МИКРОСХЕМА ИНТЕГРАЛЬНАЯ Л1839ВМ2

Техническое описание

ЩИЗ.480.334 ТО

С О Д Е Р Ж А Н И Е

Инв.№ подл. | Подп. и дата | Взам. инв.№ | Инв.№ дубл. | Подп. и дата

I. СБИС СОПРОЦЕССОРА (СПРЦ) ЛI839ВМ2

I.I. Назначение и общие характеристики

СБИС сопроцессора (СПРЦ) предназначена для выполнения команд умножения и деления целых чисел и всех команд обработки чисел с плавающей запятой (ПЗ) в системе команд VAX. СБИС СПРЦ может работать только совместно с микропроцессором КЛI839ВМI или ЛI839ВМI.

СПРЦ изготавливается по КМОП технологии с двойной металлизацией.

Размер кристалла I0,7xI0,4 мм. Конструктивно кристалл СПРЦ размещается в I32-выводном штырьковом металлокерамическом корпусе типа 6III.I32-4.

Напряжение питания +5В ± I0%.

Частота тактового сигнала синхронизации до I0 МГц.

Потребляемая мощность на максимальной частоте до I Вт.

Основные характеристики СПРЦ:

Разрядность операнда, бит	- 8, I6, 32, 64
Разрядность физического адреса, бит	- 24
Разрядность микрокоманды, бит	- 3I
Время выполнения микрокоманды (микроцикла), нс	- 200
Минимальное количество микроциклов, необходимое для выполнения команды	- 2
Время умножения 32-разрядных чисел в формате "фиксированная запятая", нс	- 800 *
Время умножения 32-разрядных чисел в формате "плавающая запятая", нс	- I500*

* Под временем выполнения операции понимается время от загрузки последнего операнда в сопроцессор до выгрузки результата из него.

Время сложения или вычитание 32-разрядных чисел в формате "плавающая запятая", нс - I500*

Условное графическое обозначение СПРЦ представлено на рис.I.

Назначение выводов представлено в табл.I.

I.2. Электрические параметры, условия эксплуатации

I.2.I. Электрические параметры приведены в табл.2.

I.2.2. Предельно допустимые режимы эксплуатации приведены в табл.3.

I.2.3. Внешние воздействующие факторы:

Синусоидальная вибрация:

диапазон частот, Гц ..... от I до 5000

амплитуда ускорения, $м.с^{-2}$ (g) ..... 400(40)

Механический удар:

одиночного действия:

пиковое ударное ускорение, $м.с^{-2}$ (g) ... I5000(I500)

длительность действия ударного ускорения, мс от 0,I до 2,0

многократного действия:

пиковое ударное ускорение, $м.с^{-2}$ (g) .... I500(I50)

длительность действия ударного ускорения, мс от I до 5

Линейное ускорение, $м.с^{-2}$ (g) ..... 5000(500)

Акустический шум:

диапазон частот, Гц 50-I0000

уровень звукового давления (относительно $2 \cdot I0^{-5}$ Па), дБ I70

Атмосферное пониженное давление:

рабочее, Па (мм рт.ст.) ..... $I,3.I0^{-4}(I0^{-6})$

предельное, Па (мм рт.ст.) ..... $I,3.I0^{-4}(I0^{-6})$

Атмосферное повышенное рабочее давление, атм 3

Повышенная температура среды:

рабочая, ^{0}C 85

предельная, ^{0}C I25

Пониженная температура среды:

рабочая, °С ..... минус 60

предельная, °С ..... минус 60

Смена температур:

от пониженной предельной температуры среды, °С минус 60

до повышенной предельной температуры среды, °С 125

Повышенная относительная влажность при 35°С, % 98

Атмосферные конденсированные осадки (роса, иней)

Соляной (морской) туман

Плесневые грибы

Контрольные среды (среды заполнения):

объемная доля компонентов контрольной среды, %:

гелиево-воздушная 90

аргоно-воздушная 90

аргоно-азотная 90

I.2.4. Минимальная наработка микросхем I00000 часов, а в облегченном режиме при U_{cc} = 5В ± 3% - I20000 часов.

I.2.5. Интенсивность отказов в течение наработки не более $I \cdot I0^{-6}$ I/ч.

I.2.6. Гамма-процентный срок сохраняемости-25 лет.

I.3. Общие принципы функционирования

Микропроцессор выполняет прием и дешифрацию команд, адресацию и выборку микрокоманд, а также адресацию данных.

В системе микрокоманд процессора имеется специальный формат микрокоманды, который предназначен для выполнения команд сопроцессора. Все микрокоманды, выбранные процессором, принимаются не только в процессор, но и в сопроцессор. При этом сопроцессор анализирует микрокоманды и выделяет из общего потока те, которые предназначены для него.

Операции приема-выдачи данных для сопроцессора могут осуществ-

Инв.№ подл. | Подп. и дата | Взам.инв.№ | Инв.№ дубл. | Подп. и дата

Условное графическое обозначение

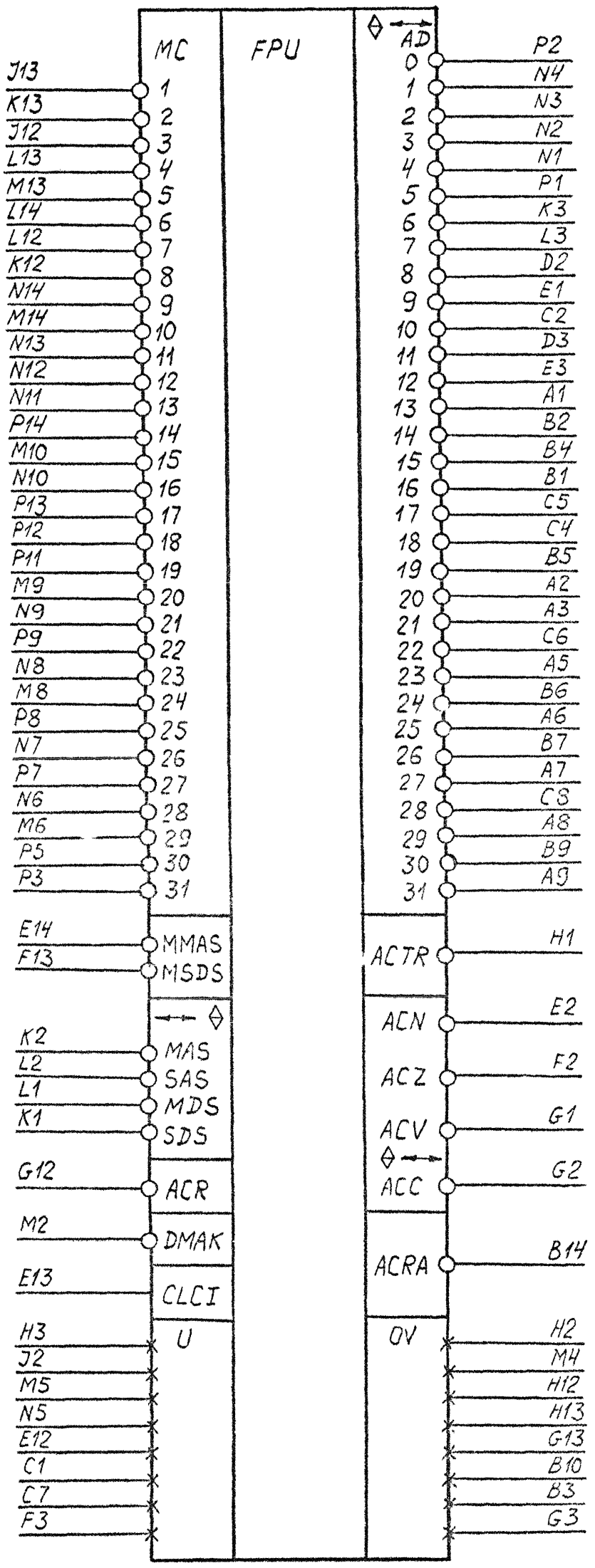


Рис. 1

Таблица 1

ТАБЛИЦА НАЗНАЧЕНИЯ ВЫВОДОВ

Номер вывода	Обозначение	Тип вывода	Наименование
JI3	MCI	Вход	Первый разряд микрокоманды
KI3	MC2	Вход	Второй разряд микрокоманды
JI2	MC3	Вход	Третий разряд микрокоманды
LI3	MC4	Вход	Четвертый разряд микрокоманды
MI3	MC5	Вход	Пятый разряд микрокоманды
LI4	MC6	Вход	Шестой разряд микрокоманды
LI2	MC7	Вход	Седьмой разряд микрокоманды
KI2	MC8	Вход	Восьмой разряд микрокоманды
NI4	MC9	Вход	Девятый разряд микрокоманды
MI4	MCIO	Вход	Десятый разряд микрокоманды
NI3	MCII	Вход	Одиннадцатый разряд микрокоманды
NI2	MCI2	Вход	Двенадцатый разряд микрокоманды
NII	MCI3	Вход	Тринадцатый разряд микрокоманды
PI4	MCI4	Вход	Четырнадцатый разряд микрокоманды
MIO	MCI5	Вход	Пятнадцатый разряд микрокоманды
NIO	MCI6	Вход	Шестнадцатый разряд микрокоманды

Продолжение табл. 1

Номер вывода	Обозначение	Тип вывода	Наименование
PI3	MCI7	Вход	Семнадцатый разряд микрокоманды
PI2	MCI8	Вход	Восемнадцатый разряд микрокоманды
PII	MCI9	Вход	Девятнадцатый разряд микрокоманды
M9	MC20	Вход	Двадцатый разряд микрокоманды
N9	MC2I	Вход	Двадцать первый разряд микрокоманды
P9	MC22	Вход	Двадцать второй разряд микрокоманды
N8	MC23	Вход	Двадцать третий разряд микрокоманды
M8	MC24	Вход	Двадцать четвертый разряд микрокоманды
P8	MC25	Вход	Двадцать пятый разряд микрокоманды
N7	MC26	Вход	Двадцать шестой разряд микрокоманды
P7	MC27	Вход	Двадцать седьмой разряд микрокоманды
N6	MC28	Вход	Двадцать восьмой разряд микрокоманды
M6	MC29	Вход	Двадцать девятый разряд микрокоманды
P5	MC30	Вход	Тридцатый разряд микрокоманды
P3	MC3I	Вход	Тридцать первый разряд микрокоманды
EI4	MMAS	Вход	Сигнал сопровождения адреса микрокоманды
FI3	MSDS	Вход	Сигнал сопровождения микрокоманды

Продолжение табл. 1

Номер вывода	Обозначение	Тип вывода	Наименование
K2	MAS	Вход-выход	Сигнал сопровождения адреса от главного устройства
L2	SAS	Вход-выход	Сигнал подтверждения приема адреса подчиненным устройством
LI	MDS	Вход-выход	Сигнал сопровождения данных от главного устройства
KI	SDS	Вход-выход	Сигнал обработки данных подчиненным устройством
GI2	ACR	Вход	Сигнал сброса сопроцессора
EI3	CLCI	Вход	Сигнал тактовой частоты
P2	AD0	Вход-выход	Нулевой разряд адреса-данных 32-разрядной шины
N4	AD1	Вход-выход	Первый разряд адреса-данных 32-разрядной шины
N3	AD2	Вход-выход	Второй разряд адреса-данных 32-разрядной шины
N2	AD3	Вход-выход	Третий разряд адреса-данных 32-разрядной шины
NI	AD4	Вход-выход	Четвертый разряд адреса-данных 32-разрядной шины
PI	AD5	Вход-выход	Пятый разряд адреса-данных 32-разрядной шины
K3	AD6	Вход-выход	Шестой разряд адреса-данных 32-разрядной шины
L3	AD7	Вход-выход	Седьмой разряд адреса-данных 32-разрядной шины
D2	AD8	Вход-выход	Восьмой разряд адреса-данных 32-разрядной шины
LI	AD9	Вход-выход	Девятый разряд адреса-данных 32-разрядной шины
C2	AD10	Вход-выход	Десятый разряд адреса-данных 32-разрядной шины

Продолжение табл. 1

Номер вывода	Обозначение	Тип вывода	Наименование
D3	AD11	Вход-выход	Одиннадцатый разряд адреса-данных 32-разрядной шины
E3	AD12	Вход-выход	Двенадцатый азряд адреса-данных 32-разрядной шины
A1	AD13	Вход-выход	Тринадцатый разряд адреса-данных 32-разрядной шины
B2	AD14	Вход-выход	Четырнадцатый разряд адреса-данных 32-разрядной шины
B4	AD15	Вход-выход	Пятнадцатый разряд адреса-данных 32-разрядной шины
B1	AD16	Вход-выход	Шестнадцатый разряд адреса-данных 32-разрядной шины
C5	AD17	Вход-выход	Семнадцатый разряд адреса-данных 32-разрядной шины
C4	AD18	Вход-выход	Восемнадцатый разряд адреса-данных 32-разрядной шины
B5	AD19	Вход-выход	Девятнадцатый разряд адреса-данных 32-разрядной шины
A2	AD20	Вход-выход	Двадцатый разряд адреса-данных 32-разрядной шины
A3	AD21	Вход-выход	Двадцать первый разряд адреса-данных 32-разрядной шины
C6	AD22	Вход-выход	Двадцать второй разряд адреса-данных 32-разрядной шины
A5	AD23	Вход-выход	Двадцать третий разряд адреса-данных 32-разрядной шины
B6	AD24	Вход-выход	Двадцать четвертый разряд адреса-данных 32-разрядной шины
A6	AD25	Вход-выход	Двадцать пятый разряд адреса-данных 32-разрядной шины
B7	AD26	Вход-выход	Двадцать шестой разряд адреса-данных 32-разрядной шины
A7	AD27	Вход-выход	Двадцать седьмой разряд адреса-данных 32-разрядной шины

Продолжение табл. 1

Номер вывода	Обозначение	Тип вывода	Наименование
C8	AD28	Вход-выход	Двадцать восьмой разряд адреса-данных 32-разрядной шины
A8	AD29	Вход-выход	Двадцать девятый разряд адреса-данных 32-разрядной шины
B9	AD30	Вход-выход	Тридцатый разряд адреса-данных 32-разрядной шины
A9	AD3I	Вход-выход	Тридцать первый разряд адреса-данных 32-разрядной шины
HI	ACTR	Выход	Сигнал захвата магистрали сопроцессором
E2	ACN	Выход	Признак "Результат отрицателен"
F2	ACZ	Выход	Признак "Результат равен нулю"
GI	ACV	Выход	Признак "Переполнение"
G2	ACC	Вход-выход	Признак "Расширение"
BI4	ACRA	Выход	Сигнал готовности сопроцессора
M2	DMAK	Вход	Сигнал предоставления прямого доступа к памяти
H3	U		Вывод питания от источника напряжения
J2	U		Вывод питания от источника напряжения
M5	U		Вывод питания от источника напряжения
N5	U		Вывод питания от источника напряжения
EI2	U		Вывод питания от источника напряжения
CI	U		Вывод питания от источника напряжения

Продолжение табл. 1

Номер вывода	Обозначение	Тип вывода	Наименование
C7	U		Вывод питания от источника напряжения
F3	U		Вывод питания от источника напряжения
H2	0V		Общий вывод
M4	0V		Общий вывод
HI2	0V		Общий вывод
HI3	0V		Общий вывод
GI3	0V		Общий вывод
BI0	0V		Общий вывод
B3	0V		Общий вывод
G3	0V		Общий вывод
JI			Не используется
J3			Не используется
MI			Не используется
M3			Не используется
P4			Не используется
P6			Не используется
M7			Не используется

Продолжение табл. 1

Номер вывода	Обозначение	Тип вывода	Наименование
PI0			Не используется
MII			Не используется
MI2			Не используется
KI4			Не используется
JI4			Не используется
HI4			Не используется
GI4			Не используется
FI4			Не используется
FI2			Не используется
DI4			Не используется
CI4			Не используется
CI3			Не используется
DI3			Не используется
DI2			Не используется
AI4			Не используется
BI3			Не используется
CI2			Не используется

