII. Если в разряде I2 - единица, вырабатываются коды, соответствующие КОИ-8, а именно: плюс - IOIO, минус - IOII и зона - OIOI.

Десятичные числа располагаются в основной памяти и адресуются адресом самого левого байта поля, которое они занимают. Длина поля может быть от одного до I6 восьмиразрядных байтов. Операнды одной команды могут иметь разную длину.

3.3. Число с плавающей запятой. Данные в операциях с плавающей запятой имеют формат фиксированной длины, который может быть или коротким, длиной в полное слово (рис. 7), или длинным, длиной в двойное слово (рис. 8).

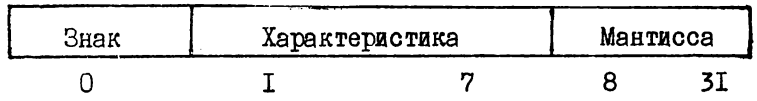



Рис. 7. Короткое число с плавающей запятой

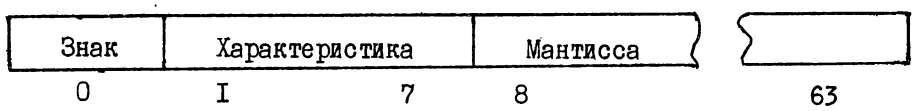



Рис. 8. Длинное число с плавающей запятой

Нулевой разряд в любом формате является знаковым. Следующие семь разрядов заняты характеристикой. Мантисса выражается шестнадцатиричными цифрами. В поле мантиссы может находиться либо шесть, либо четырнадцать шестнадцатиричных цифр.

Считается, что запятая в мантиссе сдвинута влево от старшей цифры на 64 разряда. Для получения соответствующей величины числа с плавающей запятой мантисса умножается на число I6 в степени, равной порядку. Порядок для обоих форматов указывается с помощью характеристики, которая находится в разрядах с первого по седьмой и изменяется от 0 до I27, что соответствует изменению порядка от -64 до +63.

Как для положительных, так и для отрицательных величин мантисса выражается в прямом коде. Различие в знаке отражается на значении знакового разряда. Нулевое состояние знакового разряда соответствует положительным числам, единичное состояние - отрицательным.

Число с нулевой характеристикой, нулевой мантиссой и положительным знаком называется истинным нулем.

Нормализованное число с плавающей запятой имеет отличную от нуля старшую шестнадцатиричную цифру мантиссы. Три старших двоичных разряда нормализованного числа могут быть нулями.

Диапазон абсолютных величин (M) представляемых нормализованных чисел с плавающей запятой составляет:

для коротких чисел

$$16^{-65} \leqslant M \leqslant (1-16^{-6})\ 16^{63};$$

для длинных чисел

$$16^{-65} \leqslant M \leqslant (1-16^{-14})\ 16^{63},$$

или приблизительно $2{,}4 \cdot 10^{-78} \leqslant M \leqslant 7{,}2 \cdot 10^{75}$ для обоих случаев.

Число с плавающей запятой может находиться или в регистре плавающей запятой, или в основной памяти. Короткое число с плавающей запятой занимает старшие 32 разряда регистра. Если числа располагаются в памяти, то они должны быть расположены в целочисленных границах для слов или двойных слов.

## 4. ЛОГИЧЕСКАЯ ИНФОРМАЦИЯ

4.I. Логическая информация находится в общих регистрах, в основной памяти или берется из команды. Это может быть обычное слово, двойное слово, поле переменной длины или один символ. Во всех командах, кроме команд УПАКОВАТЬ и РАСПАКОВАТЬ, оба операнда, участвующие в операции, имеют одну и ту же длину, у них нет никакой внутренней структуры и допустимы все комбинации разрядов.

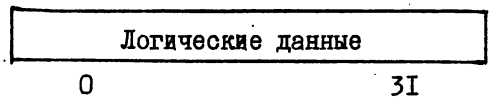



Рис. 9. Логическая информация фиксированной длины

4.2. В общих регистрах данные обычно занимают все 32 разряда (рис. 9). Все разряды, в том числе знаковый, обрабатываются одинаково. В некоторых операциях участвуют только младшие восемь разрядов регистра, а остальные 24 разряда не меняются. В некоторых операциях сдвига участвуют 64 разряда, входящие в состав пары регистров, четного и нечетного. Команда ЗАГРУЗКА АДРЕСА помещает 24-разрядный адрес в общий регистр. Старшие восемь разрядов регистра устанавливаются в нуль.

4.3. В операциях "память-регистр" данные в памяти занимают 32-разрядное слово или восьмиразрядный байт. Слово должно располагаться в памяти, начиная с границы слова, т.е. в двух младших разрядах адреса слова должны быть нули.

Операции, перемещающие данные непосредственно из команд в память, работают с одним байтом. Только один байт берется из команды и только один байт в памяти участвует в операции.

4.4. Большая часть логической информации состоит из кодов графических символов. Эта информация имеет формат переменной длины и может состоять максимально из 256 байтов (рис. I0).


Рис. I0. Логическая информация переменной длины

Данные переменной длины могут начинаться с любого байта и адресуются самым левым байтом. Поля операндов могут перекрываться. Любое перекрытие полей является допустимым.

## 5. ИНФОРМАЦИЯ О СОСТОЯНИИ СИСТЕМЫ

5.I. Информация о состоянии системы содержится в слове состояния программы (ССП). Формат ССП изображен на рис. II.

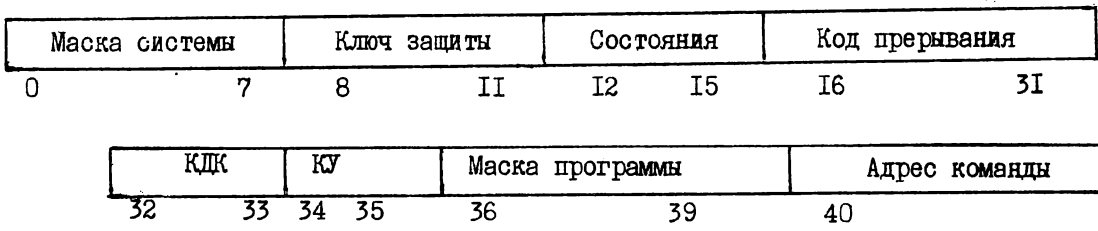



Рис. II. Формат ССП

Отдельные поля ССП имеют следующее значение.

5.I.I. Маска системы. Разряды 0–7 ССП связаны с каналами ввода-вывода и внешними сигналами. Назначение каждого разряда указано в табл. 3. Когда разряд маски равен единице, соответствующий объект может вызвать прерывание работы вычислительного устройства (ВЧУ). Если разряд маски равен нулю, соответствующий объект не может прервать работу ВЧУ; прерывания при этом хранятся в ожидании обработки.

Таблица 3

| Разряд маски системы | Источник прерывания |
|---|---|
| 0 | Мультиплексный канал |
| I | Селекторный канал I |
| 2 | Селекторный канал 2 |
| 3 | Селекторный канал 3 |
| 4 | Селекторный канал 4 |
| 5 | Селекторный канал 5 |
| 6 | Селекторный канал 6 |
| 7 | Таймер |
| 7 | Кнопка ПРЕРЫВАНИЕ на пульте управления |
| 7 | Внешний сигнал |

5.I.2. Ключ защиты. Разряды 8–II ССП являются ключом защиты для ВЧУ. При обращении к памяти этот ключ сравнивается с ключом памяти. Если средства защиты не установлены в данной машине, то разряды 8–II при загрузке должны быть нулевыми.

5.I.3. КОИ-8. Если I2-й разряд ССП равен единице, десятичный результат выдается в восьмибитном коде (КОИ-8). Если I2-й разряд ССП равен нулю, используется двоичный код для обмена информацией (ДКОИ).

5.I.4. Маска сбоя машины. Если I3-й разряд ССП равен единице, то при обнаружении ошибки схемами контроля машины происходит прерывание, выдается внешний сигнал о сбое и производятся диагностические процедуры. Если I3-й разряд равен нулю, ВЧУ замаскирован для прерываний по машинному сбою. В этом случае никаких сигналов, связанных со сбоем, не выдается, а диагностические процедуры не выполняются. Само прерывание теряется.

5.I.5. Состояние "ожидания". Если I4-й разряд ССП равен единице, ВЧУ находится в состоянии "ожидание". Если I4-й разряд ССП равен нулю, ВЧУ находится в состоянии "счет". Если ВЧУ находится в состоянии "ожидание", команды не выполняются и соответствующие обращения к памяти отсутствуют. Если ВЧУ находится в состоянии "счет", происходит нормальная выборка и выполнение команд.

5.I.6. Состояние "задача". Если I5-й разряд ССП равен единице, ВЧУ находится в состоянии "задача". Если I5-й разряд ССП равен нулю, ВЧУ находится в состоянии "супервизор".

В состоянии "задача" все команды управления каналами, средств защиты памяти и прямого управления, а также команды ЗАГРУЗКА ССП, УСТАНОВИТЬ МАСКУ СИСТЕМЫ, ДИАГНОСТИКА являются недопустимыми. Эти команды называются привилегированными. Привилегированная команда, встретившаяся в состоянии "задача", является причиной прерывания программы. В состоянии "супервизор" допустимы все команды.

5.I.7. Код прерывания. Разряды ССП I6–3I определяют, что является причиной прерывания: ввод-вывод, программа, обращение к супервизору, внешний сигнал или сигнал сбоя машины.

5.I.8. Код длины команды КДК. Код в 32-м и 33-м разрядах ССП указывает, какое число полуслов занимает последняя выполнявшаяся команда при программном прерывании или прерывании при обращении к супервизору. Содержимое этих разрядов при прерываниях по вводу-выводу, внешних прерываниях или прерываниях по машинному сбою не определено. В табл. 4 приведены значения кода длины команды.

Таблица 4

| Код длины | Разряды ССП-32-33 | Разряды команды 0-I | Длина команды | Формат |
|---|---|---|---|---|
| 0 | 00 | | Не доступна | |
| I | 0I | 00 | Одно полуслово | RR |
| 2 | I0 | 0I | Два полуслова | RX |
| 2 | I0 | I0 | Два полуслова | RS или SI |
| 3 | II | II | Три полуслова | SS |

5.I.9. Код условия КУ. В разрядах 34 и 35 ССП находится код условия (признак результата).

5.I.I0. Маска программы. Разряды 36-39 ССП - это четырехразрядная маска программных прерываний. Каждый разряд, как это видно из приведенной ниже табл. 5, связан с одним из особых случаев, возникающих при выполнении программы. Если некоторый разряд маски равен единице, то соответствующий особый случай вызывает прерывание. Если разряд маски равен нулю, то прерывание не происходит. Разряд значимости в маске, кроме того, определяет, как завершается выполнение сложения и вычитания с плавающей запятой.

Таблица 5

| Разряд маски программы | Особый случай в программе |
|---|---|
| 36 | Переполнение в операции с фиксированной запятой |
| 37 | Переполнение в десятичной операции |
| 38 | Исчезновение порядка |
| 39 | Потеря значимости |

5.I.II. Адрес команды. Разряды 40-63 ССП являются адресом команды. Этот адрес определяет крайний левый байт следующей команды.

## 6. СТРУКТУРА ОПИСАНИЙ МИКРОПРОГРАММ

6.I. Все микропрограммы именуются своими идентификаторами, т.е. условными мнемоническими обозначениями.

6.2. Каждое описание включает пункты под названиями:

Команды
Операнды
Результаты
Прерывания
Алгоритм
Диаграмма алгоритма
Регистры и триггеры

Исключение составляют описания микропрограмм ВЫБОР и ПРНКО.

6.2.I. Команды. Пункт содержит перечень команд, реализуемых данной микропрограммой с указанием для каждой команды:

- мнемонического обозначения (мнем.);
- формата;
- шестнадцатиричного входного адреса микропрограммы для данной команды (вход);
- среднего времени выполнения (без учета времени выборки команды) в микросекундах.

В приводимых в этом пункте формулах приняты следующие обозначения:

B – общее количество обработанных байтов первого операнда;

H – количество значащих шестнадцатиричных цифр (исключая впередистоящие нули) в двоичном операнде;

N – общее количество байтов первого операнда для тех инструкций, в которых оба операнда имеют одинаковую длину;

$N_1$ – общее количество байтов первого операнда;

$N_2$ – общее количество байтов второго операнда;

D – количество значащих десятичных цифр;

$N_{min}$ – наименьшее из $N_1$ и $N_2$;

$N_{abc}$ – $/N_1 - N_2/$;

S – число знаков в редактируемом поле;

Z – общее число символов начала значимости и выбора цифры в шаблоне редактирования;

R – общее число разделителей полей в шаблоне редактирования;

ОТМ – число раз занесения адреса в регистр I;

A – равно 0, если регистр I не изменяется;
равно I, если в регистр I заносится адрес;

U – количество регистров, загружаемых или записываемых в память.

6.2.2. Операнды. В пункте описаны исходные данные, которые должна обработать микропрограмма. По необходимости указывается структура исходных данных и имеющие место ограничения.

6.2.3. Результаты. Приводятся результаты выполнения микропрограммы.

6.2.4. Прерывания. Указываются прерывания, которые может вызвать данная микропрограмма. Причиной прерываний является наличие неправильных операндов или их спецификаций, а также появление особых результатов. При обнаружении причин прерывания микропрограммы, реализующие команды ЕС-1020, передают управление микропрограмм ВХПРР и EXTIO по символическим адресам:

ПНКО – операция;
ППРО – привилегированная операция;
ПНКИ – некорректность команды ВЫПОЛНИТЬ;
ПНСП – спецификация;
ПНДД – данные;
ППФЗ – переполнение с фиксированной запятой;
ПДФЗ – деление с фиксированной запятой;
ПДПП – десятичное переполнение;
ПДДН – десятичное деление;
ППЛП – переполнение порядка;
ПИСП – исчезновение порядка;
ППЗН – значимость;
ПДПЗ – деление с плавающей запятой;
ПВВВ – прерывания внешние и по вводу-выводу.

При нарушении защиты памяти и при неправильной адресации управление микропрограмме ВХПРР передается аппаратурно.

6.2.5. Алгоритм. Излагается выбранный метод реализации операций, перечисленных в пункте "Команды".

6.2.6. Диаграмма алгоритма. Пункт содержит описание диаграммы алгоритма (диаграммы алгоритмов содержатся в документе ЕІ3.055.001 ДІ). В описании даются необходимые пояснения, касающиеся общей структуры алгоритма, отдельных его блоков, а также особенностей реализации микрокомандами указанных в блоках действий. Блоки именуются своими координатами.

6.2.7. Регистры и триггеры. Приводится перечень используемых микропрограммой регистров и триггеров ВЧУ с указанием их состояния после выборки команды (исходное состояние) и способа использования (назначение).

6.2.8. Для обозначения содержимого регистра или памяти соответствующий номер регистра или адрес памяти заключается в скобки ⟨И⟩. Например, запись

$$\langle B_I \rangle + D_I$$

означает сумму базового адреса, находящегося в регистре $B_I$, и смещения $D_I$; запись

$$\langle\langle B_I \rangle + D_I \rangle$$

означает содержимое памяти по адресу, равному $\langle B_I \rangle + D_I$.

Запись состояния регистра ВЧУ в виде

$$R_2\ 0010$$

означает, что старший полубайт регистра совпадает с полем $R_2$ команды, а младший полубайт равен $0010_{(2)}$. Запись вида

$$\langle R_2 0010 \rangle$$

означает содержимое локальной памяти (ЛП) по адресу, равному $R_2 0010$.

## 7. ВЫБОРКА КОМАНД

7.1. Микропрограмма ВЫБОР.

Назначение. Выборка команд из основной памяти и передача управления микропрограммам, реализующим эти команды.

Результат. На регистрах процессора размещаются код операции, абсолютные адреса операндов, а в некоторых случаях и отдельные байты операндов. Состояние регистров и триггеров процессора после выборки команды представлено в табл. 6.

Как видно из таблицы, для всех команд, кроме команд формата RX и классов SS 2, SS 4, код операции находится в регистре Д. Для всех команд, кроме команд классов RS3 (S13), RS4 (S14), SS1 и SS3 с некорректным кодом операции, второй байт первого полуслова команды находится в регистре Л, а в регистре ГРИ формируется абсолютный адрес, компоненты которого указаны во втором полуслове команды (для команды ЗАГРУЗКА АДРЕСА вместо регистра Г используется регистр Д, так как загружаемый адрес может быть длиной в три байта). Для команд формата RR (кроме RRI) в регистре И формируется адрес локальной памяти, по которому находятся байты общего регистра с номером $R_2$. Для этих же команд и команд формата RX в регистре У формируется адрес локальной памяти, по которому находятся байты общего регистра с номером $R_1$. Абсолютный адрес, компоненты которого указаны в третьем полуслове команд классов SS2 и SS4, формируется в регистре ПТУ. Для команд десятичной арифметики в регистрах ГРИ и ПТУ формируется адрес младших байтов операндов.

Для команд формата RR (кроме RRI) и формата RX (кроме RXI) в регистры Н и З читаются байты, находящиеся в памяти по адресу, сформированному в регистре ГРИ. Байт, прочитанный в регистре Н, затем пересылается в регистр Т.

Для команд с плавающей запятой триггер БС4 устанавливается в нуль, если операнды короткие, и в единицу, если операнды длинные. Состояния триггеров БС в остальных случаях приведены лишь для сведения.

Значение счетчика адреса команд (СчАК), находящегося в регистре МФЕ, увеличивается на длину выбранной команды, а в локальную память, в буфер команды, заносится первое полуслово выбранной команды, если обращение к микрокоманде ВЫБОР произошло не из микропрограммы ИСИСЧ, реализующей команду ВЫПОЛНИТЬ. В противном случае в регистре МФЕ находится продвинутый адрес команды ВЫПОЛНИТЬ, а в буфере команды – первое полуслово этой команды.

В соответствии с кодом операции происходит передача управления в фиксированные ячейки первого модуля постоянной памяти.

Шестнадцатиричный абсолютный адрес передачи управления определяется формулами вида:

$$B+2 \cdot \{KO\},$$

где $\{KO\}$ - шестнадцатиричное значение младшей тетрады кода операции;

B - базовый шестнадцатиричный адрес.

Например, после выборки команды СЛОЖЕНИЕ ДЕСЯТИЧНОЕ (код операции равен FA) управление будет передано по адресу

$$I8I+2 \cdot A = I8I + I4 = I95$$

Прерывания. Спецификация:

а) адрес команды, находящийся в регистре МФЕ, нечетный;

б) номера общих регистров, указанные в полях $R_1$ и $R_2$, команд с плавающей запятой не равны 0,2,4,6 или адрес $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ не кратен 8 для длинных операндов (4 - для коротких операндов);

в) в командах класса RX2 адрес $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ не кратен 4.

Значение СчАК равно продвинутому адресу команды. В буфере команды - первое полуслово команды.

Операция: выбраны команды классов RS3 (SI3), RS4 (SI4), SSI или SS3.

Значение СчАК равно продвинутому адресу команды. В буфере команды - первое полуслово команды.

Таблица 6

| Формат | | | Регистры | | | | | | | | | | Триггеры | | | | Адрес микрокоманды передачи управления | Время выборки, мксек |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| КО (разряды 0-3) | Класс | Операции | Г | Р | И | П | Т | У | Д | Л | Н | Ѯ | БС2 | БС3 | БС4 | БС5 | | |
| 0000 | RRI | Переходы и переключение состояния | | | | | | | КО | $R_1P_2$ | | | 0 | 0 | | | I40+2{КО} | 5 |
| 000I | RR2 | С фиксированной запятой (для полных слов) и логические операции | | | $R_2$0010 | | $\langle R_2$0010$\rangle$ | $R_1$0010 | КО | $R_1R_2$ | $\langle R_2$0010$\rangle$ | $\langle R_2$0011$\rangle$ | 0 | 0 | 0 | 0 | I00+2{КО} | 8 |
| 00I0 | RR3 | С плавающей запятой (длинные операнды) | | | $R_2$I000 | | $\langle R_2$I000$\rangle$ | $R_1$I000 | КО | $R_1R_2$ | $\langle R_2$I000$\rangle$ | $\langle R_2$I001$\rangle$ | I | 0 | I | 0 | I0I+2{КО} | 9 |
| 00II | RR4 | С плавающей запятой (короткие операнды) | | | $R_2$I000 | | $\langle R_2$I000$\rangle$ | $R_1$I000 | КО | $R_1R_2$ | $\langle R_2$I000$\rangle$ | $\langle R_2$I001$\rangle$ | I | 0 | 0 | 0 | I0I+2{КО} | 9 |
| 0I00 | RXI | С фиксированной запятой (для полуслов) и переходы | $\langle X_2\rangle+\langle B_2\rangle+D_2$ | | | | | $R_1$0010 | КО | $R_1X_2$ | | | 0 | 0 | 0 | 0 | I20+2{КО} | I8 |
| 0I0I | RX2 | С фиксированной запятой (для полных слов) и логические операции | $\langle X_2\rangle+\langle B_2\rangle+D_2$ | | | | $\langle\langle X_2\rangle+\langle B_2\rangle+D_2\rangle$ | $R_1$0010 | КО | $R_1X_2$ | $\langle\langle X_2\rangle+\langle B_2\rangle+D_2\rangle$ | $\langle\langle X_2\rangle+\langle B_2\rangle+D_2+I\rangle$ | 0 | 0 | I | 0 | I2I+2{КО} | I8 |
| 0II0 | RX3 | С плавающей запятой (длинные операнды) | $\langle X_2\rangle+\langle B_2\rangle+D_2$ | | | | $\langle\langle X_2\rangle+\langle B_2\rangle+D_2\rangle$ | $R_1$I000 | КО | $R_1X_2$ | $\langle\langle X_2\rangle+\langle B_2\rangle+D_2\rangle$ | $\langle\langle X_2\rangle+\langle B_2\rangle+D_2+I\rangle$ | 0 | 0 | I | I | I4I+2{КО} | I9 |
| 0III | RX4 | С плавающей запятой (короткие операнды) | $\langle X_2\rangle+\langle B_2\rangle+D_2$ | | | | $\langle\langle X_2\rangle+\langle B_2\rangle+D_2\rangle$ | $R_1$I000 | КО | $R_1X_2$ | $\langle\langle X_2\rangle+\langle B_2\rangle+D_2\rangle$ | $\langle\langle X_2\rangle+\langle B_2\rangle+D_2+I\rangle$ | 0 | 0 | 0 | I | I4I+2{КО} | I9 |

Продолжение

| Формат | | | Регистры | | | | | | Триггеры | | | | Адрес микрокоманды передачи управления | Время выборки, мксек |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| КО (разряды 0-3) | Класс | Операции | Г Р И | П Т У | Д | Л | Н | З | ВС2 | ВС3 | ВС4 | ВС5 | | |
| 1000 | RS1 / SI1 | Переходы, переключение состояния и операции сдвига | $\langle B_2 \rangle + D_2$ / $\langle B_I \rangle + D_I$ | | КО | $R_I R_3$ / $I_2$ | | | 0 | 0 | 0 | 0 | $160+2\{КО\}$ | 14 |
| 1001 | RS2 / SI2 | С фиксированной запятой, логические операции и ввод/вывод | $\langle B_2 \rangle + D_2$ / $\langle B_I \rangle + D_I$ | | КО | $R_I R_3$ / $I_2$ | | | 0 | 0 | 1 | 0 | $161+2\{КО\}$ | 14 |
| 1010 | RS3 / SI3 | Операции отсутствуют | | | КО | | КО | $R_I R_3$ / $I_2$ | 0 | 1 | 1 | 0 | 014 | 11 |
| 1011 | RS4 / SI4 | Операции отсутствуют | | | КО | | КО | $R_I R_3$ / $I_2$ | 0 | 1 | 1 | 0 | 014 | 11 |
| 1100 | SS1 | Операции отсутствуют | | | КО | | КО | L | 0 | 1 | 1 | 1 | 014 | 14 |
| 1101 | SS2 | Логические операции | $\langle B_I \rangle + D_I$ | $\langle B_2 \rangle + D_2$ | | L | | | 0 | 0 | 0 | 1 | $180+2\{КО\}$ | 23 |
| 1110 | SS3 | Операции отсутствуют | | | КО | | КО | L | 0 | 1 | 1 | 1 | 014 | 14 |
| 1111 | SS4 | Десятичной арифметики | $\langle B_I \rangle + D_I + L_I$ | $\langle B_2 \rangle + D_2 + L_2$ | | $L_1 L_2$ | | | 0 | 0 | 1 | 1 | $181+2\{КО\}$ | 27 |



Примечания. 1. КО - код операции.

2. $\{КО\}$ - значение младшей тетрады КО (разряды 4-7).

3. Для команды ЗАГРУЗКА АДРЕСА (класс RX1) вместо регистра Г используется регистр Д.

4. Время выборки указано с учетом базирования, время индексирования равно 3 мксек.

Внешние и вводо-выводные прерывания: перед выборкой команды обнаружен (триггер ТВВВ установлен в единицу) хотя бы один из следующих запросов:

а) на вводо-выводное прерывание;

б) на внешнее прерывание;

в) на обновление таймера;

или нажата кнопка ОСТАНОВ на пульте управления.

Выборка команды не начинается. Значение СчАК не меняется. Запрос сохраняется.

Адресация.

Защита памяти по чтению.

Алгоритм. Обращение к микропрограмме ВЫБОР возможно аппаратурное или из микропрограмм. Из основной памяти читаются старшие байты команды. Анализируется наличие причин внешних и вводо-выводных прерываний. Адрес команды анализируется на правильность спецификации. Если причина прерывания есть, то происходит прерывание. Если нет, то происходит запись старших байтов команды в локальную память, в буфер команды, и дальнейшая выборка команды продолжается в зависимости от формата выбранной команды.

Для команд всех форматов происходит формирование результатов выборки в соответствии с табл. 6 и передача управления в соответствии с кодом операции. Дополнительно для команд классов RR3, RR4, RX3 и RX4 производится проверка на правильность спецификации номеров регистров плавающей запятой, а для команд классов RX2, RX3 и RX4 на правильность спецификации проверяется и адрес второго операнда.

Диаграмма алгоритма. Существуют десять входов в микропрограмму ВЫБОР. По адресу 000 (блок I8A0) обращаются все микропрограммы, за исключением ИСИСЧ, реализующей команду ВЫПОЛНИТЬ. Микропрограмма ИСИСЧ обращается к микропрограмме ВЫБОР по одному из девяти (в зависимости от формата и класса подчиненной команды) адресов: IA9 (блок I8AI), 0A4 (блок I8C2), 0A3 (блок I8C3), 0A6, 0A7, 0E7 (блок I9C0), 0CC (блок 20BI), IBA или IBE (блок 20EI).

Переход по классам (блок I8EI) реализован функциональным переходом по старшей тетраде кода операции. Некоторые ветки (классы) объединяются. При объединении ветвей, соответствующих формату RX (блок I9A0), информация о классе сохраняется на триггерах БС4 и БС5:

| БС4 | БС5 | Класс |
|---|---|---|
| 0 | 0 | RXI |
| I | 0 | RX2 |
| I | I | RX3 |
| 0 | I | RX4 |

Эти же триггеры хранят информацию о формате и классе при объединении ветвей, соответствующих форматам RS ( SI) и SS (блоки 20B0, 20C0, 20Д0):

| БС4 | БС5 | Формат (класс) |
|---|---|---|
| 0 | 0 | RSI (SII) |
| I | 0 | RS2 (SI2) |
| 0 | I | SS2 |
| I | I | SS4 |

На триггерах БС2 и БС4 запоминается RR2, RR3 и RR4 (блоки I8B2, I8Д2):

| БС2 | БС4 | Класс |
|---|---|---|
| 0 | 0 | RR2 |
| I | I | RR3 |
| I | 0 | RR4 |

Установкой ПКФ (блок I9БО) запоминается, что вырабатывается команда ЗАГРУЗКА АДРЕСА.

В блоке I8ДI происходит сброс триггера БС2. Микропрограмма ИСИСЧ, обращаясь к микропрограмме ВЫБОР по одному из входов 0А6, 0А7, 0Е7 (блок I9ВО), ОСС (блок 20БI), IBA или IBE (блок 20ДI), устанавливает триггер БС2 в единицу. Таким образом, нулевое состояние этого триггера (при выборке команд всех форматов, кроме RR) говорит о том, что обращение к микропрограмме ВЫБОР произошло по входу 000 (блок I8А0).

В форматах RX, SS код операции, который необходим для выхода из микропрограммы, читается из буфера команды или, в случае входа из ИСИСЧ для продолжения выборки команд формата RX, из рабочей области ЛП (06-07). Микропрограмма ИСИСЧ при обращении к ВЫБОР для продолжения выборки команд формата RX записывает код подчиненной команды в рабочую область ЛП, а для команд формата SS в буфер команды. В последнем случае микропрограмма ВЫБОР сама записывает в буфер код операции команды ВЫПОЛНИТЬ (блок 20Б6).

Выход по младшей тетраде кода операции происходит с учетом состояния триггеров БС2 (блок I8Б4) и БС4 (блоки I9В6, 20А5, 20Б7 и 20В8).

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| М Ф Е | Текущий адрес команды | Формирование адреса следующей команды (СчАК) |
| Г Р И | - | Формирование адресов операндов в соответствии с табл. 6 |
| П Т | - | В соответствии с табл. 6<br>Рабочий регистр в блоках I8Б4, I9BI, I9CI, 20CI; в остальных случаях в соответствии с табл. 6 |
| У | - | Рабочий регистр в блоках I9B2, I9C2, 20ДI, 20Д4; в остальных случаях в соответствии с табл. 6 |
| Д | - | Рабочий регистр в блоке 20С4; в остальных случаях в соответствии с табл. 6 |
| Л | - | Рабочий регистр в блоке I8С5; в остальных случаях в соответствии с табл. 6 |
| БС2 | "I", если вошли в МП ВЫБОР из МП ИСИСЧ; в остальных случаях непредсказуемо | В формате RR для отличия RR2 от RR3 и RR4; в остальных форматах как указатель выборки подчиненной команды |
| БС3 | - | Устанавливается в соответствии с табл. 6 |
| БС4 | - | В блоках I9EI, 20А5, 20С5, 20В7, 20С8 для различения форматов и классов (см. табл.6) |
| БС5 | - | В блоках I9EI, 20С2, 20С5 для различения форматов и классов (см. табл. 6) |
| ТВК | 0 | Индикатор выборки команды |
| ТАК | 0 | Индикатор того, что СчАК находится в локальной памяти |
| ПКФ | - | Индикатор команды ЗАГРУЗКА АДРЕСА |

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| ЛП 98–99 | Код (первое полуслово) предыдущей команды | Хранение кода (первого полуслова) текущей команды. Буфер инструкции |
| ЛП $8^D$, 8E, $8^F$ | Продвинутый адрес команды ВЫПОЛНИТЬ, если обращение происходит из микропрограммы ИСИСЧ | Используется для восстановления в МФЕ продвинутого адреса команды ВЫПОЛНИТЬ |
| ЛП 06, 07 | – | Хранение кода операции в случае выборки подчиненной команды формата RX |

7.2. Микропрограмма ПРНКО. Существуют такие коды операций, которые не соответствуют никакой операции или же указанной операции в данной модели нет. Если выбрана команда с некорректным кодом операции, то происходит передача управления микропрограмме ВХПР по символическому адресу ПНКО.

При некорректности старшей тетрады кода операции переход к микропрограмме ВХПР реализован в микропрограмме ВЫБОР. Для всех остальных некорректных кодов операций переход к микропрограмме ВХПР реализован в микропрограмме ПРНКО. Ниже перечислены некорректные коды операций, обслуживаемые микропрограммой ПРНКО, с указанием для каждого из них в скобках шестнадцатиричного входного адреса микропрограммы:

00(I40), 01(I42), 02(I44), 03(I46), 0B(I56), 0C(I58), $0^D$(I5A), 0E(I5C), 0E(I5E), 25(I0B), 26($I0^D$), 27(I0F), 35(I0B), 36($I0^D$), 37(I0F), $4^D$(I3A), 51(I23), 52(I25), 53(I27), 61(I43), 62(I45), 63(I47), 64(I49), 65(I4B), 66($I4^D$), 67(I4F), 71(I43), 72(I45), 73(I47), 74(I49), 75(I4B), 76($I4^D$), 77(I4F), 81(I62), 83(I66), 84(I68), 85(I6A), 99(I73), 9A(I75), 9B(I77), $^D$0(I80), $^D$8(I90), $^D$9(I92), $^D$A(I94), $^D$B(I96), F4(I89), F5(I8B), F6($I8^D$), F7(I8F), FE($I9^D$), FF(I9F), F0(I81).

## 8. СТАНДАРТНАЯ СИСТЕМА КОМАНД. АРИФМЕТИЧЕСКИЕ ОПЕРАЦИИ

### 8.1. Микропрограмма ARSR

Команды

| Название | Мнем. | Формат | | | | Вход | Время |
|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | | | |
| СЛОЖЕНИЕ | AR | RR | 1A | $R_1$ | $R_2$ | 114 | 12 |
| СЛОЖЕНИЕ КОДОВ | ALR | RR | 1E | $R_1$ | $R_2$ | 11C | 13 |
| ВЫЧИТАНИЕ | SR | RR | 1B | $R_1$ | $R_2$ | 116 | 12 |
| ВЫЧИТАНИЕ КОДОВ | SLR | RR | 1F | $R_1$ | $R_2$ | 11E | 14 |

Операнды. Два числа с фиксированной запятой длиной в слово, расположенные в общих регистрах с номерами $R_I$ и $R_2$. Для команд SR, SLR операнд в общем регистре $R_I$ - уменьшаемое; операнд в общем регистре $R_2$ - вычитаемое.

Результат. 32-разрядная алгебраическая сумма (разность) в форме числа с фиксированной запятой, расположенная в общем регистре с номером $R_I$.

Установка признака результата для команд AR, SR:

0 - сумма (разность) равна нулю;
I - сумма (разность) меньше нуля;
2 - сумма (разность) больше нуля;
3 - переполнение.

Установка признака результата для команд ALR, SLR:

0 - сумма равна нулю (перенос отсутствует , т.к. при нулевой разности всегда имеет место перенос из знакового разряда);
I - сумма (разность) не равна нулю (перенос отсутствует);
2 - сумма (разность) равна нулю (есть перенос);
3 - сумма (разность) не равна нулю (есть перенос).

Прерывания. Перевыполнение с фиксированной запятой: есть перенос только в знаковый разряд или только из знакового разряда. При переполнении знак результата противоположен истинному знаку суммы или разности. Прерывание происходит только в командах СЛОЖИТЬ и ВЫЧЕСТЬ. В командах СЛОЖИТЬ ЛОГИЧЕСКИ и ВЫЧЕСТЬ ЛОГИЧЕСКИ переполнение игнорируется.

Алгоритм. Второй операнд складывается с первым операндом (вычитается из первого), а сумма (разность) помещается на место первого операнда. В сложении (вычитании) участвуют все 32 разряда каждого операнда. Если переносы из знакового разряда и старшего разряда числа одновременно либо отсутствуют, либо присутствуют, это говорит о том, что сложение выполнено нормально.

Наличие переноса только из одного из указанных разрядов характеризует переполнение. При переполнении полученный знаковый разряд остается без изменения.

Пример (для простоты взят один байт).

```
        0III IIII          Первый операнд
        0I00 0000          Второй операнд
        0III IIII
       +0I00 0000
        I0II IIII          Сложение, сложение логическое
        I0II IIII          Результат. Имеет место переполнение
                           Признак результата:
          3                     сложения;
          I                     сложения логического
```

Диаграмма алгоритма. Для логического вычитания после установки признака результата с помощью КУ 2 (блок 22Д3) разряды 6-7 регистра БС складываются по модулю 2 с кодом I0 для верной установки признака результата.

Настройка подпрограммы (блоки 22AI, 22CI) состоит в занесении косвенной функции "+" или "-". Для операций СЛОЖЕНИЕ и ВЫЧИТАНИЕ одновременно с анализом переполнения (блок 22C4) устанавливается признак результата с помощью КУI.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| И | $R_2$0010 | Хранит текущий адрес второго операнда |
| У | $R_I$0010 | Хранит текущий адрес первого операнда |
| Д | Код операции | Хранит четный байт второго операнда |
| Л | $R_I$ $R_2$ | Хранит нечетный байт второго операнда |
| БС 2 | 0 | Индикатор команды:<br>I - ВЫЧИТАНИЕ КОДОВ;<br>0 - СЛОЖЕНИЕ КОДОВ |
| БС 4 | 0 | Индикатор конца операции:<br>I - конец операции |
| БС 5 | 0 | Индикатор логической команды:<br>I - команды ВЫЧИТАНИЕ КОДОВ, СЛОЖЕНИЕ КОДОВ;<br>0 - команды ВЫЧИТАНИЕ, СЛОЖЕНИЕ |
| Н | $\langle R_2 0010 \rangle$ | Информационный регистр |
| З | $\langle R_2 0011 \rangle$ | Информационный регистр |

8.2. Микропрограмма ASRX

Команды

| Название | Мнем. | Формат | | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | байт 3 | байт 4 | | |
| СЛОЖЕНИЕ | A | RX | 5A | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I35 | I5 |
| СЛОЖЕНИЕ КОДОВ | AL | RX | 5E | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I3I | I6 |
| ВЫЧИТАНИЕ | S | RX | 5B | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I37 | I5 |
| ВЫЧИТАНИЕ КОДОВ | SL | RX | 5F | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I3F | I7 |

Операнды. Первое слагаемое (уменьшаемое) - число с фиксированной запятой длиной в слово, расположенное в общем регистре с номером $R_I$. Второе слагаемое (вычитаемое) - число с фиксированной запятой длиной в слово, расположенное в оперативной памяти по адресу $\langle X_2 \rangle + \langle B_2 \rangle + D_2$.

Результат. 32-разрядная алгебраическая сумма (разность) в форме числа с фиксированной запятой, расположенная в регистре с номером $R_I$.