Продолжение табл.1

Номер вывода	Обозначение	Тип вывода	Наименование
AI2			Не используется
AI3			Не используется
CIO			Не используется
CII			Не используется
AII			Не используется
BI2			Не используется
AIO			Не используется
BII			Не используется
C9			Не используется
B8			Не используется
A4			Не используется
C3			Не используется
DI			Не используется
FI			Не используется

Таблица 2

Наименование параметра, единица измерения, режим измерения	Буквенное обозначение	Норма		Температура, °С
		не менее	не более	
Выходное напряжение низкого уровня, В при I_{OL} = 2,5 мА	U_{OL}	-	0,45	25± 10
			0,5	-60 85
Выходное напряжение высокого уровня, В при I_{OH} = -0,2 мА	U_{OH}	4,05	-	25±10
		4,0		-60 85
Ток потребления, мА	I_{CC}	-	8,0	25±10
			10,0	-60 85
Ток потребления динамический, мА	I_{CCO}	-	170	25±10
			200	-60 85
Ток утечки на входе, мкА	I_{LI}	-	3,0	25±10
			10,0	-60 85
Ток утечки на выходе, мкА	I_{LO}	-	10,0	25±10
			50,0	-60 85
Частота следования импульсов тактовых сигналов, МГц	f_C	10,0	-	25±10 -60 85
Входная емкость, пФ	C_I	-	12,0	25±10
Выходная емкость, пФ	C_O	-	12,0	25±10
Емкость входа/выхода, пФ	$C_{I/O}$	-	14,0	25±10

Продолжение табл.2

Наименование параметра, единица измерения, режим измерения	Буквенное обозначение	Норма		Температура, °С
		не менее	не более	
Время умножения 32-разрядных чисел в формате "фиксированная запятая", нс	t_{MUL}	-	800*	25±10 -60 85
Время умножения 32-разрядных чисел в формате "плавающая запятая", нс	t_{MULF}	-	1500*	25+10 -60 85
Время сложения 32-разрядных чисел в формате "плавающая запятая", нс	t_{ADDF}	-	1500*	25±10 -60 85

* - без времени обмена данными по магистрали

Таблица 3

Наименование параметра, единица измерения	Буквенное обозначение	Норма	
		не менее	не более
Напряжение питания, В	U_{CC}	4,5	5,5
Входное напряжение высокого уровня, В	U_{IH}	(0,85xUcc)*	Ucc**
Входное напряжение низкого уровня, В	U_{IL}	0	0,8*
Напряжение на любом входе , В	U_I	0	Ucc**
Выходной ток низкого уровня, мА	I_{OL}	-	2,5
Выходной ток высокого уровня, мА	I_{OH}	-	0,2
Емкость нагрузки, пФ	C_L	-	I00
Время фронта нарастания и спада входных сигналов, нс	t_{LH}, t_{HL}	-	$0,I \cdot T_{CLCI}$

* - с учетом всех видов помех.

** - но не более конкретного значения напряжения питания, приложенного к микросхеме.

ляться или в процедуре чтения-записи по адресу системной памяти, равному 000000, или в специальной процедуре обмена по магистрали, который называется обмен с перехватом. Если нужно передать данные из процессора в сопроцессор, то используется процедура записи по адресу системной памяти, равному 000000. Если нужно передать данные из СПРЦ в процессор, то используется процедура чтения по адресу системной памяти, равному 000000. Если нужно принять данные из памяти в СПРЦ, то используется процедура чтения с перехватом. В этой процедуре процессор выполняет адресацию, а данные принимаются не в процессор, а в сопроцессор. Если нужно переслать данные из СПРЦ в память, то используется процедура записи с перехватом. В этой процедуре процессор выполняет адресацию, а данные выдает сопроцессор.

Всего в СПРЦ выполняется 85 команд. Коды команд приведены в табл.4.

Все основные операции выполнения команд (расформирование и формирование формата ПЗ, умножение, деление) выполняются на сопроцессоре аппаратно. Сдвиг может выполняться на I-I5 разрядов за один такт. Кроме того, такие дополнительные операции, связанные с выполнением команд, как нормализация, округление, изменение знака, также выполняются аппаратно. Аппаратно выполняется также отслеживание особых случаев (переполнение, антипереполнение и т.д.).

Время выполнения некоторых операций:

умножение в форматах B,W ,L и F - 3 периода тактовой частоты(3T);

умножение в форматах D и G - I4T.

Получение одной цифры частного в операции деления - 0,5T.

Операция сложения-вычитания - 2T.

Сдвиг на I-I5 разрядов - IT.

Операции обработки мантиссы, порядка и знака выполняются одновременно. Передача состояний из СПРЦ в процессор выполняется аппаратно по выводам ACN, ACZ, ACV, ACC, которые есть и у процессора и у сопроцессора.

Таблица 4

СИСТЕМА КОМАНД

Код	Мнемоника	Формат	Назначение
53	TSTF	КОП src.rx	Тестирование в формате F
73	TSTD	-"-	Тестирование в формате D
53FD	TSTG	-"-	Тестирование в формате G
52	MNEGF	КОП src.rx,dst.wx	Пересылка с арифметическим отрицанием в формате F
72	MNEGD	-"-	Пересылка с арифметическим отрицанием в формате D
52FD	MNEGG	-"-	Пересылка с арифметическим отрицанием в формате G
50	MOVF	-"-	Пересылка слов формата F
70	MOVD	-"-	Пересылка слов формата D
50FD	MOVG	-"-	Пересылка слов формата G
4C	CVTBF	КОП src.rx,dst.wy	Преобразование байта в слово формата F
6C	CVTBD	-"-	Преобразование байта в слово формата D
4CFD	CVTBG	-"-	Преобразование байта в слово формата G
4D	CVTWF	-"-	Преобразование слова в слово формата F
6D	CVTWD	-"-	Преобразование слова в слово формата D
4DFD	CVTWG	-"-	Преобразование слова в слово формата G
4E	CVTLF	-"-	Преобразование двойного слова в слово формата F
6E	CVTLD	-"-	Преобразование двойного слова в слово формата D
4EFD	CVTLG	-"-	Преобразование двойного слова в слово формата G

Код	Мнемоника	Формат	Назначение
48	CVTFB	КОП src.rx,dst.wy	Преобразование слова формата F в байт
49	CVTFW	-"-	Преобразование слова формата F в слово
4A	CVTFL	-"-	Преобразование слова формата F в двойное слово
4B	CVTRFL	КОП src.rx,dst.wl	Преобразование слова формата F в двойное слово с округлением
56	CVTFD	КОП src.rx,dst.wy	Преобразование слова формата F в слово формата D
99FD	CVTFG	-"-	Преобразование слова формата F в слово формата G
68	CVTDB	-"-	Преобразование слова формата D в байт
69	CVTDW	-"-	Преобразование слова формата D в слово
6A	CVTDL	-"-	Преобразование слова формата D в двойное слово
6B	CVTRDL	КОП src.rx,dst.wl	Преобразование слова формата D в двойное слово с округлением
76	CVTDF	КОП src.rx,dst.wy	Преобразование слова формата D в слово формата F
48FD	CVTGB	-"-	Преобразование слова формата G в байт
49FD	CVTGW	-"-	Преобразование слова формата G в слово
4AFD	CVTGL	-"-	Преобразование слова формата G в двойное слово
4BFD	CVTRGL	КОП src.rx,dst.wl	Преобразование слова формата G в двойное слово с округлением

Код	Мнемоника	Формат	Назначение
33FD	CVTGF	КОП src.rx,dst.wy	Преобразование слова формата G в слово формата F
51	CMPF	КОП src1.rx,src2.rx	Сравнение слов формата F
71	CMPD	-"-	Сравнение слов формата D
51FD	CMPG	-"-	Сравнение слов формата G
84	MULB2	КОП mulr.rx,prod.mx	Умножение байтов двухоперандное
A4	MULW2	-"-	Умножение слов двухоперандное
C4	MULL2	-"-	Умножение двойных слов двухоперандное
86	DIVB2	КОП divr.rx,quo.mx	Деление байтов двухоперандное
A6	DIVW2	-"-	Деление слов двухоперандное
C6	DIVL2	-"-	Деление двойных слов двухоперандное
40	ADDF2	КОП addr.rx,sum.mx	Сложение двухоперандное в формате F
60	ADDD2	-"-	Сложение двухоперандное в формате D
40FD	ADDG2	-"-	Сложение двухоперандное в формате G
42	SUBF2	КОП sub.rx,dif.mx	Вычитание двухоперандное в формате F
62	SUBD2	-"-	Вычитание двухоперандное в формате D
42FD	SUBG2	-"-	Вычитание двухоперандное в формате G
44	MULF2	КОП mulr.rx,prod.mx	Умножение двухоперандное в формате F

Код	Мнемоника	Формат	Назначение
64	MULD2	КОП *mulr.rx,prod.mx*	Умножение двухоперандное в формате D
44FD	MULG2	–"–	Умножение двухоперандное в формате G
46	DIVF2	КОП *divr.rx,quo.mx*	Деление двухоперандное в формате F
66	DIVD2	–"–	Деление двухоперандное в формате D
46FD	DIVG2	–"–	Деление двухоперандное в формате G
85	MULB3	КОП *mulr.rx,muld.rx, prod.wx*	Умножение байтов трехоперандное
A5	MULW3	–"–	Умножение слов трехоперандное
C5	MULL3	–"–	Умножение двойных слов трехоперандное
87	DIVB3	КОП *divr.rx,divd.rx, quo.wx*	Деление байтов трехоперандное
A7	DIVW3	–"–	Деление слов трехоперандное
C7	DIVL3	–"–	Деление двойных слов трехоперандное
4I	ADDF3	КОП *add1.rx,add2.rx, sum.wx*	Сложение трехоперандное в формате F
6I	ADDD3	–"–	Сложение трехоперандное в формате D
4IFD	ADDG3	–"–	Сложение трехоперандное в формате G
43	SUBF3	КОП *sub.rx,min.rx, dif.wx*	Вычитание трехоперандное в формате F
63	SUBD3	–"–	Вычитание трехоперандное в формате D

Код	Мнемоника	Формат	Назначение
43F-D	SUBG3	КОП sub.rx, min.rx, dif.wx	Вычитание трехоперандное в формате G
45	MULF3	КОП mulr.rx, muld.rx, prod.wx	Умножение трехоперандное в формате F
65	MULD3	-"-	Умножение трехоперандное в формате D
45F-D	MULG3	-"-	Умножение трехоперандное в формате G
47	DIVF3	КОП divr.rx, divd.rx, quo.wx	Деление трехоперандное в формате F
67	DIVD3	-"-	Деление трехоперандное в формате D
47F-D	DIVG3	-"-	Деление трехоперандное в формате G
7A	EMUL	КОП mulr.rl, muld.rl, add.rl, prod.wq	Расширенное умножение
7B	EDIV	КОП divr.rl, divd.rq, quo.wl, rem.wl	Расширенное деление
54	EMODF	КОП mulr.rx, mulrx.rb, muld.rx, int.wl, fract.wx	Расширенное умножение в формате F с выделением целой части
74	EMODD	-"-	Расширенное умножение в формате D с выделением целой части
54F-D	EMODG	КОП mulr.rx, mulrx.rw, muld.rx, int.wl, fract.wx	Расширенное умножение в формате G с выделением целой части
79	ASHQ	КОП cnt.rb, src.rx, dst.wx	Арифметический сдвиг квадрослова
55	POLYF	КОП arg.rf, degree.rw, tbladdr.ab, {R0-3.wl}	Вычисление значения полинома в формате F

Продолжение табл. 4

Код	Мнемоника	Формат	Назначение
75	POLYD	КОП arg.rd, degree.rw, tbladdr.ab, {R0-5.we}	Вычисление значения полинома в формате D
55FD	POLYG	КОП arg.rg, degree.rw, tbladdr.ab, {R0-5.we}	Вычисление значения полинома в формате G
4F	ACBF	КОП limit.rx, add.rx, index.mx, displ.bw	Сложение слов формата F со сравнением и переход
6F	ACBD	-"-	Сложение слов формата D со сравнением и переход
4FFD	ACBG	-"-	Сложение слов формата G со сравнением и переход

Примечания:

1. X – первый тип данных определяется инструкцией
2. Y – второй тип данных определяется инструкцией
3. src – источник
4. dst – приемник
5. r – операнд доступен только по чтению
6. W – операнд доступен только по записи
7. m – выполняется модификация значения (допустимо как чтение содержимого поля, адресуемого операндом, так и запись в это поле)
8. prod – результат
9. quo – частное
10. rem – остаток
11. dif – разность
12. int – целая часть результата
13. fract – дробная часть результата
14. cnt – счетчик
15. arg – аргумент

16. *degree* - степень
17. *tbladdr* - адрес таблицы коэффициентов
18. *limit* - предельное значение содержимого счетчика
19. *index* - текущее значение
20. *displ* - адрес перехода

2. СТРУКТУРНАЯ СХЕМА

Структурная схема СБИС сопроцессора представлена на рис.2.

СБИС СПРЦ состоит из следующих основных функциональных блоков:

блок приема/выдачи данных и состояний (БПВД);

операционный блок (ОБ);

блок умножения (БУМ);

блок обработки порядка (БОП);

блок обработки особых случаев (БОС);

блок микропрограммного управления (БМУ);

интерфейсный блок (ИБ).

Все операции приема и выдачи данных выполняются блоком БПВД. Сигналы сопровождения обмена и адреса на магистрали AD отслеживаются интерфейсным блоком и, если обнаруживается условие обмена с сопроцессором, то данные принимаются или выдаются блоком БПВД. Кроме операций обмена данными, в этом блоке выполняется расформирование и формирование форматов ПЗ. Расформирование заключается в выделении из формата ПЗ DEC мантиссы, порядка и знака.

Формирование – это обратная операция.

В БПВД выполняется также формирование и выдача битов состояния N и Z , формирование условия ветвления в команде АСВ, формирование кода ошибки.

БПВД (рис. 3, 4) связан с другими блоками: 64-разрядной магистралью данных (MD) , 32-разрядной шиной принимаемых и выдаваемых данных D(0-31), 11-разрядной магистралью порядка принимаемых данных DA(4-14), 11-разрядной магистралью порядка выдаваемых данных POR(0-10) , а также различными управляющими сигналами.

Структурная схема СБИС сопроцессора

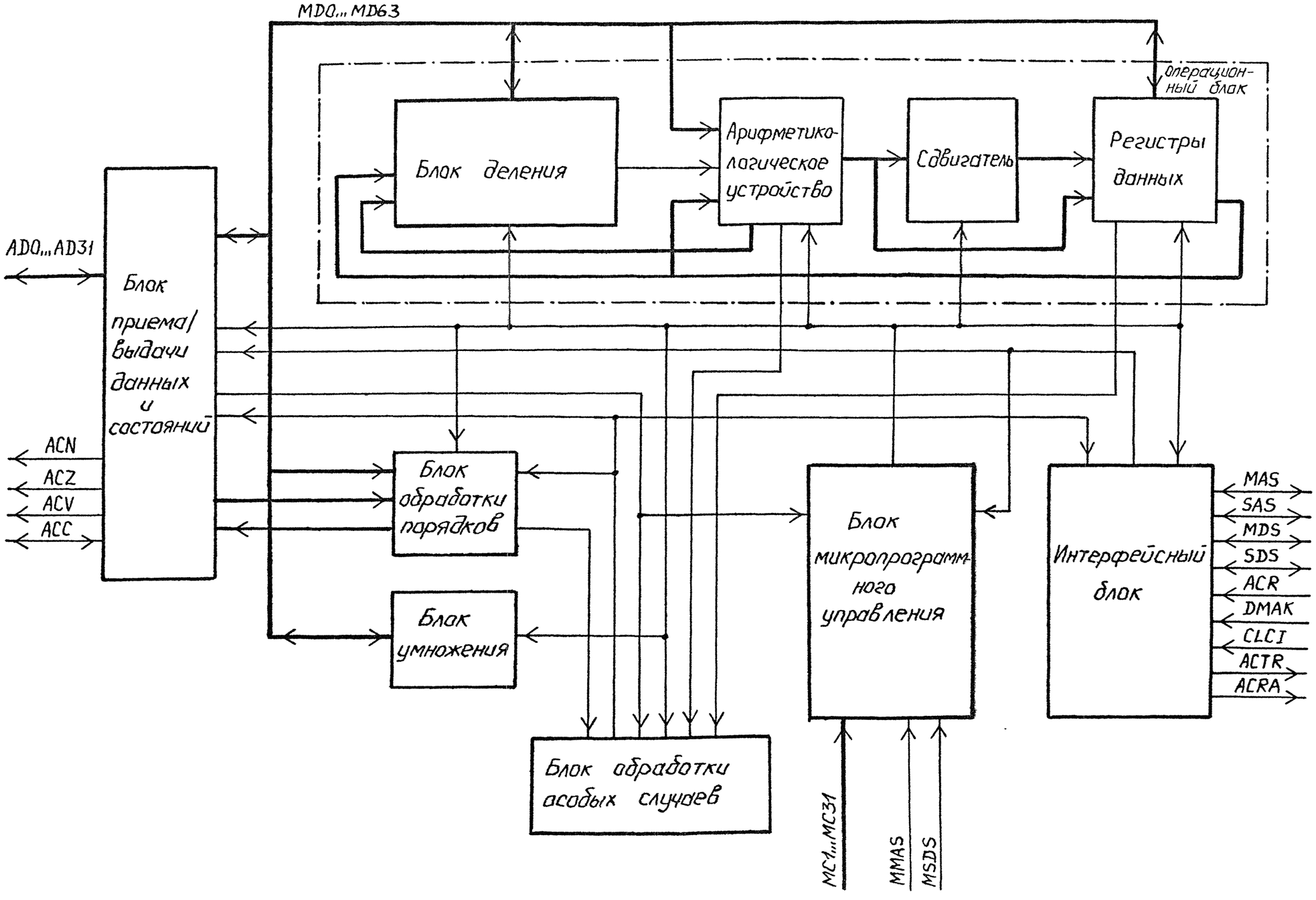
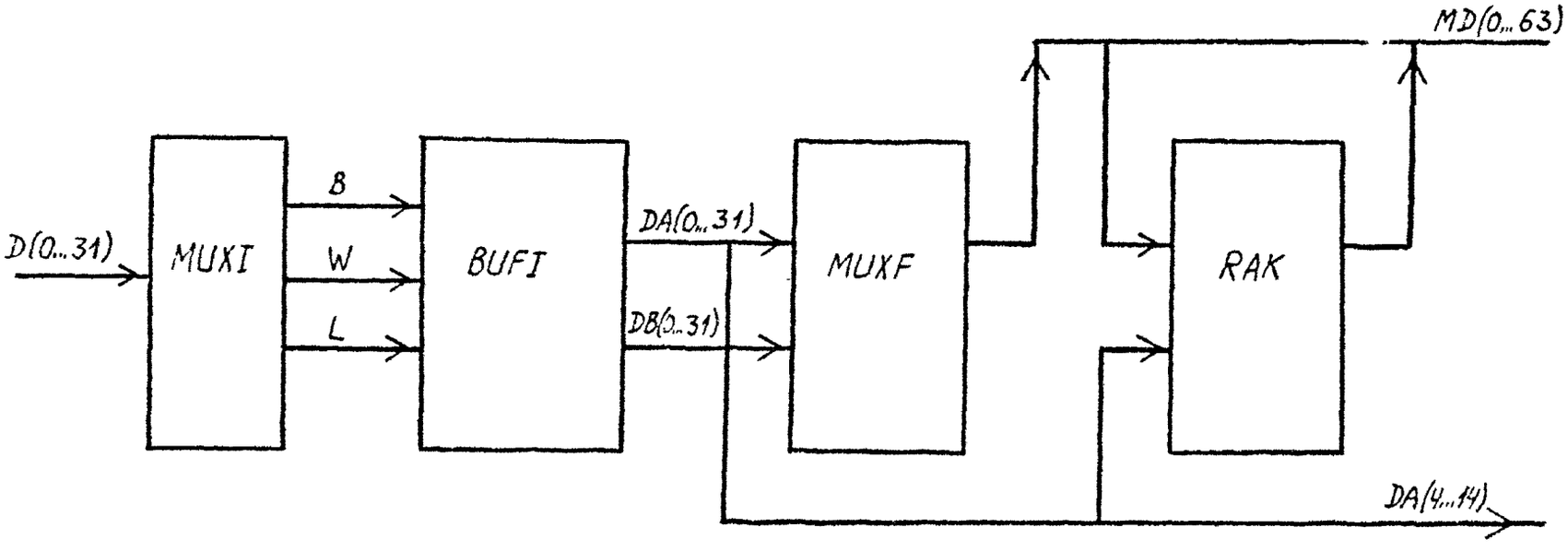


Рис.2.

Входной буфер данных и аккумулятор



MUXI - входной мультиплексор

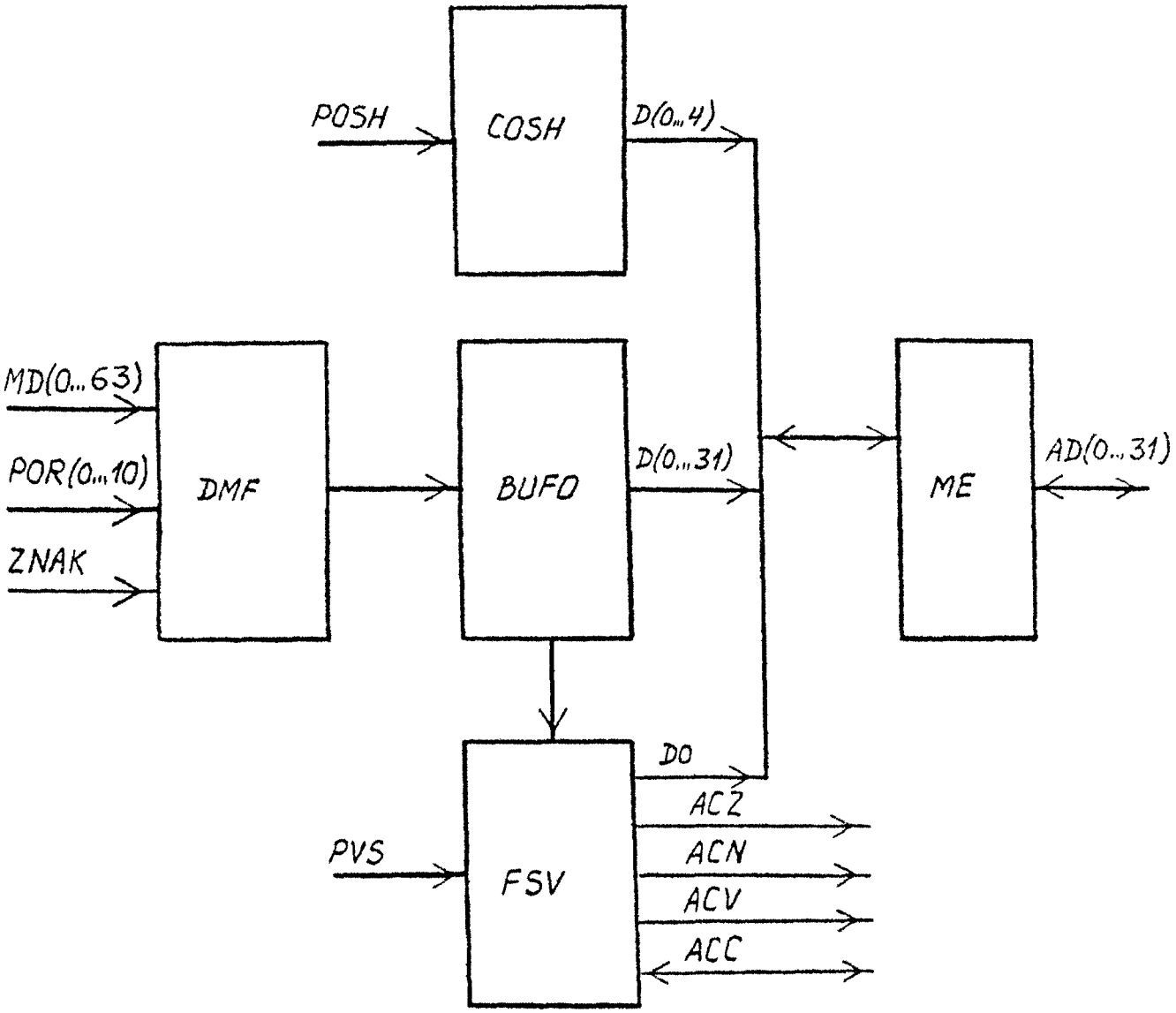
BUFI - буферные регистры

MUXF - мультиплексор форматов

RAK - регистр-аккумулятор

Рис. 3,

Устройства выдачи данных



DMF – демультиплексор форматов

BUFO – выходной буфер данных

COSH – блок формирования кода ошибки

FSV – блок формирования кодов состояний и ветвлений

ME – магистральные элементы

Рис. 4.

Все операции, кроме умножения, над целыми числами и мантиссой формата ПЗ выполняются в ОБ (рис. 5). Здесь выполняются операции сложения/вычитания, изменения знака, округления, сдвига, нормализации и деления. В операции умножения формата ПЗ ОБ принимает участие совместно с БУМ. В ОБ выполняется также формирование знака данных формата ПЗ. ОБ связан с другими блоками 64-разрядной магистралью MD и управляющими сигналами. Кроме того, в ОБ формируются некоторые признаки результата выполнения операции, такие как знак результата, арифметическое переполнение, условия округления и т.д. Эти признаки используются другими блоками СБИС СПРЦ.

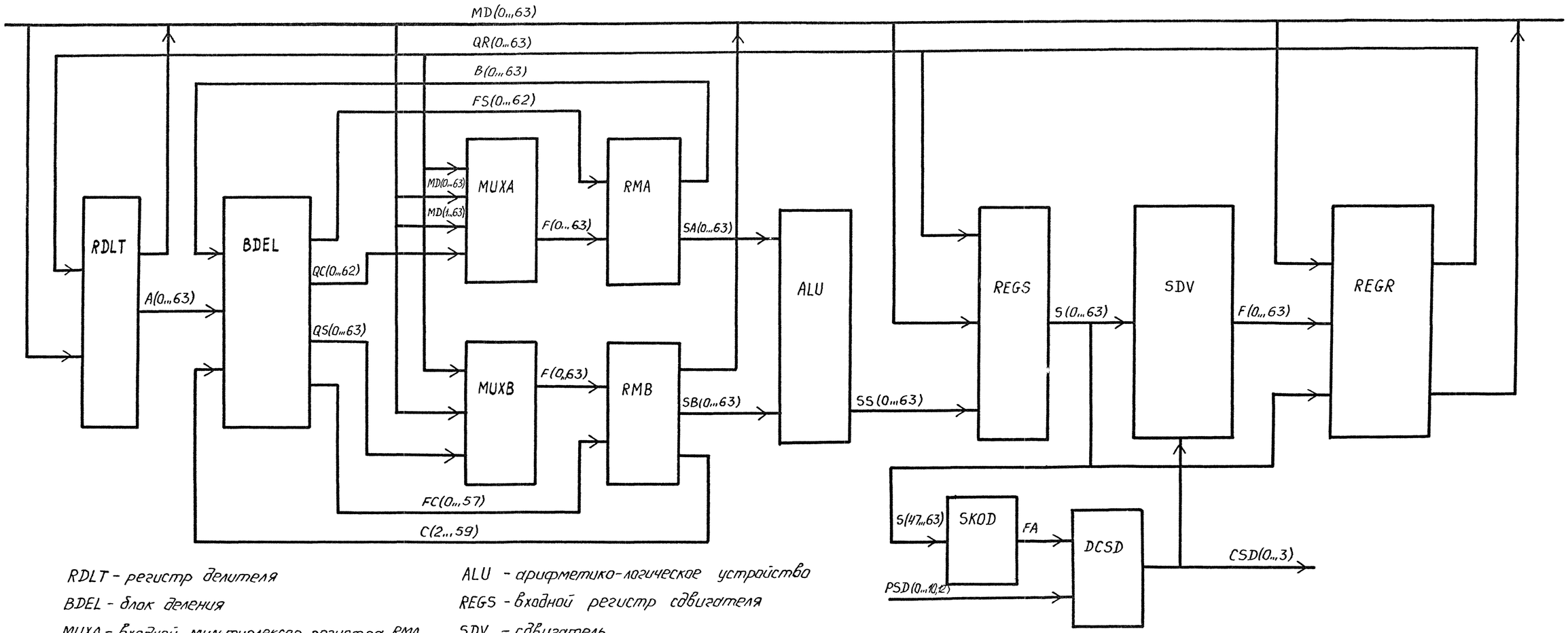
БУМ (рис.6) выполняет операции умножения. Умножение может быть как знаковым, так и кодовым. За один цикл БУМ выполняет умножение двух 32-разрядных чисел и 64-разрядный результат выдает на магистраль MD.

Умножитель построен на основе алгоритма умножения на две цифры множителя. В соответствии с этим необходимо иметь удвоенное и утроенное множимое. Удвоенное множимое получается сдвигом множимого на один разряд влево. Утроенное множимое вычисляется на специальных сумматорах, находящихся в БУМ.

Умножение 64-разрядных чисел выполняется за 4 цикла БУМ совместно с ОБ. Управление работой ОБ в этом случае выполняет БУМ. Определение факта арифметического переполнения при умножении выполняется в БПВД под управлением БУМ. Связь БУМ с другими блоками осуществляется по магистрали MD и управляющими сигналами.