Установка признака результата для команд A, S :

0 - сумма (разность) равна нулю;

I - сумма (разность) меньше нуля;

2 - сумма (разность) больше нуля;

3 - переполнение.

Установка признака результата для команд AL, SL :

0 - сумма равна нулю (перенос отсутствует), при нулевой разности всегда имеет место перенос из знакового разряда;

I - сумма (разность) не равна нулю (перенос отсутствует);

2 - сумма (разность) равна нулю (есть перенос);

3 - сумма (разность) не равна нулю (есть перенос).

Прерывания. Переполнение с фиксированной запятой: есть перенос только в знаковый разряд или только из знакового разряда. При переполнении знак результата противоположен истинному знаку суммы или разности. Прерывание происходит только в командах СЛОЖЕНИЕ и ВЫЧИТАНИЕ. В командах СЛОЖЕНИЕ КОДОВ, ВЫЧИТАНИЕ КОДОВ переполнение игнорируется.

Спецификация: если адрес второго операнда не кратен 4, то произойдет прерывание по спецификации (реализовано в микропрограмме ВЫБОР). Адресация. Защита по чтению.

Алгоритм. Второй операнд складывается с первым (вычитается из первого), и сумма (разность) помещается на место первого операнда. В сложении (вычитании) участвуют все 32 разряда каждого операнда. Если переносы из знакового разряда и старшего разряда числа одновременно либо отсутствуют, либо присутствуют, это говорит о том, что сложение выполнено нормально. Наличие переноса только из одного из указанных разрядов характеризует переполнение. При переполнении полученный знаковый разряд остается без изменения.

Пример (для простоты числа взяты длиной в один байт).

| | |
|---|---|
| 0III IIII | Первый операнд |
| 0I00 0000 | Второй операнд |
| +0III IIII<br>0I00 0000 | |
| I0II IIII | Сложение, сложение кодов |
| I0II IIII | Результат. Имеет место переполнение |
| | Признак результата: |
| 3 | сложения; |
| I | сложения кодов |

Диаграмма алгоритма. Для команды ВЫЧИТАНИЕ КОДОВ после установки признака результата с помощью КУ2 (блок 22Д3) разряды 6-7 регистра БС складываются по модулю 2 с кодом I0 для верной установки признака результата.

Настройка подпрограммы (блоки 23AI, 23CI) состоит в занесении косвенной функции "+" для команд СЛОЖЕНИЕ и СЛОЖЕНИЕ КОДОВ и "-" для команд ВЫЧИТАНИЕ и ВЫЧИТАНИЕ КОДОВ. Для команд СЛОЖЕНИЕ и ВЫЧИТАНИЕ одновременно с анализом переполнения (блок 23C4) устанавливается признак результата с помощью КУI.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ | Хранит текущий адрес второго операнда |
| У | $R_I$0OI0 | Хранит текущий адрес первого операнда |
| Д | Код операции | Хранит четный байт второго операнда |
| Л | $R_I$ $X_2$ | Хранит нечетный байт второго операнда |
| БС2 | 0 | Индикатор команд:<br>I - ВЫЧИТАНИЕ КОДОВ;<br>0 - СЛОЖЕНИЕ КОДОВ |

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| БС4 | I | Индикатор конца операции:<br>0 - конец операции |
| БС5 | 0 | Индикатор логической команды:<br>I - команды ВЫЧИТАНИЕ КОДОВ, СЛОЖЕНИЕ КОДОВ;<br>0 - команды ВЫЧИТАНИЕ, СЛОЖЕНИЕ |
| Н | $\langle\langle X_2\rangle + \langle B_2\rangle + D_2\rangle$ | Информационный регистр |
| З | $\langle\langle X_2\rangle + \langle B_2\rangle + D_2 + I\rangle$ | Информационный регистр |

8.3. Микропрограмма AHSH

Команды

| Название | Мнем. | Формат | | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | байт 3 | байт 4 | | |
| СЛОЖЕНИЕ ПОЛУСЛОВА | AH | RX | 4A | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I34 | II |
| ВЫЧИТАНИЕ ПОЛУСЛОВА | SH | RX | 4B | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I36 | II |

Операнды. Первое слагаемое (уменьшаемое) - число с фиксированной запятой длиной в слово, расположенное в общем регистре с номером $R_I$. Второе слагаемое (вычитаемое) - число с фиксированной запятой длиной в полуслово, расположенное в оперативной памяти по адресу $\langle X_2\rangle + \langle B_2\rangle + D_2$.

Результат. 32-разрядная алгебраическая сумма (разность) в форме числа с фиксированной запятой, расположенная в общем регистре с номером $R_I$.

Признак результата:

0 - сумма (разность) равна нулю;
I - сумма (разность) меньше нуля;
2 - сумма (разность) больше нуля;
3 - переполнение.

Прерывания. Спецификация: если адрес второго операнда не кратен 2, то произойдет прерывание по спецификации. Фиксированное переполнение: есть перенос в знаковый разряд или только из знакового разряда. При переполнении знак результата противоположен истинному знаку суммы или разности.

Адресация. Защита по чтению.

Алгоритм. Второй операнд складывается с первым операндом (вычитается из первого), и сумма (разность) помещается на место первого операнда.

Перед сложением (вычитанием) длина второго операнда увеличивается до получения полного слова, при этом старшим I6 разрядам полученного слова присваивается значение знакового разряда. В сложении (вычитании) участвуют все 32 разряда каждого операнда. Если переносы из знакового разряда и старшего разряда числа одновременно либо отсутствуют, либо присутствуют, это говорит о том, что сложение (вычитание) выполнено нормально. Наличие переноса только из одного из указанных разрядов характеризует переполнение. При переполнении полученный знаковый разряд суммы (разности) не изменяется.

Диаграмма алгоритма. Настройка подпрограммы (блок 24В2) состоит в занесении косвенной функции "+" для команды СЛОЖЕНИЕ ПОЛУСЛОВА и "-" для команды ВЫЧИТАНИЕ ПОЛУСЛОВА.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ | Хранит адрес полуслова |
| У | $R_I$0010 | Хранит текущий адрес первого операнда |
| Д | Код операции | Хранит четный байт второго операнда |
| Л | $R_I X_2$ | Хранит нечетный байт второго операнда |
| БС2 | 0 | Индикатор конца операции:<br>I - конец операции |
| БС3 | 0 | Индикатор значения знакового разряда полуслова:<br>0 - разряд = 0;<br>I - разряд = I |

8.4. Микропрограмма MRM

Команды

| Название | Мнем. | Формат | | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | байт 3 | байт 4 | | |
| УМНОЖЕНИЕ | MR | RR | IC | $R_I$ | $R_2$ | | | II8 | 330 |
| УМНОЖЕНИЕ | M | RX | 5C | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I39 | 330 |
| УМНОЖЕНИЕ ПОЛУСЛОВА | MH | RX | 4C | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I38 | 200 |

Операнды. Множимое - число с фиксированной запятой длиной в слово, расположенное в общем регистре с номером $R_I$+I (команды MR и M), где $R_I$ - четно или с номером $R_I$ (команда MH).

Множитель - число с фиксированной запятой длиной в слово, расположенное в регистре $R_2$ (команда MR) или в основной памяти по адресу $D_2 + \langle B_2 \rangle + \langle X_2 \rangle$ (команда M);

- число с фиксированной запятой длиной в полуслово, расположенное в основной памяти по адресу $D_2 + \langle B_2 \rangle + \langle X_2 \rangle$ (команда MH).

Результат. Команды MR и M. 64-разрядное произведение в форме числа с фиксированной запятой, расположенное в паре общих регистров с номерами $R_I$ и $R_I$ + I.

Команда MH. 32 младших разряда произведения, расположенные в общем регистре $R_I$. При наличии переполнения знаковый разряд может отличаться от истинного.

Знак произведения всегда определяется по алгебраическим правилам в зависимости от знаков сомножителей.

Признак результата всегда остается без изменения.

Прерывания. Спецификация: в командах M и MR номер $R_I$ не кратен 2, в команде MH адрес $\langle B_2 \rangle + \langle X_2 \rangle + D_2$ не кратен 2, в команде M адрес $\langle B_2 \rangle + \langle X_2 \rangle + D_2$ не кратен 4 (реализовано в микропрограмме ВЫБОР). Адресация (для команд M и MH). Защита памяти по чтению (для команд M и MH).

Алгоритм. Умножение начинается с анализа знака множимого. Если знак "-", то формируется двоичное дополнение множимого. Множимое (или его двоичное дополнение) затем увеличивается последовательно в 2,3,6 раз и полученные кратные запоминаются.

Далее последовательно анализируются шестнадцатиричные цифры множителя, начиная с младшей.

В приведенной ниже табл. 7 указаны действия, производимые для каждой цифры множителя в зависимости от предыдущей цифры. Для младшей цифры предыдущая цифра считается равной нулю.

Таблица составлена так, чтобы обеспечить минимальное количество сложений.

Таблица 7

| Цифры множителя | Предыдущая цифра множителя > 8 | Предыдущая цифра множителя < 8 |
|---|---|---|
| 0 | + IM | 0 |
| I | + 2M | + IM |
| 2 | + 3M | + 2M |
| 3 | + 2M+2M | + 3M |
| 4 | + 3M+2M | + 2M+2M |
| 5 | + 6M | + 3M+2M |
| 6 | + IM+6M | + 6M |
| 7 | + 6M+2M | + IM+6M |
| 8 | - (IM+6M) | - (6M+2M) |
| 9 | - 6M | - (IM+6M) |
| A | - (3M+2M) | - 6M |
| B | - (2M+2M) | - (3M+2M) |
| C | - 3M | - (2M+2M) |
| D | - 2M | - 3M |
| E | - IM | - 2M |
| F | 0 | - IM |

Для цифр от 8 до F сложение заменяется вычитанием. Кратные, полученные из таблицы, затем складываются или вычитаются с переносом (сдвиг на четыре разряда) в соответствии с весом каждой цифры множителя.

При обработке самой старшей шестнадцатиричной цифры множителя ее знаковый разряд считается разрядом данных и цифра обрабатывается, как любая другая. При этом в случае отрицательного множителя и положительного множимого автоматически получается отрицательное произведение. По окончании выполнения циклов сложения анализируется знак множимого. Если множимое было отрицательным, берется двоичное дополнение полученного результата. Умножение заканчивается занесением произведения в заданную пару универсальных регистров. Если множимое - максимальное отрицательное число, то алгоритм умножения в этом случае другой. Шестнадцатиричные цифры множителя не анализируются и кратные 2M, 3M и 6M множимого не формируются. Формируется и дополняется только кратное IM множимого. В этом случае множитель умножается на $2^{31}$, что означает сдвиг множителя влево на 31 разряд. Микропрограмма выполняет это путем сдвига множителя на один разряд вправо и запоминанием результата в 32 старших разрядах поля произведения. Перенос от сдвига вправо идет в старший разряд 32 младших разрядов поля произведения. Затем результат двоично дополняется, чтобы получить истинное значение произведения. Для иллюстрации метода умножения приводятся примеры, в которых для простоты взят один байт (исключение составляет пример 5).

Пример I.

| | | | |
|---|---|---|---|
| + 7 | 0000 | 0III | Множимое |
| + 33 | 00I0 | 000I | Множитель |
| IM | 0000 | 0III | |
| 2M | 0000 | IIIO | |
| 3M | 000I | 0I0I | Кратные множимого |
| 6M | 00I0 | I0I0 | |
| | 0000 | 0000 | Первая цифра множителя показывает, что должно быть добавлено кратное +IM |
| | +0000 | 0III | |
| | 0000 | 0III | |
| | 0000 | 0III | Вторая цифра множителя показывает, что должно быть добавлено кратное 2M с переносом (сдвиг на четыре разряда) |
| + | | | |
| | 0000 IIIO | | |
| | 0000 IIIO | 0III | |
| + 23I | 0000 | IIIO 0III | Произведение |

Пример 2.

| | | | |
|---|---|---|---|
| +I2 | 0000 | II00 | Множимое |
| -I5 | IIII | 000I | Множитель |
| IM | 0000 | II00 | |
| 2M | 000I | I000 | |
| 3M | 00I0 | 0I00 | Кратные множимого |
| 6M | 0I00 | I000 | |
| | 0000 | 0000 | Первая цифра множителя показывает, что должно быть добавлено кратное +IM |
| | +0000 | II00 | |
| | 0000 | II00 | |
| - | 0000 | II00 | Анализ второй цифры множителя дает - IM со сдвигом на четыре разряда |
| | 0000 II00 | | |
| | IIII II00 | II00 | |
| - I80 | IIII | II00 II00 | Произведение |

Пример 3.

| | | | |
|---|---|---|---|
| -7 | IIII | I00I | Множимое |
| +6 | 0000 | 0II0 | Множитель |
| | 0000 | 0III | Двоичное дополнение множимого |
| IM | 0000 | 0III | |
| 2M | 0000 | IIIO | |
| 3M | 000I | 0I0I | Кратные множимого |
| 6M | 00I0 | I0I0 | |

| | | | |
|---|---|---|---|
| | 0000 | 0000 | Анализ первой цифры дает +6М |
| | +0010 | 1010 | |
| | 0010 | 1010 | |
| | | | Анализ второй цифры дает +0 |
| | 1101 | 0110 | Двоичное дополнение результата |
| -42 | 1101 | 0110 | Произведение |

Пример 4.

| | | | |
|---|---|---|---|
| -7 | 1111 | 1001 | Множимое |
| -6 | 1111 | 1010 | Множитель |
| | 0000 | 0111 | Двоичное дополнение множимого |
| 1М | 0000 | 0111 | |
| 2М | 0000 | 1110 | Кратные множимого |
| 3М | 0001 | 0101 | |
| 6М | 0010 | 1010 | |
| | 0000 | 0000 | Анализ первой цифры дает -6М |
| | -0010 | 1010 | |
| | 1101 | 0110 | |
| | | | Анализ второй цифры дает +0 |
| | 0010 | 1010 | Двоичное дополнение результата |
| +42 | 0010 | 1010 | Произведение |

Пример 5.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | Множимое |
| 1010 | 1011 | 1011 | 1100 | 1101 | 1110 | 1111 | 0011 | Множитель |
| 1101 | 0101 | 1101 | 1110 | 0110 | 1111 | 0111 | 1001 | $\underbrace{100\ldots0}_{30\text{ битов}}$ Умножение на $2^{31}$ |
| 0010 | 1010 | 0010 | 0001 | 1001 | 0000 | 1000 | 0110 | $\underbrace{100\ldots0}_{30\text{ битов}}$ Результат |

Диаграмма алгоритма. При получении кратных множимого (блок 25В3) используется прямая функция "сдвиг" для 2М и 6М и косвенная функция "сложение" для 3М.

Анализ на максимальное отрицательное число (блок 25С3) осуществляется проверкой сохранения знака после получения двоичного дополнения множимого.

Одновременно с чтением множителя из локальной или оперативной памяти идет формирование счетчика: для слова он равен 4, для полуслова 2 (блоки 25С4, 25В5).

Настройка программ (блоки 26А1, 26В1, 26С1, 26Д1, 26А6, 26В6, 26С6, 26Д6) включает в себя занесение косвенной функции "+" или "-" прямо или с перекосом. Косвенная функция "-" заносится для цифр множителя $> 8$, а косвенная функция "+" заносится для цифр множителя $< 8$. Косвенная функция "прямо" заносится для младших разрядов байта множителя. Косвенная функция "с перекосом" заносится для старших разрядов байта множителя.

Подготовка следующего байта множителя (блок 27Д1) осуществляется сдвигом байтов множителя из регистров Д,Ф,Е в регистр Л.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| М | Адрес следующей команды | Используется как счетчик при обработке цифр множителя |
| Ф | | Хранит старший нечетный байт множителя |
| Е | | Хранит младший четный байт множителя |
| Г | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ - формат RX | Не используется |
| Р | | Хранит четный байт кратного множимого |
| И | $R_2$0010 - формат RR | Используется как пятый байт при получении частичного произведения |
| Т | | Хранит предыдущую цифру множителя |
| У | $R_I$0010 | Здесь находится текущий адрес байтов множимого |
| Д | Код операции | Хранит старший четный байт множителя |
| Л | | Хранит младший нечетный байт множителя |
| БС2 | 0 | Индикатор умножения слова или полуслова:<br>I - слово<br>0 - полуслово |
| БС3 | 0 | а) индикатор формата команды: I - формат RX, (блок 25В4) 0 - формат RR<br>б) индикатор количества кратных множимого:<br>I - два, 0 - одно (блок 25В6, 25Д6) |
| БС4 | 0 | а) используется для организации записи МФЕ в локальную память (блок 25В2);<br>б) индикатор младших и старших разрядов байта:<br>I - младшие разряды, 0 - старшие разряды (блок 25В6, 25Д6) |
| БС5 | 0 | Индикатор знака множимого:<br>I - знак отрицательный, 0 - знак положительный |
| ЛП | | Хранит: |
| 06-07 | - | старшие 2 байта кратного 1М |
| 16-17 | - | младшие 2 байта кратного 1М |
| 26-27 | - | старшие 2 байта кратного 2М |
| 36-37 | - | младшие 2 байта кратного 2М |
| 45-47 | - | старшие 3 байта кратного 3М |
| 56-57 | - | младшие 2 байта кратного 3М |
| 65-67 | - | старшие 3 байта кратного 6М |
| 76-77 | - | младшие 2 байта кратного 6М |
| 84-87 | - | четыре байта частичного произведения |

8.5. Микропрограмма ФДЕЛ

Команды

| Название | Мнем. | Формат | | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | байт 3 | байт 4 | | |
| ДЕЛЕНИЕ | DR | RR | 1D | $R_I$ | $R_2$ | | | IIA | 382 |
| ДЕЛЕНИЕ | D | RX | 5D | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I3B | 380 |

Операнды. Делимое - число с фиксированной запятой длиной в двоиное слово, расположенное в четно-нечетной паре общих регистров. Номер четного регистра пары равен $R_I$.

Делитель - число с фиксированной запятой длиной в слово, расположенное в общем регистре с номером $R_2$ (команда DR) или в основной памяти по адресу $\langle B_2 \rangle + \langle X_2 \rangle + D_2$ (команда D ).

Результат. Остаток и частное - числа с фиксированной запятой, длиной в слово, замещающие делимое, соответственно в четном и нечетном регистрах. Остаток имеет тот же знак, что и делимое. Знак частного определяется по алгебраическим правилам. Нулевой остаток и нулевое частное всегда положительны.

Признак результата остается без изменения.

Прерывания. Спецификация: номер $R_I$, указанный в команде, не кратен 2. Анализ адреса $\langle B_2 \rangle + \langle X_2 \rangle + D_2$ на кратность 4 реализован в микропрограмме выборки команд ВЫБОР.

Деление с фиксированной запятой: частное от деления превосходит размер общего регистра (включая случай деления на нуль). Деление подавляется. Данные в регистрах и в памяти остаются без изменения. Адресация. Защита памяти по чтению.

Алгоритм.Выбран метод деления без восстановления остатка. Если делитель равен нулю или делимое равно максимальному отрицательному числу, то имеет место переполнение: длина частного превосходит размер слова. В остальных случаях из старших 33 разрядов модуля делимого вычитается модуль делителя. Если результат отрицательный, то старшая цифра частного - нуль. Если результат положительный, то старшая цифра частного - единица.

Так как частное получается в прямом коде, то в последнем случае имеет место переполнение, если результат вычитания отличен от нуля или оставшиеся цифры делимого представляют собой число, превосходящее делитель, или знак частного - плюс. Отсутствие признаков переполнения означает, что частное равно максимальному отрицательному числу.

Если результат первого вычитания, которое назовем пробным вычитанием, отрицательный, то остальные цифры частного получаются следующим образом. Результат пробного вычитания сдвигается влево на один разряд, и в освободившийся разряд справа сносится цифра делимого. Сдвинутый результат вычитания складывается с делителем. Если результат сложения положительный, то следующая цифра частного - единица, если - отрицательный, то следующая цифра частного - нуль.

Результат сложения сдвигается влево на один разряд, и в освобождающийся разряд справа сносится следующая цифра делимого.

Если результат сложения был положительным, то делитель вычитается из сдвинутого результата сложения; если результат сложения был отрицательный, то делитель добавляется к сдвинутому результату сложения. Затем определяется цифра частного, снова сдвигается результат вычитания или сложения, сносится следующая цифра делимого и, в зависимости от знака предыдущего результата вычитания (сложения), происходит добавление или вычитание делителя и т.д., пока не будет снесена последняя цифра делимого и получена младшая цифра частного.

Если результат последнего вычитания (сложения) положительный, то он и есть остаток. Если результат отрицательный, то остаток получается добавлением к результату делителя. Остатку присваивается знак делимого. Чтобы совместить процедуры добавления делителя и присвоения остатку знака, в алгоритме удобно все операции производить не с модулем делителя, а с отрицательным делителем. Тогда остаток получается по формуле:

остаток = результат - делитель, если знак делимого плюс;

остаток = делитель - результат, если знак делимого минус.

Частному присваивается знак по алгебраическим правилам.

При сдвиге делимого или результата сложения (вычитания) влево старшую цифру делимого можно не сохранять. Знак результата пробного вычитания не всегда верно отражается в знаковом разряде. Он определяется наличием или отсутствием переноса из старшего разряда результата:

если есть перенос, то знак +;

если нет переноса, то знак -.

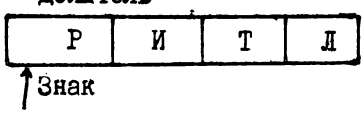
Знак всех последующих результатов вычитания (сложения) всегда получается истинным, ибо они по модулю всегда не превышают делитель.

Пример. Для простоты делимое взято 16-разрядным, а делитель 8-разрядным.

| | | | | |
|---|---|---|---|---|
| 0010 | 1111 | 1100 | 1001 | Делимое |
| 0111 | 0111 | | | Делитель |
| 1000 | 1001 | | | Отрицательный делитель |
| 0101 | 1111 | 1001 | 0010 | Сдвиг делимого на 1 разряд влево |
| +1000 | 1001 | | | |
| 1110 | 1000 | 1001 | 0010 | Результат пробного вычитания отрицателен. Частное равно 0 |
| 1101 | 0001 | 0010 | 0100 | Сдвиг |
| -1000 | 1001 | | | |
| 0100 | 1000 | 0010 | 0100 | Результат положителен. Частное равно 01 |
| 1001 | 0000 | 0100 | 1000 | Сдвиг |
| +1000 | 1001 | | | |
| 0001 | 1001 | 0100 | 1000 | Результат положителен. Частное равно 011 |
| 0011 | 0010 | 1001 | 0000 | Сдвиг |
| +1000 | 1001 | | | |
| 1011 | 1011 | 1001 | 0000 | Результат отрицателен. Частное равно 0110 |
| 0111 | 0111 | 0010 | 0000 | Сдвиг |
| -1000 | 1001 | | | |
| 1110 | 1110 | 0010 | 0000 | Результат отрицателен. Частное равно 0110 0 |
| 1101 | 1100 | 0100 | 0000 | Сдвиг |
| -1000 | 1001 | | | |
| 0101 | 0011 | 0100 | 0000 | Результат отрицателен. Частное равно 0110 01 |
| 1010 | 0110 | 1000 | 0000 | Сдвиг |
| +1000 | 1001 | | | |
| 0010 | 1111 | 1000 | 0000 | Результат положителен. Частное равно 0110 011 |
| 0101 | 1111 | 0000 | 0000 | Сдвиг |
| +1000 | 1001 | | | |
| 1110 | 1000 | 0000 | 0000 | Результат отрицателен. Частное равно 0110 0110 |

Диаграмма алгоритма. Выборка делителя из основной или локальной памяти происходит на регистры Р,И,Т,Л. Делитель выбирается справа налево. Если делитель - положительное число, то на регистры заносится его дополнение (28В2, 28Д2).

Делитель



Для формата RX, независимо от знака делителя, занесение его на регистры осуществляют одни и те же микрокоманды с выполнением лишь разных косвенных функций (сложение или вычитание). На триггере БС4 запоминается знак делителя:

ОБС4 - знак делителя +,

IБС4 - знак делителя -.

В случае формата RX знак делителя определяется по исходному состоянию триггера ТЗН.

28С2, 28С3. Содержимое регистров МФЕ заносится в локальную память. В регистр Г, который дальше используется как счетчик по модулю 8, заносится код III. Если спецификация RI правильная, то происходит чтение двух старших байтов делимого.

28В4. Два старших байта делимого (0-I) заносятся без изменения на регистры Ф и Е. В зависимости от знака делимого "+" или "-" в регистр косвенной функции заносится соответственно косвенная функция "+" или "-".

28С5, 28Д4, 28С6, 28Д5, 28Д6. Из нуля вычитается или к нулю прибавляется делимое по косвенной функции, записанной в блоке 28С4. Байты 6-7 полученного результата помещаются в рабочую область ЛП по адресу F6, байт 5 - на регистр З и в локальную память по адресу F5, а байты 0-4 сдвигаются на один разряд влево и заносятся на регистры Ф,Е,У,Д и Н (сдвигается уже положительное делимое).

Делимое

| | | | | | | в ЛП | |
|---|---|---|---|---|---|---|---|
| Ф | Е | У | Д | Н | З | F6 | F7 |
| 0 | I | 2 | 3 | 4 | 5 | 6 | 7 |

Делимое, равное максимальному отрицательному числу, обнаруживается по знаку "минус" результата взятия дополнения от отрицательного делимого. Здесь же адрес четного регистра делимого (три старших разряда) заносится на регистр П, освобождая регистр Д, и на триггере БС5 фиксируется знак исходного делимого.

ОБС5 - знак делимого +,

IБС5 - знак делимого -.

28Д7. Четыре старших байта делимого (регистры Ф,Е,У,Д), сдвинутых влево, складываются с делителем (регистры Р,И,Т,Л) по косвенной функции. Для выполнения пробного вычитания используется часть основного цикла деления. Указателем того, что эта часть цикла выполняет пробное вычитание, служит единичное состояние триггера БС2. После выполнения пробного вычитания этот триггер сбрасывается в 0.

28Д8. В блоке 28С2 в счетчик (регистр Г) был записан код III, который должен получаться после формирования нулевого разряда частного. Разряд частного еще не формировался, поэтому, чтобы не нарушить счет после пробного вычитания, управление передается блоку 28С9 - на сдвиг результата, минуя вычитание единицы из Г.

28А7, 28С9. Четыре байта результата сложения (вычитания, регистры Ф,Е,У и Д) сдвигаются на один разряд влево с учетом разряда, выдвинутого из регистра Н или регистра З. При каждом сдвиге содержимого регистров Н и З в младший разряд этих регистров заносится очередная цифра частного.

Указателем регистра служит триггер БС3:

ОБС3 - сдвигается содержимое регистра Н,

IБС3 - сдвигается содержимое регистра З.

Таким образом, в регистрах Н и З формируется сначала старшее полуслово частного (триггер НДД=0), а затем младшее полуслово частного (триггер НДД=I). Старшее полуслово частного затем запоминается в локальной памяти по адресу Е4, а младшее - по адресу Е6:

Частное

| Е4 | Е5 | Е6 | Е7 |
|---|---|---|---|
| 0 | I | 2 | 3 |

↑ Знак

В зависимости от знака предыдущего результата в регистр косвенной функции (подготовка к сложению остатка и делителя или вычитания) записывается косвенная функция:

| Остаток "+" | Остаток "-" |
|---|---|
| A+B | B-A |

28В8. Регистр Г является счетчиком до восьми количества сдвигов содержимого регистра Н или З и сформированных разрядов одного байта частного.

| N-й разряд частного | Содержимое регистра Г после помещения N-го разряда частного в регистр Н или в регистр З |
|---|---|
| 0 | III |
| I | IIO |
| 2 | IOI |
| 3 | IOO |
| 4 | OII |
| 5 | OIO |
| 6 | OOI |
| 7 | OOO |

Из счетчика до восьми (регистр Г) вычитается единица.

28А8. 28В8. Проверяется на нуль содержимое регистра Г. Нуль в регистре Г означает конец формирования байта частного в регистре Н или в регистре З. После того, как старшее полуслово частного сформировано, в регистрах Н и З читается из локальной памяти младшее полуслово делимого, и триггер НДД устанавливается в единицу.

Признаком конца деления является нулевое состояние регистра Г и единичное состояние триггера НДД. Если конец деления, то триггер БС5 устанавливается в состояние, обратное знаку делимого:

IБС5 - знак делимого "+",

OБС5 - знак делимого "-", при этом в этих же микрокомандах триггер БС4 установится в состояние, указывающее на совпадение и несовпадение знаков делимого и делителя:

OБС4 - совпадение,

IБС4 - несовпадение.

29В1. Здесь же по знакам делимого и делителя, как и в блоках 28А8 и 28С8, в соответствии с вышеизложенными правилами, устанавливаются триггеры БС4 и БС5.

29В2. Если знак частного по алгебраическим правилам должен быть плюсом (OБС4), т.е. максимальным отрицательным числом частное быть не может, то сбрасывается индикатор отсутствия прерывания (триггер БС3).

29В3,29В4. Производится сложение четырех младших байтов делимого с делителем. На регистрах Ф,Е,У и Д формируется остаток операции и параллельно в рабочую часть ЛП по адресам Е4 и Е6 записывается максимальное отрицательное число - частное.

29В7, 29В8, 29С7, 29Д6, 29Д7, 29Е7. Из регистра П три разряда (номер четного общего регистра делимого), дополненные четвертым справа нулем, записываются в регистр Д и составляют номер общего регистра остатка.

29С8, 29С9, 29Д8. При нулевом состоянии триггера БС4 (знаки делимого и делителя совпадают) частное из рабочей ЛП записывается в нечетный общий регистр делимого, по IБС4 (знаки не совпадают) в нечетный общий регистр делимого записывается дополнение полученного частного.

Замещение делимого частным выполняют одни и те же микрокоманды по косвенной функции, которая заносится в регистр косвенной функции до начала формирования конечного частного в зависимости от состояния триггера БС4:

ОБС4 - (B-A),

IБС4 - (A-B).

Адрес нечетного общего регистра делимого получается модификацией в регистре Д адреса четного регистра.

Устанавливается признак отсутствия прерывания (триггер БС3).

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| М | Адрес следующей команды | Не используется |
| Ф<br>Е | | Хранят нулевой и первый байты делимого и результата вычитания (добавления) делителя |
| Г | Если формат RX, то $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ | Счетчик до восьми количества сдвигов делимого (счетчик формирования одного байта частного) |
| Р<br>И | Если RR , то R 0010 | Хранят старшее полуслово делителя |
| П | | Хранит три старших разряда номера четного общего регистра |
| Т | Если формат RX, то $\langle X_2 \rangle + \langle B_2 \rangle + D_2$, если RR , то $\langle R_2 0010 \rangle$ | Хранит старший байт младшего полуслова делителя |
| У | R10010 | Хранит второй байт делимого и результата вычитания (добавления) делителя |
| Д | Код операции | Хранит третий байт делимого и результата вычитания (добавления) делителя |
| Л | Если формат RX, то $R_I X_2$, Если формат RR, то $R_I R_2$ | Хранит младший байт младшего полуслова делителя |
| Н<br>З | Байты делителя, прочитанные по адресу, находящемуся в регистре ГРИ | Хранят четвертый, пятый, шестой и седьмой байты делимого |
| БС2 | 0 | Организация выборки делимого. Индикатор пробного вычитания |
| БС3 | 0 | Организация выборки делимого. Индикатор сдвигаемого регистра: регистр Н или регистр З. Индикатор переполнения |
| БС4 | I для формата RX<br>0 для формата RR | Индикатор знака делителя до получения всех цифр частного.<br>Индикатор знака частного |
| БС5 | 0 | Индикатор знака делимого |
| ТЗН | Знак делителя для формата RX | Знак результата вычитания (добавления) делителя. Знак делителя для формата RX |
| НДД | - | Индикатор формирования старшего и младшего полуслова частного |
| ЛП<br>Е4-Е7 | - | Запоминание частного |
| ЛП<br>4- 7 | - | Запоминание четвертого, пятого, шестого и седьмого байтов делимого |

9.I. Микропрограмма RRI8

Команды

| Название | Мнем. | Формат | | | | Вход | Время |
|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | | |
| ЗАГРУЗКА | LR | RR | I8 | $R_I$ | $R_2$ | II0 | I0 |
| ЗАГРУЗКА ПОЛОЖИТЕЛЬНАЯ | LPR | RR | I0 | $R_I$ | $R_2$ | I00 | I5 |
| ЗАГРУЗКА ОТРИЦАТЕЛЬНАЯ | LNR | RR | II | $R_I$ | $R_2$ | I02 | I5 |
| ЗАГРУЗКА И ПРОВЕРКА | LTR | RR | I2 | $R_I$ | $R_2$ | I04 | I2 |
| ЗАГРУЗКА ДОПОЛНЕНИЯ | LCR | RR | I3 | $R_I$ | $R_2$ | I06 | I2 |

Операнды. Число с фиксированной запятой длиной в слово, расположенное в общем регистре с номером $R_2$.

Результат. Общий регистр $R_I$, содержащий заданное число с фиксированной запятой (команды LR, LTR) или абсолютное значение заданного числа (команда LPR), или дополнительный код абсолютного значения заданного числа (команда LNR), или дополнительный код заданного числа (команда LCR).

Число в общем регистре $R_2$ сохраняется. Устанавливается признак результата (исключение составляет выполнение команды LR):

0 - результат равен нулю (LPR, LNR,LTR,LCR);

I - результат меньше нуля (LNR,LTR,LCR);

2 - результат больше нуля (LPR,LTR,LCR);

3 - переполнение (LPR,LCR).

Прерывания. Переполнение с фиксированной запятой: результат выполнения команды LPR или LCR превосходит размер слова (заданное число равно максимальному отрицательному числу). Общий регистр $R_I$ содержит усеченный результат.

Алгоритм. Формирование содержимого общего регистра $R_I$ производится, начиная с младшего полуслова регистра.

Диаграмма алгоритма. Младшие и старшие полуслова результата формируются поочередно одними и теми же микрокомандами. Настройка подпрограммы (3IB2) на выполнение необходимой операции осуществляется установкой косвенной функции

(+) для команд LR и LTR,

(-) для команды LCR.

При выполнении команд LPR и LNR устанавливаемая косвенная функция зависит от знака заданного числа с фиксированной запятой:

| Команда | Знак числа+ | Знак числа- |
|---|---|---|
| LPR | (+) | (-) |
| LNR | (-) | (+) |

Установку косвенной функции для обеих команд выполняют одни и те же микрокоманды, при этом для LPR анализируется непосредственно сам знак числа, а для LNR анализируется инвертированный знак числа.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| И | $R_2$0010 | Хранит адрес младшего полуслова общего регистра $R_2$ |
| У | $R_I$0010 | Хранит текущий адрес полуслов общего регистра $R_I$ |
| Д | К0 | Хранит адрес старшего полуслова общего регистра $R_2$ |
| Л | $R_I$ $R_2$ | Используется при формировании адреса в регистре Д |
| БС2 | 0 | Индикатор команд, в которых устанавливается признак результата:<br>0 – команда LR |
| БС3 | 0 | Индикатор формируемого полуслова общего регистра $R_I$:<br>0 – младшее полуслово,<br>I – старшее полуслово |
| БС5 | 0 | Индикатор команды LNR:<br>I – команда LNR |

9.2. Микропрограмма $\zeta$ 5898

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | байт 3 | байт 4 | | |
| ЗАГРУЗКА | L | RX | 58 | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I3I | 9 |
| ЗАГРУЗКА ГРУППОВАЯ | LM | RS | 98 | $R_I$ | $R_3$ | $B_2$ | $D_2$ | I7I | 4+8 U |

Операнды. Команда L : логическая информация длиной в слово, расположенная в основной памяти по адресу $\langle X_2 \rangle + \langle B_2 \rangle + D_2$. Команда LM: логическая информация, расположенная в основной памяти, начиная с адреса $\langle X_2 \rangle + \langle B_2 \rangle + D_2$, и занимающая столько слов, сколько нужно для загрузки общих регистров, начиная с регистра с номером $R_I$ и кончая регистром с номером $R_3$ в порядке возрастания номеров. При этом за регистром с номером I5 следует регистр с номером 0.

Результат. Команда L : общий регистр с номером $R_I$, загруженный заданным словом памяти. Команда LM: общие регистры, начиная с регистра с номером $R_I$ и кончая регистром с номером $R_3$, загруженные заданными словами основной памяти. Информация в памяти не изменяется. Признак результата остается без изменений.

Прерывания. Спецификация: адрес $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ не кратен 4. (В случае команды L – реализовано в микропрограмме ВЫБОР). Адресация. Защита памяти по чтению.

Алгоритм. Загрузка общих регистров происходит, начиная со старшего полуслова общего регистра $R_I$. Для команды L полагается, что $R_3 = R_I$.

Диаграмма алгоритма. См. диаграмму.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ для команды L,<br>$\langle B_2 \rangle + D_2$ – для LM | Хранит текущий адрес основной памяти |
| Т | – | Хранит адрес локальной памяти, где находятся полуслова |
| Д | Код операции | Хранит адрес локальной памяти, где находится старшее полуслово общего регистра $R_3$ ($R_I$ для команды L ) |
| БС2 | 0 | Указатель: при формировании адреса полуслова общего регистра:<br>I – формируется адрес старшего полуслова регистра<br>0 – формируется адрес младшего полуслова регистра |

9.3. Микропрограмма RX

Команды

| Название | Мнем. | Формат | | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | байт 3 | байт 4 | | |
| ЗАГРУЗКА ПОЛУСЛОВА | LH | RX | 48 | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I30 | I0 |

Операнды. Число с фиксированной запятой длиной в полуслово, расположенное в основной памяти по адресу $\langle X_2 \rangle + \langle B_2 \rangle + D_2$.

Результат. Число с фиксированной запятой длиной в слово, расположенное в общем регистре с номером $R_I$ и равное исходному. Младшее полуслово этого числа совпадает с исходным, а старшее полуслово заполнено значением знакового разряда исходного числа. Признак результата не изменяется.

Прерывания. Спецификация: адрес $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ не кратен 2. Адресация. Защита памяти по чтению.

Алгоритм. Вначале на регистрах процессора формируется число с фиксированной запятой длиной в слово. Затем оно записывается в общий регистр, начиная со старшего полуслова.

Диаграмма алгоритма. См. диаграмму.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ | Хранит адрес основной памяти |
| Т | – | Хранит старший байт полуслова памяти |

Продолжение

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| У | $R_I$0010 | Хранит адрес младшего полуслова общего регистра |
| Д | К0 | Хранит адрес старшего полуслова общего регистра |
| Л | $R_I$ $X_2$ | Хранит младший байт полуслова памяти |
| БС2 | 0 | Индикатор конца операции:<br>0 – конец операции |

9.4. Микропрограмма RX4I

Команды

| Название | Мнем. | Формат | | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | байт 3 | байт 4 | | |
| ЗАГРУЗКА АДРЕСА | A | RX | 4I | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I22 | 6 |

Операнды. 24 – разрядный адрес, определяемый полями команды $X_2$, $B_2$ и $D_2$ и равный $X_2$ + + $B_2$ + $D_2$ (перенос за 24-й разряд игнорируется), расположенный на регистрах Д, Р и И процессора.

Результат. Общий регистр с номером $R_I$, содержащий в разрядах 0–7 нули, а в разрядах 8–31 – заданный адрес. Заданный адрес не проверяется на наличие в памяти системы, защищенность или целочисленность границ. Признак результата не изменяется.

Алгоритм. Формирование содержимого регистра $R_I$ производится, начиная с младшего полуслова регистра.

Диаграмма алгоритма. См. диаграмму.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| У | $R_I$0010 | Хранит текущий адрес полуслов общего регистра |
| Р<br>И | Два младших байта адреса | Хранит 2 младших байта адреса |
| БС2 | 0 | Индикатор конца операции:<br>0 – конец операции |
| Д | Старший байт адреса | Хранит старший байт адреса |

9.5. Микропрограмма ICRX

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | байт 3 | байт 4 | | |
| ПРОЧИТАТЬ СИМВОЛ | IC | RX | 43 | $R_I$ $X_2$ | $B_2$ | $D_2$ | I26 | 4 |

Операнды. Логическая информация длиной в байт, расположенная по адресу $\langle X_2 \rangle + \langle B_2 \rangle + D_2$.

Результат. 8 разрядов из оперативной памяти по адресу $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ помещены в младшие разряды общего регистра с номером $R_I$. Остальные разряды регистра не меняются. Признак результата остается без изменения.

Прерывания. Адресация. Защита по чтению (прерывание происходит на этапе выборки команды в микропрограмме ВЫБОР).

Алгоритм. Один байт основной памяти переписывается в разряды 24-31 общего регистра с номером $R_I$.

Диаграмма алгоритма. В зависимости от четности адреса $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ четный или нечетный байт основной памяти через регистр З записывается в байт общего регистра (блоки 35B1, 35C2).

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г Р И | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ | Хранит адрес основной памяти |
| У | $R_I$0010 | Хранит адрес байтов общего регистра |
| Д | К0 | Хранит четный байт второго операнда |

## 10. СТАНДАРТНАЯ СИСТЕМА КОМАНД. ОПЕРАЦИИ ЗАПИСИ

10.1 Микропрограмма ST

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | байт 3 | байт 4 | | |
| ЗАПИСЬ В ПАМЯТЬ | ST | RX | 50 | $R_I$ $X_2$ | $B_2$ | $D_2$ | I2I | 9 |

Операнды. Логическая информация длиной в слово, расположенная в общем регистре с номером $R_I$.

Результат. 32 разряда из общего регистра, записанные в слово памяти по адресу $\langle X_2 \rangle$ + + $\langle B_2 \rangle$ + $D_2$. Содержимое общего регистра не изменяется. Признак результата остается без изменения.

Прерывания. Адресация. Защита по записи. Спецификация: адрес $\langle X_2 \rangle$ + $\langle B_2 \rangle$ + $D_2$ не кратен 4 (реализовано в микропрограмме ВЫБОР).

Алгоритм. Четыре байта общего регистра, начиная со старших, переписывается в поле основной памяти с адресом $\langle X_2 \rangle$ + $\langle B_2 \rangle$ + $D_2$.

Диаграмма алгоритма. См. диаграмму.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г Р И | $\langle X_2 \rangle$ + $\langle B_2 \rangle$ + $D_2$ | Хранит текущий адрес основной памяти |
| У | $R_I$0010 | Хранит текущий адрес локальной памяти |
| ВС4 | I | Индикатор конца операции: 0 – конец операции |

10.2. Микропрограмма STCRX

Команды

| Название | Мнем. | Формат | | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | байт 3 | байт 4 | | |
| ЗАПИСЬ В ПАМЯТЬ СИМВОЛА | STC | RX | 42 | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I24 | 5 |

Операнды. Логическая информация длиной в байт, расположенная в разрядах 24-31 общего регистра с номером $R_I$.

Результат. 8 младших разрядов общего регистра, записанные по адресу $\langle X_2 \rangle$ + $\langle B_2 \rangle$ + $D_2$. Содержимое общего регистра не изменяется. 24 старших разряда в операции не участвуют. Признак результата остается без изменения.

Прерывания. Адресация. Защита по чтению и записи.

Алгоритм. Один байт общего регистра переписывается в поле основной памяти с адресом $\langle X_2 \rangle$ + $\langle B_2 \rangle$ + $D_2$.

Диаграмма алгоритма. В зависимости от четности адреса $\langle X_2 \rangle$ + $\langle B_2 \rangle$ + $D_2$ байт из общего регистра записывается в байт основной памяти через регистр Н (адрес четный) или регистр З (адрес нечетный) (блок 38С2).

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г Р И | $\langle X_2 \rangle$ + $\langle B_2 \rangle$ + $D_2$ | Хранит адрес основной памяти |
| У | $R_I$0010 | Хранит адрес общего регистра |
| Д | Код операции | Буфер для записываемого байта |

10.3. Микропрограмма STM

Команды

| Название | Мнем. | Формат | | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | | байт 3 | байт 4 | | |
| ЗАПИСЬ В ПАМЯТЬ ПОЛУСЛОВА | STH | RX | 40 | $R_1$ | $X_2$ | $B_2$ | $D_2$ | 120 | 4 |

Операнды. Логическая информация длиной в полуслово, расположенная в младших 16 разрядах общего регистра с номером $R_1$.

Результат. 16 младших разрядов общего регистра, записанные в полуслово памяти по адресу $\langle X_2 \rangle + \langle B_2 \rangle + D_2$. Содержимое общего регистра не изменяется. 16 старших разрядов в операции не участвуют. Признак результата остается без изменения.

Прерывания. Адресация. Защита по записи. Спецификация: адрес $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ не кратен 2.

Алгоритм. Два байта общего регистра переписываются в поле основной памяти с адресом $\langle X_2 \rangle + \langle B_2 \rangle + D_2$.

Диаграмма алгоритма. См. диаграмму.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г Р И | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ | Хранит адрес основной памяти |
| У | $R_1$0010 | Хранит адрес локальной памяти |

10.4. Микропрограмма STMRS

Команды

| Название | Мнем. | Формат | | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | | байт 3 | байт 4 | | |
| ЗАПИСЬ В ПАМЯТЬ ГРУППОВАЯ | STM | RS | 90 | $R_1$ | $R_3$ | $B_2$ | $D_2$ | 161 | 4+10U |

Операнды. Содержимое группы общих регистров, начиная с регистра с номером $R_1$ и кончая регистром с номером $R_3$. Номера регистров указывают порядок, в котором их содержимое записывается в память. За регистром с номером 15 следует регистр с номером 0.

Результат. Содержимое группы общих регистров, помещенное в соответствующее количество слов основной памяти по адресу $\langle B_2 \rangle + D_2$. Содержимое общих регистров не изменяется. Признак результата остается без изменения.

Прерывания. Адресация. Защита по записи. Спецификация: адрес $\langle B_2 \rangle + D_2$ не кратен 4.

Алгоритм. Содержимое общего регистра $R_1$, начиная со старших байтов, читается и затем записывается в основную память. Затем номер регистра $R_1$ сравнивается с номером регистра $R_3$, и если номера совпадают, то на этом операция оканчивается. При несовпадении номеров регистров номер R увеличивается на единицу и операция продолжается.

Диаграмма алгоритма. См. диаграмму.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle B_2 \rangle + D_2$ | Хранит текущий адрес основной памяти |
| Т | - | Хранит адрес общего регистра $R_3$ |
| У | - | Хранит текущие адреса общих регистров, начиная с регистра $R_I$ |
| Л | $R_I$ $R_3$ | Хранит $R_I$ $R_3$ |
| БС2 | 0 | Индикатор записи четырех байтов общего регистра:<br>I - обработаны четыре байта регистра |

## II. СТАНДАРТНАЯ СИСТЕМА КОМАНД. ОПЕРАЦИИ И, ИЛИ, ИСКЛЮЧАЮЩИЕ ИЛИ

### II.I. Микропрограмма NRORR

Команды

| Название | Мнем. | Формат | | | | Вход | Время |
|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | | |
| И | NR | RR | I4 | $R_I$ | $R_2$ | I08 | I2 |
| ИЛИ | OR | RR | I6 | $R_I$ | $R_2$ | I0C | I2 |
| ИСКЛЮЧАЮЩИЕ ИЛИ | XR | RR | I7 | $R_I$ | $R_2$ | I0E | I2 |

Операнды. Оба операнда - логические величины, не имеющие внутренней структуры, расположенные в общих регистрах с номерами $R_I$ и $R_2$.

Результат. Команда NR: поразрядное логическое произведение (И) первого и второго операндов, расположенное в общем регистре с номером $R_I$. Команда OR: поразрядная логическая сумма (ИЛИ), расположенная в общем регистре с номером $R_I$. Команда XR: поразрядная логическая сумма по модулю 2, расположенная в общем регистре с номером $R_I$.

Установка признака результата:

0 - все разряды результата равны нулю;

I - результат имеет разряды, равные единице.

Алгоритм. Команда NR. Поразрядное логическое произведение осуществляется согласно табл.8.

Таблица 8

| Разряд первого операнда | Разряд второго операнда | Результат логического произведения |
|---|---|---|
| 0 | 0 | 0 |
| 0 | I | 0 |
| I | 0 | 0 |
| I | I | I |

Команда OR. Поразрядное логическое сложение осуществляется согласно табл. 9.

Таблица 9

| Разряд первого операнда | Разряд второго операнда | Результат логического сложения |
|---|---|---|
| I | 0 | I |
| I | I | I |
| 0 | I | I |
| 0 | 0 | 0 |

Команда XR. Поразрядное суммирование по модулю 2 осуществляется согласно табл. IO.

Таблица IO

| Разряд первого операнда | Разряд второго операнда | Результат сложения по модулю 2 |
|---|---|---|
| I | I | 0 |
| 0 | I | I |
| I | 0 | I |
| 0 | 0 | 0 |

Диаграмма алгоритма. Настройка подпрограммы (блок 42CI) состоит в занесении косвенной функции логического умножения или логического сложения, или сложения по модулю 2.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| И | $R_2$00IO | Хранит текущий адрес второго операнда |
| У | $R_I$00IO | Хранит текущий адрес первого операнда |
| Л | $R_I R_2$ | Хранит четный байт второго операнда |
| Д | Код операции | Хранит нечетный байт второго операнда |
| БС2 | 0 | Индикатор конца операции: 0 – конец операции |
| Н | $\langle R_2$00IO$\rangle$ | Информационный регистр |
| З | $\langle R_2$00II$\rangle$ | Информационный регистр |

II.2. Микропрограмма NOX

Команды

| Название | Мнем. | Формат | | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | байт 3 | байт 4 | | |
| И | N | RX | 54 | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I29 | I2 |
| ИЛИ | O | RX | 56 | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I2D | I2 |
| ИСКЛЮЧАЮЩЕЕ ИЛИ | X | RX | 57 | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I2F | I2 |

Операнды. Оба операнда – логические величины, не имеющие внутренней структуры. Первый операнд расположен в общем регистре с номером $R_I$. Второй – в оперативной памяти по адресу $\langle X_2\rangle$ + + $\langle B_2\rangle$ + $D_2$.

Результат. Команда N : поразрядное логическое произведение (И) первого и второго операндов, расположенное в общем регистре с номером $R_I$. Команда O: поразрядная логическая сумма (ИЛИ), расположенная в общем регистре с номером $R_I$. Команда X: поразрядная сумма по модулю 2, расположенная в общем регистре с номером $R_I$.

Установка признака результата:

0 - все разряды результата равны нулю;

I - результат имеет разряды, равные единице.

Прерывания. Адресация. Защита по чтению. Спецификация: адрес второго операнда не кратен 4 (реализовано в ВЫБОР).

Алгоритм. Команда N . Поразрядное логическое произведение осуществляется согласно табл.II.

Таблица II

| Разряд первого операнда | Разряд второго операнда | Результат логического произведения |
|---|---|---|
| 0 | 0 | 0 |
| 0 | I | 0 |
| I | 0 | 0 |
| I | I | I |

Команда O. Поразрядное логическое сложение осуществляется согласно табл. I2.

Таблица I2

| Разряд первого операнда | Разряд второго операнда | Результат логического сложения |
|---|---|---|
| I | 0 | I |
| I | I | I |
| 0 | I | I |
| 0 | 0 | 0 |

Команда X. Поразрядное суммирование по модулю 2 осуществляется согласно табл. I3.

Таблица I3

| Разряд первого операнда | Разряд второго операнда | Результат сложения по модулю 2 |
|---|---|---|
| I | I | 0 |
| 0 | I | I |
| I | 0 | I |
| 0 | 0 | 0 |

Диаграмма алгоритма. Настройка подпрограммы (блок 43CI) состоит в занесении косвенной функции логического умножения или логического сложения, или сложения по модулю 2.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ | Хранит текущий адрес второго операнда |
| Т | - | Хранит четный байт второго операнда |
| У | $R_I$00I0 | Хранит текущий адрес первого операнда |
| Л | $R_I X_2$ | Хранит нечетный байт второго операнда |
| БС4 | I | Индикатор конца операции:<br>"0" - конец операции |
| Н | $\langle\langle X_2 \rangle + \langle B_2 \rangle + D_2 \rangle$ | Информационный регистр |
| З | $\langle\langle X_2 \rangle + \langle B_2 \rangle + D_2 + I \rangle$ | Информационный регистр |

II.3. Микропрограмма NIOIX

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | байт 3 | байт 4 | | |
| И | NI | SI | 94 | $I_2$ | $B_I$ | $D_I$ | I69 | 5 |
| ИЛИ | OI | SI | 96 | $I_2$ | $B_I$ | $D_I$ | I6D | 5 |
| ИСКЛЮЧАЮЩЕЕ ИЛИ | XI | SI | 97 | $I_2$ | $B_I$ | $D_I$ | I6F | 5 |

Операнды. Оба операнда - логические величины, не имеющие внутренней структуры, Первый операнд длиной в один байт расположен по адресу $\langle B_I \rangle + D_I$. Второй находится непосредственно в команде, в поле $I_2$.

Результат. Команда NI: поразрядное логическое произведение (И) первого и второго операндов, расположенное по адресу $\langle B_I \rangle + D_I$. Команда OI: поразрядная логическая сумма (ИЛИ), расположенная по адресу $\langle B_I \rangle + D_I$. Команда XI: поразрядная сумма по модулю 2, расположенная по адресу $\langle B_I \rangle + D_I$.

Установка признака результата:

0 - все разряды результата равны нулю;

I - результат имеет разряды, равные единице.

Прерывания. Адресация. Защита по чтению и записи.

Алгоритм. Команда NI. Поразрядное логическое произведение осуществляется согласно табл. I4.

Таблица I4

| Разряд первого операнда | Разряд второго операнда | Результат логического произведения |
|---|---|---|
| 0 | 0 | 0 |
| 0 | I | 0 |
| I | 0 | 0 |
| I | I | I |

Команда OI. Поразрядное логическое сложение осуществляется согласно табл. I5.

Таблица I5

| Разряд первого операнда | Разряд второго операнда | Результат логического сложения |
|---|---|---|
| I | 0 | I |
| I | I | I |
| 0 | I | I |
| 0 | 0 | 0 |

Команда XI. Поразрядное суммирование по модулю 2 осуществляется согласно табл. I6.

Таблица I6

| Разряд первого операнда | Разряд второго операнда | Результат сложения по модулю 2 |
|---|---|---|
| I | I | 0 |
| 0 | I | I |
| I | 0 | I |
| 0 | 0 | 0 |

Диаграмма алгоритма. Настройка подпрограммы (блок 44CI) состоит в занесении косвенной функции логического умножения или логического сложения, или сложения по модулю 2.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle B_I \rangle + D_I$ | Хранит адрес первого операнда |
| Л | $I_2$ | Хранит непосредственный операнд |

II.4. Микропрограмма SSД46

Команды

| Название | Мнем. | Формат | | | | | | | Вход |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | байт 3 | байт 4 | байт 5 | байт 6 | |
| И | NC | SS | D4 | L | $B_I$ | $D_I$ | $B_2$ | $D_2$ | I88 |
| ИЛИ | OC | SS | D6 | L | $B_I$ | $D_I$ | $B_2$ | $D_2$ | I8C |
| ИСКЛЮЧАЮЩЕЕ ИЛИ | XC | SS | D7 | L | $B_I$ | $D_I$ | $B_2$ | $D_2$ | I8E |

Время: $I3+3^N$, если оба адреса четные или нечетные; $5+5^N$ - в остальных случаях.

Операнды. Оба операнда - логические величины, не имеющие внутренней структуры. Они расположены в полях основной памяти. Адрес поля первого операнда - $D_I + \langle B_I \rangle$. Адрес поля второго операнда - $D_2 + \langle B_2 \rangle$. Длина полей определяется полем L команды: минимальная длина полей - I байт (L =0); максимальная длина полей - 256 байтов (L =255). Допускается перекрытие полей.

Результат. Поразрядное логическое произведение (команда LC) или поразрядная логическая сумма (команда OC), или поразрядная сумма по модулю 2 (команда XC) первого и второго операнда, расположенные в поле первого операнда.

Устанавливается признак результата:

0 - все разряды результата равны нулю;

I - результат имеет разряды, равные единице.

Прерывания. Адресация. Защита памяти по чтению.

Алгоритм. Команда NC: поразрядное логическое произведение осуществляется согласно табл. I7.

Таблица I7

| Разряд первого операнда | Разряд второго операнда | Результат логического произведения |
|---|---|---|
| 0 | 0 | 0 |
| 0 | I | 0 |
| I | 0 | 0 |
| I | I | I |

Команда OC. Поразрядное логическое сложение осуществляется согласно табл. I8

Таблица I8

| Разряд первого операнда | Разряд второго операнда | Результат логического сложения |
|---|---|---|
| I | 0 | I |
| I | I | I |
| 0 | I | I |
| 0 | 0 | 0 |

Команда XC. Поразрядное суммирование по модулю 2 осуществляется согласно табл. I9.

Таблица I9

| Разряд первого операнда | Разряд второго операнда | Результат сложения по модулю 2 |
|---|---|---|
| I | I | 0 |
| 0 | I | I |
| I | 0 | I |
| 0 | 0 | 0 |

Диаграмма алгоритма. Так как чтение операндов из основной памяти производится четно-нечетными парами байтов независимо от четности адреса, то это порождает четыре последовательности обработки операндов в зависимости от четности адресов их полей.

В микропрограмме имеются два основных цикла, которые реализуют случаи "чет-чет" и "нечет-чет". Случаи "нечет-нечет" и "чет-нечет" используют, соответственно, первый и второй циклы после предварительной обработки старших байтов операндов и модификации адресов их полей.

Настройка микропрограммы на выполнение определенной команды осуществляется установкой соответствующей косвенной функции (45AI, 45BI, 45CI).

45A6, 45C5, 45Д7, 45E4. При переходе от положительных значений величины L к отрицательным устанавливается в единицу триггер ПФ.

45A7. Байт, находящийся в регистре 3, запоминается на входе арифметическо-логического блока.

45C5. Анализируются триггеры ЧЕТ и ПФ.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г Р И | $\langle B_1 \rangle + D_1$ | Хранит текущий адрес байтов первого операнда |
| П Т У | $\langle B_2 \rangle + D_2$ | Хранит текущий адрес байтов второго операнда |
| Л | L | Счетчик количества необработанных байтов |
| Д | К0 | Используется для хранения байтов второго операнда |
| БС2 | 0 | Индикатор наличия необработанных байтов в цикле "чет-чет" |
| БС3 | 0 | Рабочий индикатор при модификации адреса байтов второго операнда |
| БС4 | 0 | Индикатор наличия необработанных байтов в цикле "нечет-чет" |
| Вход В БА | - | Используется также для хранения нечетного байта второго операнда в цикле "чет-чет" |

## I2. СТАНДАРТНАЯ СИСТЕМА КОМАНД. ОПЕРАЦИИ ПЕРЕКОДИРОВКИ И ПРЕОБРАЗОВАНИЯ

### I2.I. Микропрограмма TRTRT

Команды

| Название | Мнем. | Формат | | | | | | | Вход |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | байт 3 | байт 4 | байт 5 | байт 6 | |
| ПЕРЕКОДИРОВАТЬ | TR | SS | $D_C$ | L | $B_1$ | $D_1$ | $B_2$ | $D_2$ | I98 |
| ПЕРЕКОДИРОВАТЬ И ПРОВЕРИТЬ | TRT | SS | DD | L | $B_1$ | $D_1$ | $B_2$ | $D_2$ | I9A |

Время. Команда TR: $11+10^N$. Команда TRT: $11+10^N$ - если все выбранные байт-функции равны 0; 2I+9B - если есть хотя бы один ненулевой байт-функция.

Операнды. Поле аргумента с адресом старшего байта, равным $\langle B_1 \rangle + D_1$. Поле функции (словарь) - адрес старшего байта равен $\langle B_2 \rangle + D_2$. Байт-функцией данного байт-аргумента считается байт поля функции с адресом, равным полному начальному адресу поля функции плюс сам байт-аргумент, который прибавляется к младшим разрядам адреса поля функции.

Оба поля имеют одинаковую длину, определяемую полем команды: минимальная длина полей - I байт (L =0); максимальная длина полей - 256 байтов (L =255). Все значения байтов являются допустимыми.

Результат. Команда TR. Поле аргумента, каждый байт-аргумент которого замещен своим байт-функцией. Признак результата не изменяется. Поле функции не изменяется, если оно не перекрывается с полем аргумента.

Команда TRT. В общем регистре I, в его младших 24 разрядах, адрес первого байт-аргумента имеет ненулевой байт-функцию и старшие восемь разрядов при этом не меняются. Ненулевой байт-функция заносится в младшие 8 разрядов общего регистра 2. Разряды 0-23 этого регистра не меняются. Поле аргумента и поле функции не изменяется. Устанавливается признак результата:

0 - у всех байт-аргументов нулевые байт-функции;

I - ненулевой байт-функция встретился до окончания поля аргумента;

2 - ненулевой байт-функция у последнего байт-аргумента.

Прерывания. Адресация. Защита по чтению (команда ПЕРЕКОДИРОВАТЬ И ПРОВЕРИТЬ). Защита по записи и чтению (команда ПЕРЕКОДИРОВАТЬ).

Алгоритм. Байты первого операнда выбираются для перекодировки один за другим слева направо. Каждый байт-аргумент добавляется к младшим разрядам полного начального адреса второго операнда. Сумма используется как адрес байт-функции. Если команда TR, то байт-функции замещают последовательно все исходные байт-аргументы. Если команда TRT, то байт-функция анализируется.

Если байт-функция - нуль и при этом обработан последний байт-аргумент, операция заканчивается установкой признака результата. Если не последний, то происходит выборка следующего байт-аргумента и его байт-функции. Если байт-функция отлична от нуля, то операция заканчивается занесением адреса байт-аргумента в общий регистр I, байт-функции - в общий регистр 2. Устанавливается признак результата.

Диаграмма алгоритма. Признак результата (блок 48С5) устанавливается путем непосредственной засылки кодов 00, 0I, I0 в разряды 6-7 регистра БС.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| М<br>Ф<br>Е | Адрес следующей команды | Хранит текущий адрес байт-функции |
| Г<br>Р<br>И | $\langle B_I \rangle + D_I$ | Хранит текущий адрес байт-аргумента |
| П<br>Т<br>У | $\langle B_2 \rangle + D_2$ | Хранит адрес второго операнда |
| Л | L | Индикатор конца операции |
| Д | Код операции | Хранит байт-функцию |
| БС2 | 0 | Для организации цикла при засылке СчАК (блок 48С1). Индикатор четности адреса второго операнда (блок 48В3) |
| БС5 | I | Тип команды:<br>0 - команда ПЕРЕКОДИРОВАТЬ;<br>I - команда ПЕРЕКОДИРОВАТЬ и ПРОВЕРИТЬ (48В3)<br>Индикатор четности адреса байт-функции (48В4) |

I2.2. Микропрограмма RX4E

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | байт 3 | байт 4 | | |
| ПРЕОБРАЗОВАНИЕ В ДЕСЯТИЧНУЮ | CVD | RX | 4E | $R_1X_2$ | $B_2$ | $D_2$ | I3C | $36+23H+4H^2$ |

Операнды. Число с фиксированной запятой длиной в слово, расположенное в общем регистре с номером $R_1$.

Результат. Упакованное десятичное число (изображение исходного числа с фиксированной запятой в десятичной системе счисления), расположенное в поле основной памяти с адресом $\langle X_2 \rangle + \langle B_2 \rangle + D_2$. Длина поля равна восьми байтам. Код знака определяется I2-м разрядом ССП. Если в разряде I2 – нуль, вырабатываются коды, соответствующие ДКОИ, а именно: плюс – IIOO, минус – IIOI. Если в разряде I2 – единица, вырабатываются коды, соответствующие КОИ-8, а именно: плюс – IOIO, минус – IOII.

Признак результата остается без изменения.

Прерывания. Спецификация: адрес $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ не кратен 8. Адресация. Защита памяти по записи.

Алгоритм. Перевод числа из двоичной системы счисления в десятичную осуществляется по формуле

$$(\ldots 0+Xк)\cdot 2 + Xк+1 )\cdot 2+\ldots +X_{30})\cdot 2+X_{31} ,$$

где Хк – старшая двоичная цифра старшего ненулевого байта модуля исходного числа с фиксированной запятой К=0,8; I6 или 24.

Диаграмма алгоритма. Имеются четыре цикла, каждый из которых переводит только один байт. Выделение двоичных цифр Хк реализовано сдвигом влево содержимого регистра, в который занесен байт модуля исходного числа. Десятичное умножение на два реализовано десятичным сложением содержимого регистра З с самим собой. Десятичные переносы из регистра З суммируются в регистре Т. Суммирование двоичное или десятичное – в зависимости от того, в каком цикле происходит перевод, и т.д. В случае двоичного суммирования происходит корректировка результата, если он оказывается не десятичным числом. Регистр, в котором происходит двоичное суммирование, одновременно выполняет роль счетчика количества сдвигов байта модуля исходного числа.

Пример. Перевод старшего значащего байта в цикле I.

IIIIIIII – старший значащий байт (255 в десятичной системе счисления).

| Регистры | Т (двоичное удвоение) | З (десятичное удвоение) | У (сдвиг) |
|---|---|---|---|
| Первоначальное содержимое регистров | 0000 000I | 0000 0000 | IIII IIII |
| **Выполнение цикла:** | | | |
| **первое** | 0000 00I0 | 0000 000I | IIII IIIO |
| **второе** | 0000 0I00 | 0000 00II | IIII IIOO |
| **третье** | 0000 I000 | 0000 0III | IIII I000 |
| **четвертое** | 000I 0000 | 000I 0I0I | IIII 0000 |
| **пятое** | 00II 0000 | 00II 000I | IIIO 0000 |
| **шестое** | 0I00 0000 | 0II0 00II | IIOO 0000 |
| **седьмое** | I000 000I | 00I0 0III | I000 0000 |
| **восьмое** | 0000 00I0 | 0I0I 0I0I | 0000 0000 |

Результат перевода мы получим в регистрах Т, З и У, причем в регистре Т получаем десятичные цифры.

Ниже приводятся пояснения к некоторым блокам диаграммы алгоритма.

49В4. Дополнение запоминается в локальной памяти по адресам 04-07.

49Д4, 49Д5. Гашение незначащих байтов реализуется засылкой старшего значащего байта всегда в один и тот же регистр У. Следующие байты, если они есть, засылаются последовательно в регистры Л, Н и Д. Модуль двоичного числа читается из регистра, если оно положительное, или из локальной памяти, если оно отрицательное.

Сброс триггеров БС3, БС2 или ПКФ означает, что значащих цифр в двоичном числе не более трех, двух или одной.

49С5. Первый цикл выполняется, когда хотя бы один байт не равен нулю. При этом в регистре У производится сдвиг влево, в регистре З - десятичное сложение и в регистре Т - двоичное сложение (Т используется также как счетчик до восьми).

Десятичный частичный результат на выходе из этого цикла содержится в регистрах Т и З.

49В6. Второй цикл вводится, когда в исходном числе не менее двух значащих байтов (триггер-индикатор ПКФ в I). В этом цикле выполняется сдвиг влево содержимого регистра Л, десятичное сложение - в регистрах З и Т, и двоичное сложение - в регистре У, который одновременно служит счетчиком до восьми.

Десятичный частичный результат на выходе теперь находится на регистрах У, Т, З.

49Д7, 49С7. При использовании третьего цикла (триггер БС2 в I) сдвиг влево на I разряд производится в регистре Н, десятичное сложение - в регистрах З,Т,У и двоичное сложение (он же и счетчик) - в регистре Л.

Десятичный частичный результат на выходе из этого цикла содержится в регистрах Л,У,Т и З.

Так как здесь регистр Л используется как двоичный, то в этом регистре может получиться на выходе из цикла не десятичная цифра (0А-0F ). Тогда производится корректировка содержимого регистра Л посредством двоичного сложения с константой 06.

49В8, 49С8. Четвертый цикл (триггер БС3 в I) выполняет сдвиг в регистре Д, десятичное сложение в регистрах З,Т,У,Л и двоичное сложение в регистре Н (регистр Н является и счетчиком). На выходе из цикла 4 десятичный результат содержится в регистрах Н,Л,У,Т,З, причем регистр Н может содержать как десятичную цифру, так и величину 0А-I5. Если величина содержимого регистра Н находится в диапазоне 0А-0F , то она двоично складывается с константой 06 корректировки. Если содержимое регистра Н находится в промежутке I0-I5, то корректировка его производится посредством десятичного сложения с константой 06.

Окончательный десятичный результат находится на регистрах Д,Л,У,Т и З.

49С9. Последний блок выполнения микропрограммы, независимо от выполняемых циклов, - блок загрузки десятичного результата с регистров Д,Л,У,Т,З в оперативную память по адресу 02.

Регистры и триггеры.

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ | Адресный регистр основной памяти |
| Т<br>У<br>Д<br>Л<br>З<br>Н | -<br>RIOOIO<br>КО<br>$R_I X_2$<br>= | Рабочие регистры, в которых происходит сдвиг байта исходного числа и получаются десятичные цифры |

Продолжение

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| ПКФ<br>БС2<br>БС3 | <br>0<br>0 | Индикаторы количества значащих байтов в исходном числе:<br>ПКФ=I-количество значащих цифр I<br>БС2=I-количество значащих цифр 2<br>БС3=I-количество значащих цифр 3 |
| БС4 | 0 | Организация получения дополнения исходного числа.<br>Организация выхода из микропрограммы |

I2.3. Микропрограмма RX4F

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | байт 3 | байт 4 | | |
| ПРЕОБРАЗОВАНИЕ В ДВОИЧНУЮ | CVB | RX | 4F | $R_I$ $X_2$ | $B_2$ | $D_2$ | I3E | 32+26D |

Операнды. Упакованное десятичное число, расположенное в поле основной памяти по адресу $\langle X_2\rangle + \langle B_2\rangle + D_2$. Длина поля равна двойному слову.

Результат. Число с фиксированной запятой в общем регистре $R_I$, являющееся изображением десятичного числа в двоичной системе счисления. Отрицательные числа представляются в дополнительном коде. Наименьшее десятичное число, которое может быть помещено в общий регистр, равно 2I47483648, наибольшее - 2I47483647. Для любого десятичного числа, выходящего за указанные пределы, в регистр помещаются 32 младших разряда числа с фиксированной запятой.

Признак результата остается без изменения.

Прерывания. Спецификация: адрес основной памяти $\langle X_2\rangle + \langle B_2\rangle + D_2$ не кратен 8. Данные: код цифры не соответствует кодам 0000-I00I или код знака отличен от кодов I0I0-IIII. Деление с фиксированной запятой: десятичное число не может быть точно представлено в общем регистре в форме числа с фиксированной запятой. Адресация. Защита памяти по чтению.

Алгоритм. Перевод числа из десятичной системы счисления в двоичную осуществляется по формуле

$$(\ldots(0+Xк)\ I0+Xк+I)\ I0+\ldots + X_{I4})\ I0 + X_{I5}\ ,$$

где Хк - старшая отличная от нуля цифра десятичного числа;

К = 0;I ... I4.

Диаграмма алгоритма. Выделение десятичных цифр Хк реализовано пересылкой десятичных цифр, прочитанных из основной памяти, по одной в младшие четыре разряда регистра Д (блок 50С2). Исключение составляет самая старшая значащая цифра, которая непосредственно двоично добавляется к нулевому содержимому общего регистра (блок 50А2). Засылкой цифр с регистров Н и З в регистр Д управляют триггеры БС2 и БС3:

ОБС3, ОБС2 - засылаются старшие разряды Н;

ОБС3, IБС2 - засылаются младшие разряды Н;

IБС3, IБС2 - засылаются старшие разряды З;

IБС3, ОБС2 - засылаются младшие разряды З.

После засылки в регистр Д разрядов регистра З каждого полуслова уменьшается на единицу значение счетчика (регистр П) полуслов операнда.

Умножение на $10_{(10)}$ частичного результата, находящегося в регистре $R_I$, происходит по формуле:

$$\langle R_I \rangle \cdot 8 + \langle R_I \rangle \cdot 2,$$

где $\langle R_I \rangle$ - содержимое регистра $R_I$ в следующей последовательности. Содержимое регистра $R_I$ сдвигается на один разряд вправо и загружается в регистры Т,Ф,Е и Л (блок 50В2). Выдвинутый разряд запоминается в старшем разряде буфера перекоса (блок 50Д2). Затем происходит одновременное сложение содержимого регистра Д с перекошенным содержимым регистров Т,Ф,Е и Л и сдвигаемым на один разряд влево содержимым общего регистра $R_I$ (блок 50С3).

Пример. Для простоты операнд и регистр $R_I$ взяты длиной в один байт.

| | | |
|---|---|---|
| 0001 | 1001 | Десятичное число |
| 0000 | 0001 | Регистр $R_I$ после добавления старшей десятичной цифры |
| 0000 | 0000 | $\langle R_I \rangle$, сдвинутое вправо на 1 разряд |
| | 1000 | Буфер перекоса |
| 0000 | 1001 | Регистр Д |
| 0000 | 0010 | $\langle R_I \rangle$, сдвинутое на 1 разряд влево |
| 0001 | 0011 | Результат |

50В4. Выход из цикла перевода десятичных цифр происходит по нулевому значению регистра П.

50В5, 50В6. Так как в регистре $R_I$ получается модуль двоичного числа без знака, то значение $\geqslant 2^{31}$ при положительном операнде означает переполнение.

50С5, 50В7. Функцию косвенного индикатора переполнения выполняет пятый разряд регистра Л, основного - триггер БС5.

Регистры и триггеры

| Регистры и триггеры | Состояние после выборки | Назначение |
|---|---|---|
| М<br>Ф<br>Е | Адрес следующей команды | Используются для запоминания второго и третьего сдвинутых на один разряд вправо байтов частичного результата (из $R_I$) |
| Г<br>Р<br>И | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ | Хранит адрес полуслов второго операнда |
| П | | Счетчик полуслов второго операнда (счет до четырех) |
| Т | | Используется для запоминания сдвинутого на 1 разряд вправо старшего байта частичного результата (из $R_I$) |
| У | $R_I$0010 | Хранит адрес полуслов первого операнда ($R_I$) |
| Д | К0 | Для хранения обрабатываемой десятичной цифры |
| Л | $R_I$ $X_2$ | Запоминается сдвинутый на 1 разряд вправо младший байт частичного результата (из $R_I$). Косвенный индикатор переполнения (разряд 5) |

Продолжение

| Регистры и триггеры | Состояние после выборки | Назначение |
|---|---|---|
| БС2 | 0 | Управляют засылкой одной из четырех цифр (02) с регистров Н и З в регистр Д. |
| БС3 | 0 | |
| БС4 | 0 | Индикатор наличия переноса после сложения (х8) значения младшего байта частичного результата с десятичной цифрой |
| БС5 | 0 | Индикатор переполнения; потери значащих цифр в процессе выполнения микропрограммы |

I2.4. Микропрограмма РАСК

Команды

| Название | Мнем. | Формат | | | | | | | Вход |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | байт 3 | байт 4 | байт 5 | байт 6 | |
| УПАКОВАТЬ | РАСК | SS | F2 | $L_1$ $L_2$ | $B_1$ | $D_1$ | $B_2$ | $D_2$ | I85 |
| РАСПАКОВАТЬ | UNPK | SS | F3 | $L_1$ $L_2$ | $B_1$ | $D_1$ | $B_2$ | $D_2$ | I87 |

Время: $I4 + 4\,N_I$.

Операнды. Логическая информация переменной длины, расположенная в основной памяти в поле с адресом $\langle B_2 \rangle + D_2$. Длина поля $L_2 + I$ байтов.