БОП (рис.7) выполняет все операции по вычислению и обработке порядка формата ПЗ, а также управление работой сдвигателя в ОБ в операциях сдвига и нормализации. БОП принимает порядок операндов в командах ПЗ по магистрали DA (4-I4) , а параметр

Структурная схема операционного блока



RDLT - регистр делителя

BDEL - блок деления

MUXA - входной мультиплексор регистра RMA

MUXB - входной мультиплексор регистра RMB

RMA - регистр мантиссы A

RMB - регистр мантиссы B

ALU - арифметико-логическое устройство

REGS - входной регистр сдвигателя

SDV - сдвигатель

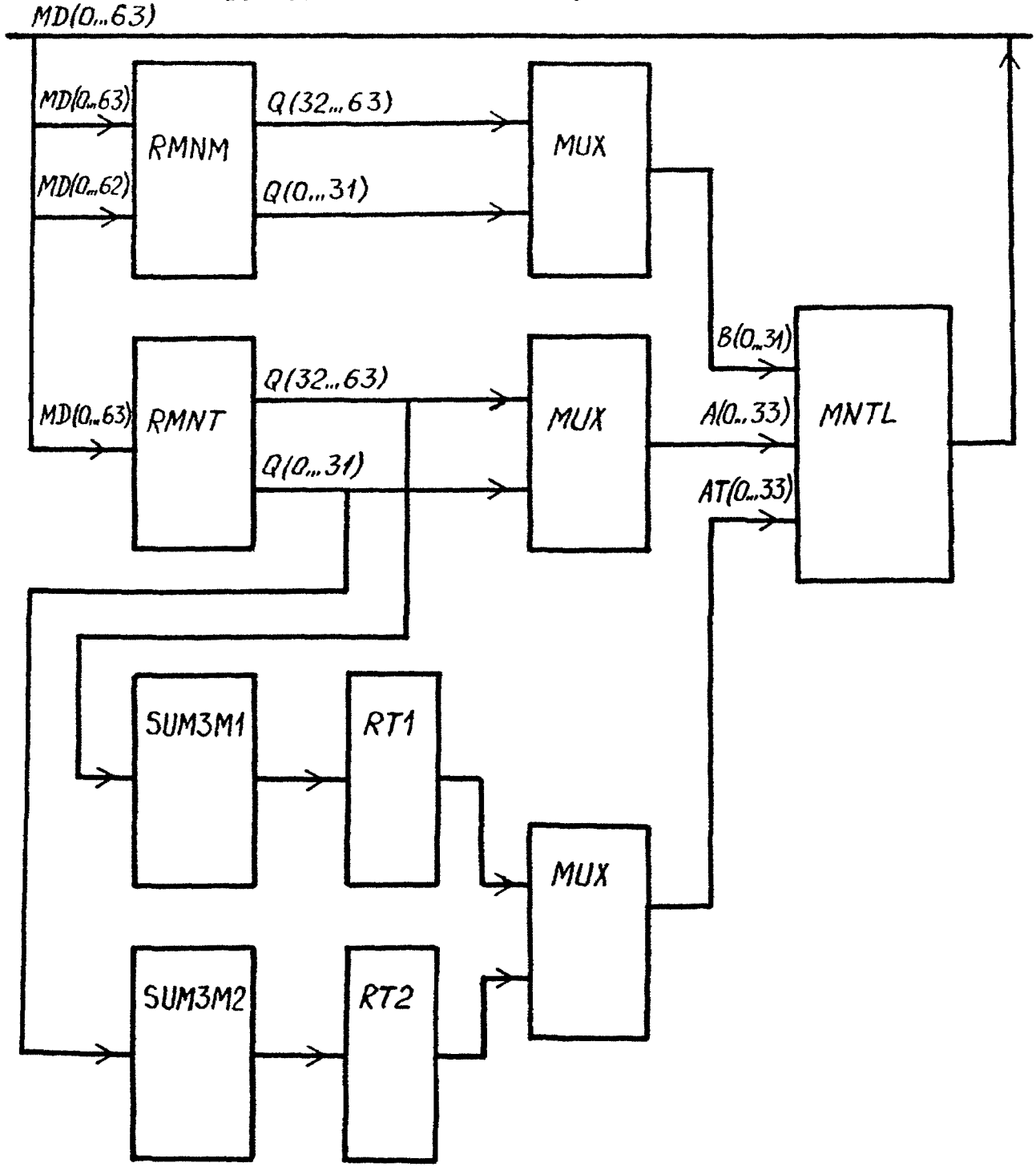
REGR - регистр результата

SKOD - схема определения количества одинаковых цифр

DCSD - дешифратор сдвига

Рис. 5.

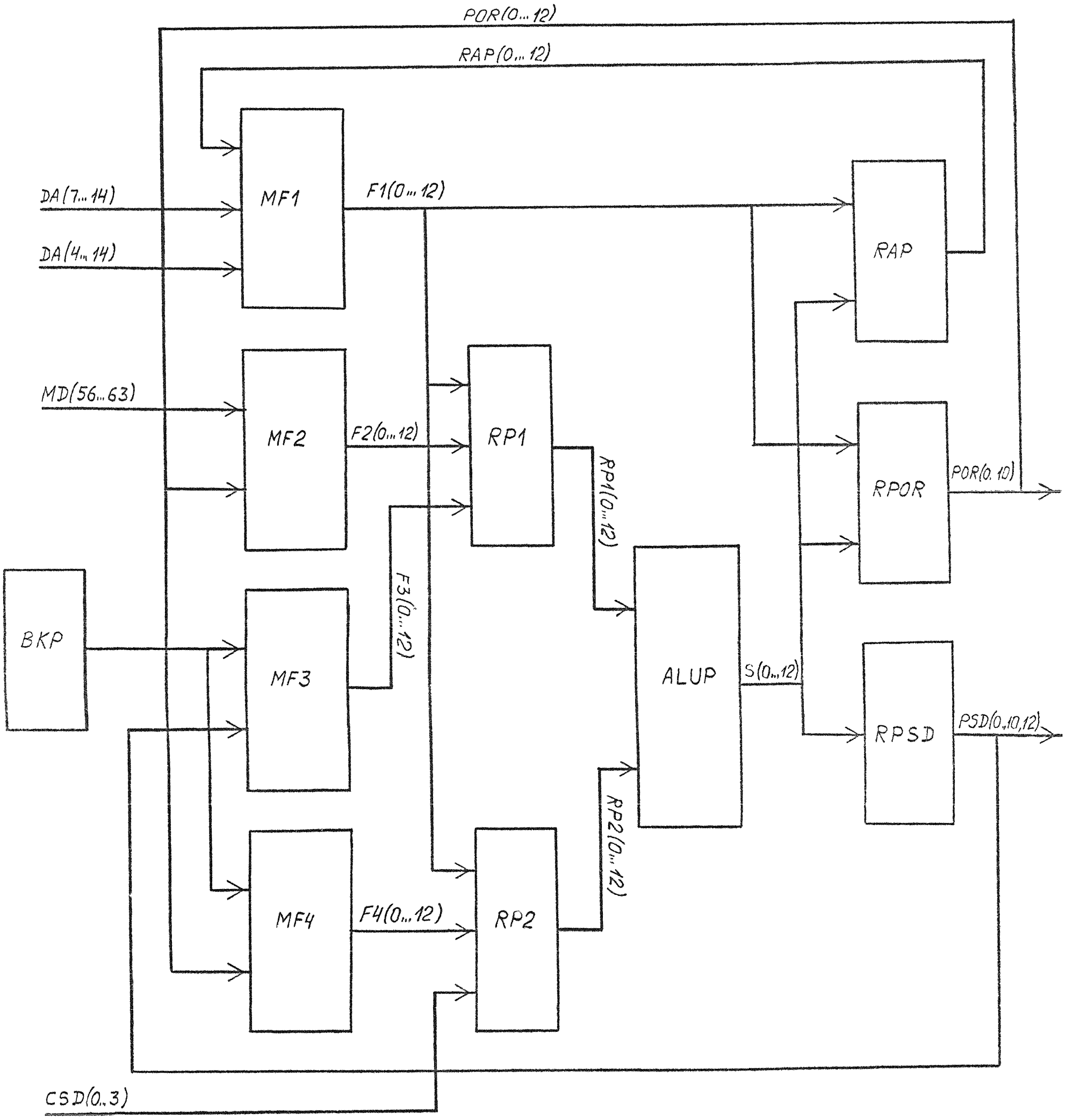
Структурная схема блока умножения



RMNM – регистр множимого
RMNT – регистр множителя
MUX – мультиплексор
SUM3M1 – сумматор старшей половины утроеннного множителя
SUM3M2 – сумматор младшей половины утроенного множителя
RT1, RT2 – регистры утроенного множителя
MNTL – умножитель

Рис. 6.

Блок обработки порядка



BKP – блок констант порядка

MF1...MF4 – мультиплексоры

RP1 – регистр порядка I

RP2 – регистр порядка 2

ALUP – арифметико-логическое устройство БОП

RAP – регистр-аккумулятор порядка

RPOR – регистр порядка результата

RPSD – регистр параметра сдвига

Рис. 7

сдвига в команде *ASHQ* по магистрали *MD* (56 - 63) из БПВД.

Вычислительный порядок передается из БОП в БПВД по магистрали POR (0-10). При выполнении сдвига и нормализации параметр сдвига передается в БОП из ОБ по 4-разрядной магистрали параметра сдвига *CSD* (0-3). Из БОП передается для управления сдвигателем в ОБ 12-разрядный код сдвига (*PSD* (0-10, 12)). В БОП вырабатываются также различные признаки, связанные с обработкой порядка и сдвигом, которые используются для отслеживания особых случаев (переполнение, антипереполнение), возникающих при выполнении команд. Кроме этого, БОП связан с другими блоками различными управляющими сигналами.

Все особые случаи, возникающие при выполнении команд, отслеживаются в БОС. К этим особым случаям относятся:

целое переполнение;

переполнение в командах ПЗ;

антипереполнение;

целое деление на ноль;

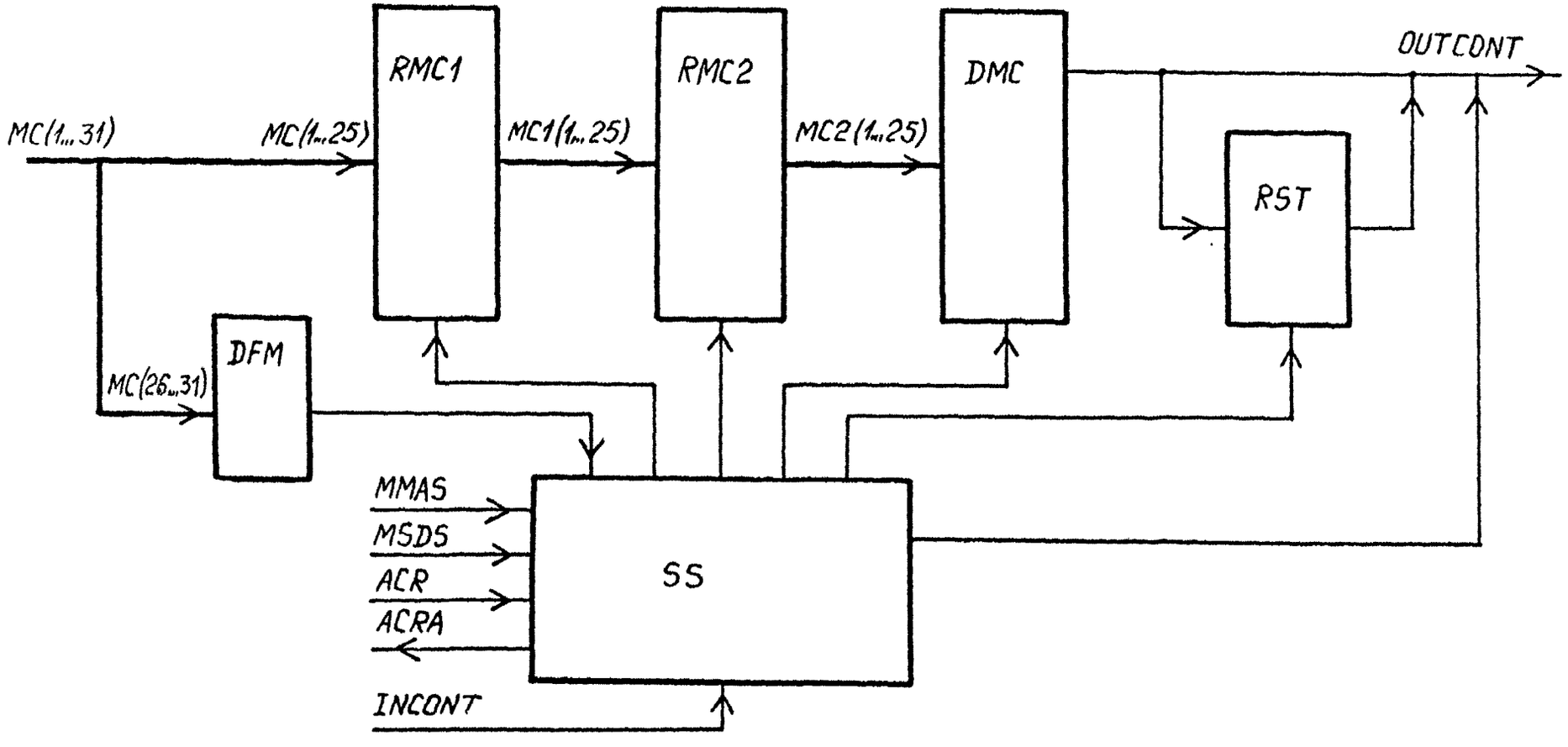
деление на ноль в командах ПЗ;

резервный операнд.

Сигнал разрешения антипереполнения (бит *FU* РСП) передается из процессора в сопроцессор по внешнему выводу ACC. По этому же выводу передается из СПРЦ в процессор сигнал фатальной ошибки. Кроме сигнала фатальной ошибки в центральный процессор из сопроцессора передается бит арифметической ошибки.

Операции приема и дешифрации микрокоманд выполняет БМУ (рис.8). Адресацию микрокоманд выполняет процессор. При этом все микрокоманды, передаваемые по магистрали микрокоманд, принимаются в СПРЦ , где из всего потока выбираются микрокоманды, имеющие формат 11101О.

Структурная схема блока микропрограммного управления



SS – схема синхронизации
RMC1 – регистр микрокоманд I
RMC2 – регистр микрокоманд 2
DFM – дешифратор формата микрокоманд
RST – регистр хранения

Рис.8

Микрокоманды с этим форматом вызывают торможение процес - сора, а сопроцессор начинает ее выполнение. Поля микрокоманды дешифрируются и различные управляющие сигналы передаются из БМУ в другие блоки. Кроме дешифрации микрокоманды БМУ вырабатывает сигналы синхронизации приема и обработки микрокоманды, выполняет согласование сигналов обработки с приемом данных и управляет выдачей состояний из БПВД при завершении команды.

Интерфейсный блок (ИБ) сопроцессора предназначен для организации обменов сопроцессора с процессором и сопроцессора с памятью.

3. СТРУКТУРА РЕГИСТРОВ БИС СПРЦ И СИСТЕМА МЕЖРЕГИСТРОВЫХ СВЯЗЕЙ

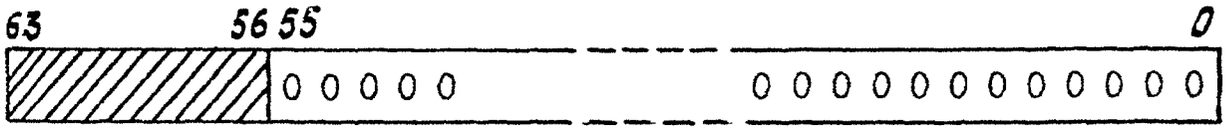
Данные поступают в СПРЦ на входной буфер операндов (рис.3) С внешней магистрали *AD* данные поступают на внутреннюю шину данных *D*(0-31) и через входной мультиплексор *(MUXI)* записываются на буферные регистры *(BUFI)* .

При приеме данные всех форматов преобразуются в 64-разрядный формат. Преобразование выполняется на входном мультиплексоре *(MUXI)* и на мультиплексоре форматов *(MUXF)* .

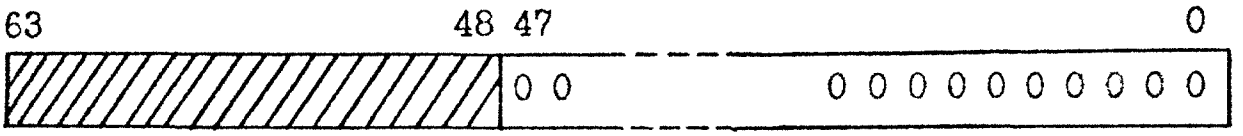
На мультиплексоре *MUXI* выполняется перемещение данных типа "байт" и "слово" из младших разрядов шины *D*(0-31) в старшие разряды буферных регистров. Младшие разряды при этом заполняются нулями. Входной буфер построен как четырехрегистровый стек "первый вошел - первый вышел". Разрядность каждого из четырех регистров равна 32. Выход первого регистра соединен с мультиплексором *MUXF* шиной *DA*(0-31), а выход второго - шиной *DB*(0-31). На мультиплексоре, в зависимости от формата данных, выполняются необходимые перестановки битов информации и дополнение определенных полей нулями. С мультиплексора *MUXF* данные читаются на 64-разрядную магистраль *MD* (0-63). Кроме того, через ответвление шины *DA* (4-14) в блок обработки порядка передается порядок чисел в формате ПЗ.