Результат. Младший байт операнда, в котором старшая и младшая тетрады меняются местами, заносится в младший байт поля результата (адрес поля результата $\langle B_1 \rangle + D_1$; длина поля результата $L_1 + I$ байтов).

Для команды РАСК старшие тетрады следующих байтов операнда игнорируются, а младшие тетрады записываются в байты поля результата по две тетрады в один байт.

Для команды UNPK обе тетрады следующих байтов операнда дополняются слева кодом в соответствии с I2-м разрядом ССП и записываются в байты поля результата.

Микропрограмма кончает работу формированием старшего байта результата. Если необходимо, операнд дополняется слева нулями. Тетрады операнда, не помещающиеся в поле результата, игнорируются.

Прерывания. Адресация. Защита памяти по записи и чтению.

Алгоритм. Каждый байт результата записывается в память сразу же, как только выбраны необходимые байты операнда.

Диаграмма алгоритма. 5IA5,5IA7,5IB5, 5IC5, 5IC7, 5ID5, 5ID7, 5IB7. Для выхода из циклов организованы счетчики длины первого и второго операнда. Счетчики занимают старшую и младшую тетраду регистра Л. Вычитание константы (I или 2) производится всегда из старших четырех разрядов счетчика, причем содержимое регистра Л при этом передается накрест.

Выход из цикла производится, когда после очередного вычитания константы появляется единица переноса (содержимое счетчика становится отрицательным).

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г Р И | $\langle B_1 \rangle + D_1 + L_1$ | Хранит текущий адрес байтов поля результата |
| П Т У | $\langle B_2 \rangle + D_2 + L_2$ | Хранит текущий адрес байтов операнда |
| Д | К0 | Для запоминания байтов операнда. В UNPK используется для хранения кода зоны: F0 или 50 |
| Л | $L_1 L_2$ | Счетчик числа обработанных байтов операнда и полученных байтов результата |
| БС3 | 0 | Индикатор четности адреса конца операнда:<br>0БС3 - адрес четный,<br>1БС3 - адрес нечетный |
| БС4 | 1 | Индикатор четности адреса конца поля результата:<br>0БС4 - четный,<br>1БС4 - нечетный |
| БС5 | 1 | Индикатор команды РАСК (1БС5) или UNPK (0БС5). В команде UNPK указывает вид зоны:<br>1БС5 - зона "F",<br>0БС5 - зона "5" |

## 13. СТАНДАРТНАЯ СИСТЕМА КОМАНД. ОПЕРАЦИИ ПЕРЕКЛЮЧЕНИЯ СОСТОЯНИЯ

### 13.1. Микропрограмма МПРОГ

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | | | | |
| УСТАНОВИТЬ МАСКУ ПРОГРАММЫ | SPM | RR | 04 | $R_1$ | | | 148 | 7 |

Операнды. Новые признаки результата и маска программы, расположенные в разрядах 2-7 общего регистра $R_1$.

Результат. Операнд замещает признак результата и маску программы в ССП.

Диаграмма алгоритма. См. диаграмму.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| У | - | Хранит адрес операнда |
| Л | $R_I$ (в старшем полубайте) | Рабочие регистры |
| Д | КО | |
| БС6<br>БС7 | Признак результата ССП | Заносится новый признак результата в ССП |
| ЛП<br>8С | Маска программы ССП | Заносится новая маска программы в ССП |

I3.2. Микропрограмма МСИСТ

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | байт 3 | байт 4 | | |
| УСТАНОВИТЬ МАСКУ СИСТЕМЫ | SSM | SI | 80 | | $B_I$ | $D_I$ | I60 | 9 |

Операнды. Новая маска системы, расположенная в байте основной памяти по адресу $\langle B_I \rangle + D_I$.

Результат. Операнд замещает маску системы в ССП. В соответствии с разрядами 0,I,2 и 7 маски устанавливаются управляющие триггеры маски системы ССП БР0, БРI, БР2 и БР7.

Прерывания. Привилегированная операция: ВЧУ находится в состоянии "задача" (I5-й разряд ССП равен I).

Адресация. Защита памяти по чтению.

Диаграмма алгоритма. См. диаграмму.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle B_I \rangle + D_I$ | Хранит адрес операнда |
| Л | - | Рабочие регистры |
| Д | КО | |
| БР0<br>БРI<br>БР2<br>БР7 | Маска системы | Заносятся разряды 0,I,2,7 новой маски системы |
| ЛП<br>88 | Маска системы ССП | Заносится новая маска системы |

I3.3. Микропрограмма СУПЕР

Команды

| Название | Мнем. | Формат | | | Вход | Время |
|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | |
| ОБРАЩЕНИЕ К СУПЕРВИЗОРУ | SVC | RR | 0A | $R_I$ $R_2$ | I54 | 5 |

Операнды. Код прерывания, расположенный в байте 2 команды. Шестнадцатиричная константа 22 – адрес ячейки основной памяти, в которой хранится код прерывания старого ССП прерывания при обращении к супервизору.

Результат. Код прерывания засылается в регистр Д. Нуль засылается в регистр Л. Адрес ячейки основной памяти, равный 22, засылается в регистр ГРИ ВЧУ. Передается управление микропрограмме ЗСССП по адресу 038.

Примечание. Микропрограмма ЗСССП записывает содержимое регистров Л и Д в разряды I6-3I старого ССП, которое сохраняется в памяти в процессе прерывания и формирует остальное содержимое старого ССП. Микропрограмма КЗССП загружает новое ССП прерывания при обращении к супервизору. Описания микропрограмм ЗСССП и КЗССП даны в EI3.055.00I T0I.

Диаграмма алгоритма. См. диаграмму.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | - | Хранит адрес основной памяти (22) |
| Л | $R_I$ $R_2$ | В него засылается 0 |
| Д | К0 | В него засылается код прерывания |

## I4. СТАНДАРТНАЯ СИСТЕМА КОМАНД. ОПЕРАЦИИ ПЕРЕСЫЛКИ

I4.I. Микропрограмма SI92

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | байт 3 | байт 4 | | |
| ПЕРЕСЫЛКА | MVI | SI | 92 | $I_2$ | $B_I$ | $D_I$ | I65 | 4 |

Операнды. Восьмиразрядный байт, заданный в поле $I_2$ команды.

Результат. Восьмиразрядный байт, заданный в поле $I_2$ команды, записывается в основную память по адресу $\langle B_I \rangle + D_I$.

Признак результата остается без изменения.

Алгоритм. В зависимости от четности адреса основной памяти одновременно с пересылкой операнда регенерируется нечетный (адрес - четный) или четный (адрес - нечетный) байты основной памяти.

Диаграмма алгоритма. После выборки команды MVI операнд уже находится в регистре Л ВЧУ.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle B_2 \rangle + D_I$ | Хранит адрес байта основной памяти |
| Л | $I_2$ | Хранит $I_2$ |

I4.2. Микропрограмма SS

Команды

| Название | Мнем. | Формат | | | | | | | Вход |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | байт 3 | байт 4 | байт 5 | байт 6 | |
| ПЕРЕСЫЛКА | M VC | SS | $D_2$ | L | $B_I$ | $D_I$ | $B_2$ | $D_2$ | I84 |

Время: I4+2N , если оба адреса четные или нечетные;

7+5N - в остальных случаях.

Операнды. Логическая информация переменной длины, расположенная в поле основной памяти по адресу $\langle B_2 \rangle + D_2$. Длина поля определяется полем L команды. Минимальная длина поля I байт (L =0). Максимальная длина поля 256 байтов (L =255).

Результат. Байты операнда переписаны в поле основной памяти по адресу $\langle B_I \rangle + D_I$.

Признак результата остается без изменения.

Прерывания. Адресация. Защита памяти по чтению и записи.

Алгоритм. Пересылка байтов операнда идет слева направо, начиная со старших байтов. Если адреса полей операнда и результата оба четные или нечетные, то пересылается сразу по два байта.

Диаграмма алгоритма. Так как чтение операндов из основной памяти происходит четно-нечетными парами байтов независимо от четности адресов, то это порождает четыре последовательности обработки операндов.

В микропрограмме имеются два основных цикла, которые реализуют случаи "чет-чет" и "нечет-чет". Случаи "нечет-нечет" и "чет-нечет" используют соответственно первый или второй циклы после предварительной пересылки старшего байта.

58А4, 58В4, 58Д6, 58Е8. При переходе от положительных значений величины к отрицательным устанавливается в единицу триггер ПКФ.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle B_I \rangle + D_I$ | Хранит текущий адрес байтов результата |
| П<br>Т<br>У | $\langle B_2 \rangle + D_2$ | Хранит текущий адрес байтов операнда |

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Л | L | Счетчик необработанных байтов |
| Д | – | Запоминание нечетного байта операнда |
| БС2 | 0 | Рабочий индикатор при модификации адресов операндов и результата |

14.3. Микропрограмма D3DI

Команды

| Название | Мнем. | Формат | | | | | | | Вход |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | байт 3 | байт 4 | байт 5 | байт 6 | |
| ПЕРЕСЫЛКА ЦИФР | MVN | SS | DI | L | $B_I$ | $D_I$ | $B_2$ | $D_2$ | I82 |
| ПЕРЕСЫЛКА ЗОН | MVZ | SS | D3 | L | $B_I$ | $D_I$ | $B_2$ | $D_2$ | I86 |

Время: 20,5+3,5N , если адреса полей операндов оба четные или нечетные; I6+5N – в остальных случаях.

Операнды. Оба операнда – логическая информация переменной длины. Адрес поля первого операнда – $\langle B_I \rangle + D_I$. Адрес поля второго операнда – $\langle B_2 \rangle + D_2$. Длина полей определяется полем L команды. Максимальная длина полей I байт (L =0). Максимальная длина полей 256 байтов (L = 255). Допустимо любое перекрытие полей.

Результат. Младшие (команда MVN ) или старшие (команда MVZ ) тетрады каждого байта первого операнда замещены младшими (команда MVN ) или старшими (команда MVZ ) тетрадами соответствующих байтов второго операнда.

Признак результата остается без изменения.

Прерывания. Адресация. Защита памяти по чтению и записи.

Алгоритм. Пересылка младших или старших тетрад байтов второго операнда в байты первого операнда происходит слева направо, начиная со старших байтов. Если адреса полей операндов оба четные или нечетные, то замещаются младшие (старшие) тетрады сразу двух байтов первого операнда.

Диаграмма алгоритма. Так как чтение операндов из основной памяти производится всегда четно-нечетными парами байтов, то это порождает четыре последовательности обработки операндов в зависимости от четности адресов полей операндов (блок 59C2).

В микропрограмме имеются два основных цикла, которые реализуют случаи "чет-чет" и "нечет-чет". Случаи "нечет-нечет" и "чет-нечет" используют соответственно первый или второй циклы.

59A7, 59C5, 60B4, 60C5, 6IB8, 6IД8. При переходе от положительных значений величины L к отрицательным должен установиться в единицу триггер ПКФ.

59CI. Здесь же СчАК запоминается в локальной памяти.

59B9. Предварительно СчАК восстанавливается в регистре МФЕ.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г Р И | $\langle B_I \rangle + D_I$ | Хранит текущий адрес байтов первого операнда |

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| П<br>Т<br>У | $\langle B_2 \rangle + D_2$ | Хранит текущий адрес байтов второго операнда |
| М<br>Ф<br>Е | Адрес следующей команды | Не используются<br>Счетчик количества необработанных байтов |
| Д<br>Л | -<br>L | Рабочие регистры, в которых запоминаются младшие или старшие тетрады байтов второго операнда |
| БС2 | 0 | Используется при анализе на четность адресов полей операндов |
| БС3 | 0 | Индикатор выполняемой команды:<br>0БС3 - MVZ ,<br>1БС3 - MVN |
| БС4 | 0 | Используется при модификации текущих адресов байтов операндов |

I4.4. Микропрограмма RINGO

Команды

| Название | Мнем. | Формат | | | | | | | Вход |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | байт 3 | байт 4 | байт 5 | байт 6 | |
| ПЕРЕСЫЛКА СО СДВИГОМ | MVO | SS | $F_1$ | $L_1$ $L_2$ | $B_1$ | $D_1$ | $B_2$ | $D_2$ | I83 |

Время: $9\,N_1$, если $L_1 \leqslant L_2$

$5+6\,N_2+3\,N_1$, если $L_1 > L_2$.

Операнды. Логическая информация переменной длины, расположенная в двух полях основной памяти с адресами $\langle B_1 \rangle + D_1$ и $\langle B_2 \rangle + D_2$ соответственно. Длина поля первого операнда равна $L_1 + 1$, длина поля второго операнда $L_2 + 1$.

Результат. Младшие четыре разряда первого операнда пристраиваются в качестве младших разрядов ко второму операнду. Результат помещается в поле первого операнда. Значащие цифры второго операнда, не помещающиеся в поле первого операнда, игнорируются. Если надо, второй операнд дополняется нулями слева от старших разрядов.

Алгоритм. Поля операндов обрабатываются справа налево. Формирование байтов результата осуществляется путем сдвига второго операнда на четыре разряда влево.

Диаграмма алгоритма

В блоке 62BI признаки устанавливаются следующим образом:

| ГРИ | ПТУ | БС3 | БС4 |
|---|---|---|---|
| чет | чет | I | I |
| чет | нечет | I | 0 |
| нечет | чет | 0 | 0 |
| нечет | нечет | 0 | I |

При формировании байтов результата используется режим перекоса.

Байт операнда берется из регистра Н, если адрес его четен, или из регистра $z$, если адрес его нечетен.

Запись байта результата происходит через регистр Н, если адрес байта результата четен, в противном случае, через регистр $z$.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г Р И | $\langle B_1 \rangle + D_1 + L_1$ | Хранит текущий адрес байтов первого операнда |
| П Т У | $\langle B_2 \rangle + D_2 + L_2$ | Хранит текущий адрес байтов второго операнда |
| Л | $L_1$ $L_2$ | Счетчик количества пересылаемых значащих байтов второго операнда |
| Д | КО | Буфер байта результата. Счетчик пересылки нулевых байтов |
| БС2 | 0 | 1БС2 - признак, определяющий, надо ли дополнять нулями первый операнд |
| БС3 | 0 | Индикатор четности адресов байтов операндов |
| БС4 | 1 | |

## 15. СТАНДАРТНАЯ СИСТЕМА КОМАНД. ОПЕРАЦИИ ПЕРЕХОДОВ

### 15.1. Микропрограмма ИСИСЧ

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | байт 3 | байт 4 | | |
| ВЫПОЛНИТЬ | EX | RX | 44 | $R_1$ $X_2$ | $B_2$ | $D_2$ | 128 | 15+ время выполнения подчиненной команды |
| ПЕРЕХОД ПО СЧЕТЧИКУ | BCTR | RR | 06 | $R_1$ $R_2$ | | | 14C | 19 |
| ПЕРЕХОД ПО СЧЕТЧИКУ | BCT | RX | 46 | $R_1$ $X_2$ | $B_2$ | $D_2$ | 12C | 13 |

Операнды. Команда EX: константа модификации в разрядах 24-31 общего регистра $R_1$, если $R_1 \neq 0$; адрес подчиненной команды, равный $\langle X_2 \rangle + \langle B_2 \rangle + D_2$; продвинутый адрес команды EX в разрядах 40-63 ССП.

Команды BCTR, BCT: счетчик - число с фиксированной запятой - в общем регистре $R_1$; адрес перехода, расположенный в команде BCTR в разрядах 8-31 общего регистра $R_2$ (если $R_2 \neq 0$), или равный $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ в команде BCT.

Результат. Команда EX. Код операции подчиненной команды заносится в регистр Д ВЧУ и в локальную память процессора по адресу 98, если подчиненная команда имеет формат SS . В регистре Д результат логического сложения разрядов 8–I5 подчиненной команды с константой модификации или разряды 8–I5 подчиненной команды, если $R_I$ = 0. Если подчиненная команда имеет формат, отличный от RR, то в регистр МФЕ процессора занесен адрес подчиненной команды, увеличенный на 2, а исходное значение МФЕ (продвинутый адрес команды EX) переписано в рабочую область локальной памяти по адресам 8Д, 8Е и 8F. Происходит передача управления микропрограмме выборки команд ВЫБОР по адресу, определяемому значением старшей тетрады кода операции подчиненной команды:

| Старшая тетрада КО | Адрес микропрограммы ВЫБОР |
|---|---|
| 0 | IA9 |
| I | 0A3 |
| 2 | 0A4 |
| 3 | 0A4 |
| 4 | 0E7 |
| 5 | 0A6 |
| 6 | 0A7 |
| 7 | 0A7 |
| 8 | 0CC |
| 9 | 0CC |
| D | IBA |
| F | IBE |

Примечание. В микропрограмме ВЫБОР доводится до конца выборка подчиненной команды. В ССП возвращается подвинутый адрес команды EX и передается управление микропрограмме, реализующей соответствующую подчиненную команду. Таким образом, если команда, на которую указывает команда EX, представляет собой команду ПЕРЕХОДА с возвратом или если возникает прерывание, то будет запомнен продвинутый адрес и код длины команды EX. Если команда EX указывает на команду перехода и условия перехода удовлетворены, то продвинутый адрес команды EX в ССП будет замещен адресом перехода указанной команды.

Команды BCTR, BCT. Значение счетчика уменьшено на единицу, новое значение счетчика проверяется на равенство нулю. Заносится адрес перехода в разряды 40–63 ССП, если новое значение счетчика не нуль. В команде BCTR, если $R_I = R_2$, в качестве адреса перехода будет взято исходное значение общего регистра; если же $R_2$ = 0, то адрес в ССП не меняется.

Прерывания. (Возможны только при выполнении команды EX).

Спецификация: адрес подчиненной команды $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ не кратен 2.

Некорректность команды ВЫПОЛНИТЬ: подчиненной командой является тоже команда EX.

Операция: старшая тетрада кода операции подчиненной команды равна A, B, C или E. Эти коды операций не соответствуют никакой операции.

Адресация. Защита памяти по чтению.

Алгоритм. Выполнение команд BCTR, BCT и команды EX происходит совершенно независимо, то есть различными микрокомандами.

Диаграмма алгоритма. См. диаграмму.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Команда EX<br>М<br>Ф<br>Е | 40–63 разряды ССП | В него засылается адрес подчиненной команды |

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г Р И | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ | Хранит адрес подчиненной команды |
| Т | | Рабочий регистр |
| Л | $R_I$ $X_2$ | Разряды (или промодифицированные разряды) 8–I5 подчиненной команды |
| Д | КО команды EX | КО подчиненной команды |
| БС2 | 0 | Для всех форматов, кроме RR, IБС2 – признак подчиненной команды при переходе к микропрограмме ВЫБОР |
| Команды BCTR. BCT | | |
| М Ф Е | 40–63 разряды ССП | В него засылается адрес перехода |
| Г Р И | BCT: $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ | Хранит адрес перехода |
| БС2 | 0 | IБС2 – признак команды BCT |

I5.2. Микропрограмма ИНДЕК

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | байт 3 | байт 4 | | |
| ПЕРЕХОД ПО ИНДЕКСУ "БОЛЬШЕ" | BXH | Rs | 87 | $R_I$ $R_3$ | $B_2$ | $D_2$ | I6C | 28 |
| ПЕРЕХОД ПО ИНДЕКСУ "МЕНЬШЕ ИЛИ РАВНО" | BXLE | RS | 86 | $R_I$ $R_3$ | $B_2$ | $D_2$ | I6E | 28 |

Операнды. Индекс – число с фиксированной запятой, расположенное в общем регистре $R_I$.

Приращение – число с фиксированной запятой, расположенное в общем регистре $R_3$.

Константа сравнения – число с фиксированной запятой, расположенное в общем регистре $R_3$, если $R_3$ – нечетно, или в общем регистре с номером $R_3 + I$, если $R_3$ – четно.

Адрес перехода, равный $\langle B_2 \rangle + D_2$.

Результат. Новое значение индекса, равное алгебраической сумме исходного значения и приращения.

Сравнение нового значения индекса с константой. Если общие регистры индекса и константы совпадают, то в качестве константы берется исходное значение индекса.

Занесение адреса перехода в разряды 40–63 ССП, если новое значение индекса больше константы (BXH) или меньше, или равно константе (BXLE). В остальных случаях состояние ССП не изменяется.

Алгоритм. Сравнение с константой осуществляется параллельно с вычислением нового значения индекса и реализовано вычитанием нового значения индекса из константы. Переполнение, возникающее при сложении индекса с приращением, игнорируется.

Диаграмма алгоритма. Вычитание индекса из константы (блок 66А2) выполняется по косвенной функции (-). Триггеры ТЗН и ПЕР определяют результат этого вычитания.

| ПЕР | ТЗН | Результат |
|---|---|---|
| 0 | 0 | Константа больше индекса |
| 0 | I | Константа меньше индекса |
| I | 0 | Константа меньше индекса |
| I | I | Константа больше индекса |

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| М<br>Ф<br>Е | Разряды 40–63 ССП | В него заносится адрес перехода |
| Г<br>Р<br>И | $\langle B_2 \rangle + D_2$ | Хранит адрес перехода |
| Т | - | Используется в качестве адресного регистра |
| У | - | Хранят сформированные байты индекса |
| Д | К0 | Хранит сформированный байт индекса |
| БС2 | 0 | Индикатор окончания вычисления индекса |
| БС3 | 0 | 0 - BXH, I - BXLE |

I5.3. Микропрограмма ВОЗВР

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | байт 3 | байт 4 | | |
| ПЕРЕХОД С ВОЗВРАТОМ | BALR | RR | 05 | $R_I$ $R_2$ | | | I4A | 22 |
| ПЕРЕХОД С ВОЗВРАТОМ | BAL | RX | 45 | $R_I$ $X_2$ | $B_2$ | $D_2$ | I2A | I5 |

Операнды. Первый операнд: правые 32 разряда ССП, содержащие код длины команды, признак результата, маску программы и адрес команды BALR или BAL , увеличенный соответственно на 2 или на 4. (Если команды являются подчиненными команде ВЫПОЛНИТЬ, то код длины команды в ССП будет I0, а адрес команды - это адрес команды ВЫПОЛНИТЬ, увеличенный на 4).

Второй операнд: адрес перехода, находящийся в разрядах с 8 по 3I общего регистра $R_2$, если $R_2 \neq 0$ (BALR), или равный $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ (BAL ).

Результат. Первый операнд (разряды ССП) записывается в общий регистр $R_I$. В разряды ССП, с 40 по 63, записывается второй операнд (адрес перехода). Если в команде BALR $R_I = R_2$, то в ССП будет записано начальное содержимое разрядов 8–3I регистра $R_2$. Содержимое ССП не изменяется, если в команде BALR $R_2 = 0$.

Алгоритм. Для команды формата RR перед запоминанием информации возврата читается и запоминается на регистрах адрес перехода. Затем для обеих команд формируется информация возврата, состоящая из кода условия, кода длины команды, маски программы и продвинутого адреса команды.

Диаграмма алгоритма. См. диаграмму.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| М<br>Ф<br>Е | ССП разряды 40–63 | В него засылается адрес перехода |
| Г<br>Р<br>И | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ для RX | Хранит адрес перехода |
| Т | | Используется при формировании информации возврата |
| У<br>Д | КО | Используются в качестве адресных регистров при обращении к ЛП |
| БС2 | 0 | Признак формата выполняемой команды:<br>0 – RR<br>I – RX |
| БС6<br>БС7 | Признак результата | Признак результата |
| ЛП<br>8С<br>98 | Маска программы<br>Длина команды | Маска программы<br>Длина команды |

I5.4. Микропрограмма УСЛОВ

Команды

| Название | Мнем. | Формат | | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | байт 3 | байт 4 | | |
| УСЛОВНЫЙ ПЕРЕХОД | BCR | RR | 07 | $M_I$ | $R_2$ | | | I4E | II |
| УСЛОВНЫЙ ПЕРЕХОД | BC | RX | 47 | $M_I$ | $X_2$ | $B_2$ | $D_2$ | I2E | II |

Операнды. Признак результата, находящийся в разрядах 34, 35 ССП. Маска, содержащаяся в поле $M_I$ команд. Разряды маски, равные единице, указывают на необходимость проверки признака результата на определенные значения в соответствии с табл. 20.

Таблица 20

| Поле маски | Признак результата |
|---|---|
| I000 | 0 |
| 0I00 | I |
| 00I0 | 2 |
| 000I | 3 |

Например, маска 0I0I указывает на проверку признака результата на равенство единице или трем.

Адрес перехода, содержащийся в разрядах с 8 по 3I общего регистра $R_2$ (BCR), если $R_2 \neq 0$, или равный $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ (BC).

Результат. Если признак результата совпадает хотя бы с одним из проверяемых его значений, то адрес перехода записывается в разряды 40–63 ССП.

Если все биты маски равны нулю или признак результата не совпадает ни с одним из проверяемых его значений, то ССП не изменяется.

Алгоритм. По значению признака результата проверяется равенство единице соответствующего разряда маски.

Диаграмма алгоритма. Ветвление по признаку результата реализовано функциональным переходом.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| М<br>Ф<br>Е | Разряды 40–63 ССП | Заносится адрес перехода |
| Г<br>Р<br>И | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ для ВС | Адрес перехода для команды ВСР |
| Л | Для RR – $M_I R_2$; для RX – $M_I X_2$ | Хранит $M_I R_2$ или $M_I X_2$ |
| Д | КО | В нем формируется информация для функционального перехода |
| БС2 | 0 | Индикатор формата команды.<br>0 – RR<br>I – RX |
| БС6<br>БС7 | Признак результата | Признак результата |

## I6. СТАНДАРТНАЯ СИСТЕМА КОМАНД. ОПЕРАЦИИ СДВИГОВ

### I6.I. Микропрограмма SHIFT

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | байт 3 | байт 4 | | |
| СДВИГ ВПРАВО КОДОВ | SRL | RS | 88 | $R_I$ | $B_2$ | $D_2$ | I70 | 38 |
| СДВИГ ВЛЕВО КОДОВ | SLL | RS | 89 | $R_I$ | $B_2$ | $D_2$ | I72 | 41 |
| СДВИГ ВПРАВО | SRA | RS | 8A | $R_I$ | $B_2$ | $D_2$ | I74 | 40 |
| СДВИГ ВЛЕВО | SLA | RS | 8B | $R_I$ | $B_2$ | $D_2$ | I76 | 48 |
| СДВИГ ВПРАВО ДВОЙНОЙ КОДОВ | SRDL | RS | 8C | $R_I$ | $B_2$ | $D_2$ | I78 | 66 |
| СДВИГ ВЛЕВО ДВОЙНОЙ КОДОВ | SLDL | RS | 8D | $R_I$ | $B_2$ | $D_2$ | I7A | 66 |
| СДВИГ ВПРАВО ДВОЙНОЙ | SRDA | RS | 8E | $R_I$ | $B_2$ | $D_2$ | I7C | 69 |
| СДВИГ ВЛЕВО ДВОЙНОЙ | SLDA | RS | 8F | $R_I$ | $B_2$ | $D_2$ | I7E | 79 |

Операнды. Число с фиксированной запятой длиной в слово (команды SRA и SLA) или длиной в двойное слово (команды SRDA, SLDA).

Логическая информация длиной в слово (команды SRL и SLL ) или длиной в двойное слово (команды SRDL и SLDL ).

Операнд длиной в слово располагается в общем регистре с номером $R_I$, операнд длиной в двойное слово занимает четно-нечетную пару общих регистров с номером четного регистра, равным $R_I$. Величина сдвига определяется шестью младшими разрядами адреса $\langle B_2 \rangle + D_2$.

Результат. В командах SRL (SRDL ) и SLL (SLDL ) исходная логическая информация, сдвинутая соответственно влево или вправо на количество разрядов, определяемых величиной сдвига. Разряды, выдвигаемые за пределы регистра (пары регистров) теряются. Освобождаемые разряды заполняются нулями. Признак результата остается без изменения.

В командах SRA (SRDA) и SLA ( SLDA) знаковый разряд исходного числа в сдвигах не участвует. Освобождаемые разряды заполняются при сдвиге вправо значением знакового разряда, а при сдвиге влево - нулями. Если при сдвиге влево среди выдвинутых разрядов есть хотя бы один, отличный от знакового, то это означает переполнение. Признак результата принимает значения:

0, если результат равен нулю;
1, если результат меньше нуля;
2, если результат больше нуля;
3, если имеет место переполнение.

Прерывания. Переполнение с фиксированной запятой: при выполнении команды SLA или SLDA произошло переполнение. Операция завершается. Спецификация: в командах SRDL , SLDL , SRDA и SLDA значение $R_I$ нечетное. Операция подавляется. Операнды в памяти не изменяются.

Алгоритм. Если величина сдвига превышает 15, то производятся вначале сдвиги всего операнда на 16 разрядов до тех пор, пока величина сдвига не станет $\leqslant$ 15.

Сдвиги на величину $\leqslant$ 15 осуществляются по табл. 21.

Таблица 21

| Величина сдвига | Сдвиг влево | Сдвиг вправо |
|---|---|---|
| 0000 | | |
| 0001 | (1L ) | (1R) |
| 0010 | (1L ) + (1L ) | (4L·1L·8R) + (1L) |
| 0011 | (4L ) + (1R) | (4L·1L·8R) |
| 0100 | (4L ) | (4L·8R) |
| 0101 | (4L·1L) | (4L·8R)+(1R) |
| 0110 | (4L·1L)+(1L) | (1L·8R)+(1L) |
| 0111 | (8L)+(1R) | (1L·8R) |
| 1000 | (8L) | (8R) |
| 1001 | (8L·1L) | (8R)+(1R) |
| 1010 | (8L·1L)+(1L) | (4L·1L·16R)+(1L) |
| 1011 | (4L·8L)+(1R) | (4L·1L·16R) |
| 1100 | (4L·8L) | (4L·16R) |
| 1101 | (4L·1L·8L) | (4L·16R)+(1R) |
| 1110 | (4L·1L·8L)+(1L) | (16R)+(1L)+(1L ) |
| 1111 | (16L) + (1R) | (16R) + (1L) |

L означает сдвиг влево, R - сдвиг вправо. Выражение в круглых скобках означает одновременность выполнения сдвига. Например, (4L·1L) + (1L ) означает, что выполняется сдвиг на 5 разрядов влево, а затем еще на разряд.

Если нужен добавочный сдвиг ((1L ) или (1R)) , то его направление указывает младший разряд двоичного числа, записанного в первом столбце табл. 21.

1 - добавочный сдвиг вправо,
0 - добавочный сдвиг влево.

Исключение составляет сдвиг вправо на 15 разрядов.

Диаграмма алгоритма. Настройка подпрограмм 72EI, 72E3, 72E5, 73EI, 73E3, 73E5 на выполнение соответствующего сдвига осуществляется установкой косвенных функций "транзит" или "сдвиг влево". В подпрограммах 72EI, 72E5, 73EI, 73E5 используется режим перекоса.

Сдвиги на I6 и 8 разрядов производятся путем переадресации байтов общего регистра.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle B_2 \rangle + D_2$ | Счетчик сдвигов на I6<br>Рабочий регистр<br>Хранит величину сдвига |
| Т<br>У | | Рабочие регистры |
| Д | КО | Рабочий регистр (адресный) |
| Л | $R_I$ | Хранит $R_I$ |
| БС2 | 0 | IБС2 - указатель добавочного сдвига |
| БС3 | 0 | IБС3 - указатель команд SRA, SRDA, SLA, SLDA (арифметический сдвиг) |
| БС4 | 0 | IБС4 - указатель сдвигов на I5L , I4R, I5R, (блоки 7IДI, 70Д8). Используется также для организации циклов (см. блоки 7IA5, 72EI, 72E3, 73E5, 73EI, 73E3, 73E5) |
| БС5 | 0 | IБС5 - указатель команд SRDL ,SLDL , SRDA, SLDA (двойное слово) |
| БС6 | | Для команд, устанавливающих признак результата: |
| БС7 | | БС7 - указатель знака операнда (IБС7 - знак операнда +).<br>Для команд арифметического сдвига влево IБС6 указатель переполнения |
| НДД | 0 | IНДД - указатель команд SLL , SLA, SLDL , SLDA (сдвиг влево) |
| ППФ | | Используется при организации циклов наряду с БС4 |

## I7. СТАНДАРТНАЯ СИСТЕМА КОМАНД. ОПЕРАЦИИ СРАВНЕНИЯ И ПРОВЕРКИ

### I7.I. Микропрограмма CRRR

Команды

| Название | Мнем. | Формат | | | | Вход | Время |
|---|---|---|---|---|---|---|---|
| | | | байт I | Байт 2 | | | |
| СРАВНЕНИЕ | CR | RR | I9 | $R_I$ | $R_2$ | II2 | I2 |

Операнды. Два числа с фиксированной запятой длиной в слово, расположенные в общих регистрах с номерами $R_I$ и $R_2$.

Результат. Установка признака результата:

0 – числа равны;

1 – первое число меньше второго;

2 – первое число больше второго.

Алгоритм. Сравнение чисел осуществляется вычитанием. Второе число двоично вычитается из первого. Если разность равна нулю, то числа равны; если разность меньше нуля, то первое число меньше второго; если разность больше нуля, то первое число больше второго.

Диаграмма алгоритма. Вычитание операндов (блок 75С1) реализовано с помощью косвенной функции "–".

В результате вычитания чисел может произойти переполнение. Если переполнение возникло, то полученный знак разности отличается от истинного. Для получения истинного знака разности, который совпадает со знаком первого операнда, старший байт первого операнда складывается с нулем с помощью прямой функции "+". Тем самым гасится и переполнение. Затем признак результата формируется установкой КУ1.

Пример (для простоты взят один байт).

| | |
|---|---|
| 0100 0000 | Первое число |
| 1001 0000 | Второе число |
| | Вычитание: |
| 0111 0000 | двоичное дополнение второго числа |
| 0100 0000 | |
| +0111 0000 | Сложение |
| 1011 0000 | |
| 1011 0000 | Результат. Имеет место переполнение. Знак неверен |
| 0100 0000 | |
| +0000 0000 | Знак верен. Переполнение погашено |
| 0100 0000 | |
| 2 | Признак результата: первое число больше второго |

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| И | $R_2$0010 | Хранит текущий адрес второго числа |
| У | $R_1$0010 | Хранит текущий адрес первого числа |
| Д | Код операции | Хранит нечетный байт второго числа |
| Л | $R_1R_2$ | Хранит четный байт второго числа |
| БС2 | 0 | Индикатор конца операции: |
| | | 1 – конец операции |
| Н | $\langle R_2 0010 \rangle$ | Информационный регистр |
| З | $\langle R_2 0011 \rangle$ | Информационный регистр |

17.2. Микропрограмма CRX

Команды

| Название | Мнем. | Формат | | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | байт 3 | | байт 4 | | |
| СРАВНЕНИЕ | С | RX | 59 | $R_1$ | $X_2$ | $B_2$ | $D_2$ | 133 | 13 |

Операнды. Два числа с фиксированной запятой длиной в слово. Первое число расположено в общем регистре с номером $R_I$. Второе - в оперативной памяти по адресу $\langle X_2 \rangle + \langle B_2 \rangle + D_2$.

Результат. Установка признака результата:

0 - числа равны;

1 - первое число меньше второго;

2 - первое число больше второго.

Прерывания. Адресация. Защита по чтению. Спецификация: адрес второго числа не кратен 4 (реализовано в микропрограмме ВЫБОР).

Алгоритм. Сравнение чисел осуществляется вычитанием. Второе число двоично вычитается из первого. Если разность равна нулю, то числа равны; если разность меньше нуля, то первое число меньше второго; если разность больше нуля, то первое число больше второго.

Диаграмма алгоритма. Вычитание операндов (блок 76С1) реализовано с помощью косвенной функции "-". В результате вычитания чисел может произойти переполнение. Если переполнение возникло, то полученный знак разности отличается от истинного. Для получения истинного знака разности, который совпадает со знаком первого операнда, старший байт первого операнда складывается с нулем при помощи функции "+". Тем самым гасится и переполнение. Затем признак результата формируется установкой КУ1 (блок 76С3).

Пример (для простоты взят один байт).

| | |
|---|---|
| 0100 0000 | Первое число |
| 1001 0000 | Второе число |
| | Вычитание: |
| 0111 0000 | двоичное дополнение второго числа |
| 0100 0000 | |
| +0111 0000 | Сложение |
| 1011 0000 | |
| 1011 0000 | Результат. Имеет место переполнение. Знак неверен |
| 0100 0000 | |
| +0000 0000 | |
| 0100 0000 | Знак верен. Переполнение погашено |
| 2 | Признак результата: первое число больше второго |

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| И | $R_2$0010 | Хранит текущий адрес второго числа |
| У | $R_I$0010 | Хранит текущий адрес первого числа |
| Д | Код операции | Хранит нечетный байт первого числа |
| Л | $R_I X_2$ | Хранит четный байт первого числа |
| БС2 | 0 | Индикатор конца операции:<br>1 - конец операции |

17.3. Микропрограмма CHRX

Команды

| Название | Мнем. | Формат | | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | | байт 3 | байт 4 | | |
| СРАВНЕНИЕ ПОЛУСЛОВА | CH | RX | 49 | $R_I$ | $X_2$ | $B_2$ | $D_2$ | 132 | 11 |

Операнды. Первый операнд - число с фиксированной запятой длиной в слово, расположенное в общем регистре с номером $R_1$. Второй операнд - число с фиксированной запятой длиной в полуслово, расположенное в оперативной памяти по адресу $\langle X_2 \rangle + \langle B_2 \rangle + D_2$.

Результат. Установка признака результата:

0 - числа равны;

1 - первое число меньше второго;

2 - первое число больше второго.

Прерывания. Адресация. Защита по чтению. Спецификация: адрес полуслова не кратен 2.

Алгоритм. Сравнение чисел осуществляется вычитанием. Второе число двоично вычитается из первого. Если разность равна нулю, то числа равны; если разность меньше нуля, то первое число меньше второго; если разность больше нуля, то первое число больше второго.

Перед вычитанием длина второго числа увеличивается до получения полного слова, при этом старшим 16 разрядам полученного слова присваивается значение знакового разряда.

Диаграмма алгоритма. Вычитание числа (блок 77Д3) реализовано с помощью косвенной функции "-". В результате вычитания чисел может произойти переполнение. Если оно возникло, то полученный знак разности отличается от истинного. Для получения истинного знака разности, который совпадает со знаком первого операнда, полученный результат складывается с нулем с помощью прямой функции "+". Тем самым гасится и переполнение. Затем признак результата формируется установкой КУ1 (блок 77Д5).

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ | Хранит текущий адрес второго операнда |
| У | $R_1$0010 | Хранит текущий адрес первого операнда |
| Д | Код операции | Хранит нечетный байт второго операнда |
| Л | - | Хранит четный байт второго операнда |
| БС2 | 0 | Индикатор конца операции:<br>1 - конец операции |
| БС5 | 0 | Индикатор значения знакового разряда полуслова:<br>0 - разряд = 0<br>1 - разряд = 1 |

17.4. Микропрограмма CLRR

Команды

| Название | Мнем. | Формат | | | | Вход | Время |
|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | | | |
| СРАВНЕНИЕ КОДОВ | CLR | RR | 15 | $R_1$ | $R_2$ | 10A | 5+3B |

Операнды. Оба операнда - целые двоичные числа в прямом коде без знака длиной в слово, расположенные в общих регистрах с номерами $R_1$ и $R_2$.

Результат. Установка признака результата:

0 - операнды равны;

1 - первый операнд меньше;

2 - первый операнд больше.

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| БС2 | 0 | Индикатор конца операции:<br>0 – конец операции в случае совпадения всех байтов |

I7.5. Микропрограмма CLRX

Команды

| Название | Мнем. | Формат | | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | | байт 3 | байт 4 | | |
| СРАВНЕНИЕ КОДОВ | CL | RX | 55 | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I2B | 7+2B<br>7, если B = I |

Операнды. Оба операнда – целые двоичные числа в прямом коде без знака длиной в слово. Первый операнд расположен в общем регистре с номером $R_I$. Второй – в основной памяти по адресу $\langle X_2 \rangle + \langle B_2 \rangle + D_2$.

Результат. Установка признака результата:

0 – операнды равны;

I – первый операнд меньше;

2 – первый операнд больше.

Прерывания. Адресация. Защита по чтению. Спецификация: адрес $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ не кратен 4 (реализовано в микропрограмме ВЫБОР).

Алгоритм. Сравнение операндов осуществляется слева направо, начиная со старших байтов. Байт второго операнда двоично вычитается из байта первого операнда. Если результат больше нуля, то первый операнд больше второго; если результат меньше нуля, то первый операнд меньше второго. Если результат равен нулю, то происходит вычитание следующих байтов и т.д.

Примеры (для простоты взяты два байта).

```
I.  IIII IIIO 000I 00IO        Первый операнд
    IOII IIIO OIII OIII        Второй операнд

a)  IIII IIIO
   -IOII IIIO
    OIOO 0000                  Результат первого вычитания больше нуля, значит первый операнд больше второго

2.  OIOI OIII IIII IIIO        Первый операнд
    OIOI OIII IIII IIII        Второй операнд

a)  OIOI OIII
   -OIOI OIII
    0000 0000                  Результат первого вычитания равен нулю, необходимо сравнение следующих байтов

б)  IIII IIIO
   -IIII IIII
(пере-
нос) IIII IIII                 Результат меньше нуля, значит первый операнд меньше второго
```

Диаграмма алгоритма. Вычитание байтов операндов реализовано с помощью косвенной функции "_" в два этапа. На первом этапе (блоки 79С1, 79С3) операция выполняется над байтами операндов. Если ТРКФ ≠ 0, то для получения верного результата эта же функция выполняется дополнительно над нулевыми байтами (блок 79Д5).

Пример (для простоты взят один байт).

| | | | | |
|---|---|---|---|---|
| 0011 | 1111 | | | Первый операнд |
| 1100 | 1001 | | | Второй операнд |
| 0011 | 0111 | | | Первый этап вычитания: двоичное дополнение второго операнда (ТПКФ=0). |
| +0011 | 0111 | | | Сложение |
| 0111 | 0110 | | | Результат выполнения функции АЛУ над байтами операндов (ТРКФ=1). В действительности второй операнд больше первого |
| 1111 | 1111 | | | Второй этап вычитания: обратный код нуля (т.к. ТПКФ установится в единицу). |
| +0000 | 0000 | | | Сложение |
| 1111 | 1111 | | | Результат второго этапа |
| 1111 | 1111 | 0111 | 0110 | Результат вычитания |

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г Р И | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ | Хранит текущий адрес второго операнда |
| У | $R_I$0010 | Хранит текущий адрес первого операнда |
| Д | Код операции | Хранит четный байт второго операнда |
| Л | $R_I X_2$ | Хранит нечетный байт второго операнда |
| БС4 | 1 | Индикатор конца операции: 0 - конец операции в случае совпадения всех байтов |
| Н | $\langle\langle X_2 \rangle + \langle B_2 \rangle + D_2 \rangle$ | Информационный регистр |
| З | $\langle\langle X_2 \rangle + \langle B_2 \rangle + D_2 + 1 \rangle$ | Информационный регистр |

## 17.6. Микропрограмма CLSI

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | байт 3 | байт 4 | | |
| СРАВНЕНИЕ КОДОВ | CLI | SI | 95 | $I_2$ | $B_I$ | $D_I$ | I6B | 5 |

Операнды. Оба операнда - целые двоичные числа длиной в байт в прямом коде без знака. Первый операнд находится в основной памяти по адресу $\langle B_I \rangle + D_I$. Второй - непосредственно в команде, в поле $I_2$.

Результат. Установка признака результата:

0 - операнды равны;

I - первый операнд меньше;

2 - первый операнд больше.

Прерывания. Адресация. Защита по чтению.

Алгоритм. Байт второго операнда двоично вычитается из байта первого операнда. Если результат больше нуля, то первый операнд больше второго; если результат меньше нуля, то первый операнд меньше второго. Если результат равен нулю, то операнды равны.

Примеры (для простоты взят один байт).

| | | | |
|---|---|---|---|
| I. | IIII | IIIO | Первый операнд |
| | IOII | IIIO | Второй операнд |
| | IIII | IIIO | |
| | - IOII | IIIO | |
| | OIOO | OOOO | Результат вычитания больше нуля, значит первый операнд больше второго |
| 2. | OIOI | OIII | Первый операнд |
| | OIOI | OIII | Второй операнд |
| | OIOI | OIII | |
| | - OIOI | OIII | |
| | OOOO | OOOO | Результат равен нулю, значит операнды равны |

Диаграмма алгоритма. Вычитание байтов операндов реализовано с помощью косвенной функции "-" в два этапа. На первом этапе (блок 80CI) операция выполняется над байтами операндов. На втором этапе для получения верного результата эта же функция выполняется дополнительно над нулевыми байтами (блок 80C2).

Пример. (Для простоты взят один байт).

| | |
|---|---|
| OOII IIII | Первый операнд |
| IIOO IOOI | Второй операнд |
| | Первый этап вычитания: |
| OOII OIII | двоичное дополнение второго операнда |
| + OOII IIII | |
| OOII OIII | Сложение |
| OIII OIIO | Результат выполнения функции АЛУ над байтами операндов |
| IIII IIII | Второй этап вычитания: |
| | обратный код нуля (т.к. ТПКФ установится в единицу) |
| + 0000 0000 | |
| IIII IIII | Сложение |
| IIII IIII | Результат второго этапа |
| IIII IIII OIII OIII | Результат вычитания. Первый операнд меньше второго. |

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle B_I \rangle + {}^{D}I$ | Хранит адрес основной памяти |
| Л | $I_2$ | Хранит $I_2$ |

I7.7. Микропрограмма CLCSS

Команды

| Название | Мнем. | Формат | | | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | байт 3 | байт 4 | байт 5 | байт 6 | | |
| СРАВНЕНИЕ КОДОВ | CLC | SS | D5 | L | $B_1$ | $D_1$ | $B_2$ | $D_2$ | 18A | 44-5B |

Операнды. Два поля байтов одинаковой длины. Адрес первого поля – $D_1 + \langle B_1 \rangle$ . Адрес второго поля – $D_2 + \langle B_2 \rangle$ . Длина полей определяется полем L команды: минимальная длина полей – I байт (L=0); максимальная длина полей – 256 байтов (L=255).

Оба операнда рассматриваются как целые двоичные числа в прямом коде без знака.

Результат. Установка признака результата: 0 – операнды равны; I – первый операнд меньше; 2 – первый операнд больше.

Прерывания. Адресация. Защита по чтению.

Алгоритм. Сравнение операндов осуществляется слева направо, начиная со старших байтов. Байт второго операнда двоично вычитается из байта первого операнда. Если результат больше нуля, то первый операнд больше второго; если результат меньше нуля, то первый операнд меньше второго. Если результат равен нулю, то происходит вычитание следующих байтов и т.д.

Примеры (для простоты взяты два байта).

```
I.  IIII IIIO 000I 00IO        Первый операнд
    IOII IIIO OIII OIII        Второй операнд

a)  IIII IIIO
   -IOII IIIO
    OIOO 0000                  Результат первого вычитания больше нуля, значит первый
                               операнд больше
2.  OIOI OIII IIII IIIO        Первый операнд
    OIOI OIII IIII IIII        Второй операнд

a)  OIOI OIII
   -OIOI OIII
    0000 0000                  Результат первого вычитания – нуль, необходимо сравне-
                               ние следующих байтов

б)             IIII IIIO
              -IIII IIII
    (перенос)  IIII IIII       Результат меньше нуля, значит первый операнд меньше
```

Диаграмма алгоритма. Так как чтение операндов из памяти производится четно-нечетными парами байтов независимо от четности адреса, то это порождает четыре последовательности обработки операндов в зависимости от четности адресов (блоки 8IBI, 8ICI, 8IДI).

Вычитание байтов операндов, рассматриваемых как положительные двоичные числа, реализовано с помощью косвенной функции АЛУ (–) в два этапа.

На первом этапе (блоки 8IAI, 8IA7, 8IB2, 8IB4, 8IД2, 8IEI) операция АЛУ выполняется над байтами операндов.

Если ТРКФ $\neq$ 0, то для получения верного результата эта же функция АЛУ выполняется дополнительно над нулевыми байтами (блоки 8IA3, 8IC3, 8IC9).

Пример (для простоты взят один байт).

```
OOII IIII        Первый операнд
IIOO IOOI        Второй операнд
```

| | | | | |
|---|---|---|---|---|
| | | | | Первый этап вычитания: |
| 0011 | 0111 | | | дополнительный код (ТПКФ=0). |
| 0011 | 1111 | | | |
| +0011 | 0111 | | | Сложение |
| 0111 | 0110 | | | Результат выполнения функции АЛУ над байтами операндов ТРКФ=1 |
| | | | | В действительности второй операнд больше первого |
| 1111 | 1111 | | | Второй этап вычитания: обратный код нуля (т.к. ТПКФ установится в единицу). |
| 0000 | 0000 | | | |
| +1111 | 1111 | | | Сложение |
| 1111 | 1111 | | | Результат второго этапа |
| 1111 | 1111 | 0111 | 0110 | Результат вычитания |

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г Р И | $\langle B_1 \rangle + D_1$ | Хранит текущий адрес байтов первого операнда |
| П Т У | $\langle B_2 \rangle + D_2$ | Хранит текущий адрес байтов второго операнда |
| Л | L | Индикатор конца операции |
| Д | - | Хранит нечетный байт второго операнда |

17.8. Микропрограмма TMSI

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | байт 3 | байт 4 | | |
| ПРОВЕРИТЬ ПО МАСКЕ | TM | SI | 9I | $I_2$ | $B_1$ | $D_1$ | I63 | 5 |

Операнды. Первый операнд - логическая информация длиной в байт, расположенная в основной памяти по адресу $\langle B_1 \rangle + D_1$. Второй операнд - логическая информация длиной в байт, расположенная непосредственно в команде в поле $I_2$, используется как восьмиразрядная маска, т.е. разряд этого байта, равный единице, указывает на то, что соответствующий разряд байта в памяти выбирается для анализа. Если разряд маски равен нулю, то соответствующий разряд байтов памяти игнорируется.

Результат. Устанавливается признак результата:

0 - выбранные разряды или маска полностью равны нулю;

1 - выбранные разряды равны и нулям и единицам;

3 - все выбранные разряды равны единице.

Прерывания. Адресация. Защита по чтению.

Алгоритм. Из памяти читается байт. При помощи операции логического умножения выделяются заданные маской разряды. Если результат операции - нуль (нулевая маска или все выбранные разряды равны нулю), операция на этом оканчивается; устанавливается признак результата 0. Если результат операции - не нуль, то сложением этого результата по модулю два с маской проверяется не равны ли все выбранные разряды единице.

Диаграмма алгоритма. Признак результата устанавливается с помощью КУ2 (блок 82В5), если все выделенные разряды равны нулю или выделенные разряды равны единицам и нулям.

Когда все выделенные разряды равны единицам, признак результата устанавливается отдельной микрокомандой (блок 82С5).

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle B_I \rangle + {}^{D}I$ | Хранит адрес основной памяти |
| У | - | Хранит байт результата операции логического умножения |
| Л | $I_2$ | Используется как маска |

I7.9. Микропрограмма ПРИУС

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | байт 3 | байт 4 | | |
| ПРОВЕРИТЬ И УСТАНОВИТЬ | TS | SI | 93 | | $B_I$ | ${}^{D}I$ | I67 | 7 |

Операнды. Логическая информация длиной в один байт, расположенная в основной памяти по адресу $\langle B_I \rangle + {}^{D}I$.

Результат. В зависимости от крайнего левого разряда (разряда 0) адресованного байта устанавливается признак результата:

0 - крайний левый разряд указанного байта равен 0;

I - крайний левый разряд указанного байта равен I.

Во все разряды байта заносятся единицы.

Прерывания. Адресация. Защита памяти по записи и чтению.

Диаграмма алгоритма. См. диаграмму.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle B_I \rangle + {}^{D}I$ | Хранит адрес операнда |
| БС6<br>БС7 | Признак результата | Индикатор результата |

# I8. ДЕСЯТИЧНАЯ АРИФМЕТИКА. АРИФМЕТИЧЕСКИЕ ОПЕРАЦИИ И ОПЕРАЦИИ СРАВНЕНИЯ

## I8.I. Микропрограмма СЛОЖД

Команды

| Название | Мнем. | Формат | | | | | | | | Вход |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | байт 3 | байт 4 | байт 5 | байт 6 | |
| СЛОЖЕНИЕ ДЕСЯТИЧНОЕ | AP | AP | FA | $L_I$ | $L_2$ | $B_I$ | $D_I$ | $B_2$ | $D_2$ | I95 |
| ВЫЧИТАНИЕ ДЕСЯТИЧНОЕ | SP | SS | FB | $L_I$ | $L_2$ | $B_I$ | $D_I$ | $B_2$ | $D_2$ | I97 |

Время: $47+3{,}2\ N_{min}+ 2{,}2\ N_{abs} + 0{,}2\ N_1$ .

Операнды. Первый операнд – десятичное число в упакованном формате длиной $L_I$ + I байтов, расположенное в основной памяти по адресу $\langle B_I \rangle + D_I$.

Второй операнд – десятичное число в упакованном формате длиной $L_2$ + I байтов, расположенное по адресу $\langle B_2 \rangle + D_2$.

Результат. Алгебраическая сумма (разность) первого и второго числа в упакованном формате длиной $L_I$+ I байтов по адресу $\langle B_I \rangle + D_I$. Код знака результата определяется I2-м разрядом ССП. При переполнении теряются старшие значащие цифры результата. Нулевая сумма всегда положительна. Когда старшие цифры теряются из-за переполнения, нулевой результат имеет знак точной суммы.

Устанавливается признак результата:

0 – сумма (разность) равна нулю;

I – сумма (разность) меньше нуля;

2 – сумма (разность) больше нуля;

3 – переполнение.

Прерывания. Данные: неверен код цифры или знака в одном из операндов; результат непредсказуем.

Десятичное переполнение: в поле первого операнда не помещаются все значащие цифры суммы. Переполнение может порождаться одной из двух причин: во-первых, потерей переноса из старшего десятичного разряда поля результата, во-вторых, получением результата, не укладывающегося в отведенное ему поле.

Защита памяти по записи и чтению. Адресация.

Алгоритм. Второе число складывается с первым (при ВЫЧИТАНИИ ДЕСЯТИЧНОМ вычитается из первого), а сумма (разность) помещается на место первого числа.

Сложение (вычитание) алгебраическое с учетом всех цифр каждого числа. Все знаки и цифры проверяются на осмысленность.

ВЫЧИТАНИЕ ДЕСЯТИЧНОЕ сводится к СЛОЖЕНИЮ ДЕСЯТИЧНОМУ путем изменения знака вычитаемого на противоположный (в основной памяти знак второго операнда остается без изменения).

Оба числа участвуют в вычислениях в прямом коде. Если знаки чисел одинаковые, то числа складываются и их сумме присваивается их общий знак. Если знаки чисел различные, то производится вычитание второго числа из первого, когда $L_I \geqslant L_2$ и первого из второго, когда $L_2 > L_I$. Результату предварительно присваивается знак уменьшаемого. Если оказалось, что уменьшаемое по абсолютной величине меньше вычитаемого, то результат получается в дополнительном коде, а затем переводится в прямой, и предварительно присвоенный знак результата меняется на противоположный.

Если длины операндов различны, то более короткое число дополняется слева нулями, и сложение (вычитание) доводится до конца. В случае, если второе число длиннее первого, байты, полученные при сложении (вычитании) избыточных цифр второго числа с нулями, дополняющими первое число, в основную память не записываются, но анализируются для выяснения наличия переполнения.

Если $L_1 \geqslant L_2$, то о переполнении свидетельствует наличие переноса из старшей десятичной цифры результата.

После получения окончательной суммы (разности) формируется признак результата.

Диаграмма алгоритма. 85С1, 85В1. Знак второго числа определяется путем сравнения его кода последовательно с кодами 1011 и 1101. При совпадении устанавливается признак 1ВС3 (минус), при несовпадении проверяется, будет ли код знака больше 9, и устанавливается признак 0ВС3 (плюс).

85В2, 85С2. Устанавливается признак 1ВС5, если адрес конца первого числа нечетный и 0ВС5 - если он четный; определяется знак первого числа, проверяется правильность кода знака (тем же способом, что и второго числа) и устанавливается признак 1ВС2, если знак минус, 0ВС2 - если знак плюс.

85В4. Если знаки одинаковые, то заносится косвенная функция "+", устанавливается признак знака результата, совпадающий с признаком знака первого числа: 0ВС2 - плюс, 1ВС2 - минус.

85Е3. Заносится косвенная функция "-:" и устанавливается признак знака результата, совпадающий с признаком знака первого числа.

85Д4. При $L_1 < L_2$ заносится косвенная функция ":-" и устанавливается признак знака результата, противоположный признаку знака первого числа (т.е. знак совпадает со знаком более длинного операнда).

85С5. Вычисляется младший байт результата; формируется по признаку ВС2 и по 12-му разряду ССП знак результата. Младшая цифра результата со знаком записывается в основную память на свое место. Запоминается в локальной памяти адрес конца первого операнда.

В связи с тем, что из основной памяти всегда считывается четно-нечетная пара байтов, а операнды переменной длины расположены в памяти произвольно, сложение выполняется по двум различным циклам, в зависимости от четности адресов концов операндов: один цикл для случая, если эти адреса одинаковой четности, и второй - если различной четности.

85С7. Анализируется триггер НДД.

85С6, 85С8. Если $L_1 = L_2$, то вычисление результата заканчивается непосредственно в одном из циклов в блоке 85С5. Если $L_1 < L_2$, то после получения $L_1 + 1$ младших байтов результата устанавливается признак 1ВС3, если все полученные байты результата нулевые.

86С1. Появление 1ПКФ после вычисления последнего байта результата говорит о переполнении, если выполнялось сложение, и о том, что результат получился в дополнительном коде, если выполнялось вычитание.

86Д1. Проверяется, какая косвенная функция выполнялась. Проверка осуществляется при помощи выполнения косвенной функции над константами 80 и 80. При появлении 1ПКФ фиксируется, что выполнялось сложение.

86Д2. Если выполнялось вычитание, то результат получился в дополнительном коде. Для перевода его в прямой код необходимо восстановить адрес конца первого операнда в регистре ГРИ. Этот адрес был записан предварительно в локальной памяти. После этого знак результата и признак знака результата меняется на противоположный.

86Д4. Осуществляется вычисление дополнения результата путем вычитания его из нуля и запись его в основную память.

86В1. Если после вычисления последнего, старшего, байта результата переноса нет (0ПКФ), то проверяется, не нулевой ли результат получился путем анализа триггера РКФ.

86Д2, 86Д3, 86В3. Если результат равен нулю, а знак его минус, то необходимо заменить знак на противоположный, для чего восстанавливается в регистре ГРИ адрес конца результата. Знак меняется на противоположный, затем устанавливается признак результата.

86А8. Осуществляется переход в зависимости от состояния триггеров РКФ и ПКФ на различные случаи выработки признака результата. Здесь триггер РКФ указывает на равенство нулю старших байтов результата, которые не перемещаются в поле первого операнда. При 0РКФ и 0ПКФ переполнения нет и результат получится в прямом коде. При 1РКФ и 1ПКФ результат получится в дополнительном коде. В случаях 1РКФ, 0ПКФ и 0РКФ, 1ПКФ фиксируется переполнение.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| М | Адрес следующей команды | Не используется |
| Ф | | Счетчик длины второго операнда |
| Е | | Счетчик длины первого операнда |
| Г Р И | $\langle B_1 \rangle + D_1 + L_1$ | Для хранения адреса обрабатываемого байта первого операнда |
| П Т У | $\langle B_2 \rangle + D_2 + L_2$ | Для хранения адреса обрабатываемого байта второго операнда |
| Д | Код операции | Для запоминания байта операнда в цикле сложения (вычитания) |
| Л | $L_1 L_2$ | При вычислении дополнения результата - счетчик длины (в старших четырех разрядах) |
| БС2 | 0 | Индикатор команды: 1БС2 - вычитание, 0БС2 - сложение; индикатор знака первого операнда, знака результата: 0БС2 - плюс, 1БС2 - минус |
| БС3 | 0 | Индикатор знака второго операнда: 0БС3 - плюс, 1БС3 - минус, при $L_1 < L_2$ используется для запоминания равенства нулю младших $L_1 + 1$ байтов результата: 0БС3 - результат нулевой |
| БС4 | 1 | Индикатор четности адреса конца второго операнда: 1БС4 - нечетный, 0БС4 - четный |
| БС5 | 1 | Индикатор четности адреса конца первого операнда: 1БС5 - нечетный, 0БС5 - четный |
| ЛП E6,F6,F7 | - | Запоминается адрес конца первого операнда |

I8.2. Микропрограмма F8SS

Команды

| Название | Мнем. | Формат | | | | | | | | Вход |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | | байт 3 | байт 4 | байт 5 | байт 6 | |
| СЛОЖЕНИЕ С ОЧИСТКОЙ | ZAP | SS | F8 | $L_1$ | $L_2$ | $B_1$ | $D_1$ | $B_2$ | $D_2$ | I9I |

Время: $40 + 4N_{min} + 2{,}5N_{abs}$.

Операнды. Упакованное десятичное число, расположенное в поле основной памяти по адресу $\langle B_2 \rangle + D_2$. Длина поля - $(L_2 + 1)$ байтов.

Результат. Если коды знака и цифр в исходном числе правильные, причем поле операнда и поле результата не перекрываются или перекрываются так, что самый правый байт поля результата совпадает с самым правым байтом поля операнда или находится правее него, то исходное число будет переписано в поле основной памяти длиной ($L_I$ + I) байтов по адресу $\langle B_I \rangle$ + $D_I$. Если необходимо, исходное число дополняется слева нулями. Значащие цифры операнда, не помещающиеся в поле результата, теряются. Нулевой результат всегда положителен. Однако, если нулевой результат получился вследствие потери значащих цифр операнда, то нулевой результат сохраняет знак исходного числа. Код знака результата устанавливается в соответствии с состоянием I2-го разряда ССП.

Устанавливается признак результата:

0 - результат нуль,
I - результат меньше нуля,
2 - результат больше нуля,
3 - переполнение.

Прерывания. Данные: код знака или цифры исходного числа является неправильным или поле результата и поле операнда перекрываются так, что самый правый байт поля результата находится левее самого правого байта поля операнда.

Десятичное переполнение: при помещении операнда в поле результата произошла потеря значащих цифр операнда.

Адресация.

Защита памяти по записи или чтению.

Алгоритм. В микропрограмме имеются два основных цикла, которые реализуют случаи: адрес младшего байта поля результата нечетный, адрес младшего байта поля операнда нечетный ("нечет-нечет") и адрес младшего байта поля результата четный, адрес младшего байта поля операнда нечетный ("чет-нечет").

Случаи "чет-чет" и "нечет-чет" используют соответственно первый и второй основные циклы после пересылки младшего байта и модификации адресов на -I.

В цикле "нечет-нечет" формируются и записываются сразу два байта результата; в цикле "чет-нечет" идет побайтное формирование результата.

Выявление неправильного перекрытия полей обеспечивается:

в случае "чет-нечет", если адрес младшего байта операнда на единицу больше адреса младшего байта результата, засылкой на место знака результата знаковой комбинации IIII до выборки операнда; в остальных случаях засылкой в поле результата знака операнда вместе с младшей цифрой на этапе обработки самого младшего байта операнда при первом выполнении цикла или во входах в циклы (случаи "чет-чет" и "нечет-чет").

Диаграмма алгоритма. Проверка корректности цифр операнда происходит путем десятичного сложения их с нулем по косвенной функции. Наличие хотя бы одной нецифровой комбинации (исключая знак) устанавливает триггер НДД в единицу.

Значения L I ( L 2 ), равные -I и -2, обнаруживаются по состоянию триггеров ЧЕТ и ПШФ:

- 2 (ОЧЕТ, IПШФ),
- I (IЧЕТ, IПШФ),

после операции вычитания из L I ( L 2 ) цифры 2.

Значения L I ( L 2 ), равные -I и больше -I, обнаруживаются по состоянию триггера ПШФ:

- I (IПШФ),

после вычитания из L I ( L 2 ) цифры I.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| М | Адрес следующей команды | Не используется |
| Ф | | Хранит $L_2$ |
| Е | | Здесь запоминается байт операнда с четным адресом (цикл "чет-нечет") |
| Г Р И | $\langle B_I \rangle + D_I + L_I$ | Хранит текущий адрес байтов поля результата |
| П Т У | $\langle B_2 \rangle + D_2 + L_2$ | Хранит текущий адрес байтов операнда |
| Д | - | Хранит $L_I$ |
| Л | $L_I$ $L_2$ | Используется при проверке корректности знака операнда и при формировании знака результата |
| БС2 | 0 | Индикатор четности адреса младшего байта поля результата |
| БС3 | 0 | Индикатор выхода из цикла (ОБС3) "чет-нечет" |
| БС4 | I | Индикатор первого выполнения циклов |
| Вход А | - | Здесь запоминается байт операнда с нечетным адресом (цикл "чет-нечет") |
| ПКФ | - | Индикатор переполнения (ПКФ - переполнение) |
| F5 - F7 | - | Хранит $\langle B_I \rangle + D_I + L_I$ |

I8.3. Микропрограмма УМДЕД

Команды

| Название | Мнем. | Формат | | | | | | | Вход |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | байт 3 | байт 4 | байт 5 | байт 6 | |
| УМНОЖЕНИЕ ДЕСЯТИЧНОЕ | MP | SS | FS | $L_I$ $L_2$ | $B_I$ | $D_I$ | $B_2$ | $D_2$ | I99 |
| ДЕЛЕНИЕ ДЕСЯТИЧНОЕ | DP | SS | FD | $L_I$ $L_2$ | $B_I$ | $D_I$ | $B_2$ | $D_2$ | I9B |

Время. Команда MP: $82+9N_I + 9N_2 + (2N_2 - I) \cdot (27+6N_I - 3N_2)$, команда DP: $2(N_I - N_2) \cdot (100 + 19N_2)$.

Операнды. Множимое (команда MP) или делимое (команда DP) - упакованное десятичное число, расположенное в поле основной памяти по адресу $\langle B_I \rangle + D_I$; длина поля $L_I + I$ байтов.

Множитель (команда MP) или делитель (команда DP) - упакованное десятичное число, расположенное в поле основной памяти по адресу $\langle B_2 \rangle + D_2$; длина поля $L_2 + I$ байтов.

Результат. Команда MP. Произведение - упакованное десятичное число, расположенное в поле множимого.

Команда DP. Целая часть частного и остаток - упакованные десятичные числа, расположенные в поле делимого. Частное расположено в левой части поля. Остаток расположен в правой части поля и имеет длину, равную длине поля делителя. Частное и остаток занимают все поле делимого.

Знаки произведения и частного определяются по алгебраическим правилам. Знак остатка тот же, что и знак делимого. Эти правила имеют место даже для нулевых результатов. Код знака соответствует I2-му разряду ССП.

Прерывания. Спецификация: длина множителя или делителя больше чем I5 цифр плюс знак или больше, чем длина множимого или делимого. Команда не выполняется, поэтому данные в памяти остаются без изменения.

Данные: знак или цифра операнда имеют неправильную кодировку; в старших разрядах множимого недостаточное количество нулей (меньше размера поля множителя). Операция прекращается, результат непредсказуем.

Десятичное деление: частное не помещается в отведенное ему поле; к этому случаю относится деление на нуль. Деление не выполняется, поэтому операнды в памяти не изменяются.

Адресация. Защита памяти по записи и чтению.

Алгоритм. Умножение десятичное производится путем последовательных сложений (вычитаний) согласно табл. 22.

Таблица 22

| Цифра множителя | Прибавление в сумматор |
|---|---|
| 0 | - |
| I | +IM |
| 2 | +2M |
| 3 | +2M+IM |
| 4 | +2M+2M |
| 5 | +2M+2M+IM |
| 6 | -2M-2M |
| 7 | -2M-IM |
| 8 | -2M |
| 9 | -IM |

В приведенной таблице М - множимое, 2М - удвоенное множимое. Если предыдущая (справа) цифра множителя была больше 5 (выполнялось вычитание), то следующая цифра множителя увеличивается на единицу. После умножения на очередную цифру множителя кратные множимого, прибавляемые в сумматор, сдвигаются на одну десятичную цифру влево.

Пример умножения десятичного.

```
   0000I2IA          Множимое
      9I7A           Множитель
IM 0000I2I
2M 0000242           Кратные множимого

  -0000000
  -0000242           Вычитание 2М и IM (умножение на 7, см. табл. 22)
   9999758
  -0000I2I
   9999637           Добавление 2М вместо IM, т.к. предыдущая цифра множителя (7)
  +000242            больше пяти (умножение на 2)
  -0002057           Вычитание IM (умножение на 9)
   00I2I
   9989957           Добавление IM, т.к. старшая цифра множителя была больше 5 (умно-
  +0I2I               жение на I)
   0II0957
   0II0957A          Произведение
```

Алгоритм деления десятичного. Перед выполнением деления производится пробное вычитание делителя из делимого. При этом старшая цифра делителя подписывается под второй слева цифрой делимого. Если результат пробного вычитания положителен или равен нулю, то осуществляется выход на прерывание, операнды в памяти остаются без изменения.

Деление производится по алгоритму без восстановления остатка. Старшая цифра частного определяется путем вычитания делителя из делимого, причем старшая цифра делителя подписывается под третьей слева цифрой делимого. Вычитание повторяется до тех пор, пока результат не станет отрицательным. Число вычитаний (не считая последнего) дает старшую цифру частного.

Следующая цифра частного вычисляется путем прибавления делителя, сдвинутого на один десятичный разряд вправо по сравнению с предыдущим положением делителя, к числу в поле делимого. Прибавление производится до тех пор, пока результат не станет положительным (т.е. не появится единица переноса). Количество сложений (без последнего) является дополнением соответствующей цифры частного до 9.

Следующая цифра частного вычисляется вычитанием еще на один разряд сдвинутого делителя, а последующая - снова сложением и т.д. до тех пор, пока первые края делимого и делителя не выравняются. Последний остаток оказывается отрицательным и его необходимо восстановить, т.е. еще раз прибавить делитель к числу в поле делимого.

Пример деления десятичного.

| | |
|---|---|
| 0I54675A | Делимое |
| 550A | Делитель |
| −0I54675 / 550 | Пробное вычитание |
| 9604675 | Результат отрицательный |
| −0I54675 / 550 | Вычисление старшей цифры частного |
| −0099675 / 550 | |
| −0044675 / 550 | |
| 9989675 | Старшая цифра частного равна 2. |
| + 550 | Вычисление следующей цифры. |
| +9995I75 / 550 | |
| −0000675 / 550 | |
| − 00I25 / 550 | |
| | Младшая цифра частного равна I. |
| + 99575 / 550 | Остаток отрицательный, его нужно восстановить |
| 00I25 | Остаток |
| 28IAI25A | Результат |

После получения очередных двух цифр частного они записываются на свое место в поле делимого.

Диаграмма алгоритма. 92C4, 92B4. Для определения знака множителя (делителя) код знака последовательно сравнивается с кодами IOII и IIOI. При совпадении устанавливается признак минуса - IBC4, при несовпадении код знака проверяется на корректность и устанавливается признак плюса - OBC4.

92B5. Множитель (делитель) в локальную память переписывается всегда так, чтобы адрес конца его был нечетным, независимо от положения его в основной памяти. Поэтому в зависимости от четности адреса конца множителя имеются два отдельных цикла для переписи его в ЛП.

Множитель (делитель) записывается в следующую область локальной памяти (в ячейке ЛП с адресом 97 расположен младший байт, 0,1,2,3,4,5,6,7 – байты множителя или делителя):

| | | | |
|---|---|---|---|
| 96 | I | 0 | 97 |
| A6 | 3 | 2 | A7 |
| B6 | 5 | 4 | B7 |
| C6 | 7 | 6 | C7 |

92Д7. Формируется адрес начала множимого и пропускаются $L_2$ + I начальных байта его через БА.

92С7. Предварительно читается из локальной памяти и восстанавливается в регистре ГРИ адрес конца множимого. Множимое и удвоенное множимое (без знаков) заносятся в ЛП столбиком сверху вниз, начиная с младших байтов.

На рис. I2 показано размещение множимого и удвоенного множимого в ЛП, если адрес конца множимого в основной памяти нечетный. Если же он четный, то в этом случае множимое (М) и удвоенное множимое (2М) размещаются как показано на рис. I3. Таким образом, четность адресов в ЛП сохраняется. Если длина множимого меньше I6 байтов, то часть этого поля, предназначенная для старших байтов, остается незанятой.

| | | | | |
|---|---|---|---|---|
| 04 | I | 0 | 05 | |
| I4 | 3 | 2 | I5 | |
| 24 | 5 | 4 | 25 | |
| 34 | 7 | 6 | 35 | 2М |
| 44 | 9 | 8 | 45 | |
| 54 | II | I0 | 55 | |
| 64 | I3 | I2 | 65 | |
| 74 | I5 | I4 | 75 | |
| 84 | I | 0 | 85 | |
| 94 | 3 | 2 | 95 | |
| A4 | 5 | 4 | A5 | |
| B4 | 7 | 6 | B5 | IM |
| C4 | 9 | 8 | C5 | |
| D4 | II | I0 | D5 | |
| E4 | I3 | I2 | E5 | |
| F4 | I5 | I4 | F5 | |

Рис. I2. Адрес нечетный

| | | | | |
|---|---|---|---|---|
| 04 | 0 | I5 | 05 | |
| | 2 | I | | |
| | 4 | 3 | | |
| | 6 | 5 | | 2М |
| | 8 | 7 | | |
| | I0 | 9 | | |
| | I2 | II | | |
| 74 | I4 | I3 | 75 | |
| 84 | 0 | I5 | 85 | |
| | 2 | I | | |
| | 4 | 3 | | |
| | 6 | 5 | | IM |
| | 8 | 7 | | |
| | I0 | 9 | | |
| | I2 | II | | |
| F4 | I4 | I3 | F5 | |

Рис. I3. Адрес четный

Во время переписывания множимого и удвоенного множимого в ЛП в регистре Ф – счетчик длины; в регистре Д – адрес удвоенного множимого в ЛП; в регистре Е – адрес множимого в ЛП.

92С8. В поле ЛП, показанное на рис. I4, записываются нули.

| | | | |
|---|---|---|---|
| 06 | I | 0 | 07 |
| I6 | 3 | 2 | I7 |
| 26 | 5 | 4 | 27 |
| 36 | 7 | 6 | 37 |
| 46 | 9 | 8 | 47 |
| 56 | II | I0 | 57 |
| 66 | I3 | I2 | 67 |
| 76 | I5 | I4 | 77 |
| 86 | | | 87 |

а)

| | | | |
|---|---|---|---|
| 06 | 0 | | 07 |
| I6 | 2 | I | I7 |
| 26 | 4 | 3 | 27 |
| 36 | 6 | 5 | 37 |
| 46 | 8 | 7 | 47 |
| 56 | I0 | 9 | 57 |
| 66 | I2 | II | 67 |
| 76 | I4 | I3 | 77 |
| 86 | | I5 | 87 |

б)

Рис. I4. Сумматор

а – адрес множимого нечетный,

б – адрес множимого четный.

Это сумматор, здесь будет накапливаться произведение (0-15-й байты). Положение разрядов произведения показано на рис. 14. Оно зависит от четности адреса конца множимого.