На магистраль *MD* с мультиплексора *MUXF* данные читаются в следующем виде:

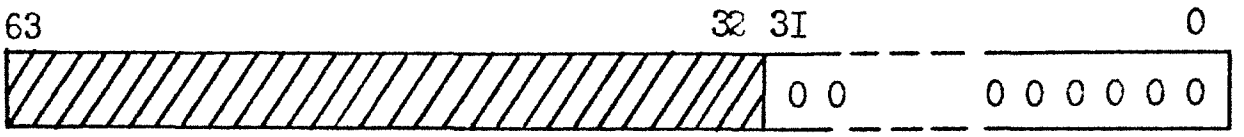
I. Байт



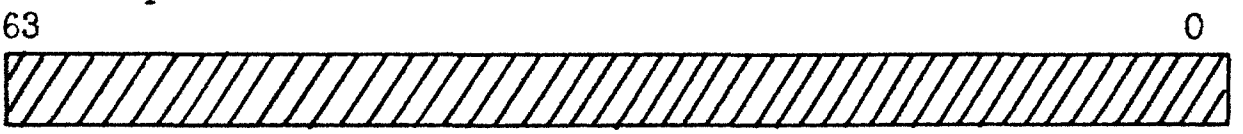
2. Слово



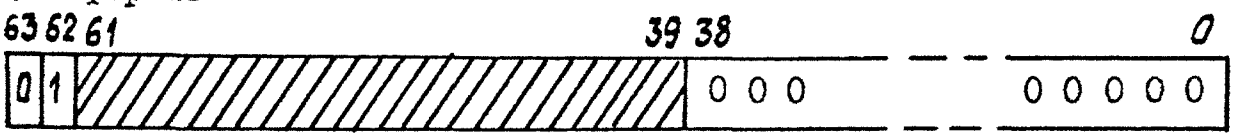
3. Двойное слово



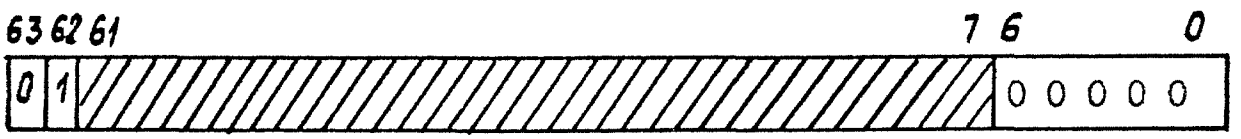
4. Четверное слово



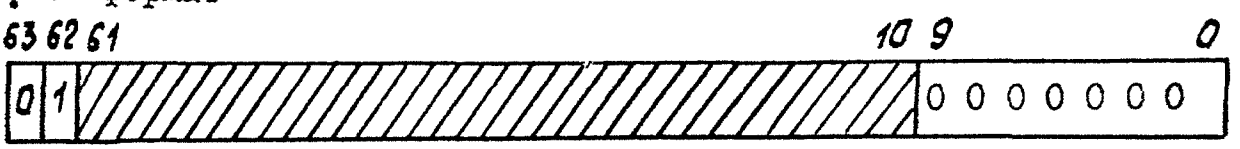
5. F-формат



6. D -формат



7. G -формат



Для хранения данных в некоторых командах используется регистр-аккумулятор (RAK). Данные в него могут записываться или с шины DA (0-31) или с магистрали MD (0-63). С шины DA (0-31) данные записываются в разряды аккумулятора 0-31. При этом в разрядах 32-63 записывается знак, т.е. 31 разряд. С магистрали MD данные записываются без модификации. Из аккумулятора данные могут читаться на магистраль MD .

Основные операции обработки данных выполняются в операционном блоке (рис. 5). В него входят три операционных устройства: блок деления (BDEL), арифметико-логическое устрой-

ство (ALU) и сдвигатель (SDV). Для хранения операндов и результатов операции используются регистры: регистр делителя (RDLT), регистр мантиссы A (RMA), регистр мантиссы B (RMB), регистр сдвигателя (REGS) и регистр результата (REGR). Все регистры доступны по записи с магистрали MD. При этом при записи данных с магистрали MD в регистр результата по адресу 010_2 запись происходит в оба регистра REGS и REGR.

Кроме того, при записи данных с магистрали MD в регистры RMA, RMB, RDLT по адресам соответственно 000_2, 001_2, 110_2 запись выполняется не только в адресуемый регистр, но и в регистры REGS и REGR. В регистр RMA запись с MD может производиться как прямо, так и со сдвигом вправо на один разряд.

Регистры RDLT, RMA, RMB, REGS могут принимать данные с выхода регистра REGR по шине QR(0–63). При этом RDLT принимает данные по этой шине со сдвигом на один разряд вправо.

При выполнении всех арифметических и логических операций регистры RMA и RMB используются как входные регистры ALU, а REGS – как выходной регистр. Запись результата операции ALU одновременно происходит и в регистр REGR. При выполнении деления регистры RMA и RMB подключаются к блоку деления.

В процессе выполнения деления промежуточные результаты поступают на RMA по шине FS(0–62), а на RMB – по шине FC (0–57). С регистра RMA данные поступают на BDEL по шине B (0–63), а с регистра RMB – по шине C (2–59). Результат деления поступает с блока деления на регистр RMA по шине QC (0–62) и на RMB – по шине QS(0–63). При выполнении операции сдвига регистр REGS служит входным регистром сдви-

гателя, а регистр REGR - выходным. Для выполнения сдвига и нормализации используются вспомогательные блоки: схема определения количества одинаковых цифр (SKOD) и дешифратор сдвига (DCSD). SKOD анализирует I6 старших цифр, без учета знака, на регистре REGS . Количество одинаковых цифр выдается пятиразрядным кодом на шину FA . При этом, если знак числа положительный, подсчитывается количество старших нулей, а если отрицательный - количество старших единиц. На схеме DCSD из кода FA при нормализации или параметра сдвига PSD(0-I0,I2) при сдвиге, формируется код сдвига CSD(0-3), который используется для управления сдвигателем.

Все операции по обработке порядка и вычислению параметра сдвига выполняются на блоке обработки порядка (рис. 7' Вычислительные операции выполняются на арифметико-логическом устройстве БОП (ALUP). При преобразовании порядка используются некоторые константы. Эти константы хранятся на блоке констант порядка (БКР). Для хранения данных используются регистры: регистр порядка I (RP1), регистр порядка II (RP2), регистр-аккумулятор порядка (RAP), регистр порядка результата (RPOR),регистр параметра сдвига (RPSD).

Устройства и регистры связаны между собой через систему мультиплексоров и шин .

Регистр RP1 доступен по записи:

с шины DA для загрузки порядка данных в формате ПЗ;

с шины MD для загрузки параметра сдвига в команде ASHQ ;

с блока констант;

с регистров RAP, RPOR, RPSD.

Регистр RP2 доступен по записи:

с шины DA ;

с блока констант;

с шины CSD;

с регистров RAP, RPOR.

Регистры RP1 и RP2 – это входные регистры для ALUP . Регистр RPSD является выходным регистром для ALUP . В него происходит запись в любом цикле ALUP . Регистры RAP и RPOR тоже доступны по записи с выхода ALUP . Кроме того, эти регистры доступны по записи с шины DA .

Константы БОП

Код												Назначение
II	I0	9	8	7	6	5	4	3	2	I	0	
0	0	0	0	I	0	0	0	0	I	I	I	Преобразование B ← F, D
0	I	0	0	0	0	0	0	0	I	I	I	Преобразование B ← G
0	0	0	0	I	0	0	0	I	I	I	I	Преобразование W ← F, D
0	I	0	0	0	0	0	0	I	I	I	I	Преобразование W ← G
0	0	0	0	I	0	0	I	I	I	I	I	Преобразование L ← F, D
0	I	0	0	0	0	0	I	I	I	I	I	Преобразование L ← G
0	0	0	0	I	0	0	0	0	0	0	0	Умножение/деление F, D
0	I	0	0	0	0	0	0	0	0	0	0	Умножение/деление G
0	0	0	0	0	0	0	0	I	0	0	0	Сдвиг делимого в команде DIVB
0	0	0	0	0	0	0	I	0	0	0	0	Сдвиг делимого в команде DIVW
0	0	0	0	0	0	I	0	0	0	0	0	Сдвиг делимого в команде DIVL
0	0	0	0	0	0	0	I	I	0	0	0	Сдвиг множимого в команде MULB
0	0	0	0	0	0	0	I	0	0	0	0	Сдвиг множимого в команде MULW

Операция умножения выполняется на блоке умножения (рис. 6). Для приема операндов используются регистры: регистр множимого (RMNM) и регистр множителя (RMNT). С этих регистров на умножитель (MNTL) подается информация или со старших 32 разрядов или с младших 32 разрядов. Кроме этих регистров в блоке имеется еще два регистра утроенного множителя: RT1 для старшей части и RT2 для младшей части. Через мультиплексор информация с этих регистров также поступает на MNTL. Так как операция умножения симметрична, то умножитель MNTL использует содержимое регистра RMNT в качестве множимого, регистров RT1,RT2 в качестве утроенного множимого, а RMNM в качестве множителя. Такое несоответствие между названием регистров и их функцией в процессе умножения объясняется тем, что по системе команд первый операнд команды называется множителем, второй – множимым.

Все регистры загружаются с шины MD. На RMNM информация записывается или прямо, или со сдвигом на I разряд влево. Результат умножения выдается на шину MD.

Завершаются все команды формированием результата и его выдачей через устройство выдачи данных (рис. 4). Результат поступает с шины MD на демультиплексор форматов (DMF) и с него на выходной буфер данных (BUFO). Далее данные попадают на внутреннюю шину данных D(0-3I) и далее через магистральные элементы (ME) на внешнюю магистраль AD. При выполнении команды может возникнуть фатальная ошибка. В этом случае данные не выдаются. Вид ошибки определяется ее кодом. Этот код формируется на специальном устройстве (COSH). На это устройство поступают признаки ошибок (POSH) и код ошибки выдается на магистраль D(0-3I).

Кроме данных в каждой команде формируются состояния, а в

команде ACB – признак ветвления. Это выполняется на устройстве формирования состояний и ветвлений (FSV). На это устройство поступают признаки ветвления и состояний (PVS). Признак ветвления выдается на магистраль D(0-31) в нулевой разряд, признак состояния – на внешние выводы ACZ, ACN,ACV. На вывод ACC поступает сигнал фатальной ошибки в случае ее возникновения.

Коды ошибок

AD					Наименование
04	03	02	01	00	
0	0	0	0	1	Резервный операнд
0	0	0	1	0	Целое переполнение
0	0	1	0	0	Целое деление на ноль
1	0	0	0	0	Переполнение ПЗ
1	0	0	1	0	Деление на ноль ПЗ
1	0	1	0	0	Антипереполнение ПЗ

На вывод ACN поступает бит состояния " N ", на вывод ACZ – бит состояния " Z ". Код на выводах ACV и ACC является управляющим для процессора. Если ACV=0 и ACC=0 , то это значит, что арифметической ошибки нет. Если ACV=0 и ACC = 1 , то это значит, что возникла фатальная ошибка и процессор воспринимает этот код как сигнал фатального прерывания. Если ACV=1 , то независимо от состояния ACC фатального

прерывания не возникает.

ACV	ACC	
0	0	- нет ошибки
0	I	- резервный операнд, - деление на ноль ПЗ, - переполнение ПЗ, - антипереполнение
I	0	- целое переполнение
I	I	- целое деление на ноль.

4. ИНТЕРФЕЙСНЫЙ БЛОК

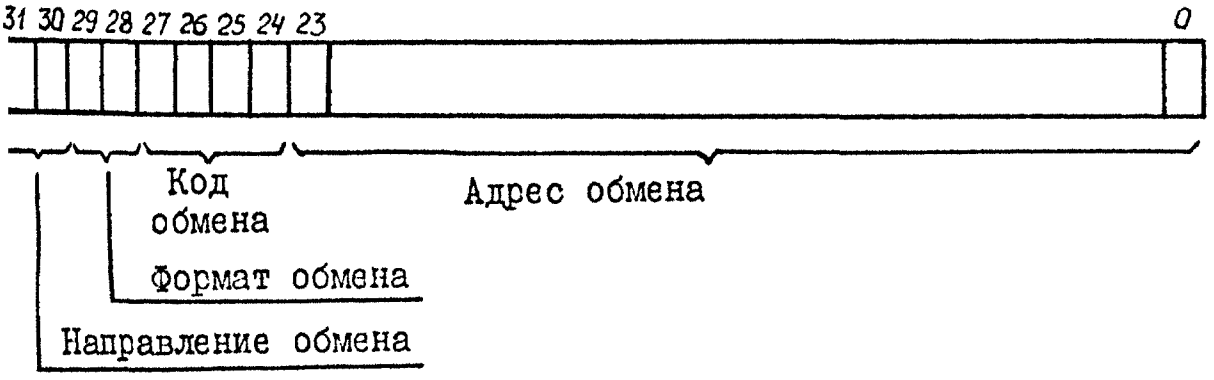
4.I. Структура интерфейсного блока

Интерфейсный блок обслуживает обмены на магистрали AD , относящиеся к СПРЦ, обеспечивая при этом прием и выдачу данных. Структурная схема интерфейсного блока представлена на рис. 9 .

Выделение из общего потока обменов обмена, относящегося к сопроцессору, осуществляется интерфейсным блоком по адресу на магистрали AD .

Ниже приведен формат адреса, выставляемого процессором на магистрали AD.

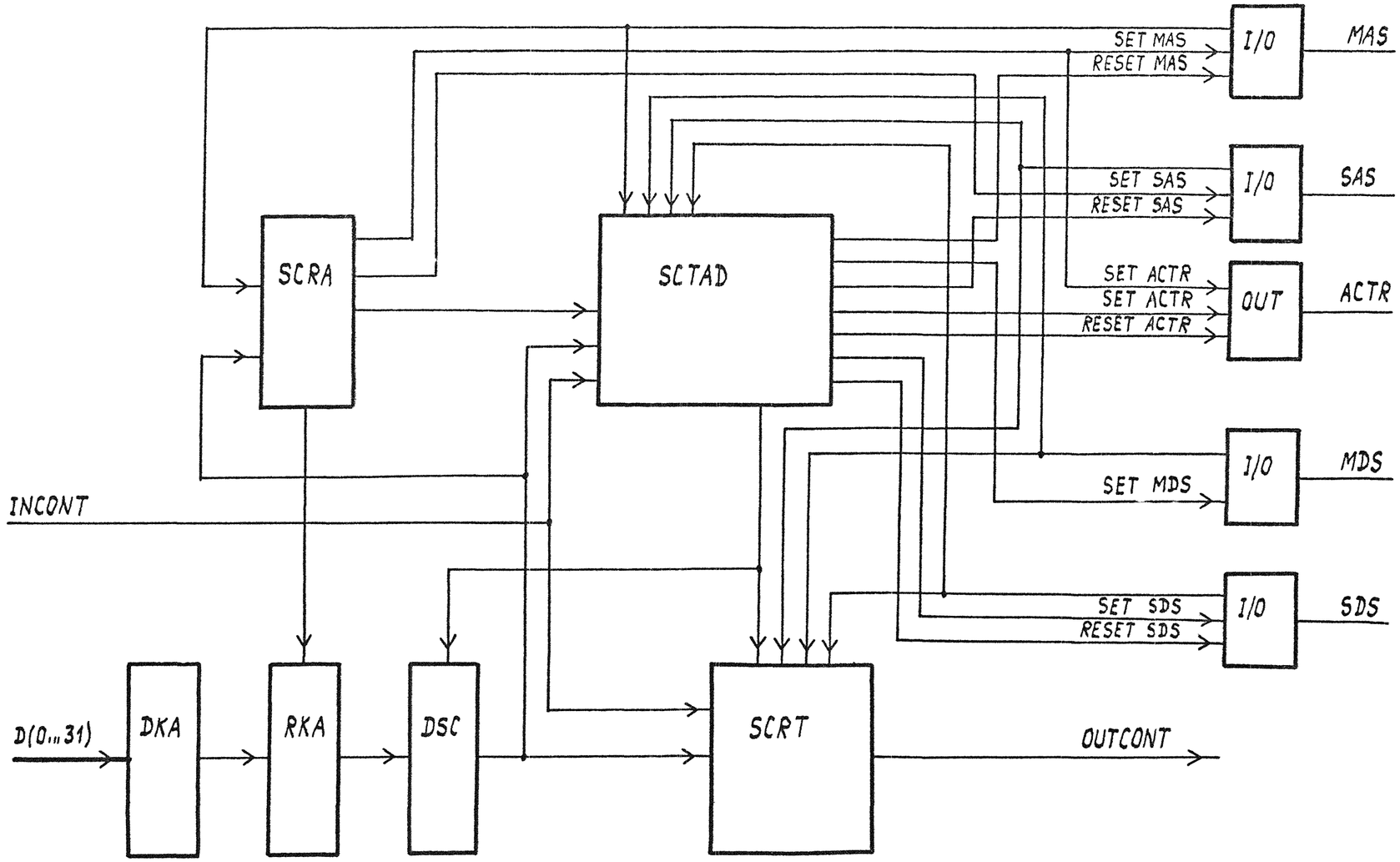
Формат представлен в прямом коде (магистраль AD – инверсная магистраль)



Поле – Направление обмена

AD3I	AD30	Тип обмена
0	0	Обмен с основной памятью
0	I	Обмен с основной памятью с перехватом обмена СПРЦ
I	0	Обмен с системными регистрами
I	I	Обмен с системной памятью

Структурная схема интерфейсного блока



DKA – дешифратор кода обмена и адреса
RKA – регистр кода обмена и адреса
DSC – дешифратор сигналов управления
SCRT – схема управления приемом и выдачей данных
SCRA – схема управления приемом адреса
SCTAD – схема управления обменом по внешней шине AD
I/O – элемент входа/выхода
OUT – элемент выхода

Рис. 9.