В дальнейшем, непосредственно при умножении, регистры процессора используются для следующих целей:

Ф - для хранения очередной цифры множителя;

Е - для хранения адреса множимого или удвоенного множимого, которое прибавляется в сумматор;

Р - для хранения адреса поля, в котором накапливается произведение;

И - в старших 4 разрядах - счетчик пар цифр множителя, на которые уже произведено умножение (необходим для определения адреса поля сумматора, начиная с которого выполняется прибавление множимого или удвоенного множимого); в младших 4 разрядах располагается знак произведения;

Т - для хранения адреса байтов множителя, содержащих цифру, на которую производится умножение;

У - для промежуточных вычислений;

Л - для хранения $2L_2\ L_1$ и для организации счетчика цифр множителя в старших 4 разрядах;

Д - счетчик количества складываемых байтов.

93В1. Управление умножением ведется с помощью индикаторов: БС5 и БС2 (вначале устанавливаем ОБС5 и ОБС2).

Читаются два байта множителя по адресу, помещенному в регистр Т, и из четырех цифр выбирается одна по индикаторам БС5 и БС2 и помещается в регистр Ф.

93А0. Проверяется правильность кода этой десятичной цифры и всех предыдущих цифр множимого по состоянию триггера НДД.

93В2. По индикатору БС4 определяется, была ли предыдущая цифра множителя больше 5 (1БС4) или меньше, или равна 5 (ОБС4).

93В3. Определяется, больше ли 5 цифра, записанная в регистр Ф.

93А3. В регистре Ф получается дополнение этой цифры до десяти, заносится косвенная функция "-:" и устанавливается признак 1БС4.

93С3. Устанавливается признак ОБС4 и заносится косвенная функция "+:".

93В4. Определяется и заносится в регистр Д длина поля, прибавляемого в сумматор, а в регистр Р - адрес поля сумматора, с которого начинается сложение. Для формирования их используется содержимое регистра И - счетчик пар цифр множителя.

93В5. Определяется, больше цифра в регистре Ф двух или нет.

93А5. Если больше, то в регистр Е заносим константу 04 - адрес удвоенного множимого.

93С5, 93С6. Проверяется больше ли цифра в регистре Ф единицы. В регистр Е заносится константа 84 (адрес множимого).

93В6. Сдвиг кратных множимого осуществляется с помощью использования четырех типов сложения и изменения адреса поля сумматора, начиная с которого выполняется сложение (вычитание). Выбор одного из четырех типов сложения: без сдвига, со сдвигом на одну десятичную цифру влево, со сдвигом на две десятичные цифры влево, со сдвигом на три десятичные цифры влево осуществляется с помощью тех же индикаторов БС5 и БС2, которые управляют умножением.

93С9. Типы сложения без сдвига и со сдвигом на одну десятичную цифру выполняются по одному и тому же циклу, только во втором случае предварительно в поле УСТАНОВ микрокоманды устанавливается перекос косвенной функции (ПК). Точно так же со сдвигом на две и на три цифры.

93Д0. После окончания умножения на данную цифру множителя осуществляется подготовка к умножению на следующую цифру множителя: установка соответствующим образом индикаторов БС5 и БС2, прибавление 1 в счетчик И, модификация, если нужно, адреса множителя в регистре Т.

Управление умножением с помощью указателей БС5 и БС2 иллюстрируется табл. 23.

Таблица 23

| Начальное состояние | | Положение цифры множителя в регистре Н или З | Сдвиги | Действия, выполняемые после умножения на цифру множителя |
|---|---|---|---|---|
| БС5 | БС2 | | | |
| 0 | 0 | З старшие | Без сдвига | Установка 1БС2 |
| 0 | 1 | Н младшие | Со сдвигом на одну цифру | Установка 1БС5, увеличение содержимого счетчика в регистре И на единицу |
| 1 | 1 | Н старшие | Со сдвигом на две цифры | Установка ОБС2, модификация адреса множителя |
| 1 | 0 | З младшие | Со сдвигом на три цифры | Установка ОБС5. Увеличение счетчика в регистре И на единицу |

93Д1. После умножения на очередную цифру множителя из счетчика количества цифр множителя, организованного в старших четырех разрядах регистра Л, вычитается единица. Если в результате вычитания содержимое счетчика не станет отрицательным, то осуществляется переход к выбору следующей цифры множителя.

93Е1. Анализируется указатель БС4.

93Д2. Для этого в счетчик Л, в старшие четыре разряда, заносится 0000, устанавливается ОБС4, в регистр Ф заносится 1.

93Е2. Умножение закончено. Адрес конца множимого читается из локальной памяти и заносится в регистр ПТУ. В регистр Е заносится константа 06 и там организуется текущий адрес сумматора. Затем произведение из сумматора переписывается на место множимого в основную память. Ему присваивается знак, который хранится в младших четырех разрядах регистра И.

94В1. Вычисляется количество цифр частного 2 $(L_1 - L_2)$-1 и помещается в регистр Е. Адрес начала делимого помещается в локальную память по адресу Е7, F6, F7.

94В2. Формируется адрес, с которого необходимо начинать пробное вычитание. Для этого из адреса конца делимого вычитается разность $L_1 - L_2$, которая подготавливается в регистре Д. Пробное вычитание выполняется по одной из двух ветвей микропрограммы в зависимости от четности полученного адреса, с которого начинается пробное вычитание. Результат пробного вычитания никуда не заносится, после окончания пробного вычитания анализируется триггер ПКФ: если ОПКФ, то осуществляется переход к программе прерываний (ПДДН), иначе переходим непосредственно к делению. Здесь же проверяется и деление на 0: если делитель равен нулю, то после пробного вычитания будет выход на прерывание (ПДДН).

94С2, 94Д2. Деление начинается с определения не младшая ли цифра частного будет вычисляться. Для этого из счетчика цифр частного в регистре Е вычитается 1. По ОРПФ определяется, что цифра последняя (младшая), и устанавливается признак 1БС5. Если это еще не последняя цифра частного, то устанавливается ОБС5.

94С3. В регистре ГРИ подготавливается начальный адрес поля делимого, с которого необходимо начать вычитание. В старшие четыре разряда регистра Т записывается $L_2$ и там организуется счетчик вычитаемых байтов, по которому организуется выход из цикла.

В младших четырех разрядах регистра Т хранится число (изменяющееся в ходе деления), с помощью которого корректируется начальный адрес в поле делимого. При вычислении последней цифры (признак 1БС5) знак делимого заменяется нулем.

94С4. Деление выполняется по алгоритму без восстановления остатка. Если старшую цифру частного считать первой, то нечетные цифры вычисляются путем вычитания делителя из поля делимого до тех пор, пока остаток не будет отрицательным. Цифра частного в этом случае непосредственно накапливается в регистре Ф. Четные цифры частного определяются с помощью прибавления делителя к полю делимого столько раз, пока не появится 1ПКФ. В этом случае в регистре Ф накапливается не сама цифра частного, а ее дополнение до девяти.

94Д4, 94Е4. После вычисления очередной цифры частного необходимо произвести сдвиги делителя вправо.

В микропрограмме сам делитель не сдвигается, а сдвиг достигается путем выбора типа вычитания: без сдвига, со сдвигом на одну цифру, со сдвигом на две цифры, со сдвигом на три цифры и изменением адреса поля делимого, с которого начинается вычитание.

Управление этим процессом ведется с помощью указателей БС3 и БС2 согласно табл. 24.

Таблица 24

| Начальное состояние | | Тип вычитания, выполняемый для определения очередной цифры | Действия, выполняемые после вычисления цифры частного |
|---|---|---|---|
| БС3 | БС2 | | |
| 0 | 0 | Со сдвигом на три цифры | Установка 1БС2. Подготовка адреса для записи двух цифр частного в память, занесение косвенной функции "-:", занесение нуля в регистр Ф. Увеличение числа в регистре Т на единицу |
| 0 | 1 | Со сдвигом на две цифры | Установка 1БС3. Сдвиг полученной цифры частного в старшие разряды регистра Ф. Занесение косвенной функции "+:" |
| 1 | 1 | Со сдвигом на одну цифру | Установка 0БС2. Подготовка адреса для записи двух цифр частного в память, занесение косвенной функции "-:", занесение нуля в регистр Ф, увеличение числа в регистре Т на единицу |
| 1 | 0 | Без сдвига | Установка 0БС3. Сдвиг полученной цифры частного в старшие разряды регистра Ф. Занесение косвенной функции "+:" |

94С8. После вычисления последней цифры частного (признак 1БС5), а она всегда вычисляется вычитанием без сдвига или со сдвигом на 2 цифры, необходимо восстановить остаток. При 1БС5 выбор пути на восстановление остатка или на конец деления осуществляется с помощью указателя БС4:

если 0БС4 - на восстановление остатка,

если 1БС4 - на конец.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| М | Счетчик адреса команд | Не используется |
| Ф | | Хранится очередная цифра множителя. Накапливается цифра частного |
| Е | | Текущий адрес множимого или удвоенного множимого |
| Г | $\langle B_1 \rangle + {}^{D}{}_1 + L_1$ | Не используется |
| Р | | Текущий адрес сумматора |
| И | | В четырех младших разрядах хранится знак произведения. В старших разрядах - счетчик пар цифр множителя |
| П | $\langle B_2 \rangle + {}^{D}{}_2 + L_2$ | Не используется |
| Т | | Текущий адрес множителя. Хранится число для корректировки адреса поля делимого, с которого нужно начать вычитание (сложение). В старших четырех разрядах - счетчик вычитаемых байтов |

Продолжение

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| У | - | Хранятся байты операндов при промежуточных вычислениях |
| Д | Код операции | Счетчик количества вычитаемых (складываемых) байтов в команде деления |
| Л | $L_1$ $L_2$ | Счетчик цифр множителя (в старших четырех разрядах) |
| БС2 | ОБС2 | Индикатор, управляющий умножением. Индикатор, управляющий делением |
| БС3 | ОБС3 | Индикатор команды: ОБС3 - умножение, ІБС3 - деление; индикатор, управляющий делением |
| БС4 | ІБС4 | Индикатор знака второго операнда: ОБС4 - плюс, ІБС4 - минус; индикатор того, что предыдущая цифра была больше 5 (ІБС4) |
| БС5 | ІБС5 | Индикатор знака первого операнда: ОБС5 - плюс, ІБС5 - минус. Индикатор, управляющий умножением, признак того, что вычисляется младшая цифра частного (ІБС5) |
| ЛШ D6 | - | Запоминается $L_2$ |
| 96-97 A6-A7 B6-B7 C6-C7 | - | Хранится множитель или делитель |
| E7 F6-F7 | - | Хранится адрес конца первого операнда |
| D7 | - | Хранится знак результата |
| 04-05 I4-I5 24-25 34-35 44-45 54-55 64-65 74-75 | - | Хранится удвоенное множимое 2М |
| 84-85 94-95 A4-A5 B4-B5 C4-C5 D4-D5 E4-E5 F4-F5 | - | Хранится множимое |
| 06-07 I6-I7 26-27 36-37 46-47 56-57 66-67 76-77 | - | Поле сумматора при умножении |

I8.4. Микропрограмма F9SS

Команды

| Название | Мнем. | Формат | | | | | | | | Вход |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | байт 3 | байт 4 | байт 5 | байт 6 | |
| СРАВНЕНИЕ ДЕСЯТИЧНОЕ | СР | SS | F9 | $L_I$ | $L_2$ | $B_I$ | $D_I$ | $B_2$ | $D_2$ | I93 |

Время $24+2\ N_I + 2\ N_2$.

Операнды. Первый операнд - десятичное число в упакованном формате длиной $L_I$ + I байтов, расположенное в основной памяти по адресу $\langle B_I \rangle + D_I$.

Второй операнд - десятичное число в упакованном формате длиной $L_2$ + I байтов, расположенное в основной памяти по адресу $\langle B_2 \rangle + D_2$.

Результат. После сравнения устанавливается признак результата:

0 - операнды равны;

I - первый операнд меньше;

2 - первый операнд больше.

Прерывания. Данные: неверны коды знаков и цифр в операндах. Адресация. Защита памяти по чтению.

Алгоритм. Первое число сравнивается со вторым с учетом всех знаков и цифр. Проверяется допустимость кодов знаков и цифр. Признак результата отражает результат сравнения.

При одинаковых знаках чисел сравнение идет путем вычитания второго числа из первого, и если $L_I \neq L_2$, то более короткий операнд дополняется слева нулями.

Если знаки различные и хотя бы одно число отлично от нуля, то большим считается число со знаком плюс.

Нули с противоположными знаками считаются равными.

Диаграмма алгоритма. 95CI, 95BI. Устанавливается признак IBC4, если адрес конца первого операнда нечетный, и OBC4 - если он четный. Знак сравнивается с константами IIOI и IOII, и если совпадает с одной из них, то устанавливается признак IBC3 (минус), если же не совпадает, то проверяется, больше ли код знака 9, и устанавливается признак OBC3 (плюс).

Если код знака меньше или равен девяти, то фиксируются неправильные десятичные данные и осуществляется переход в программу прерываний.

95C2, 95B2. Определяется четность адреса конца второго операнда и устанавливается указатель IBC5, если он нечетный, и OBC5 - если четный. Знак второго операнда определяется так же, как и первого, и устанавливается признак IBC2, если знак минус и OBC2 - если знак плюс.

95B3, 95B4. Если знаки чисел разные, то каждый байт с помощью косвенной функции десятично складывается с нулем, чтобы проверить правильность всех цифровых кодов.

Счетчик длины операндов организуется в старших четырех разрядах регистра Л.

95A5, 95B5. После того, как все байты первого операнда обработаны, пропускается второй операнд, используются те же ветви микропрограммы, для чего адрес помещается в регистры ГРИ, и $L_2$ заносится в старшие разряды регистра Л.

95A6. После окончания обработки второго операнда устанавливается признак результата. Если хотя бы один из операндов отличен от нуля, то признак результата равен единице, когда знак первого числа минус , и признак результата равен двум, когда знак первого числа плюс. Если оба числа равны нулю, признак результата равен нулю.

95A4. Проверяется триггер НДД.

95ДO, 95ДI, 95EI. При одинаковых знаках операндов в регистре Л сохраняется $L_I\ L_2$, если $L_I \leq L_2$, и помещаем туда $L_2\ L_I$, если $L_I > L_2$. При этом устанавливается признак IBC2, если $L_I > L_2$, или OBC2, если $L_I \leq L_2$.

95Д2, 95С4, 95Д4. В зависимости от четности адресов концов операндов происходит переход на четыре ветви, которые затем объединяются в два цикла, по которым идет вычитание второго операнда из первого. Результат вычитания никуда не заносится. Из счетчика длины операндов в регистре Л одновременно вычитаются единицы из четырех младших и четырех старших разрядов. При появлении 1ШФ после очередного вычитания константы $II_{16}$ организуется выход из циклов.

95С5, 95Д5. После выхода из циклов проверяется, одинаковой ли длины были операнды. Если $L_1 = L_2$, то перед последним вычитанием константы $II_{16}$ из содержимого регистра Л там было записано $00_{16}$. Поэтому при выходе из цикла в регистре Л получится константа $EF_{16}$. Содержимое регистра Л сравнивается с константой Е. Если совпадает, то осуществляется переход на выработку признака результата и окончание команды.

95С6. По признаку 1БС2 происходит разветвление на два дополнительных цикла (блоки 95В7, 95С7), в которых продолжается обработка старших байтов более длинного операнда.

95Е6. Проверяется триггер НДД.

95Е7. Признак результата устанавливается по состоянию триггеров 1ПКФ, 1РКФ и индикатору знака операндов БС3.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г Р И | $\langle B_1 \rangle + D_1 + L_1$ | Хранится адрес обрабатываемого байта первого операнда |
| П Т У | $\langle B_2 \rangle + D_2 + L_2$ | Хранится адрес обрабатываемого байта второго операнда |
| Д | Код операции | Для хранения байта операнда при вычитании |
| Л | $L_1$ $L_2$ | Счетчик длины операндов |
| БС2 | 0 | Индикатор знака второго операнда: 0БС2 - плюс, 1БС2 - минус. Если знаки операндов одинаковые, то индикатор относительной длины операндов: 1БС2, при $L_1 > L_2$, 0БС2, при $L_1 \leq L_2$ |
| БС3 | 0 | Индикатор знака первого операнда: 0БС3 - плюс, 1БС3 - минус |
| БС4 | 1 | Индикатор четности адреса конца первого операнда: 1БС3 - нечетный, 0БС3 - четный; индикатор выхода из цикла при прогоне операндов через арифметический блок, когда знаки операндов различны |
| БС5 | 1 | Индикатор четности адреса конца второго операнда: 0БС5 - четный, 1БС5 - нечетный |

# I9. ДЕСЯТИЧНАЯ АРИФМЕТИКА. ОПЕРАЦИИ РЕДАКТИРОВАНИЯ

## I9.I. Микропрограмма ДЕДF

Команды

| Название | Мнем. | Формат | | | | | | | Вход |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | байт 3 | байт 4 | байт 5 | байт 6 | |
| ОТРЕДАКТИРОВАТЬ | EDD | SS | DE | L | $B_I$ | $D_I$ | $B_2$ | $D_2$ | I9C |
| ОТРЕДАКТИРОВАТЬ И ОТМЕТИТЬ | EDMK | SS | DF | L | $B_I$ | $D_I$ | $B_2$ | $D_2$ | I9E |

Время.

EDD : I9+7N +2,5Z + R + 5 $N_2$ - 0,5S

EDMK: I9+7N +2,5Z + R + 5 $N_2$ - 0,5S + 50TM + 3A.

Операнды. Исходные данные - одно или несколько упакованных десятичных чисел, расположенных в основной памяти, начиная с адреса $\langle B_2 \rangle + D_2$.

Образец - набор произвольных сигналов, расположенный в поле основной памяти длиною в L +I байтов по адресу $\langle B_I \rangle + D_I$. Три символа образца имеют специальное назначение:

20 - символ выбора цифры,

2I - символ начала значимости,

22 - символ разделения полей

(20, 2I и 22 - шестнадцатиричные коды).

Эти три символа замещаются либо цифрой исходных данных, либо символом-заполнителем, в качестве которого используется первый символ образца.

Допускается перекрытие полей исходных данных и образца.

Результат. Формат исходных данных меняется с упакованного на формат с зоной. Кроме того, происходит редактирование по образцу. Результат замещает образец. Перекрытие полей дает результаты, которые нельзя предсказать заранее.

Оба операнда обрабатываются слева направо - символ за символом. Три фактора влияют на то, какой символ помещается в поле первого операнда: цифра, получаемая из поля исходных данных, символ образца и состояние триггера, называемого S-триггером.

S - триггер используется для того, чтобы управлять занесением или замещением цифр исходных данных и символов образца. Замещаются цифры исходных данных в том случае, когда нужно погасить (уничтожить) нули слева. Те цифры, которые заносятся в результат, будут именоваться значащими независимо от того, равны они нулю или нет. Символы образца замещаются или заносятся в память, если они зависят от значимости (например, знаки пунктуации) или от знака (например, символы кредита).

S - триггер используется также для того, чтобы записать знак исходного числа и соответствующим образом установить признак результата.

S - триггер устанавливается в нуль в начале операции, а затем его состояние меняется в зависимости от исходного числа и символов образца.

Символ выбора цифры (20) вызывает выборку очередной цифры исходных данных. Цифра исходных данных замещает символ образца, если S-триггер в единице или эта цифра не нуль. Если ненулевая цифра вставляется при нулевом S-триггере, то S -триггер устанавливается в единицу, чтобы указать, что последующие цифры значащие. Если и в S -триггере нуль и цифра исходных данных равна нулю, символ образца замещается символом - заполнителем.

Когда цифра исходных данных помещается в поле результата, то к ней добавляется зона. Если в I2-м разряде ССП стоит нуль, вырабатывается код зоны IIII, соответствующий ДКОИ. Если в I2-м разряде ССП - единица, вырабатывается код зоны 0I0I, соответствующий КОИ-8. Цифры исходных данных анализируются только один раз. Они выбираются из поля по восемь разрядов сразу. Левые четыре разряда анализируются первыми, а правые четыре разряда хранятся до следующего симво-

ла образца, который вызывает анализ цифры. Однако проверка, не находится ли в правых четырех разрядах код знака, производится сразу же после анализа левых четырех разрядов.

В результате проверки любой из кодов знака плюс (1010, 1100, 1110, 1111) устанавливает S -триггер в нуль, тогда как коды знака минус (1011 и 1101) оставляют S -триггер неизменным. Когда в правых четырех разрядах встречается один из кодов знака, эти разряды больше не рассматриваются как цифра, а для анализа следующей цифры из образца выбирается новый символ. Знак плюс устанавливает S -триггер в нуль, даже в том случае, когда триггер был установлен в единицу ненулевой цифрой из того же байта исходных данных или символом начала значимости для этой цифры.

Символ начала значимости (21) выполняет те же функции и, кроме того, указывает установкой S-триггера в единичное состояние, что следующие цифры значащие.

Символ разделения полей (22) ограничивает отдельные поля при редактировании нескольких полей с помощью одной команды. Этот символ замещается символом-заполнителем. S -триггер устанавливается в нуль, в результате чего проверка на нули начинается сначала.

Все другие символы образца обрабатываются одинаково: если S -триггер в единице, символ образца не меняется; если S-триггер в нуле, символ образца замещается символом-заполнителем.

В случае команды ЕРМК дополнительно в общий регистр 1 заносится адрес каждой первой значащей цифры результата. Адрес байта заносится в разряды 8-31 этого регистра. Разряды 0-7 не меняются. Адрес байта заносится каждый раз, когда S-триггер находится в нуле и в поле результата помещается ненулевая цифра. Если же значимость вводится символом начала значимости, стоящим в образце, и при этом в результат помещается символ-заполнитель, то адрес байта не заносится.

Устанавливается признак результата. Признак результата устанавливается в процессе анализа полей, определенных символами разделения полей, независимо от числа встреченных кодов знака. При операциях редактирования над несколькими полями установленный признак результата относится только к полю, стоящему за последним символом разделения полей. Если последний символ образца является символом разделителей полей, признак результата устанавливается равным нулю.

С помощью признака результата регистрируется знак числа последнего редактированного поля и равенство этого числа нулю, что облегчает гашение (т.е. замену пробелами) полей нулевых чисел. Для этого все цифры в процессе работы проверяются на комбинацию 0000, и в конце операции в признаке результата фиксируется факт наличия или отсутствия в исходных данных сплошных нулей.

Признак результата устанавливается равным 0 при нулевом поле в исходных данных вне зависимости от состояния S-триггера, а также когда исходные данные не исследуются, т.е. в образце нет ни символов выбора цифры, ни символов начала значимости.

Признак результата устанавливается равным 1, если содержимое поля результата меньше нуля. Признак результата устанавливается равным 2 при ненулевом поле исходных данных и единичном состоянии S -триггера. Это означает, что результат больше нуля.

Ниже, в табл. 25, приведены некоторые сведения об операции редактирования. В двух левых столбцах даются символы образца и их коды. В следующих столбцах приведено значение цифр и S - триггера, которые используются для выбора одного из возможных действий. В самом правом столбце указано новое состояние S -триггера.

Таблица 25

| Код символа | Название и назначение | Анализируется ли цифра | Состояние триггера | Значение цифры | Символ, помещаемый в результат | Переключение триггера |
|---|---|---|---|---|---|---|
| 0010 0000 | Выборка цифры | Да | S=1<br>S=0<br>S=0 | <br>d ≠ 0<br>d =0 | Цифра<br>Цифра<br>Заполнитель | S=1 |
| 0010 0001 | Начало значимости | Да | S=1<br>S=0<br>S=0 | <br>d ≠ 0<br>d =0 | Цифра<br>Цифра<br>Заполнитель | S=1<br>S=1 |

| Код символа | Название и назначение | Анализируется ли цифра | Состояние триггера | Значение цифры | Символ, помещаемый в результат | Переключение триггера |
|---|---|---|---|---|---|---|
| 0010 0010 | Разделитель полей | Нет | | | Заполнитель | S=0 |
| Другие | Включение текста | Нет | S=1<br>S=0 | | Оставить<br>Заполнитель | |

Обозначения:

d - цифра исходных данных;

S - S -триггер (1-использованы знак "минус", цифры или символ образца; 0 - использованы знак "плюс", символ-заполнитель);

цифра - цифра исходных данных замещает символ образца;

заполнитель - символ-заполнитель замещает символ образца;

оставить - символ образца не меняется.

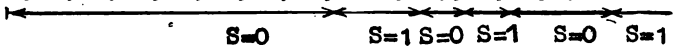
Пример.

Образец:

23 AB CD 00 01 20 22 21 EF 20 20 22 22 21

Результат:

23 23 23 23 23 23 23 23 EF 50 53 23 23 58




Исходные данные:

00 0A 3B 89

Код зоны равен 0101=5.

Признак результата равен 1

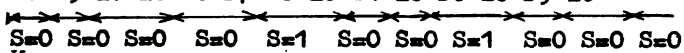
При выполнении операции ОТРЕДАКТИРОВАТЬ И ОТМЕТИТЬ в общий регистр 1 заносится адрес 14-го байта результата.

Образец:

20 21 20 22 A0 21 20 22 20 21 20 22 20 AB

Результат:

F8 F9 20 20 F7 F0 20 F4 20 F0 20 F9 20




Исходные данные:

8C 9A 0E 7B 0B 4F 00 9E

Код зоны равен 1111=F

Признак результата равен 2

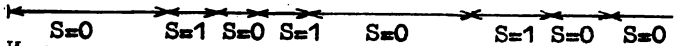
Команда ОТРЕДАКТИРОВАТЬ И ОТМЕТИТЬ в общий регистр 1 заносит адрес 13-го байта результата

Образец:

22 21 22 20 20 20 21 22 21 21 21 2B 22 2C

Результат:

22 58 22 51 52 53 54 22 55 56 57 2B 22 22




Исходные данные:

8A 1B 2C 3D 4E 5F6F 7D

Код зоны равен 0101=5

Признак результата равен 0

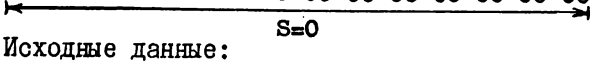
Команда ОТРЕДАКТИРОВАТЬ И ОТМЕТИТЬ в общий регистр 1 заносит адрес 11-го байта результата

Образец:

CC 20 20 20 20 AB 20 CD EF 00 20 01 22 20

Результат:

CC CC CC CC CC CC CC CC CC CC CC CC CC CC




Исходные данные:

00 00 00 00 00

Признак результата равен 0

Команда ОТРЕДАКТИРОВАТЬ И ОТМЕТИТЬ содержимое общего регистра 1 не изменяет

Прерывания. Данные: у исходных данных (в левом полубайте) неверен код десятичной цифры. Выполнение операции прекращается.

Адресация. Защита памяти по чтению и записи.

Алгоритм. Замещение образца отредактированным операндом ведется последовательно по одному байту слева направо в таком порядке. На регистры Н и З читается четно-нечетная пара байтов образца. Первый байт образца, заполнитель, запоминается. Формируется содержимое регистра Н по трем факторам: замещаемому символу образца, состоянию S -триггера и цифре исходных данных. Четно-нечетная пара помещается в поле образца, тем самым нечетный байт образца регенерируется, а четный символ образца замещается отредактированным. Затем аналогично формируется содержимое регистра З. Сформированная пара помещается в поле образца. Здесь четный байт, сформированный на предыдущем этапе, регенерируется, нечетный - замещает символ образца.

В случае команды ОТРЕДАКТИРОВАТЬ И ОТМЕТИТЬ, кроме замещения образца, при необходимости в общий регистр I заносится адрес каждой первой значащей цифры результата.

Диаграмма алгоритма. В микропрограмме имеются две ветки: ветвь формирования содержимого регистра Н и ветвь формирования содержимого регистра З. Ниже даются пояснения к некоторым блокам диаграммы.

97AI. 97BI. Здесь же происходит запоминание значения СчАК (состояния МФЕ) в локальной памяти.

97A6. При чтении байтов исходных данных будет запорчена пара байтов образца, находящаяся в регистрах Н и З. Здесь из основной памяти снова читается в регистры Н и З прежняя пара байтов образца.

97C4. Модификацией адреса на +I обеспечивается повторное чтение одной и той же пары байтов.

97Д9. 97E9. Состояния IБС3 и IНДД (одновременно) не имеют места.

98C4. По состоянию триггера БС5 ( S-триггер) устанавливается ТЗН. Двоичным сложением, заведомо не дающим переполнения, гасится возможно установленный триггер переполнения. Устанавливается признак результата с помощью установки КУI.

99BI. Так как проверка на цифровую комбинацию, зонирование и занесение в образец как левой так и правой цифры байта исходных данных идет по одной и той же ветви и предполагает, что цифра занимает старшие разряды регистра Ф, то при обнаружении в правом полубайте исходных данных цифры она пересылается на место старших разрядов регистра Ф. Устанавливается индикатор наличия необработанных байтов исходных данных, означающий, что цифра расположена в младшем полубайте исходных данных.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| М | Адрес следующей команды | Используется для запоминания байта исходных данных |
| Ф<br>Е | | Для запоминания старших разрядов адреса первой значащей цифры отредактированного числа |
| Г<br>Р<br>И | $\langle B_I \rangle + D_I$ | Хранит адрес байтов образца |
| П<br>Т<br>У | $\langle B_2 \rangle + D_2$ | Хранит адрес байтов источника |
| Д | - | Запоминание символа-заполнителя. Рабочий регистр при установке признака результата и запоминании I2-го разряда ССП |

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Л | L | Счетчик обрабатываемых байтов образца |
| Н | - | Хранит обрабатываемый байт образца |
| З | - | Хранит обрабатываемый байт образца |
| БС3 | 0 | Индикатор наличия необработанных прочитанных цифр исходных данных:<br>0 - нет,<br>I - есть |
| БС2 | 0 | Индикатор четности адреса цифры исходных данных:<br>0 - "чет",<br>I - "нечет" |
| БС4 | 0 | Хранит I2-й разряд ССП |
| БС5 | I | S-триггер |
| БС6 | - | Индикатор занесения адреса в общий регистр I |
| БС7 | - | Индикатор выполняемой команды:<br>0 - отредактировать,<br>I - отредактировать и отметить |
| РКФ | - | Индикатор ненулевой цифры в исходных данных |
| ПКФ | - | Индикатор четности адреса байта образца:<br>I - "чет",<br>0 - "нечет" |

## 20. ПЛАВАЮЩАЯ ЗАПЯТАЯ. АРИФМЕТИЧЕСКИЕ ОПЕРАЦИИ И ОПЕРАЦИИ СРАВНЕНИЯ

### 20.I. Микропрограмма ASCFP

Команды

| Название | Мнем. | Формат | | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | байт 3 | байт 4 | | |
| СЛОЖЕНИЕ С НОРМАЛИЗАЦИЕЙ (ДЛИННОЕ) | $A^DR$ | FR | 2A | $R_I$ | $R_2$ | | | II5 | 84 |
| СЛОЖЕНИЕ С НОРМАЛИЗАЦИЕЙ (ДЛИННОЕ) | $A^D$ | RX | 6A | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I55 | 88 |
| СЛОЖЕНИЕ С НОРМАЛИЗАЦИЕЙ (КОРОТКОЕ) | AER | RR | 3A | $R_I$ | $R_2$ | | | II5 | 52 |
| СЛОЖЕНИЕ С НОРМАЛИЗАЦИЕЙ (КОРОТКОЕ) | AE | RX | 7A | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I55 | 5I |
| СЛОЖЕНИЕ БЕЗ НОРМАЛИЗАЦИИ (ДЛИННОЕ) | AWR | RR | 2E | $R_I$ | $R_2$ | | | II$^D$ | 70 |

Продолжение

| Название | Мнем. | Формат | | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | | байт 3 | байт 4 | | |
| СЛОЖЕНИЕ БЕЗ НОРМАЛИЗАЦИИ (ДЛИННОЕ) | AW | RX | 6E | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I5D | 63 |
| СЛОЖЕНИЕ БЕЗ НОРМАЛИЗАЦИИ (КОРОТКОЕ) | AWR | RR | 3E | $R_I$ | $R_2$ | | | IID | 46 |
| СЛОЖЕНИЕ БЕЗ НОРМАЛИЗАЦИИ (КОРОТКОЕ) | AV | RX | 7E | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I5D | 46 |
| ВЫЧИТАНИЕ С НОРМАЛИЗАЦИЕЙ (ДЛИННОЕ) | SDR | RR | 2B | $R_I$ | $R_2$ | | | II7 | 89 |
| ВЫЧИТАНИЕ С НОРМАЛИЗАЦИЕЙ (ДЛИННОЕ) | SD | RX | 6B | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I57 | 88 |
| ВЫЧИТАНИЕ С НОРМАЛИЗАЦИЕЙ (КОРОТКОЕ) | SER | RR | 3B | $R_I$ | $R_2$ | | | II7 | 58 |
| ВЫЧИТАНИЕ С НОРМАЛИЗАЦИЕЙ (КОРОТКОЕ) | SE | RX | 7B | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I57 | 54 |
| ВЫЧИТАНИЕ БЕЗ НОРМАЛИЗАЦИИ (ДЛИННОЕ) | SWR | RR | 2F | $R_I$ | $R_2$ | | | IIF | 64 |
| ВЫЧИТАНИЕ БЕЗ НОРМАЛИЗАЦИИ (ДЛИННОЕ) | SW | RX | 6F | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I5F | 85 |
| ВЫЧИТАНИЕ БЕЗ НОРМАЛИЗАЦИИ (КОРОТКОЕ) | SVR | RR | 3F | $R_I$ | $R_2$ | | | IIF | 48 |
| ВЫЧИТАНИЕ БЕЗ НОРМАЛИЗАЦИИ (КОРОТКОЕ) | SV | RX | 7F | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I5F | 5I |
| СРАВНЕНИЕ (ДЛИННОЕ) | CDR | RR | 29 | $R_I$ | $R_2$ | | | II3 | 49 |
| СРАВНЕНИЕ (ДЛИННОЕ) | CD | RX | 69 | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I53 | 53 |
| СРАВНЕНИЕ (КОРОТКОЕ) | CER | RR | 39 | $R_I$ | $R_2$ | | | II3 | 35 |
| СРАВНЕНИЕ (КОРОТКОЕ) | CE | RX | 79 | $R_I$ | $X_2$ | $B_2$ | $D_2$ | I53 | 33 |

Операнды. Два коротких (команды AER, AE, AVR, AV, SER, SE, SVR, SV , CER, CE) или два длинных (команды ADR, AD, AWR, AW, SDR, SD , SWR, SW , CDR, CD ) числа с плавающей запятой, одно из которых расположено всегда в регистре плавающей запятой с номером $R_I$, а второе – в регистре плавающей запятой с номером $R_2$ (формат RR) или в основной памяти по адресу $\langle X_2 \rangle$ + + $\langle B_2 \rangle$ + $D_2$.

Результат. Нормализованная (команды ADR, AD , AER, AE) или ненормализованная (команды AWR, AW , AVR, AV ) алгебраическая сумма двух чисел с плавающей запятой, или нормализованная (команды DR, SD , SER, SE ), или ненормализованная (команды SWR, SW , SVR, SV ) алгебраическая разность первого и второго числа, расположенные в регистре плавающей запятой с номером $R_I$. Результаты в операциях над короткими операндами имеют мантиссу из шести цифр, а над длинными – из четырнадцати цифр. Но промежуточные результаты имеют дополнительную цифру, которая служит для увеличения точности конечного результата.

Число с нулевой характеристикой, нулевой мантиссой и положительным знаком называется истинным нулем. Истинный нуль может появиться в результате при определенных величинах операндов.

Устанавливается признак результата:

0 – мантисса результата равна нулю;

I – результат меньше нуля;

2 – результат больше нуля.

В командах сравнения (команды CDR, CD, CER, CE) первое число сравнивается со вторым и в зависимости от результата сравнения устанавливается признак результата:

0 - числа равны,

I - первое число меньше,

2 - первое число больше.

Прерывания. Спецификация: короткий операнд не располагается в границах 32-разрядных слов или длинный операнд не располагается в границах 64-разрядных слов, или при указании номера регистра плавающей запятой указан номер, отличный от I,2,4,6; команда не выполняется, признак результата и данные в регистрах и в памяти остаются без изменения.

Переполнение порядка: в порядке результата при сложении и вычитании возникает переполнение, а мантисса результата отлична от нуля. Мантисса результата нормализована и правильна, знак правильный, а характеристика на $128_{10}$ меньше, чем правильная.

Исчезновение порядка: при выполнении операций сложения и вычитания с нормализацией результата порядок стал меньше нуля, а мантисса результата отлична от нуля.

При единичном разряде маски исчезновения порядка мантисса нормализована и правильна, знак правилен, характеристика на $128_{10}$ больше, чем правильная. При нулевом разряде маски операция завершается тем, что на место результата записывается истинный нуль.

Значимость: при сложении или вычитании мантисса результата равна нулю. Значение разряда маски потери значимости влияет на результат. При нулевом значении разряда маски выполнение операции завершается тем, что на место результата записывается истинный нуль. При единичном значении разряда маски характеристика и знак мантиссы остаются без изменения.

Адресация (только для команд формата RX). Защита памяти по чтению (только для команд формата RX).

Алгоритм. В микропрограмме операция ВЫЧИТАНИЕ приводится к операции СЛОЖЕНИЕ путем инвертирования знака второго операнда. Сложение двух чисел с плавающей запятой заключается в выравнивании характеристик и сложении мантисс. Характеристики операндов вычитаются, и мантисса операнда с меньшей характеристикой сдвигается вправо на количество цифр, равное величине разности характеристик.

Затем, в зависимости от знаков операндов, производится сложение или вычитание мантисс и получается промежуточная сумма. Промежуточная сумма имеет дополнительную цифру, которая появляется при сдвиге одной из мантисс вправо, только одна такая цифра принимает участие в сложении мантисс. Эта цифра равна нулю, если отсутствовало выравнивание характеристик. Если при сложении возникает переполнение, промежуточная сумма сдвигается вправо на одну цифру, при этом характеристика увеличивается на единицу.

После сложения мантисс для операций с нормализацией промежуточная сумма сдвигается влево для получения нормализованной мантиссы, в освобождающиеся разряды записываются нули, а характеристика уменьшается на число единиц, равное числу сдвигов. Если сдвиги влево промежуточной суммы отсутствовали, промежуточная сумма обрезается до нужного числа разрядов. Знак суммы определяется по алгебраическим правилам.