Поле – формат обмена

AD29	AD28	Тип формата обмена
0	0	Двойное слово
0	I	Слово
I	0	Четверное слово
I	I	Байт

Поле – код обмена

AD27	AD25	AD24	Тип обмена
0	0	I	Чтение данных
0	I	0	Запись данных
0	I	I	Чтение-модификация-запись
I	0	I	Чтение команды

Разряд AD26 в адресной части не используется.

К обменам с сопроцессором, как было сказано выше, относятся:

I) Чтение с перехватом данных в форматах байт, слово, двойное слово, четверное слово

```
A                                   A
D                                   D
3                                   0
I
0III 000I XXXX XXXX XXXX XXXX XXXX XXXX
0I0I 000I XXXX XXXX XXXX XXXX XXXX XXXX
0I00 000I XXXX XXXX XXXX XXXX XXXX XXXX
0II0 000I XXXX XXXX XXXX XXXX XXXX XXXX
```

2) Запись с перехватом данных в форматах байт, слово, двойное слово, четверное слово

```
A                                                   A
D                                                   D
3                                                   0
I
OIII 00IO XXXX XXXX XXXX XXXX XXXX XXXX
OIOI 00IO XXXX XXXX XXXX XXXX XXXX XXXX
OIOO 00IO XXXX XXXX XXXX XXXX XXXX XXXX
OIIO 00IO XXXX XXXX XXXX XXXX XXXX XXXX
```

3) Чтение-модификация-запись с перехватом данных в форматах байт, слово, двойное слово

```
A                                                   A
D                                                   D
3                                                   0
I
OIII 00II XXXX XXXX XXXX XXXX XXXX XXXX
OIOI 00II XXXX XXXX XXXX XXXX XXXX XXXX
OIOO 00II XXXX XXXX XXXX XXXX XXXX XXXX
```

4) Чтение данных в форматах байт, слово, двойное слово, четверное слово из системной памяти по адресу ноль (выходной буфер операндов СПРЦ)

```
A                                                   A
D                                                   D
3                                                   0
I
IIII 000I 0000 0000 0000 0000 0000 0000
IIOI 000I 0000 0000 0000 0000 0000 0000
IIOO 000I 0000 0000 0000 0000 0000 0000
IIIO 000I 0000 0000 0000 0000 0000 0000
```

5) Запись данных в форматах байт, слово, двойное слово, четверное слово в системную память по адресу ноль (входной буфер операндов СПРЦ)

```
A                                                   A
D                                                   D
3                                                   0
I
IIII 00IO 0000 0000 0000 0000 0000 0000
IIOI 00IO 0000 0000 0000 0000 0000 0000
IIOO 00IO 0000 0000 0000 0000 0000 0000
IIIO 00IO 0000 0000 0000 0000 0000 0000
```

6) Чтение данных в форматах двойное слово из системной памяти по адресу один и два (регистр ошибки и регистр ветвления СПРЦ соответственно)

A D 3 I	A D 0
II00 000I 0000 0000 0000 0000 0000 0000	
II00 000I 0000 0000 0000 0000 0000 0000	

В исходном состоянии, т.е. в случае, если интерфейсный блок не занят обслуживанием обмена сопроцессора, благодаря сигналам управления, вырабатываемым схемой управления приемом и выдачей данных (SCRT), элементы входа/выхода AD сопроцессора установлены на прием. Следовательно, информация с магистрали AD через шину D(0-3I) поступает на дешифратор кода обмена и адреса (DKA), при этом регистр кода обмена и адреса (RKA) открыт на прием.

При появлении интерфейсного сигнала от процессора MAS , сопровождающего адрес по магистрали AD, запускается схема управления приемом адреса (SCRA). Адрес по шине D(0-3I) поступает на дешифратор DKA.

DKA выделяет направление обмена, формат обмена и код обмена. При наличии кода обмена с системной памятью анализируется адрес. Если разряды адреса с 4 по 23 равны нулю, то DKA выделяет признак обмена с системной памятью. Если разряды адреса с 0 по 3 равны 000I, то выделяется признак обмена с регистром ошибки. Если разряды с 0 по 3 равны 00I0, то выделяется признак обмена с регистром ветвления. Если разряды адреса с 8 по II содержат хотя бы одну I, то выделяется признак сброса БМУ. Признаки-характеристики обмена с выхода DKA записываются в регистр RKA. Через период после появления сигнала MAS схема SCRA закрывает запись в регистр RKA.

Таким образом в регистре RKA могут храниться следующие

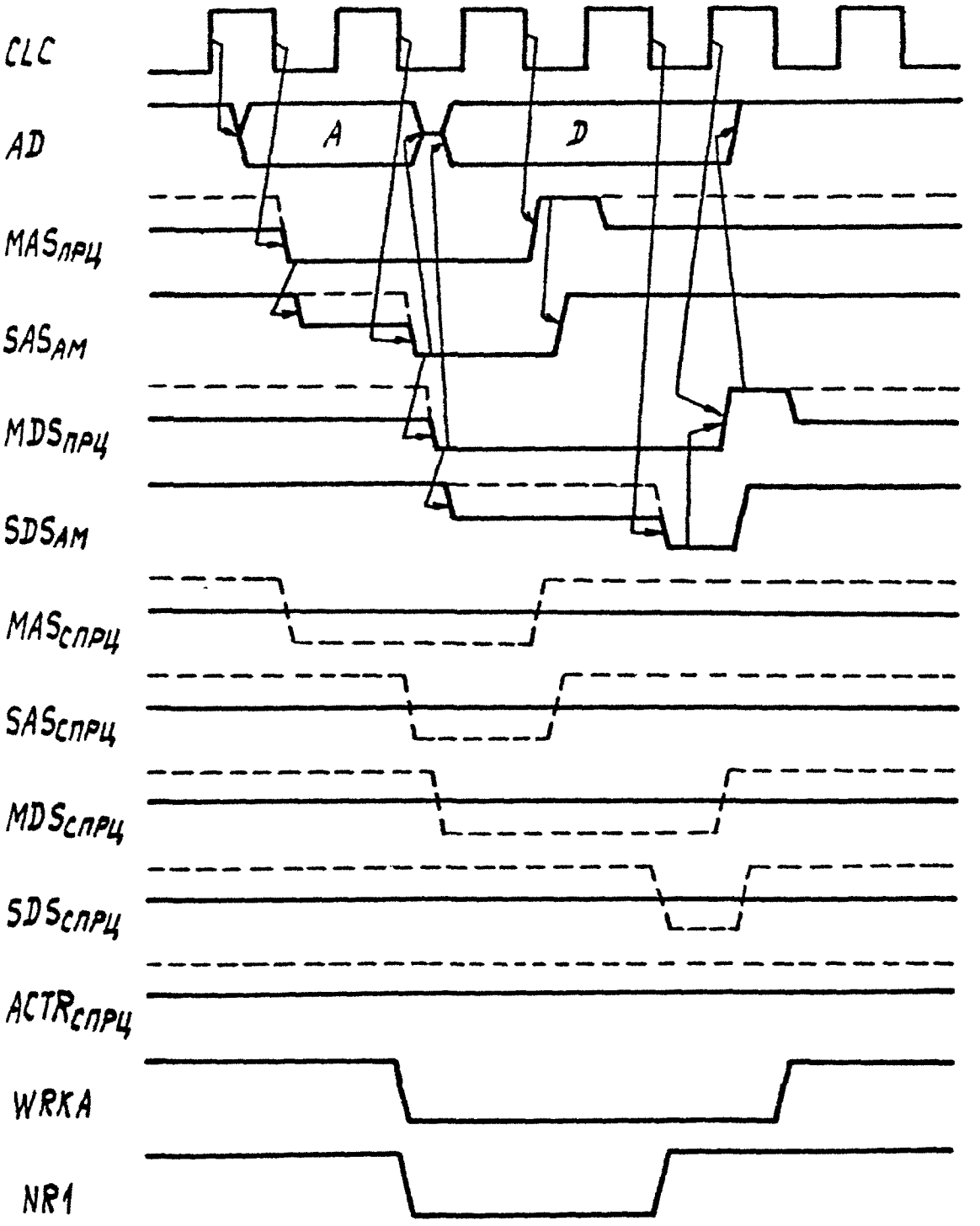
признаки: два разряда формата обмена; признаки чтения, записи, обмена с четверным словом, обмена с перехватом, обмена с системной памятью, расположенной в сопроцессоре, а также признаки регистров ошибок и ветвления и сброса БМУ.

Информация с РКА поступает на дешифратор управляющих сигналов (DSC), выходные сигналы которого обеспечивают нормальную работу схем синхронизации интерфейсного блока. Если на выходе DSC отсутствуют сигналы обмена с системной памятью или обмена с перехватом, то работа SCRA после записи в регистр РКА тормозится, при этом схема управления обменом по магистрали AD (SCTAD), а следовательно и схема управления приемом и выдачей данных (SCRT) не запускаются. Схема SCRA останется в заторможенном состоянии до конца текущего обмена, сброс ее произойдет снятием сигнала MAS . Если в регистре РКА будет взведен признак сброса БМУ, то по окончании записи в РКА интерфейсный блок выработает сигнал сброса БМУ NRI. Длительность сигнала NRI равна длительности сигнала MAS . Временная диаграмма представлена на рис.10.

В случае появления на выходе DSC признака обмена с перехватом или обмена с системной памятью и при наличии сигнала SAS SCRA запускает SCTAD , который в свою очередь совместно с входными управляющими сигналами INCONT управляют работой SCRT.

На временных диаграммах сплошной линией указаны потенциалы на выводах микросхем, как если бы они были отсоединены от цепей, связывающих одноименные выводы друг с другом. Сплошная средняя линия указывает на то, что конкретный вывод в данный момент времени находится в отключенном состоянии. Пунктиром указаны потенциалы на цепях, объединяющих одноименные сигналы. Механизм работы интерфейсных сигналов требует, чтобы на цепи, содержащие между собой выводы микросхем MAS, SAS , MDS , SDS , ACTR, PGB, PCRQ, SEL, DMAK, было подано напряжение питания через резисторы.

Временная диаграмма формирования сигнала NRI СПРЦ при чтении регистра SCBB, находящегося в системной памяти по адресу 300(I6)



SAS_{AM}, SDS_{AM} – сигналы микросхемы "Адаптер магистралей"

Рис. I0.

Минимальное значение резистора 2,7 кОм, мощность 0,125 Вт.

На временных диаграммах наряду с внешними сигналами представлены внутренние сигналы сопроцессора:

- WRKA – сигнал записи в регистр кода обмена и адреса интерфейсного блока;
- WBPR – сигнал записи в выходной буфер операндов сопроцессора;
- ZPB – сигнал записи во входной буфер операндов сопроцессора;
- DOUT – сигнал переключения выходных элементов AD сопроцессора из третьего состояния в состояние выдачи данных;
- NRN – импульс сброса признаков;
- NRI – импульс сброса БМУ сопроцессора.

4.2. Адресное чтение с перехватом данных в формате байт, слово, двойное слово

Эта процедура используется в случае, если данные в сопроцессор в формате байт, слово или двойное слово требуется принять из памяти. Адресную часть обмена выполняет процессор, после чего обмен перехватывает сопроцессор, который и принимает данные. Временная диаграмма обмена представлена на рис. IL

В фазе высокого уровня сигнала тактовой частоты CLC процессор выставляет на магистраль AD адрес, после чего в фазе низкого уровня сигнал CLC выставляет низким уровнем сигнал сопровождения адреса $MAS_{ПРЦ}$. Сигнал MAS переводит выходной элемент микросхемы "контроллер памяти" (КП) SAS в третье (отключенное) состояние. Высокий уровень на цепи, объединяющей выводы SAS микросхем, после того как вывод $SAS_{КП}$ переключится в третье состояние, поддерживается подачей питания через резистор на эту цепь. После появления сигнала $MAS_{ПРЦ}$ контроллер памяти и сопроцессор в течение периода тактовой частоты занимаются дешифрацией адреса, в результате которой в фазе низкого уровня CLC контроллер памяти выставляет низким уровнем сигнал подтверждения приема адреса $SAS_{КП}$, а сопроцессор низким уровнем выставляет свой сигнал $MAS_{СПРЦ}$ и сигнал перехвата $ACTR_{СПРЦ}$. По сигналу $SAS_{КП}$ процессор безусловно снимает с магистрали AD адрес и переводит свои элементы входа/выхода AD в третье состояние, сопроцессор же выставляет низким уровнем сигнал готовности к приему данных $MDS_{СПРЦ}$. Если процессор не успеет принять сигнал $SAS_{КП}$ в той же фазе низкого уровня CLC, что и выдавался сигнал $SAS_{КП}$, то сигнал $MDS_{СПРЦ}$ будет выдан в следующей фазе низкого уровня CLC, т.е. с задержкой в пе-

Временная диаграмма адресного чтения с перехватом данных в формате байт, слово, двойное слово

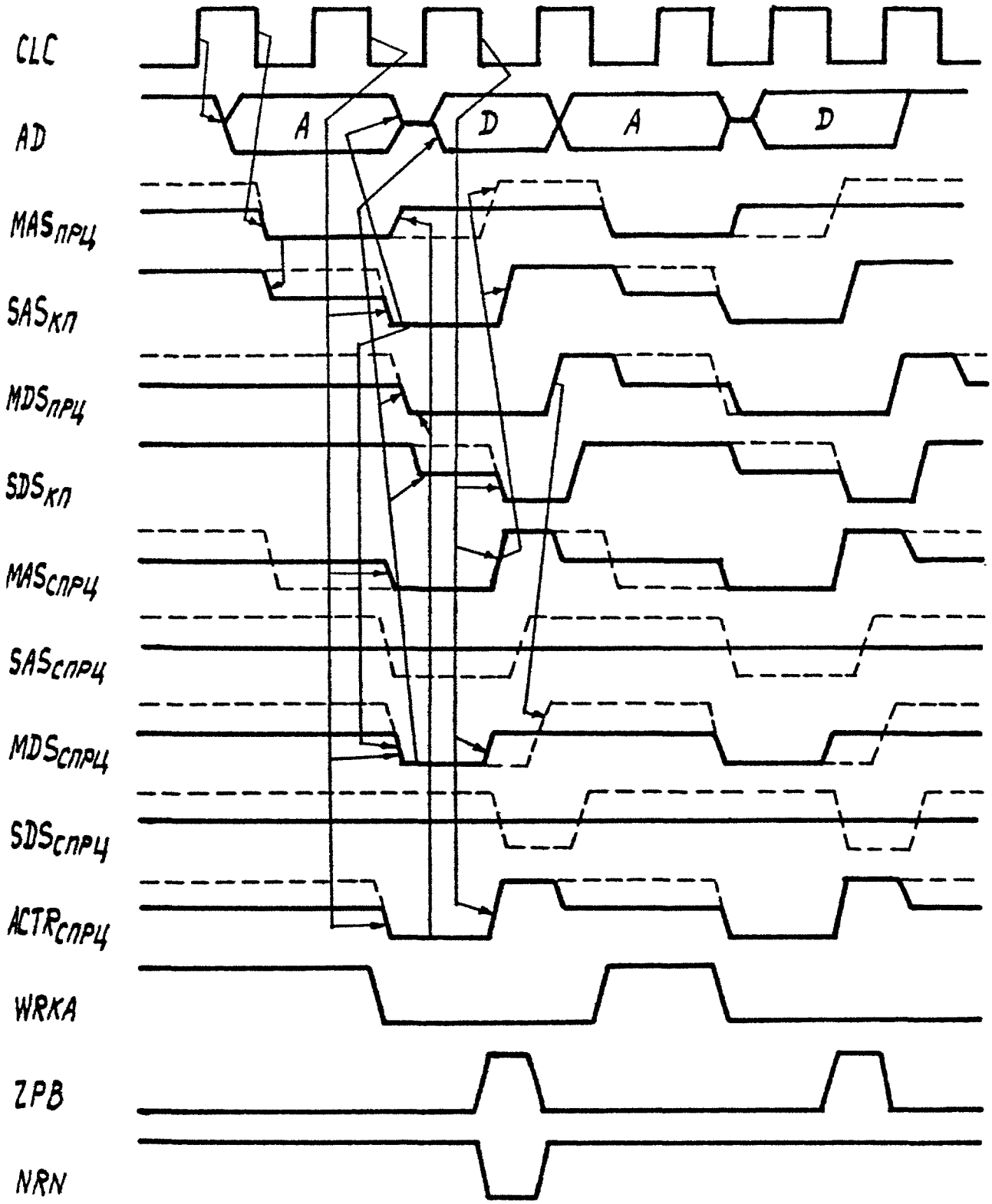


Рис. II.

риод тактовой частоты. По сигналу $ACTR_{СПРЦ}$ процессор переводит в третье состояние выходной элемент $MAS_{ПРЦ}$, а при наличии сигнала $MDS_{СПРЦ}$ выставляет низким уровнем свой сигнал $MDS_{ПРЦ}$. Наличие низкого уровня на цепи, объединяющей выводы MDS микросхем, приводит к тому, что выходной элемент SDS контроллера памяти переводится в третье состояние.

Высокий уровень на цепи, объединяющей выводы SDS микросхем, после того как выходной элемент $SDS_{КП}$ переключается в третье состояние, поддерживается подачей питания через резистор на нее. Кроме этого, выходные элементы AD контроллера памяти переводятся из третьего состояния в состояние выдачи данных. Через период после выдачи сигнала $MDS_{СПРЦ}$ в фазе низкого уровня сигнала CLC сопроцессор переводит в третье состояние сигнал $MDS_{СПРЦ}$ и снимает (устанавливает высоким уровнем) сигналы $ACTR_{СПРЦ}$ и $MAS_{СПРЦ}$. При этом сопроцессор активно поддерживает высокий уровень сигналов $MAS_{СПРЦ}$ и $ACTR_{СПРЦ}$ только лишь в течение полупериода CLC, после чего переводит их в третье состояние. Высокий уровень на цепях, объединяющих выводы микросхем MAS и $ACTR$ поддерживается подачей на них питания через резисторы. Как только на цепи, объединяющей выводы микросхем MAS, устанавливается высокий уровень, контроллер памяти снимает сигнал $SAS_{КП}$ (устанавливает высоким уровнем). В этой же фазе низкого уровня сигнала CLC, если готовы данные, контроллер памяти выставляет низким уровнем сигнал $SDS_{КП}$, в противном случае сигнал SDS будет выставлен позже по мере готовности данных в фазе низкого уровня сигнала CLC. Через период, после выдачи сигнала $MDS_{СПРЦ}$ интерфейсный блок сопроцессора начинает вырабатывать импульсы записи данных во входной буфер операндов. Эти импульсы формиру-

ются в фазе низкого уровня сигнала CLC и кратны полупериоду тактовой частоты. Последний импульс формируется на фоне прихода сигнала $SDS_{КП}$. Т.е., если сигнал SDS появился через период после выставления сопроцессором сигнала $MDS_{СПРЦ}$, то интерфейсный блок формирует один импульс записи во входной буфер операндов, если через два периода, то два импульса записи и т.д. Достоверные данные принимаются по последнему импульсу записи во входной буфер операндов. Кроме этого, если чтение с перехватом является первым обменом команды, то через период CLC после выдачи сигнала $MDS_{СПРЦ}$ интерфейсный блок формирует полупериодный импульс сброса NRN некоторых признаков СПРЦ. К примеру, если в предшествующей команде был взведен признак целого переполнения, а бита разрешения на прерывание по целому переполнению в регистре состояния ЦПР нет, то до начала выполнения следующей команды этот признак необходимо сбросить. Его сброс произойдет по сигналу NRN.

Истинные данные на магистрали AD от контроллера памяти должны появиться не позднее окончания первой половины полупериода фазы низкого уровня сигнала CLC, в которой выдается сигнал $SDS_{КП}$ при работе на максимальной тактовой частоте (10 МГц). Если центральный процессор успеет принять сигнал $SDS_{КП}$ в той же фазе низкого уровня сигнала CLC, что и выдавался сигнал $SDS_{КП}$, то центральный процессор в фазе высокого уровня сигнала CLC снимет (установит высоким уровнем) сигнал $MDS_{ПРЦ}$, в противном случае сигнал $MDS_{ПРЦ}$ снимется через период CLC. Высокий уровень на выводе $MDS_{ПРЦ}$ будет поддерживаться процессором активно в течение полупериода тактовой частоты, после чего вывод $MDS_{ПРЦ}$ переводится в третье состояние, при этом высокий уровень на цепи, объединяющей выводы MDS микросхем, будет поддерживаться подачей на нее питания через резистор. Как

только на цепи, объединяющей выводы MDS микросхем, установится высокий уровень, так сразу контроллер памяти снимет (установит высоким уровнем) сигнал $SDS_{КЛ}$ и переведет свои выводы AD в третье состояние, при этом выводы AD процессора переводятся в активное состояние (выдается следующий адрес или активно устанавливается высокий уровень). На этом выполнение данной процедуры заканчивается.

4.3. Адресная запись с перехватом данных в формате байт, слово, двойное слово

Эта процедура используется в случае, если данные в формате байт, слово, двойное слово требуется записать в память из сопроцессора. Адресную часть обмена выполняет центральный процессор, после чего обмен перехватывает сопроцессор, который и выдает данные. Временная диаграмма обмена представлена на рис. I2.

В фазе высокого уровня сигнала CLC центральный процессор выставляет на магистраль AD адрес, после чего в фазе низкого уровня сигнала CLC выставляет низким уровнем сигнал сопровождения адреса $MAS_{ПРЦ}$. Низкий уровень на цепи, объединяющей выводы MAS микросхем, переводит выходной элемент контроллера памяти SAS в третье состояние, при этом высокий уровень на цепи, объединяющей выводы SAS микросхем, поддерживается подачей на нее питания через резистор. После появления сигнала $MAS_{ПРЦ}$ контроллер памяти и сопроцессор в течение периода тактовой частоты занимаются дешифрацией адреса, в результате которой в фазе низкого уровня CLC контроллер памяти выставляет низким уровнем сигнал подтверждения приема адреса SAS, а сопроцессор низким уровнем выставляет свой сигнал $MAS_{СПРЦ}$ и сигнал подтверждения перехвата $ACTR_{СПРЦ}$. По сигналу $SAS_{КЛ}$ центральный процессор безусловно снимает с магистрали AD адрес и перево -

Временная диаграмма адресной записи с перехватом данных в формате байт, слово, двойное слово

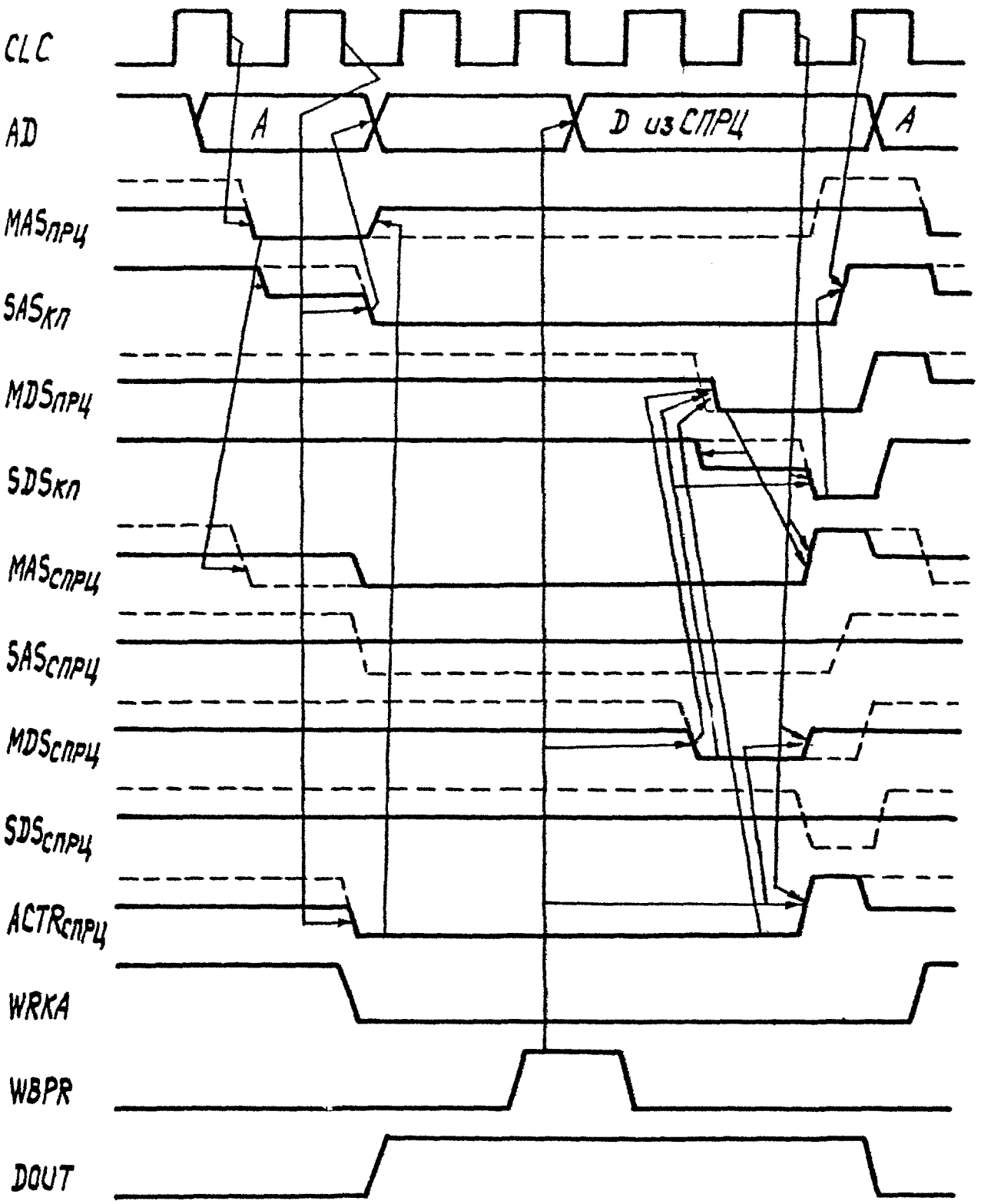


Рис. I2.

дит свои элементы входа/выхода AD в третье состояние, сопроцессор же переводит свои элементы AD на выдачу содержимого выходного буфера операндов. По сигналу $ACTR_{СПРЦ}$ центральный процессор переводит в третье состояние выходной элемент $MAS_{ПРЦ}$.

Если к моменту прихода сигнала $SAS_{КП}$, сопроцессор уже произвел запись результата в выходной буфер операндов, то через период тактовой частоты в фазе низкого уровня CLC сопроцессор выставит низким уровнем сигнал $MDS_{СПРЦ}$. В противном случае сопроцессор выставит сигнал $MDS_{СПРЦ}$ в фазе низкого уровня CLC через полтора периода тактовой частоты относительно импульса записи в выходной буфер операндов. Истинные данные появятся на магистрали AD за период до появления сигнала сопровождения данных $MDS_{СПРЦ}$.

При наличии сигналов $MDS_{СПРЦ}$ и $ACTR_{СПРЦ}$ центральный процессор выставляет свой сигнал $MDS_{ПРЦ}$. Наличие низкого уровня на цепи, объединяющей выводы MDS микросхем, приводит к тому, что выходной элемент SDS контроллера памяти переводится в третье состояние, при этом высокий уровень на цепи, объединяющей выводы SDS микросхем, поддерживается подачей на нее питания через резистор.

Через период тактовой частоты после выдачи сигнала $MDS_{СПРЦ}$ в фазе низкого уровня сигнала CLC сопроцессор переводит в третье состояние вывод $MDS_{СПРЦ}$ и снимает (устанавливает высоким уровнем) сигналы $ACTR_{СПРЦ}$ и $MAS_{СПРЦ}$. При этом сопроцессор активно поддерживает высокий уровень сигналов $MAS_{СПРЦ}$ и $ACTR_{СПРЦ}$ только лишь в течение полупериода тактовой частоты, после чего переводит их в третье состояние. Высокий уровень на цепях, объединяющих выводы микросхем MAS и $ACTR$, после того как они переключились в третье состояние, поддерживается

подачей на них питания через резисторы. Как только на цепи, объединяющей выводы микросхем MAS, устанавливается высокий уровень, контроллер памяти снимает сигнал $SAS_{КП}$ (устанавливает высоким уровнем). В этой же фазе низкого уровня сигнала CLC, если данные записаны в память, контроллер памяти выставляет низким уровнем сигнал подтверждения приема данных $SDS_{КП}$, в противном случае сигнал $SDS_{КП}$ будет выставлен позже по мере записи данных в фазе низкого уровня CLC.

Если сигнал $SDS_{КП}$ будет принят на центральный процессор в той же фазе низкого уровня CLC, что и выдавался, то центральный процессор в фазе высокого уровня сигнала CLC снимет (установит высоким уровнем) сигнал $MDS_{ПРЦ}$, в противном случае сигнал $MDS_{ПРЦ}$ снимется с задержкой в период CLC. Высокий уровень на выводе $MDS_{ПРЦ}$ будет поддерживаться процессором активно в течение полупериода тактовой частоты, после чего вывод $MDS_{ПРЦ}$ переводится в третье состояние, а высокий уровень на цепи, объединяющей выводы MDS микросхем, будет поддерживаться подачей на нее питания через резистор. Как только на цепи, объединяющей выводы микросхем MDS, установится высокий уровень, так сразу контроллер памяти снимет (установит высоким уровнем) сигнал $SDS_{КП}$, сопроцессор переведет свои выводы AD в третье состояние, а процессор переведет свои выводы AD в активное состояние (выдается следующий адрес или активно устанавливается высокий уровень). На этом выполнение данной процедуры заканчивается.