Пример. **Сложение с нормализацией.**

| | |
|---|---|
| 86 **FFFFF FFFFFFFF** | Первое число |
| 09 00000 8765432I | Второе число |
| 89 **000FF FFFFFFFF** [F] | Выравнивание порядков |
| | [ ] - дополнительная цифра |
| 00000 8765432I | Вычитание мантисс, так как знаки операндов различны |
| -**000FF FFFFFFFF** [F] | |
| **FFF**00 8765432I [I] | Результат получился в дополнительном коде |
| 00000000000000 | |
| -**FFF**00 F8765432I [I] | Перевод мантиссы результата в прямой код |
| 000FF0789ABCDE [F] | |

| | |
|---|---|
| 89 000FF0789ABCDE [F] | Присвоение результату знака и характеристики |
| 86 FF0789ABCDEF00 | Нормализация результата |

Пример. **Вычитание с нормализацией.**

| | |
|---|---|
| 0BA870I0 | Первый операнд |
| 8DFFDI34 | Второй операнд |
| 0DFFDI34 | Инвертирование знака второго операнда |
| 0D00A870 [I] | Выравнивание порядков |
| 00A870<br>+FFDI34 | Сложение мантисс, так как знаки операндов стали одинаковыми |
| 0079A4 | Имеет место переполнение |
| I00079A | Сдвиг мантиссы вправо на один разряд, так как имело место переполнение |
| 0EI0079A | Результат. Порядок увеличен на единицу, так как имел место сдвиг мантиссы вправо |

Операция СРАВНЕНИЕ начинается с анализа знаков операндов. При неравных знаках, если мантисса хотя бы одного операнда не равна нулю, то большим считается тот операнд, у которого положительный знак. Если знаки операндов равны, то производится выравнивание порядков и вычитание мантисс с учетом дополнительной цифры. Числа с нулевыми мантиссами считаются равными даже если они отличаются знаками и характеристиками.

Пример. **Сравнение.**

| | |
|---|---|
| 40 000000IIIIIII | Первый операнд |
| F5 000FFF0000FFFF | Второй операнд |
| 2 | Признак результата, так как знаки разные и знак "+" у первого числа |

Пример. **Сравнение.**

| | |
|---|---|
| 25 00DC42 | Первый операнд |
| 24 0FFIII | Второй операнд |
| 25 00FFII [I] | Знаки у операндов совпадают. Производим выравнивание порядков |
| 00DC42<br>-00FFII [I] | |
| FFDD30 [F] | Вычитание мантисс |
| I | Признак результата, так как вычитаем второе число из первого и результат получился в дополнительном коде |

Диаграмма алгоритма. Подготовка к сложению (вычитанию) мантисс (блоки I02A4, I02B4, I02Д4) реализована установкой соответствующей косвенной функции в зависимости от знаков обоих операндов и четности разницы порядков. При этом полагается, что в дальнейшем будет сдвигаться второй операнд. Косвенная функция заносится с перекосом: если разница порядков - нечетное число цифр. Этим осуществляется сдвиг операнда на одну цифру вправо. Ниже приведен выбор косвенной функции в зависимости от знаков операндов:

| Знак первого операнда | Знак второго операнда | Косвенная функция |
|---|---|---|
| + | + | ЗКФ (+) |
| + | – | ЗКФ (–) |
| – | + | ЗКФ (– –) |
| – | – | ЗКФ (+) |

Блоки I03A2 и I03E2. Сдвиг мантиссы операнда с меньшим порядком вправо реализуется генерированием соответствующего адреса сдвигаемого операнда. Адрес байта, который после сдвига будет младшим, находится вычитанием из адреса младшего байта сдвигаемого операнда целой части от деления разницы порядков на 2.

Блоки I03A4, I03B4, I03Д4, I03E4. Чтобы сделать выполнение функции над байтами сдвигаемого и несдвигаемого операндов независимым от четности разницы порядков, после чтения каждой пары байтов сдвигаемого операнда они запоминаются в регистрах Л и Д или Д и Т. Регистры Л и Д используются, когда разница порядков – четное число цифр, четное число целых байтов и нечетное число цифр, нечетное число целых байтов. Регистры Д и Т используются, если разница порядков – четное число цифр, нечетное число целых байтов и нечетное число цифр, четное число целых байтов. Для коротких операндов в старшем полубайте запоминается дополнительная цифра.

На рис. I5, I6, I7, I8 показано, каким образом запоминаются байты сдвигаемого операнда в регистрах и каким образом выполняется функция над байтами несдвигаемого операнда, которые всегда находятся в регистрах Н и З, и над байтами сдвигаемого операнда, которые находятся в регистрах Л и Д или Д и Т.

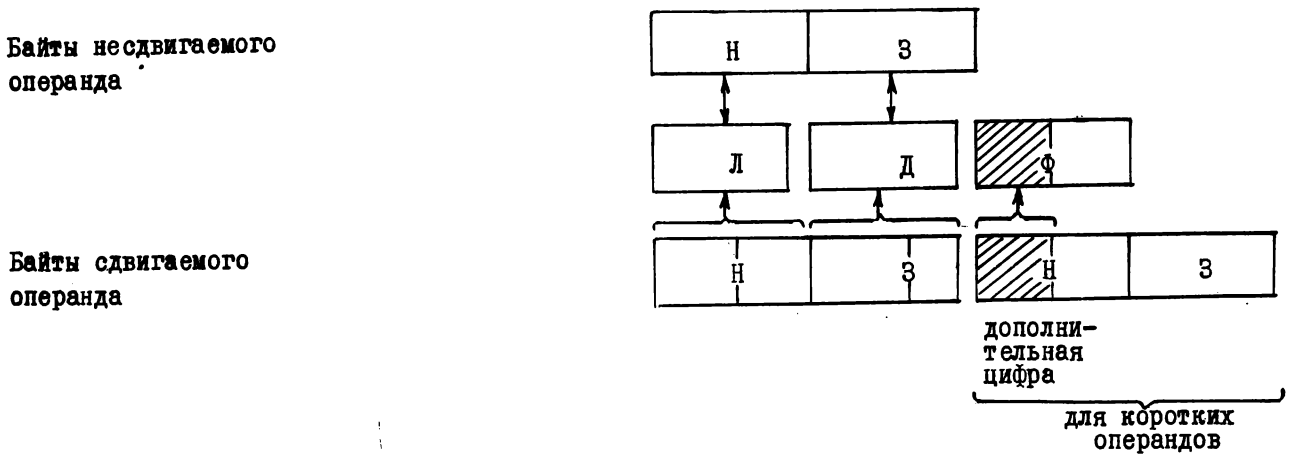



Рис. I5. Разница порядков – четное число цифр, четное число целых байтов

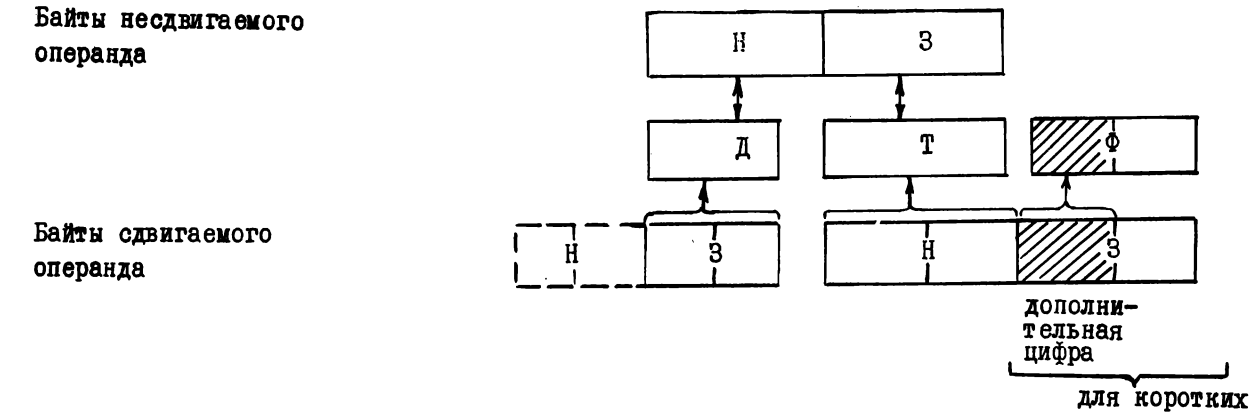



Рис. I6. Разница порядков – четное число цифр, нечетное число целых байтов

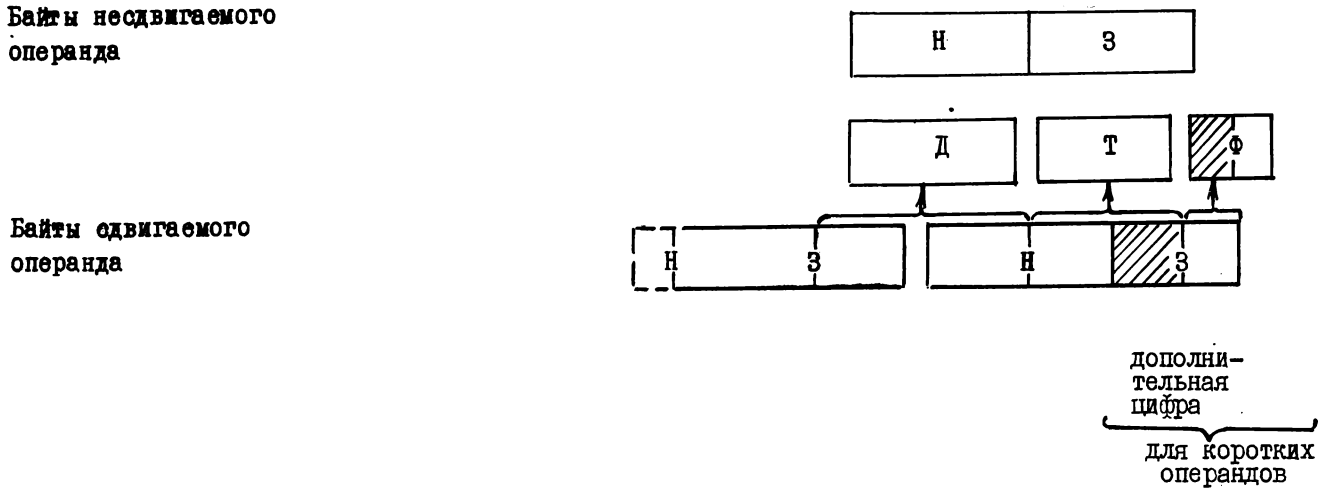



Рис. I7. Разница порядков - нечетное число цифр, четное число целых байтов

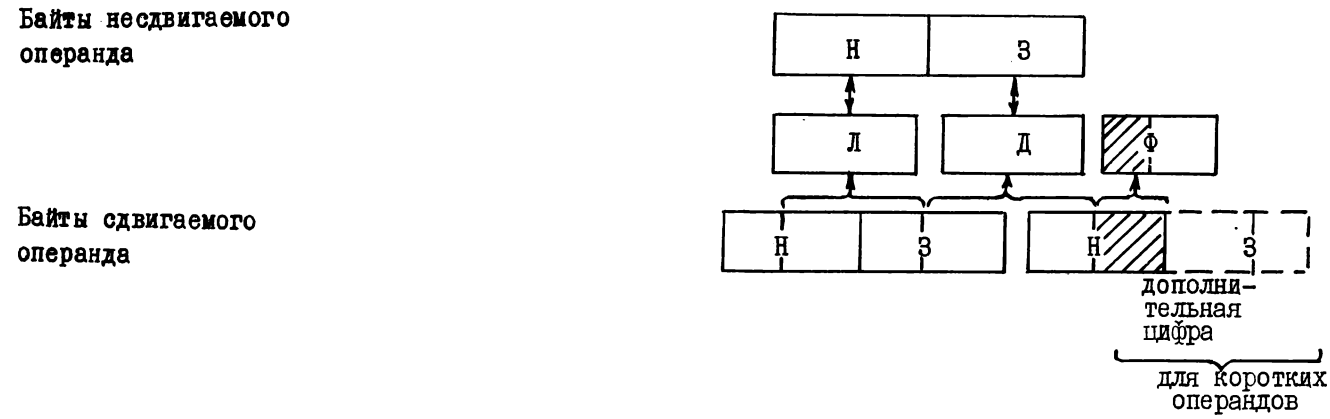



Рис. I8. Разница порядков - нечетное число цифр, нечетное число целых байтов

В микропрограмме блоки I03A4 и I03B4 фактически представляют собой один блок, т.е. блок I03A4 содержит в себе блок I03B4. Точно также блоки I03Д4 и I03E4 составляют один блок. Выход из блоков I03C4 и I03E4 осуществляется по адресу старшего байта сдвигаемого операнда. После выхода из этих блоков выполняется цикл (блоки I03B4 и I03Д4), в котором оставшиеся байты несдвигаемого операнда складываются (вычитаются) с нулями. Выход из этого цикла производится по адресу старшего байта результата. В случае, когда разница порядков больше 6 (для коротких операндов) и больше I3 (для длинных), выполняется сразу цикл сложения (вычитания) мантиссы несдвигаемого операнда с нулем.

В блоках I03A5 и I03E5 результат не записывается в память, а служит для выработки признака результата.

В блоках I03A4, I03B4, I03Д4, I03E4, I03A5, I03E5 на вход B подается операнд, который сдвигается, а именно: в блоках I03A4, I03B4, I03A5 - первый операнд, в блоках I03Д4, I03E4, I03E5 - второй операнд.

Блок I03Д6 формирует знак результата, если выполнялась функция вычитания. Знак результата формируется следующим образом.

Занесение косвенной функции осуществлялось исходя из предположения, что сдвигаться будет второй операнд. Если в действительности сдвигался второй операнд, то знак и код результата (прямой или дополнительный) определяются по состоянию триггера переноса, а именно:

| ТПКФ | Знак результата | Код результата |
|---|---|---|
| 0 | + | Прямой |
| I | – | Дополнительный |

Если сдвигался первый операнд, то:

| ТПКФ | Знак результата | Код результата |
|---|---|---|
| 0 | – | Прямой |
| I | + | Дополнительный |

В блоках I04EI и I04E2 производится нормализация. Результат заносится в регистры процессора (Л, Д, И, Ф, Н и З). При этом, если число нулевых старших цифр нечетно, то при занесении осуществляется сдвиг на одну цифру влево. Затем результат пересылается в память таким образом, чтобы старшая цифра была ненулевой.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| М | Адрес следующей микрокоманды | Не используется |
| Ф | | Рабочий регистр в блоке I04E2<br>Запоминание дополнительной цифры при коротких операндах в блоках I03A4, I03B4, I03Д4, I03E4, I03A5, I03E5 |
| Е | | Хранит адрес сдвигаемого операнда в блоке I03A2<br>Хранит адрес промежуточного результата в блоках I03Д4, I03E4<br>Запоминание порядка результата в блоках I03C5, I03Д5 |
| Г | Для формата RX: $\langle X_2 \rangle + B_2 + D_2$<br>Для формата RR: $R_2$ I000 | Не используется |
| Р | | Запоминание старшей цифры результата в блоках I03A4, I03B4, I03Д4, I03E4<br>Рабочий регистр в блоке I04E2 |
| И | | Адресный регистр оперативной памяти<br>Рабочий регистр в блоке I04E2 |
| П | – | Запоминание знака результата в блоках I02B6, I02C6, I03B6 |
| Т | $\langle R_2$ I000$\rangle$ | Запоминание разницы порядков в блоке I02C7<br>Рабочий регистр в блоках I03A4, I03B4, I03Д4, I03E4, I03A5, I03E5<br>Запоминание количества нулевых старших цифр в блоке I04EI<br>Хранит адрес промежуточного результата в блоках I03E7, I04E2 |

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| У | $\langle R_I\ I000 \rangle$ | Хранит адрес несдвигаемого операнда в блоке I03E2<br>Хранит адрес промежуточного результата в блоках I03A4, I03B4<br>Хранит адрес конечного результата в блоках I03E7, I04E2 |
| Д | Код операции | Запоминание порядка второго операнда в блоке I02CI<br>Рабочий регистр в блоках I03A4, I03B4, I03Д4, I03E4, I03A5, I03E5, I04E2 |
| Л | Для формата RR: $R_I\ R_2$<br>Для формата RX: $R_I\ X_2$ | Рабочий регистр в блоках I03A4, I03B4, I03Д4, I03E4, I03A5, I03E5, I04E2 |
| БС2 | 0 – формат RX<br>I – формат RR | Индикатор формата команды<br>Рабочий индикатор в блоках I03A7, I04EI |
| БС3 | 0 | 0 – операция сложения и вычитания<br>I – операции сравнения<br>Рабочий индикатор в блоке I04EI |
| БС4 | 0 – короткий операнд<br>I – длинный операнд | Индикатор длины операнда |
| БС5 | 0 – операция с нормализацией<br>I – операция без нормализации | Индикатор нормализации результата<br>Рабочий индикатор в блоке I04EI |

## 20.2. Микропрограммы DINOR, MFP, DFP, RDIN

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | байт 3 | байт 4 | | |
| УМНОЖЕНИЕ (КОРОТКОЕ) | MER | RR | 3C | $R_I\ R_2$ | | | II9 | 480 |
| УМНОЖЕНИЕ (КОРОТКОЕ) | ME | RX | 7C | $R_I\ X_2$ | $B_2$ | $D_2$ | I59 | 480 |
| УМНОЖЕНИЕ (ДЛИННОЕ) | MDR | RR | 2C | $R_I\ R_2$ | | | II9 | I230 |
| УМНОЖЕНИЕ (ДЛИННОЕ) | MD | RX | 6C | $R_I\ X_2$ | $B_2$ | $D_2$ | I59 | I230 |
| ДЕЛЕНИЕ (КОРОТКОЕ) | DER | RR | 3D | $R_I\ R_2$ | | | IIB | 380 |
| ДЕЛЕНИЕ (КОРОТКОЕ) | DE | RX | 7D | $R_I\ X_2$ | $B_2$ | $D_2$ | I5B | 380 |
| ДЕЛЕНИЕ (ДЛИННОЕ) | DDR | RR | 2D | $R_I\ R_2$ | | | IIB | 2050 |
| ДЕЛЕНИЕ (ДЛИННОЕ) | DD | RX | 6D | $R_I\ X_2$ | $B_2$ | $D_2$ | I5B | 2050 |

Операнды. Два коротких (команды MER, ME, DER, DE) или длинных (команды MDR, MD , DDR, DD) числа с плавающей запятой, одно из которых расположено в регистре плавающей запятой $R_1$, а второе в регистре плавающей запятой $R_2$ (команды формата RR) или в основной памяти по адресу $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ (команды формата RX).

Результат. Команды умножения. Нормализованное произведение исходных чисел, записанное в регистре плавающей запятой $R_1$. Мантисса произведения оказывается нормализованной за счет предварительной нормализации исходных операндов и нормализации промежуточного произведения, если в этом есть необходимость. Для длинных операндов перед нормализацией мантисса обрезается до I5 разрядов. Для коротких операндов (мантиссы из шести цифр) мантисса произведения имеет I4 цифр, причем две младшие цифры мантиссы всегда равны нулю. Знак произведения определяется по алгебраическим правилам. Если все цифры мантиссы промежуточного произведения равны нулю, то результат делается равным истинному нулю.

Команды деления. Нормализованное частное от деления числа, расположенного в регистре $R_1$ (делимое) на число, расположенное в регистре $R_2$ или в основной памяти (делитель), записанное в регистр $R_1$. В случае коротких операндов младшие 32 разряда регистра $R_1$ игнорируются и остаются без изменения. Мантисса оказывается нормализованной за счет обязательной предварительной нормализации исходных чисел. В образовании частного принимают участие все цифры мантиссы делимого, даже если нормализованная мантисса делимого больше нормализованной мантиссы делителя. Необходимость в нормализации промежуточного частного никогда не возникает, хотя может потребоваться сдвиг вправо. При сдвигах характеристика промежуточного произведения корректируется. Мантисса частного обрезается до нужного числа цифр. Знак частного определяется по алгебраическим правилам. Если мантисса делимого равна нулю, то результат операции - истинный нуль.

Прерывания. Спецификация: указан номер регистра плавающей запятой, отличный от 0,2,4,6, или в командах формата RX короткий (длинный) операнд не расположен на границе слова (двойного слова). Реализуется в микропрограмме ВЫБОР.

Исчезновение порядка: характеристика окончательного результата меньше нуля. При единичном значении маски исчезновения порядка мантисса нормализована и правильна, знак правилен, характеристика на $128_{10}$ больше, чем правильная. При нулевом разряде маски операция завершается тем, что на место результата записывается истинный нуль. Сигнал об исчезновении порядка не возникает при предварительной нормализации операндов и для промежуточного частного.

Переполнение порядка: значение характеристики конечного результата превышает I27. Операция доводится до конца. Мантисса результата правильная и нормализованная, знак правильный и характеристика на $128_{10}$ меньше, чем истинная. Переполнение порядка отсутствует, если значение характеристики превышает I27 только для промежуточного произведения, а характеристика конечного результата благодаря нормализации находится в пределах допустимых значений.

Деление с плавающей запятой: мантисса делителя равна нулю; делимое остается без изменения.

Адресация. Защита памяти по чтению (только для MD , ME, DD , DE).

Алгоритм. Во всех командах умножения и деления с плавающей запятой операнды нормализуются: мантисса операнда сдвигается влево на количество старших нулевых шестнадцатиричных разрядов, а из характеристики вычитается число, равное количеству старших нулевых шестнадцатиричных разрядов. Нормализованные таким образом операнды запоминаются. Если в процессе нормализации оказалось, что мантисса операнда равна нулю, то в зависимости от выполняемой операции получится или нулевой результат, или осуществляется переход на программу прерываний по некорректности деления.

Если мантисса первого операнда равна нулю, то результат всегда принимается равным истинному нулю.

При выполнении команд деления из характеристики нормализованного делимого вычитается характеристика нормализованного делителя, мантисса делимого делится на мантиссу делителя, окончательно формируется характеристика и знак результата, определяются случаи переполнения или исчезновения порядка, и, в зависимости от этого, формируется код выхода на микропрограмму ВЫБОР или на соответствующее прерывание. Перед делением мантисс образуются кратные нормализованной мантиссы делителя: Д, 2Д, 4Д, 8Д, где Д - мантисса делителя, чтобы при двоичном делении не было

необходимости в сдвигах на один разряд. Алгоритм деления мантисс основан на алгоритме двоичного деления без восстановления остатка. Цикл деления заключается в вычитании (сложении) из делимого кратных делителя 8Д, 4Д, 2Д и Д. После каждого вычитания (сложения) определяется бит частного: если остаток положительный, то бит частного равен единице, если отрицательный - то нулю. При отрицательном остатке выполняется сложение, а при положительном - вычитание кратных делителя из остатка. После получения четырех битов частного остаток сдвигается на четыре бита влево, и цикл деления повторяется.

Если старшая шестнадцатиричная цифра при таком делении получилась равной нулю, то она игнорируется, и старшей цифрой мантиссы будет вторая вычисленная цифра. Характеристика результата в этом случае равна разности характеристик делимого и делителя, увеличенной на 64.

Если же старшая шестнадцатиричная цифра частного при делении мантисс отлична от нуля, то вычисленная характеристика результата увеличивается на единицу.

Пример выполнения деления (все числа в шестнадцатиричной системе счисления).

| | |
|---|---|
| 3B000278 | Делимое |
| 3503456I | Делитель |
| 38278000 | Нормализованное делимое |
| 343556I0 | Нормализованный делитель |
| 38-34+40=44 | Предварительно вычисленная характеристика результата |
| 278000 | Мантисса нормализованного делимого |
| 3456I0 | Мантисса нормализованного делителя |
| 3456I0=Д | |
| 68AC20=2Д | |
| DI5840=4Д | |
| IA2B080=8Д | |

Получение цифр мантиссы частного

| | |
|---|---|
| 00278000 | |
| − 0IA2B080 (8Д) | |
| FE84CF80 | Старший бит результата равен 0 |
| + 00DI5840 (4Д) | |
| FF5627C0 | Следующий бит результата равен 0 |
| + 0068AC20 (2Д) | |
| FFBED3E0 | Следующий бит результата равен 0 |
| + 003456I0 (IД) | |
| FFF329F0 | Следующий бит результата равен 0 |
| | Сдвиг остатка на четыре бита влево |
| FF329F00 | |
| + 0IA2B080 (8Д) | |
| 00D54F80 | Следующий бит результата равен I |
| − 00DI5840 (4Д) | |
| 0003F740 | Следующий бит результата равен I |
| − 0068AC20 (2Д) | |
| FF9B4B20 | Следующий бит результата равен 0 |
| + 003456I0 (IД) | |
| FFCFAI30 | Следующий бит результата равен 0 |
| | Сдвиг остатка на четыре бита влево |

```
  FCFAI300
+ 0IA2B080 (8Д)
  FE9CC380          Следующий бит результата равен 0
+ 00DI5840 (4Д)
  FF6EIBC0          Следующий бит результата равен 0
+ 0068AC20 (2Д)
  FFD6C7E0          Следующий бит результата равен 0
+ 003456I0 (ID)
  000BIDF0          Следующий бит результата равен I
  00BIDF00
- 0IA2B080 (8Д)     Сдвиг остатка на четыре бита влево
  FF0F2E80          Следующий бит результата равен 0
+ 00DI5840 (4Д)
  FFE086C0          Следующий бит результата равен 0
+ 0068AC20(2Д)
  004932E0          Следующий бит результата равен I
- 003456I0 (ID)
  00I4DCD0          Следующий бит результата равен I
                    Сдвиг остатка на четыре бита влево

  0I4DCD00
- 0IA2B080 (8Д)
  FFABIC80          Следующий бит результата равен 0
+ 00DI5840 (4Д)
  007C74C0          Следующий бит результата равен I
- 0068AC20 (2Д)
  00I3C8A0          Следующий бит результата равен I
- 003456I0 (ID)
  FFDF7290          Следующий бит результата равен 0
                    Сдвиг остатка на четыре бита влево

  FDF72900
+ 0IA2B080 (8Д)
  FF99D980          Следующий бит результата равен 0
+ 00DI5840 (4Д)
  006B3IC0          Следующий бит результата равен I
- 0068AC20 (2Д)
  000285A0          Следующий бит результата равен I
- 003456I0(ID)
  FFCE2F90          Следующий бит результата равен 0
                    Сдвиг остатка на четыре бита влево

  FCE2F900
+ 0IA2B080 (8Д)
  FE85A980          Следующий бит результата равен 0
+ 00DI5840 (4Д)
  FF570IC0          Следующий бит результата равен 0
+ 0068AC20 (2Д)
  FFBFADE0          Следующий бит результата равен 0
+ 003456I0 (ID)
  FFF403F0          Следующий бит результата равен 0
```

Так как старшая шестнадцатиричная цифра мантиссы результата равна нулю, то она игнорируется. Результат деления со знаком и характеристикой:

44 CI3660

Умножение с плавающей запятой заключается в перемножении мантисс нормализованных операндов и сложении их порядков.

Алгоритм умножения мантисс операндов представляет последовательный анализ цифр множителя, и, в зависимости от значения анализируемой цифры, происходит сложение (или вычитание) кратных множимого с промежуточным произведением в соответствии с табл. 26.

Таблица 26

| Цифра множителя | Умножение на предыдущую цифру выполнялось при помощи сложения | Умножение на предыдущую цифру выполнялось при помощи вычитания |
|---|---|---|
| 0 | - | +M |
| I | +M | +2M |
| 2 | +2M | +2M+M |
| 3 | +2M+M | +2M+2M |
| 4 | +2M+2M | +5M |
| 5 | +5M | +5M+M |
| 6 | +5M+M | +5M+2M |
| 7 | +5M+2M | +5M+2M+M |
| 8 | +5M+2M+M | -5M-2M |
| 9 | -5M-2M | -5M-M |
| A | -5M-M | -5M |
| B | -5M | -2M-2M |
| C | -2M-2M | -2M-M |
| D | -2M-M | -2M |
| E | -2M | -M |
| F | -M | - |

Поле промежуточного произведения (сумматор) находится на месте первого операнда в регистре плавающей запятой; перед началом умножения мантисс в него записываются нули. В рабочей области формируются кратные мантиссы множимого: M, 2M, 5M.

После умножения на одну цифру множителя промежуточное произведение сдвигается на четыре разряда вправо.

Если умножение на старшую цифру множителя велось с помощью вычитания, то после сдвига промежуточного произведения к нему прибавляется еще множимое (M).

Характеристика результата равна сумме характеристик порядков, уменьшенной на число 64. Если после того, как умножение на все цифры множителя закончено, окажется, что старшая цифра мантиссы результата равна нулю, то результат нормализуется: мантисса сдвигается на одну шестнадцатиричную цифру влево, характеристика уменьшается на единицу.

При вычислении характеристики происходит анализ ее на переполнение или исчезновение порядка.

Пример умножения с плавающей запятой.

| | |
|---|---|
| 65000I2I | Множимое |
| 4200IFI2 | Множитель |
| 62I2I000 | Нормализованное множимое |
| 40IFI200 | Нормализованный множитель |

Кратные мантиссы множимого, дополненной нулями до I4 цифр.
На место характеристики записаны нули.

00I2I00000000000 = **M**,
00242000000000000 = **2M**,
005A500000000000 = **5M**.

Две младшие цифры мантиссы нормализованного множителя равны нулю, поэтому при умножении на них ничего в сумматор не прибавляется, производится только сдвиг вправо. В примере показывается перемножение мантисс, начиная с умножения на младшую значащую цифру множителя.

| | |
|---|---|
| 0000000000000000<br>+00242000000000000<br>00242000000000000 | Прибавление 2M |
| | Сдвиг вправо |
| 0002420000000000<br>+00I2I000000000000<br>00I4520000000000 | Прибавление M |
| | Сдвиг вправо |
| 000I452000000000<br>-00I2I000000000000<br>FFEF352000000000 | Вычитание M |
| | Сдвиг вправо |
| FFEF35200000000<br>+00242000000000000<br>0023I35200000000 | Прибавление 2M |
| | Сдвиг вправо |
| 00023I3520000000 | Мантисса промежуточного произведения |

Вычисление характеристики промежуточного произведения: 62+40–40 = 62.

После нормализации промежуточного произведения получается результат: 6I23I35200000000.

Диаграмма алгоритма. I06CI. Мантисса второго операнда прогоняется через арифметический блок для определения количества начальных нулевых шестнадцатиричных цифр. Значение счетчика адреса команды запоминается в локальной памяти.

I06C3, I06C4. Нормализация операнда происходит по одной из 4 ветвей в зависимости от количества старших шестнадцатиричных нулевых разрядов в мантиссе.

Нормализация первого операнда происходит так же, как и второго.

Нормализованные мантиссы и порядок запоминаются в рабочей области локальной памяти.

I07B2. Мантисса нормализованного делимого пересылается в регистры Л, Р, И. Цикл пересылки осуществляется при помощи адресных регистров Д и Т. Выход из цикла – при помощи установки и анализа триггера БС2.

I07C2. Формирование кратных мантиссы делителя (8Д,4Д,2Д) осуществляется при помощи цикла, который выполняет сдвиг на I разряд влево. При организации цикла регистр Д используется как адресный регистр байтов числа, которое сдвигается, а регистр Т – как адресный регистр байтов результата сдвига. Устанавливается признак формирования старшего байта мантиссы результата (ОБС5).

I07B3, I07Д3. Поскольку мантисса нормализованного делимого, а затем и остаток для длинных и коротких операндов находятся в различных местах, то деление проходит по различным веткам микропрограммы для каждого случая.

Цикл деления заключается в вычитании (сложении) из делимого кратных делителя 8Д,4Д,2Д и Д в следующей последовательности: вычитание (сложение) 8Д из делимого, анализ наличия переноса, запись старшего бита в цифре результата. Это повторяется еще три раза, но используются кратные делителя 4Д, 2Д и Д. Таким способом формируется цифра результата, состоящая из четырех битов. Биты частного накапливаются в регистре Е; после каждого вычитания (сложения) содержимое регистра Е сдвигается на один разряд влево, а в младший разряд заносится 0 или I в зависимости от выполняемого действия и наличия переноса из старшего разряда. Выбор цифры результата иллюстрируется табл. 27.

Таблица 27

| Действие | БС2 | Перенос из старшего разряда | Цифра результата | Заносится косвенная функция |
|---|---|---|---|---|
| Сложение | ОБС2 | есть | I | - |
| Сложение | ОБС2 | нет | 0 | + |
| Вычитание | IБС2 | есть | 0 | + |
| Вычитание | IБС2 | нет | I | - |

Циклы деления для коротких и длинных операндов организованы одинаковым способом. В них регистр Д используется как адресный регистр кратных делителя, регистр Т для длинных операндов используется как адресный регистр остатка.

Триггер БС2 указывает, какое действие выполняется:

IБС2 - вычитание,

ОБС2 - сложение.

Выход из цикла организован при помощи сравнения адресов.

I07C4. Проверяется указатель формирования старшего байта мантиссы частного БС5.

I07B4. Для того, чтобы проверить весь ли байт результата готов (2 шестнадцатиричные цифры вычислены или нет), вычитается единица из содержимого регистра Г и анализируется полученный результат. В регистр Г перед началом вычисления очередного байта результата заносится константа, равная 2 (блоки I07B6, I07B7, I07C2).

I07C4, I07Д4. Если вычисляется старший байт мантиссы частного, то признак формирования старшего байта - ОБС5, но так как может оказаться, что старшая цифра результата равна 0 и придется вычислять еще одну цифру для получения старшего байта, то вводится еще признак (БС3) для указания, окончательно ли сформирован старший байт мантиссы и характеристика результата.

I07C5. При вычислении характеристики результата к разности характеристик прибавляется 64.

I07E6. При вычислении характеристики результата к разности характеристик прибавляется 65.

I07E9. В регистр Г заносится константа I. Устанавливается признак ОБС3, так как характеристика результата уже вычислена.

I07E4. Устанавливаются признаки того, что характеристика и старший байт мантиссы результата уже вычислены и записаны в память: (IБС5, IБС3); в регистр Г заносится константа 2.

I08B6. Функциональный переход на различные прерывания или выборку по константам, заготовленным заранее в регистре Ф:

константа 3 - переход к ПИСП,

константа 4 - переход на ВЫБОР,

константа 5 - обнуление результата и переход на ВЫБОР,

константа 6 - переход на ПДПЗ,

константа 7 - переход на ПППП.

Предварительно в регистрах МФЕ восстанавливается СчАК.

I09AI. Кратные множимого М, 2М, 5М формируются в следующей последовательности, мантисса множимого сдвигается на один разряд влево для получения 2М, затем 2М сдвигается на один разряд влево и формируется 4М, после этого 4М и М складываются для получения 5М. 5М записывается в рабочую область на место 4М. Триггеры БС3 и БС4 используются для организации однократного или двукратного прохождения цикла сдвига в зависимости от длины операнда, регистры Т и Д используются как адресные.

I09B0. В регистр плавающей запятой $R_I$, на место множимого, записываются нули для того, чтобы в нем организовать сумматор. Регистр У используется как адресный, триггер БС2 служит для организации двукратного прохождения цикла.

I09C0. Считываются два младших байта мантиссы множителя и записываются в регистры Ф и Е.

I09ДI. По цифре множителя, расположенной в младшей тетраде регистра Е, и по индикатору предыдущего действия, триггеру БС2, осуществляется переход на 32 направления.

Если умножение на предыдущую цифру множителя велось с помощью сложения, то триггер БС2 установлен в I, если при помощи вычитания, то - в 0.

I09A2, I09B2, I09C2, I09A3, I09C3, I09Д3, I09A4, I09B4, I09C4, I09Д4, I09A5, I09B5, I09C5, I09Д5. Настройка осуществляется занесением косвенной функции "+" или "-", а также занесением адреса соответствующего кратного множимого в адресный регистр Т.

I09C7. Регистры Т и У используются как адресные регистры.

I09C8. Анализируется триггер БС5, IБС5 означает, что умножение закончено.

I09Д8. Осуществляется второй переход по цифре множителя и указателю предыдущего действия БС2 на 32 направления, где (лист II0, лист III) определяется, закончено ли умножение на данную цифру множителя или еще нужно прибавить (вычесть) какое-либо кратное множимого в сумматор, и ведется подготовка к этому сложению (вычитанию).

II0E4, IIIC7. Когда выполнение умножения на данную цифру закончено, устанавливается признак предыдущего действия IБС2, если выполнялось сложение, и 0БС2, если выполнялось вычитание.

II2Д1. В регистре Л организован счетчик цифр множителя. В него заносится константа 4 или 2, равная количеству цифр множителя, помещенных в регистры Ф и Е. После умножения на очередную цифру множителя из счетчика вычитается единица, а содержимое регистров Ф и Е сдвигается на четыре разряда вправо, и следующая цифра множителя оказывается в младшей тетраде регистра Е.

Когда после вычитания единицы в регистре Л остается нуль, считываются следующие байты множителя и помещаются в регистры Ф и Е, вновь заносится константа в счетчик (регистр Л), и цикл умножения повторяется.

II2AI. Установка IБС5.

II2A3. Анализируется триггер БС5.

II2B3. Анализируется триггер БС2.

II2Д8. Функциональный переход по заранее подготовленной константе на микропрограмму ВЫБОР или соответствующее прерывание.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение в микропрограмме | | |
|---|---|---|---|---|
| | | DINOR | DFP | MFP |
| М | Счетчик адреса команды | Не используется | Не используется | Не используется |
| Ф | | Запоминается знак и характеристика операнда | Хранятся константы для функционального перехода на выходы из микропрограммы | Хранятся цифры множителя |
| Е | | Запоминается характеристика операнда без знака | Накапливаются биты частного | Хранятся цифры множителя |
| Г | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ или $R_2$ I000 | Хранится текущий адрес байтов нормализуемого операнда | Используется как счетчик при вычислении цифр частного | Не используется |
| Р | | | При делении коротких чисел здесь располагается байт остатка | Хранятся байты чисел при вычислениях |
| И | | | При делении коротких чисел здесь располагается байт остатка | Хранятся байты чисел при вычислениях |

| Регистры и триггеры | Исходное состояние | Назначение в микропрограмме | | |
|---|---|---|---|---|
| | | DINOR | DFP | MFP |
| П | | Заносятся три младших разряда текущего адреса байтов нормализуемого операнда. Анализируется для выхода из цикла | П=I - признак неправильной характеристики результата | Не используется |
| Т | $\langle R_2\ 1000\rangle$ | Хранится текущий адрес байтов нормализованного числа | Адресный регистр результата при формировании кратных делителя, при пересылке длинного делимого в ЛП при делении чисел. При делении коротких чисел располагается старший байт остатка | Хранятся константы для функционального перехода на выходы из микропрограммы |
| У | $R_I$ 1000 | Во время нормализации второго операнда хранится адрес первого операнда | | Адресный регистр байтов промежуточного произведения |
| Л | - | Запоминается количество сдвигов при нормализации | При делении коротких операндов располагается байт остатка | Счетчик цифр множителя |
| Д | Код операции | Хранится адрес начала операнда; хранятся байты мантиссы делителя при сдвиге ее на одну цифру влево | Адресный регистр | Адресный регистр байтов множителя при умножении. Адресный регистр кратных множимого при их формировании |
| БС2 | 1БС2 при формате RR<br>0БС2 при формате RX | Индикатор формата команды:<br>1БС2 - RR<br>0БС2 - RX | Для организации двойного прохождения цикла формирования кратных делителя в случае длинных операндов. В цикле деления:<br>1БС2 - вычитание<br>0БС2 - сложение | Индикатор предыдущего действия:<br>1БС2 - сложение<br>0БС2 - вычитание |
| БС3 | 0БС3 | Используется при выполнении сдвига мантиссы на 3 и 4 шестнадцатиричные цифры | 0БС3 - признак того, что характеристика результата вычислена. Индикатор четности адреса полученного байта результата:<br>0БС3 - нечетный,<br>1БС3 - четный | Для организации двукратного прохождения циклов |

| Регистры и триггеры | Исходное состояние | Назначение в микропрограмме | | |
|---|---|---|---|---|
| | | DINOR | DFP | MFP |
| БС4 | 1БС4 - длинные операнды<br>0БС4 - короткие операнды | Индикатор длины операндов:<br>1БС4 - длинные<br>0БС4 - короткие | Индикатор длины операндов:<br>1БС4 - длинные<br>0БС4 - короткие | Индикатор длины операндов:<br>1БС4 - длинные<br>0БС4 - короткие |
| БС5 | 0БС5 - в формате RR<br>1БС5 - в формате RX | Индикатор нормализуемого операнда:<br>1БС5 - первый операнд,<br>0БС5 - второй операнд | Индикатор формирования старшего байта мантиссы частного:<br>0БС5 - формируется старший байт | Индикатор умножения на последнюю цифру множителя |
| ЛП<br>04-05<br>14-16<br>24-25<br>34-35 | - | Мантисса нормализованного множимого | Мантисса нормализованного делимого (длинного)<br>Остаток при делении | Мантисса множимого (нормализованного):<br>М |
| 06-07<br>16-17<br>26-27<br>36-37 | - | Мантисса нормализованного делимого | Увосьмеренная мантисса нормализованного делителя:<br>8Д | Не используются |
| 44-45<br>54-55<br>64-65<br>74-75 | - | Не используются | Не используются | Удвоенная мантисса нормализованного множимого:<br>2М |
| 46-47<br>56-57<br>66-67<br>76-77 | - | Не используются | Учетверенная мантисса нормализованного делителя:<br>4Д | Не используются |
| 84-85<br>94-95<br>A4-A5<br>B4-B5 | - | Не используются | Не используются | Упятеренная мантисса нормализованного множимого:<br>5М |
| 86-87<br>96-97<br>A6-A7<br>B6-B7 | - | Не используются | Удвоенная мантисса нормализованного делителя:<br>2Д | Не используются |
| C4-C5<br>D4-D5<br>E4-E5<br>F4-F5 | - | Мантисса нормализованного множителя | Не используются | Мантисса нормализованного множителя |
| C6-C7<br>D6-D7<br>E6-E7<br>F6-F7 | - | Мантисса нормализованного делителя | Мантисса нормализованного делителя:<br>М | Не используются |

Продолжение

| Регистры и триггеры | Исходное состояние | Назначение в микропрограмме | | |
|---|---|---|---|---|
| | | DINOR | DFP | MFP |
| IE | - | Не используются | Характеристика нормализованного делимого | Характеристика нормализованного множимого |
| IF | - | Не используются | Характеристика нормализованного делителя | Характеристика нормализованного множителя |
| 3E | - | Не используются | Знак результата | Знак результата |

20.3. Микропрограмма HFLP

Команды

| Название | Мнемоника | Формат | | | Вход | Время |
|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | |
| ПОПОЛАМ (КОРОТКОЕ) | HER | RR | 34 | $R_I$ $R_2$ | I09 | 36 |
| ПОПОЛАМ (ДЛИННОЕ) | HDR | RR | 24 | $R_I$ $R_2$ | I09 | 66 |

Операнды. Короткое (HER) или длинное (HDR) число с плавающей запятой, расположенное в регистр плавающей запятой с номером $R_2$.

Результат. Короткое (HER) или длинное (HDR) число с плавающей запятой, равное исходному числу, уменьшенному вдвое и нормализованному. Промежуточный результат имеет дополнительную цифру мантиссы, которая учитывается при нормализации. Результат помещается в регистр плавающей запятой с номером $R_I$.

Если мантисса исходного числа равна нулю, то знак, порядок и мантисса результата заполняются нулями.

Прерывания. Спецификация: указан номер регистра плавающей запятой, отличный от 0,2,4,6 (реализовано в микропрограмме ВЫБОР).

Исчезновение порядка: нормализация вызывает сокращение характеристики до величины ниже нуля. Если маска исчезновения порядка равна нулю, то знак, мантисса и характеристика становятся нулевыми.

При единичном значении маски возникает прерывание программы. Результат нормализуется; знак и мантисса его остаются правильными, а характеристика устанавливается на $128_{10}$ больше действительного значения.

Алгоритм. Мантисса сдвигается вправо на один двоичный разряд, начиная со старших байтов; знак не изменяется. Содержимое младшего бита помещается в старший разряд дополнительной цифры.

Управление передается микропрограмме ASCFP, которая производит нормализацию промежуточного результата.

Диаграмма алгоритма. См. диаграмму.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г | | Рабочий регистр |
| Р | | Рабочий регистр |
| И | $R_2$1000 | Хранит текущий адрес байтов операнда |
| П | | Хранит знак результата |

Продолжение

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Т | $\langle R_2\ 1000 \rangle$ | $\langle R_2 1000 \rangle$ |
| У | $R_2\ 1000$ | Хранит текущий адрес байтов промежуточного результата |
| М | Счетчик адреса команды | Не используется |
| Ф | | Хранит дополнительную цифру |
| Е | | Рабочий регистр |
| БС4 | 1БС4 - длинный операнд<br>0БС4 - короткий операнд | Индикатор длины операндов |

## 21. ПЛАВАЮЩАЯ ЗАПЯТАЯ. ОПЕРАЦИИ ЗАГРУЗКИ

### 21.1. Микропрограмма LFP

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | байт 3 | байт 4 | | |
| ЗАГРУЗКА (КОРОТКАЯ) | LER | RR | 38 | $R_1\ R_2$ | | | 111 | 10 |
| ЗАГРУЗКА (КОРОТКАЯ) | LE | RX | 78 | $R_1\ X_2$ | $B_2$ | $D_2$ | 151 | 8 |
| ЗАГРУЗКА (ДЛИННАЯ) | LDR | RR | 28 | $R_1\ R_2$ | | | 111 | 22 |
| ЗАГРУЗКА (ДЛИННАЯ) | LD | RX | 68 | $R_1\ X_2$ | $B_2$ | $D_2$ | 151 | 21 |
| ЗАГРУЗКА И ПРОВЕРКА (КОРОТКАЯ) | LTER | RR | 32 | $R_1\ R_2$ | | | 105 | 10 |
| ЗАГРУЗКА И ПРОВЕРКА (ДЛИННАЯ) | LTDR | RR | 22 | $R_1\ R_2$ | | | 105 | 22 |
| ЗАГРУЗКА ДОПОЛНЕНИЯ (КОРОТКАЯ) | LCER | RR | 33 | $R_1\ R_2$ | | | 107 | 10 |
| ЗАГРУЗКА ДОПОЛНЕНИЯ (ДЛИННАЯ) | LCDR | RR | 23 | $R_1\ R_2$ | | | 107 | 22 |
| ЗАГРУЗКА ПОЛОЖИТЕЛЬНАЯ (КОРОТКАЯ) | LPER | RR | 30 | $R_1\ R_2$ | | | 101 | 10 |
| ЗАГРУЗКА ПОЛОЖИТЕЛЬНАЯ (ДЛИННАЯ) | LPDR | RR | 20 | $R_1\ R_2$ | | | 101 | 22 |
| ЗАГРУЗКА ОТРИЦАТЕЛЬНАЯ (КОРОТКАЯ) | LNER | RR | 31 | $R_1\ R_2$ | | | 103 | 10 |
| ЗАГРУЗКА ОТРИЦАТЕЛЬНАЯ (ДЛИННАЯ) | LNDR | RR | 21 | $R_1\ R_2$ | | | 103 | 22 |

Операнды. Короткое (команды LER, LE, LTER, LCER, LPER, LNER) или длинное (команды LDR, LD, LTDR, LCDR, LPDR, LNDR) число с плавающей запятой, расположенное в командах LE и LD в основной памяти по адресу $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ или в регистре плавающей запятой с номером $R_2$ во всех остальных случаях.

Результат. Исходное число, переписанное в регистр плавающей запятой с номером $R_I$:

а) без изменения в командах LPER, LPDR;

б) с отрицательным знаком (даже если мантисса результата равна нулю) в командах LNER, LNDR;

в) с положительным знаком в командах LPER, LPDR;

г) с противоположным знаком в командах LCER, LCDR.

В командах LE, LD , LER, LDR признак результата не изменяется. В остальных командах устанавливается признак результата:

0 - мантисса результата равна нулю,

1 - результат меньше нуля,

2 - результат больше нуля.

Прерывания. Спецификация: в команде указан номер регистра плавающей запятой, отличный от 0,2,4,6, или (в командах LE и LD ) адрес основной памяти $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ не соответствует границе слова или двойного слова в зависимости от длины операнда (реализовано в микропрограмме ВЫБОР).

Адресация (только для LE и LD ).

Защита памяти по чтению.

Алгоритм. Формирование содержимого регистра $R_I$ начинается со старших байтов.

Диаграмма алгоритма. 1I5C5. Выход из цикла осуществляется посредством анализа текущего адреса байтов результата.

1I5A6. Установка признака результата производится путем прогонки старшего байта результата, находящегося в регистре Л, через арифметическо-логический блок, после чего устанавливается признак результата по микрооперации КУ1.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ | Хранится текущий адрес байтов операнда |
| У | $R_I$1000 | Хранится текущий адрес байтов результата |
| Л | $R_I X_2$ | Запоминается старший байт результата |
| БС2 | 1БС2 - признак формата RR<br>0БС2 - признак формата RX | 1БС2 - признак формата RR<br>0БС2 - признак формата RX |
| БС3 | 0 | Индикатор формирования признака результата:<br>1БС3 - признак результата устанавливается |
| БС4 | 1БС4 - признак длинного операнда<br>0БС4 - признак короткого операнда | 1БС4 - признак длинного операнда<br>0БС4 - признак короткого операнда |

## 22. ПЛАВАЮЩАЯ ЗАПЯТАЯ. ОПЕРАЦИИ ЗАПИСИ

### 22.1. Микропрограмма SFP

Команды

| Название | Мнем. | Формат | | | | | Вход | Время |
|---|---|---|---|---|---|---|---|---|
| | | | байт 1 | байт 2 | байт 3 | байт 4 | | |
| ЗАПИСЬ В ПАМЯТЬ (КОРОТКАЯ) | STE | RX | 70 | $R_1$ $X_2$ | $B_2$ | $D_2$ | I4I | I0 |
| ЗАПИСЬ В ПАМЯТЬ (ДЛИННАЯ) | STD | RX | 60 | $R_1$ $X_2$ | $B_2$ | $D_2$ | I4I | I9 |

Операнды. Короткое (команда STE) или длинное (команда STD ) число с плавающей запятой, расположенное в регистре плавающей запятой с номером $R_1$.

Результат. Операнд переписывается в основную память по адресу $\langle X_2 \rangle + \langle B_2 \rangle + D_2$.

Прерывания. Спецификация: указан номер регистра плавающей запятой, отличный от 0, 2, 4, 6 или адрес результата не совпадает с границей слова ( STE) или двойного слова ( STD ) (реализовано в микропрограмме ВЫБОР).

Адресация. Защита памяти по записи.

Алгоритм. Пересылка операнда происходит, начиная со старших байтов.

Диаграмма алгоритма. См. диаграмму.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle X_2 \rangle + \langle B_2 \rangle + D_2$ | Хранится текущий адрес байтов операнда |
| У | $R_1$I000 | Хранится текущий адрес байтов операнда |
| БС3 | ОБС3 | ОБС3 - признак того, что переписывается первая или третья пара байтов<br>IБС3 - признак того, что переписывается вторая или четвертая пара байтов |
| БС4 | IБС4 - длинные операнды<br>ОБС4 - короткие операнды | ОБС4 - пересылка закончена,<br>IБС4 - пересылка не закончена |

## 23. ПРИВИЛЕГИРОВАННЫЕ КОМАНДЫ. ЗАЩИТА ПАМЯТИ. ОПЕРАЦИИ НАД КЛЮЧАМИ

### 23.1. Микропрограмма КЛЮЧП

Команды

| Название | Мнем. | Формат | | | Вход | Время |
|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | | |
| УСТАНОВИТЬ КЛЮЧ ПАМЯТИ | SSK | RR | 08 | $R_1$ $R_2$ | I50 | I9 |
| ПРОЧИТАТЬ КЛЮЧ ПАМЯТИ | ISK | RR | 09 | $R_1$ $R_2$ | I52 | I3 |

Операнды. Старшие I3 разрядов адреса блока основной памяти объемом в 2048 байтов, расположенные в разрядах 8-20 общего регистра $R_2$. В связи с тем, что максимальный объем памяти в модели равен 256К, разряды 8-I3 игнорируются. Новое значение ключа памяти адресуемого блока, расположенное в разрядах 24-28 общего регистра $R_I$ (команда SSK) или его текущее значение, расположенное в запоминающем устройстве ключей памяти (команда ISK).

Результат. В разряды 24-28 общего регистра $R_I$ записывается его текущее значение, а в разряды 29-3I - нуль (команда ISK) или же ключу адресуемого блока памяти и его дублю в мультиплексной памяти присваивается новое значение.

Примечание. В мультиплексной памяти отведена область, где дублируется содержимое запоминающего устройства ключей памяти. На рис. I9 показано формирование адреса ячейки, содержащей дубль ключа данного блока памяти.

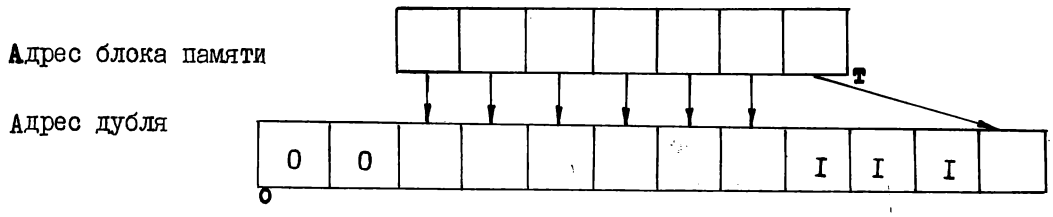



Рис. I9. Формирование адреса дубля ключа памяти

Прерывания. Привилегированная операция: I5-й разряд ССП равен I, т.е. процессор находится в состоянии "задача".

Спецификация: разряды 28-3I общего регистра $R_2$ не равны нулю.

Диаграмма алгоритма. В блоке II9AI наличие блока защиты определяется по единичному состоянию триггера блока защиты (ТБЗ). После установки ключа памяти (блок II9B4) восстанавливается ключ защиты в ССП.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | | Хранит адрес блока памяти |
| Т<br>Л | | Рабочие регистры |
| Д | КО | Адресный регистр |
| БС2 | 0 | Определяет выполняемую команду<br>ISK - I,<br>SSK - 0 |
| БЗ (0-4) | Информационный регистр ЗУКП | Заносится ( SSK) или читается (ISK) ключ памяти |
| БЗ (4-8) | Ключ защиты ССП | Ключ защиты ССП |

## 24. КОМАНДЫ ПРЯМОГО УПРАВЛЕНИЯ

### 24.I. Микропрограмма ПРУПР

Команды

| Название | Мнем. | Формат | | | | | Вход |
|---|---|---|---|---|---|---|---|
| | | | байт I | байт 2 | байт 3 | байт 4 | |
| ПРЯМАЯ ЗАПИСЬ | WRD | SI | 84 | $I_2$ | $B_I$ | $D_I$ | I68 |
| ПРЯМОЕ ЧТЕНИЕ | RDD | SI | 85 | $I_I$ | $B_2$ | $D_I$ | I6A |

Время: а) ПРЯМАЯ ЗАПИСЬ – 6 мксек;

б) ПРЯМОЕ ЧТЕНИЕ – 7 мксек ($X \leqslant 6$) – X+I ($X > 6$) мксек.

Время, через которое станет разрешающим уровень на линии запрещения чтения с момента начала выполнения команды.

Операнды. Разряды 8–I5 команды, находящиеся в регистре Л. Байт данных, расположенный в основной памяти по адресу $\langle B_I \rangle$ + $Д_I$ (команда WRD) или находящийся на восьми входных потенциальных линиях (команда RDD).

Байт данных, находящийся на потенциальных линиях, разрядом четности не сопровождается.

Результат. Содержимое разрядов 8–I5 из команды посылается на восемь выходных импульсных линий в виде синхросигналов длительностью от 0,5 до I,0 мксек. Одновременно с этими синхросигналами выдается специальный синхросигнал длительностью 0,5–I,0 мксек по девятой линии (линии чтения извне в команде RDD или линии записи вовне в команде WRD, которые представляют собой разные линии). Восемь выходных импульсных сигнальных линий не имеют разряда четности.

В команде WRD восемь двоичных разрядов байта, выбранного из памяти, поступают на восемь выходных потенциальных сигнальных линий. Эти сигналы остаются без изменения до следующей команды. Передние фронты потенциальных сигналов данных и синхросигналов совпадают.

В команде RDD байт данных принимается с восьми входных потенциальных линий, если на линии запрещения чтения разрешающий уровень.

Отсутствие сигнала запрещения чтения проверяется после завершения сигнала чтения извне. Сигнал запрещения чтения должен отсутствовать, по крайней мере, еще 0,5 мксек, чтобы чтение произошло. Когда данные заносятся в память, вырабатывается разряд четности.

Если сигнал запрещения чтения не снят, выполнение команды не заканчивается.

Прерывания. Привилегированная операция: процессор находится в состоянии "проблема".

Адресация. Защита памяти по чтению (команда WRD) или по записи (команда RDD).

Диаграмма алгоритма. Так как при прямом управлении передается один байт информации, а из основной памяти читаются два байта, то перед передачей байта на выходные потенциальные линии (WRD) или перед записью байта в память (RDD) определяется четность адреса (блоки I2IB7 и I2IД4).

При прямом чтении принимается один байт информации без контрольного разряда, поэтому, чтобы не было сбоя по контролю перед подачей байта на вход БА, триггер ТТС устанавливается в "I" (блок I2IB5). Контрольный разряд формируется в БА.

Регистры и триггеры

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| Г<br>Р<br>И | $\langle B_I \rangle$ + ДI | Хранится адрес байта в основной памяти |
| Л | I2 | Байт в команде |
| Д | Код операции | Служит буферным регистром в команде RDD |

Продолжение

| Регистры и триггеры | Исходное состояние | Назначение |
|---|---|---|
| БС2 | 0 – в команде WRD<br>I – в команде RDD | Указатель команды прямого чтения (IБС2) и прямой записи (OБС2) |
| ТПУ | 0 | Установкой ГТПУ осуществляется выдача сигнала ПРЯМАЯ ЗАПИСЬ (если OБС2) или ПРЯМОЕ ЧТЕНИЕ (если IБС2) |
| ТПЧ | | Указатель завершения сигнала чтения извне и отсутствия сигнала запрещения чтения (ГТПЧ) |
| РПЧ | | Служит для приема байта с входных потенциальных линий |
| РПЗ | | Служит для выдачи байта на выходные потенциальные линии |
| ТГС | 0 | IТГС блокирует контроль по нечету |

## ПРИЛОЖЕНИЕ. ВРЕМЯ ВЫПОЛНЕНИЯ КОМАНД

### Обозначения

B – общее количество обработанных байтов первого операнда;
H – количество значащих шестнадцатиричных цифр (исключая впередистоящие нули) в двоичном операнде;
N – общее количество байтов первого операнда для тех инструкций, в которых оба операнда имеют одинаковую длину;
$N_1$ – общее количество байтов первого операнда;
$N_2$ – общее количество байтов второго операнда;
D – количество значащих десятичных цифр;
$N_{min}$ – наименьшее из $N_1$ и $N_2$;
$N_{abs}$ – $/N_1 - N_2/$;
S – число знаков в редактируемом поле;
Z – общее число символов начала значимости и выбора цифры в шаблоне редактирования;
R – общее число разделителей полей в шаблоне редактирования;
ОТМ – число раз занесения адреса в регистр I;
A – равно 0, если регистр I не изменяется и равно I, если в регистр I заносится адрес;
U – количество регистров, загружаемых или записываемых в память.

Примечание. В табл. I, 2, 3 и 4 времена выполнения приводятся с учетом базирования адресов и без учета индексирования (для команд формата РХ). Время базирования одного адреса – 4 мксек. Время индексирования адреса – 4 мксек.

Таблица I

Стандартная система команд

| Команда | | | Время выполнения, мксек |
|---|---|---|---|
| Название | Мнем. | Формат | |
| ВЫПОЛНИТЬ | EX | RX | 33 + время выполнения подчиненной команды |
| ВЫЧИТАНИЕ | SR | RR | 20 |

Продолжение

| Команда | | | Время выполнения, мксек |
|---|---|---|---|
| Название | Мнем. | Формат | |
| ВЫЧИТАНИЕ | S | RX | 33 |
| ВЫЧИТАНИЕ КОДОВ | SLR | RR | 22 |
| ВЫЧИТАНИЕ КОДОВ | SL | RX | 35 |
| ВЫЧИТАНИЕ ПОЛУСЛОВА | SH | RX | 29 |
| ДЕЛЕНИЕ | DR | RR | 390 |
| ДЕЛЕНИЕ | D | RX | 398 |
| ЗАГРУЗКА | LR | RR | I8 |
| ЗАГРУЗКА | L | RX | 27 |
| ЗАГРУЗКА АДРЕСА | LA | RX | 24 |
| ЗАГРУЗКА ГРУППОВАЯ | LM | RS | I8 - 8U |
| ЗАГРУЗКА ДОПОЛНЕНИЯ | LCR | RR | 2I |
| ЗАГРУЗКА И ПРОВЕРКА | LTR | RR | 2I |
| ЗАГРУЗКА ОТРИЦАТЕЛЬНАЯ | LNR | RR | 23 |
| ЗАГРУЗКА ПОЛОЖИТЕЛЬНАЯ | LPR | RR | 23 |
| ЗАГРУЗКА ПОЛУСЛОВА | LH | RX | 28 |
| ЗАПИСЬ В ПАМЯТЬ | ST | RX | 27 |
| ЗАПИСЬ В ПАМЯТЬ ГРУППОВАЯ | STM | RS | I8 + IOU |
| ЗАПИСЬ В ПАМЯТЬ ПОЛУСЛОВА | STH | RX | 22 |
| ЗАПИСЬ В ПАМЯТЬ СИМВОЛА | STC | RX | 23 |
| И | NR | RR | 20 |
| И | N | RX | 30 |
| И | NI | SI | I9 |
| И | NC | SS | 36+3N , если оба адреса четные или нечетные<br>28+5N - в остальных случаях |
| ИЛИ | OR | RR | 20 |
| ИЛИ | O | RX | 30 |
| ИЛИ | OI | SI | I9 |
| ИЛИ | OC | SS | 36+3N , если оба адреса четные или нечетные<br>28+N - в остальных случаях |
| ИСКЛЮЧАЮЩЕЕ ИЛИ | XR | RR | 20 |
| ИСКЛЮЧАЮЩЕЕ ИЛИ | X | RX | 30 |
| ИСКЛЮЧАЮЩЕЕ ИЛИ | XI | SI | I9 |
| ИСКЛЮЧАЮЩЕЕ ИЛИ | XC | SS | 36+3N , если оба адреса четные или нечетные<br>28+5N - в остальных случаях |
| ОБРАЩЕНИЕ К СУПЕРВИЗОРУ | SVC | RR | 59 |
| ПЕРЕКОДИРОВАТЬ | TR | SS | 34+ION |
| ПЕРЕКОДИРОВАТЬ И ПРОВЕРИТЬ | TRT | SS | 57+ION , если все байт-функции нули<br>44+9B - есть хотя бы одна ненулевая байт-функция |
| ПЕРЕСЫЛКА | MVI | SI | I8 |
| ПЕРЕСЫЛКА | MVC | SS | 37+3N , если оба адреса четные или нечетные |

| Команда | | | Время выполнения, мксек |
|---|---|---|---|
| Название | Мнем. | Формат | |
| | | | 30+5N – в остальных случаях |
| ПЕРЕСЫЛКА ЗОН | MVZ | SS | 43,5+3,5N , если оба адреса четные или нечетные |
| | | | 39+5N , в остальных случаях |
| ПЕРЕСЫЛКА СО СДВИГОМ | MVO | SS | $27+9N_I$, если $N_I \leqslant N_2$ |
| | | | $32+6N_2+3N_I$, если $N_I > N_2$ |
| ПЕРЕСЫЛКА ЦИФР | MVN | SS | 43,5+3,5N , если оба адреса четные или нечетные |
| | | | 39+5N – в остальных случаях |
| ПЕРЕХОД ПО СЧЕТЧИКУ | BCTR | RR | 24 |
| ПЕРЕХОД ПО СЧЕТЧИКУ | BCT | RX | 39 |
| ПЕРЕХОД ПО ИНДЕКСУ БОЛЬШЕ | BXH | RS | 42 |
| ПЕРЕХОД ПО ИНДЕКСУ МЕНЬШЕ ИЛИ РАВНО | BXLE | RS | 42 |
| ПЕРЕХОД С ВОЗВРАТОМ | BALR | RR | 27 |
| ПЕРЕХОД С ВОЗВРАТОМ | BAL | RX | 33 |
| ПРЕОБРАЗОВАНИЕ В ДВОИЧНУЮ | CVB | RX | $50 + 26^D$ |
| ПРЕОБРАЗОВАНИЕ В ДЕСЯТИЧНУЮ | CVD | RX | $54 + 23H + 4H^2$ |
| ПРОВЕРИТЬ ПО МАСКЕ | TM | SI | 19 |
| ПРОВЕРИТЬ И УСТАНОВИТЬ | TS | SI | 21 |
| ПРОЧИТАТЬ СИМВОЛ | IC | RX | 22 |
| РАСПАКОВАТЬ | UNPK | SS | $41+4N_I$, при $N_I > I$; 34 при $N_I = I$ |
| СДВИГ ВЛЕВО | SLA | RS | 62 |
| СДВИГ ВЛЕВО ДВОЙНОЙ | SLDA | RS | 93 |
| СДВИГ ВЛЕВО ДВОЙНОЙ КОДОВ | SLDL | RS | 80 |
| СДВИГ ВЛЕВО КОДОВ | SLL | RS | 55 |
| СДВИГ ВПРАВО | SRA | RS | 54 |
| СДВИГ ВПРАВО ДВОЙНОЙ | SRDA | RS | 83 |
| СДВИГ ВПРАВО ДВОЙНОЙ КОДОВ | SRDL | RS | 80 |
| СДВИГ ВПРАВО КОДОВ | SRL | RS | 52 |
| СЛОЖЕНИЕ КОДОВ | ALR | RR | 21 |
| СЛОЖЕНИЕ КОДОВ | AL | RX | 34 |
| СЛОЖЕНИЕ ПОЛУСЛОВА | AH | RX | 29 |
| СЛОЖЕНИЕ | AR | RR | 20 |
| СЛОЖЕНИЕ | A | RX | 33 |
| СРАВНЕНИЕ | CR | RR | 20 |
| СРАВНЕНИЕ | C | RX | 31 |
| СРАВНЕНИЕ КОДОВ | CLR | RR | 13+3B |
| СРАВНЕНИЕ КОДОВ | CL | RX | 25, если B =1 |
| | | | 25+2B – в остальных случаях |
| СРАВНЕНИЕ КОДОВ | CLI | SI | 19 |
| СРАВНЕНИЕ КОДОВ | CLC | SS | 67+5B |
| СРАВНЕНИЕ ПОЛУСЛОВА | CH | RX | 29 |
| УМНОЖЕНИЕ | MR | RR | 338 |
| УМНОЖЕНИЕ | M | RX | 348 |
| УМНОЖЕНИЕ ПОЛУСЛОВА | MH | RX | 218 |
| УПАКОВАТЬ | PACK | SS | $41+4N_I$ при $N_I > I$; 34 при $N_I = I$ |
| УСЛОВНЫЙ ПЕРЕХОД | BCR | RR | 16 |
| УСЛОВНЫЙ ПЕРЕХОД | BC | RX | 29 |
| УСТАНОВИТЬ МАСКУ ПРОГРАММЫ | SPM | RR | 12 |
| УСТАНОВИТЬ МАСКУ СИСТЕМЫ | SSM | SI | 23 |

Таблица 2

Команды десятичной арифметики

| Команда | | | Время выполнения, мксек |
|---|---|---|---|
| Название | Мнем. | Формат | |
| ВЫЧИТАНИЕ ДЕСЯТИЧНОЕ | SP | SS | $70+3{,}2N_{min}+2{,}2N_{abs}+0{,}2\ N_I$ |
| ДЕЛЕНИЕ ДЕСЯТИЧНОЕ | DP | SS | $2(N_I - N_2)\ (100+19\ N_2)$ |
| ОТРЕДАКТИРОВАТЬ | ED | SS | $46+7\ N+2{,}5\ Z+5\ N_2-0{,}5S$ |
| ОТРЕДАКТИРОВАТЬ И ОТМЕТИТЬ | EDMK | SS | $42+7\ N+2{,}5\ Z+R+5\ N_2-0{,}5S+50TM+3A$ |
| СЛОЖЕНИЕ ДЕСЯТИЧНОЕ | AP | SS | $74+3{,}2N_{min}+2{,}2\ N_{abs}+0{,}2\ N_I$ |
| СЛОЖЕНИЕ С ОЧИСТКОЙ | ZAP | SS | $65+3N_{min}+2{,}5N_{abs}$, если оба адреса четные или нечетные; $70+4{,}5N_{min}+2{,}5N_{abs}$ – в остальных случаях |
| СРАВНЕНИЕ ДЕСЯТИЧНОЕ | CP | SS | $51+2\ N_I+2\ N_2$ |
| УМНОЖЕНИЕ ДЕСЯТИЧНОЕ | MP | SS | $109+9\ N_I+9\ N_2+(2\ N_2-1)\ (27+6\ N_I-3\ N_2)$ |

Таблица 3

Команды арифметики с плавающей запятой

| Команда | | | Время выполнения, мксек |
|---|---|---|---|
| Название | Мнем. | Формат | |
| ВЫЧИТАНИЕ БЕЗ НОРМАЛИЗАЦИИ (ДЛИННОЕ) | SWR | RR | 73 |
| ВЫЧИТАНИЕ БЕЗ НОРМАЛИЗАЦИИ (ДЛИННОЕ) | SW | RX | 54 |
| ВЫЧИТАНИЕ БЕЗ НОРМАЛИЗАЦИИ (КОРОТКОЕ) | SVR | RR | 57 |
| ВЫЧИТАНИЕ БЕЗ НОРМАЛИЗАЦИИ (КОРОТКОЕ) | SV | RX | 70 |
| ВЫЧИТАНИЕ С НОРМАЛИЗАЦИЕЙ (ДЛИННОЕ) | SDR | RR | 98 |
| ВЫЧИТАНИЕ С НОРМАЛИЗАЦИЕЙ (ДЛИННОЕ) | SD | RX | 107 |
| ВЫЧИТАНИЕ С НОРМАЛИЗАЦИЕЙ (КОРОТКОЕ) | SER | RR | 67 |
| ВЫЧИТАНИЕ С НОРМАЛИЗАЦИЕЙ (КОРОТКОЕ) | SE | RX | 53 |
| ДЕЛЕНИЕ (ДЛИННОЕ) | DDR | RR | 2059 |
| ДЕЛЕНИЕ (ДЛИННОЕ) | DD | RX | 2069 |
| ДЕЛЕНИЕ (КОРОТКОЕ) | DER | RR | 389 |
| ДЕЛЕНИЕ (КОРОТКОЕ) | DE | RX | 399 |
| ЗАГРУЗКА (ДЛИННАЯ) | LDR | RR | 31 |
| ЗАГРУЗКА (КОРОТКАЯ) | LER | RR | 19 |
| ЗАГРУЗКА (ДЛИННАЯ) | LD | RX | 40 |
| ЗАГРУЗКА (КОРОТКАЯ) | LE | RX | 27 |
| ЗАГРУЗКА ДОПОЛНЕНИЯ (ДЛИННАЯ) | LCDR | RR | 30 |
| ЗАГРУЗКА ДОПОЛНЕНИЯ (КОРОТКАЯ) | LCER | RR | 19 |
| ЗАГРУЗКА И ПРОВЕРКА (ДЛИННАЯ) | LTDR | RR | 31 |
| ЗАГРУЗКА И ПРОВЕРКА (КОРОТКАЯ) | LTER | RR | 19 |
| ЗАГРУЗКА ОТРИЦАТЕЛЬНАЯ (ДЛИННАЯ) | LNDR | RR | 31 |
| ЗАГРУЗКА ОТРИЦАТЕЛЬНАЯ (КОРОТКАЯ) | LNER | RR | 19 |
| ЗАГРУЗКА ПОЛОЖИТЕЛЬНАЯ (ДЛИННАЯ) | LPDR | RR | 31 |
| ЗАГРУЗКА ПОЛОЖИТЕЛЬНАЯ (КОРОТКАЯ) | LPER | RR | 19 |
| ЗАПИСЬ В ПАМЯТЬ (ДЛИННАЯ) | STD | RX | 38 |

Продолжение

| Команда | | | Время выполнения, мксек |
|---|---|---|---|
| Название | Мнем. | Формат | |
| ЗАПИСЬ В ПАМЯТЬ (КОРОТКАЯ) | STE | RX | 29 |
| ПОПОЛАМ (ДЛИННОЕ) | HDR | RR | 75 |
| ПОПОЛАМ (КОРОТКОЕ) | HER | RR | 45 |
| СЛОЖЕНИЕ БЕЗ НОРМАЛИЗАЦИИ (ДЛИННОЕ) | AWR | RR | 79 |
| СЛОЖЕНИЕ БЕЗ НОРМАЛИЗАЦИИ (ДЛИННОЕ) | AW | RX | 82 |
| СЛОЖЕНИЕ БЕЗ НОРМАЛИЗАЦИИ (КОРОТКОЕ) | AUR | RR | 55 |
| СЛОЖЕНИЕ БЕЗ НОРМАЛИЗАЦИИ (КОРОТКОЕ) | AU | RX | 65 |
| СЛОЖЕНИЕ С НОРМАЛИЗАЦИЕЙ (ДЛИННОЕ) | ADR | RR | 93 |
| СЛОЖЕНИЕ С НОРМАЛИЗАЦИЕЙ (ДЛИННОЕ) | AD | RX | I07 |
| СЛОЖЕНИЕ С НОРМАЛИЗАЦИЕЙ (КОРОТКОЕ) | AER | RR | 62 |
| СЛОЖЕНИЕ С НОРМАЛИЗАЦИЕЙ (КОРОТКОЕ) | AE | RX | 70 |
| СРАВНЕНИЕ (ДЛИННОЕ) | CDR | RR | 58 |
| СРАВНЕНИЕ (ДЛИННОЕ) | CD | RX | 72 |
| СРАВНЕНИЕ (КОРОТКОЕ) | CER | RR | 44 |
| СРАВНЕНИЕ (КОРОТКОЕ) | CE | RX | 52 |
| УМНОЖЕНИЕ (ДЛИННОЕ) | MDR | RR | I239 |
| УМНОЖЕНИЕ (ДЛИННОЕ) | MD | RX | I248 |
| УМНОЖЕНИЕ (КОРОТКОЕ) | MER | RR | 489 |
| УМНОЖЕНИЕ (КОРОТКОЕ) | ME | RX | 498 |

Таблица 4

Команды прямого управления

| Команда | | | Время выполнения, мксек |
|---|---|---|---|
| Название | Мнем. | Формат | |
| ПРЯМАЯ ЗАПИСЬ | WRD | SI | 20 |
| ПРЯМОЕ ЧТЕНИЕ | RDD | SI | 2I |

Лист регистрации изменений

| Изменение | Номера листов (страниц) | | | | Всего листов (страниц) в документе | № документа | Входящий № сопроводительного документа и дата | Подпись Дата |
|---|---|---|---|---|---|---|---|---|
| | измененных | замененных | новых | изъятых | | | | |
| | | | | | | | | |

Внешторгиздат. Изд. № 397 ICT.

Типография ВТИ. Заказ № 1427