4.4. Адресное чтение-модификация-запись с перехватом данных в формате байт, слово, двойное слово

Эта процедура используется в случае, если данные в формате байт, слово, двойное слово из ячейки памяти необходимо прочитать на сопроцессор, модифицировать и записать из сопроцессора в ту

же ячейку памяти. Адресную часть обмена выполняет центральный процессор, после чего сопроцессор осуществляет прием и выдачу данных. Временная диаграмма обмена представлена на рис.I3.

Фаза чтения данной процедуры идентична процедуре адресного чтения с перехватом данных в формате байт, слово, двойное слово, исключение составляет лишь то, что в фазе чтения не снимается сигнал $MAS_{СПРЦ}$, а следовательно и $SAS_{КП}$. Снятие этих сигналов происходит в фазе записи. По снятию сигнала $MDS_{ПРЦ}$ (установка высоким уровнем) в фазе чтения, выходные элементы AD процессора остаются в третьем состоянии, контроллера памяти переключаются в третье состояние, а сопроцессора переключаются на выдачу содержимого выходного буфера операндов на магистраль AD, т.е. кончается фаза чтения и начинается фаза записи данной процедуры. Через I/2 периода тактовой частоты относительно снятия сигнала $MDS_{ПРЦ}$ в фазе низкого уровня CLC сопроцессор устанавливает низким уровнем сигнал подтверждения перехвата $ACTR_{СПРЦ}$ в фазе записи, после чего интерфейсный блок сопроцессора переходит в ожидание сигнала записи результата в выходной буфер операндов.

Минимальное время между снятием сигнала $MDS_{ПРЦ}$ в фазе чтения и сигналом записи в выходной буфер операндов результата сопроцессора в фазе записи составляет 2Т тактовой частоты. После появления сигнала записи в выходной буфер операндов результата сопроцессора выполнения фазы записи данной процедуры происходит идентично окончанию процедуры адресной записи с перехватом данных в формате байт, слово, двойное слово.

4.5. Адресная запись с перехватом данных в формате четверное слово

Эта процедура используется в случае, если в память из сопроцессора требуется записать данные в " D " или " G " плава -

Временная диаграмма чтения-модификации-записи с перехватом данных в формате байт, слово, двойное слово

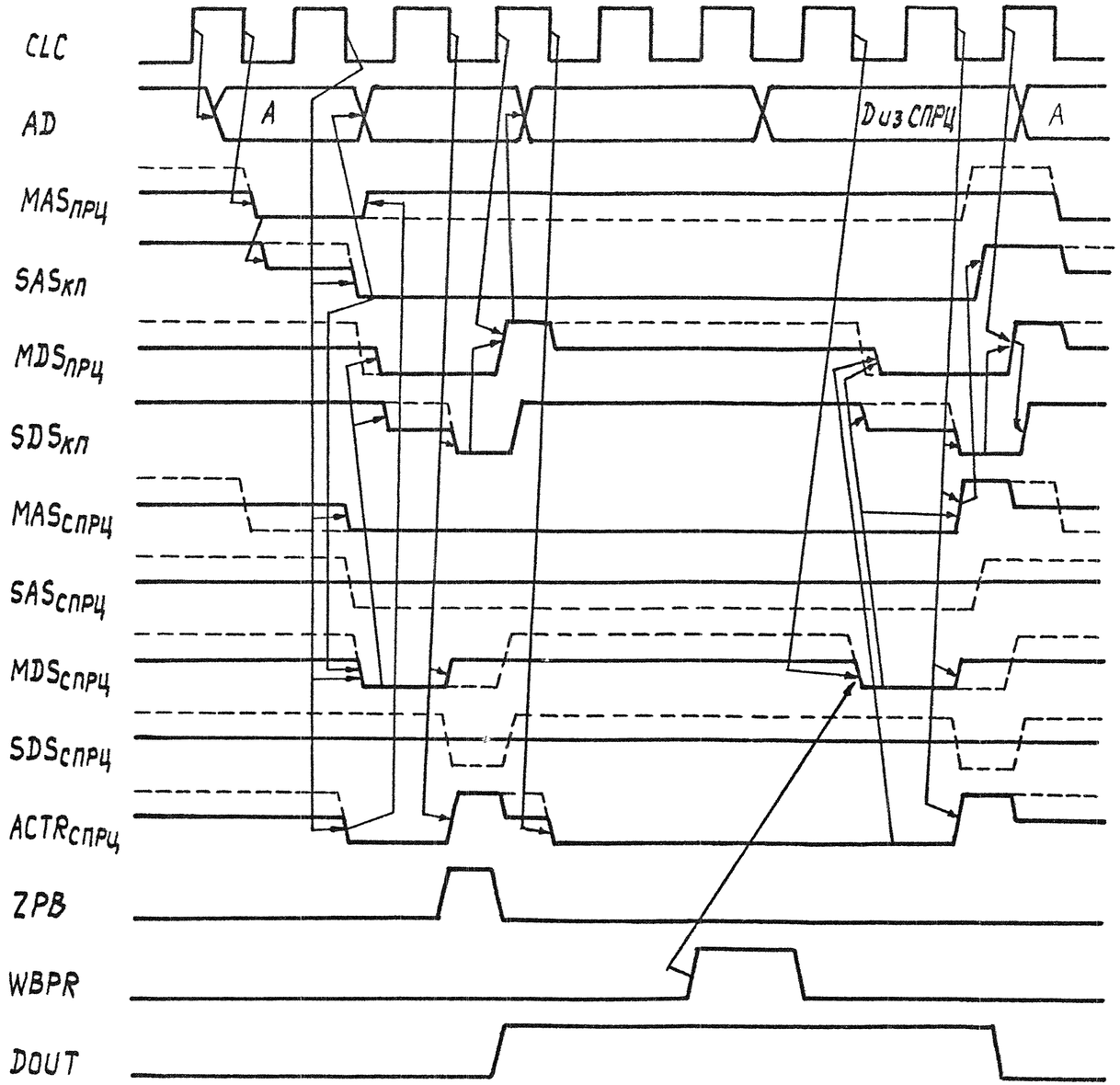


Рис. I3.

ющем формате, а также четверное слово целочисленного формата. Адресную часть обмена выполняет центральный процессор, после чего обмен перехватывает сопроцессор, который и выдает данные. Временная диаграмма обмена представлена на рис. I4.

Первая половина данного обмена, а именно запись младших 32 разрядов 64-разрядного операнда, выполняется идентично процедуре адресной записи с перехватом данных в формате двойное слово. Исключение составляет то, что сигнал $MAS_{СПРЦ}$, а следовательно, и сигнал $SAS_{КП}$, не снимаются на фоне записи младшей половины результата. Кроме этого, после снятия сигнала сопровождения младшей половины результата $MDS_{ПРЦ}$ выходные элементы AD процессора и сопроцессора не изменяют своих состояний (элементы AD процессора - в третьем состоянии, элементы AD сопроцессора в активном состоянии), при этом на выходные элементы AD сопроцессора будут поданы разряды старшей половины результата.

Через полпериода тактовой частоты относительно снятия сигнала $MDS_{ПРЦ}$, сопровождающего запись младших разрядов результата в память, в фазе низкого уровня сигнала CLC сопроцессор выставляет низким уровнем сигнал перехвата при записи старшей половины результата $ACTR_{СПРЦ}$. Еще через полпериода тактовой частоты в фазе высокого уровня сигнала CLC сопроцессор выдает низким уровнем сигнал сопровождения старшей половины результата $MDS_{СПРЦ}$. Истинное значение старшей половины результата появляется на магистрали AD за период тактовой частоты до появления сигнала $MDS_{СПРЦ}$. Наличие низкого уровня на цепи, объединяющей выводы MDS микросхем, приводит к тому, что выходной элемент SDS контроллера памяти переводится в третье состояние. Высокий уровень на цепи, объединяющей выводы микросхем SDS, после того как вывод $SDS_{КП}$ перешел в третье

Временная диаграмма адресной записи с перехватом данных
в формате четверное слово

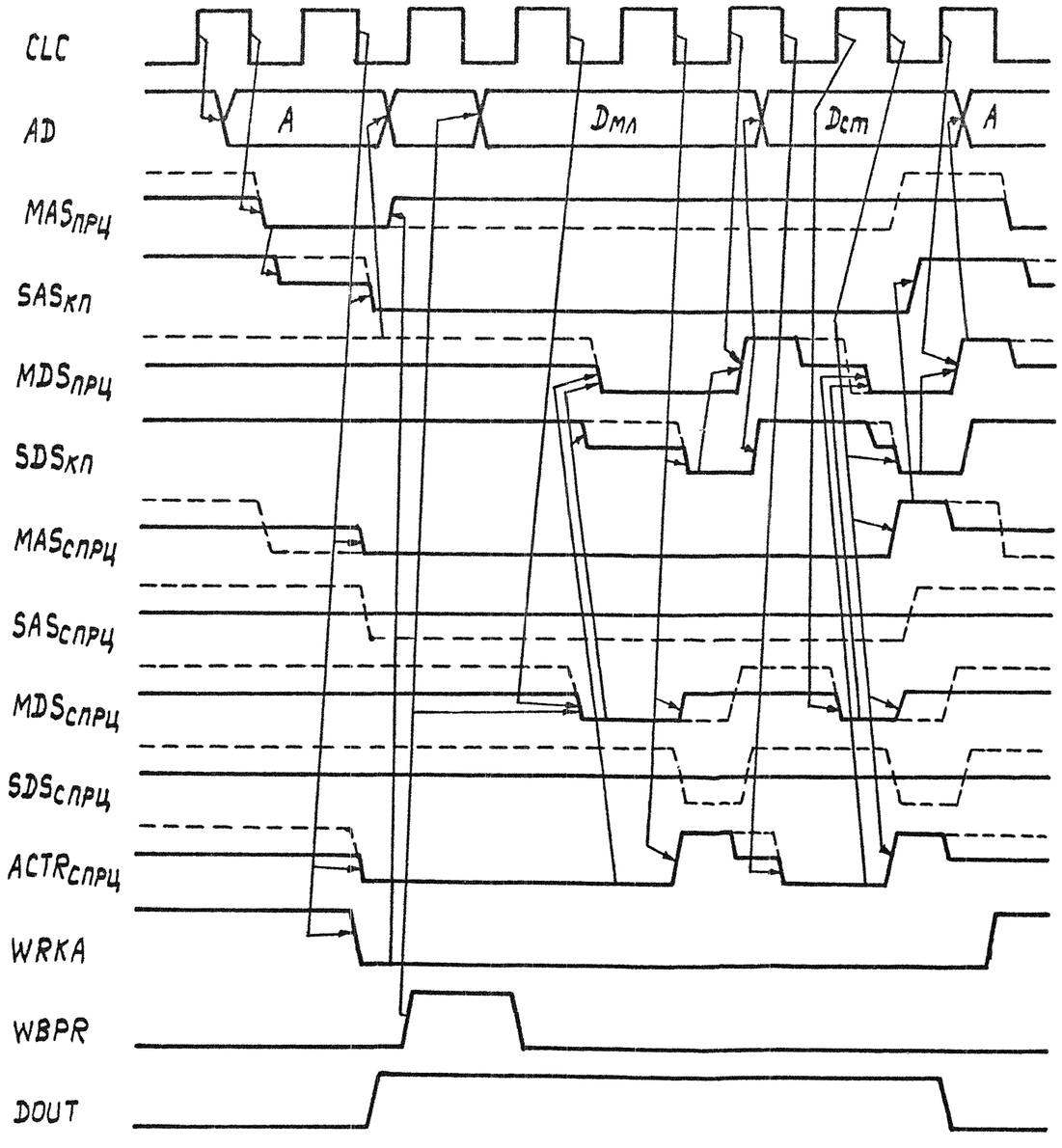


Рис. I4.

состояние, поддерживается подачей на нее питания через резистор. Наличие сигналов $ACTR_{СПРЦ}$ и $MDS_{СПРЦ}$ приводит к тому, что процессор выставляет свой сигнал $MDS_{ПРЦ}$.

В следующий полупериод тактовой частоты, т.е. в фазе низкого уровня сигнала CLC , сопроцессор переводит вывод $MDS_{СПРЦ}$ в третье состояние, и снимает (устанавливает высоким уровнем) сигналы $ACTR_{СПРЦ}$ и $MAS_{СПРЦ}$. Сопроцессор активно поддерживает высокий уровень на выводах $MAS_{СПРЦ}$ и $ACTR_{СПРЦ}$ только в течение полупериода тактовой частоты, после чего переводит их в третье состояние. Высокий уровень на цепях, объединяющих выводы микросхем MAS и $ACTR$, после того как выводы сопроцессора $MAS_{СПРЦ}$ и $ACTR_{СПРЦ}$ были переведены в третье состояние, поддерживается подачей питания на эти цепи через резисторы. Как только на цепи, объединяющей выводы микросхем MAS , устанавливается высокий уровень, контроллер памяти снимает (устанавливает высоким уровнем) сигнал $SAS_{КП}$. В этой же фазе низкого уровня сигнала CLC , если старшая половина результата записа - лась в память, контроллер памяти выставляет низким уровнем сигнал $SDS_{КП}$. В противном случае сигнал $SDS_{КП}$ будет выставлен позже, по мере того как произойдет запись данных в память, в фазе низкого уровня сигнала CLC . Если сигнал $SDS_{КП}$ успеет поступить на центральный процессор и будет принят в той же фазе низкого уровня CLC, что и выдавался, то центральный процессор снимет (установит высоким уровнем) в фазе высокого уровня CLC сигнал $MDS_{ПРЦ}$, в противном случае сигнал $MDS_{ПРЦ}$ снимется с задержкой в один период тактовой частоты. Высокий уровень на выводе $MDS_{ПРЦ}$ будет поддерживаться процессором активно в течение полупериода тактовой частоты, после чего вывод $MDS_{ПРЦ}$ переводится в третье состояние. Высокий уровень на цепи, объединяющей выводы MDS микросхем, после того, как вывод $MDS_{ПРЦ}$ будет

переведен в третье состояние, поддерживается подачей на нее питания через резистор. Как только на цепи, объединяющей выводы микросхем MDS, установится высокий уровень, контроллер памяти снимет (установит высоким уровнем) сигнал $SDS_{КП}$, сопроцессор переведет свои выводы AD в третье состояние, а процессор переведет свои выводы AD в активное состояние (выдается следующий адрес или активно устанавливается высокий уровень). На этом выполнение данной процедуры заканчивается.

4.6. Адресное чтение с перехватом данных в формате четверное слово

Эта процедура используется, если в сопроцессор требуется принять данные в "D" или "G" плавающем формате, а также четверное слово целочисленного формата из памяти. Адресную часть обмена выполняет центральный процессор, после чего обмен перехватывает сопроцессор, который и принимает данные. Временная диаграмма обмена представлена на рис. I5.

Первая половина данного обмена, а именно прием младших 32 разрядов 64-разрядного операнда, выполняется идентично процедуре адресного чтения с перехватом данных в формате двойное слово. Исключение составляет то, что сигнал $MAS_{СПРЦ}$, а следовательно, и сигнал $SAS_{КП}$, не снимаются на фоне приема младшей половины операнда. Кроме этого, после снятия сигнала $MDS_{ПРЦ}$, сопровождающего младшую половину данных, выходные элементы AD процессора и сопроцессора не изменяют своих состояний (третье состояние), а контроллеры памяти переходят в третье состояние.

Через полпериода тактовой частоты относительно снятия сигнала $MDS_{ПРЦ}$, сопровождающего прием младшей половины операнда, в фазе низкого уровня сигнала CLC сопроцессор выставляет низким уровнем сигнал перехвата при приеме старшей половины результата $ACTR_{СПРЦ}$. Еще через полпериода в фазе высокого

Временная диаграмма адресного чтения с перехватом данных в формате четверное слово

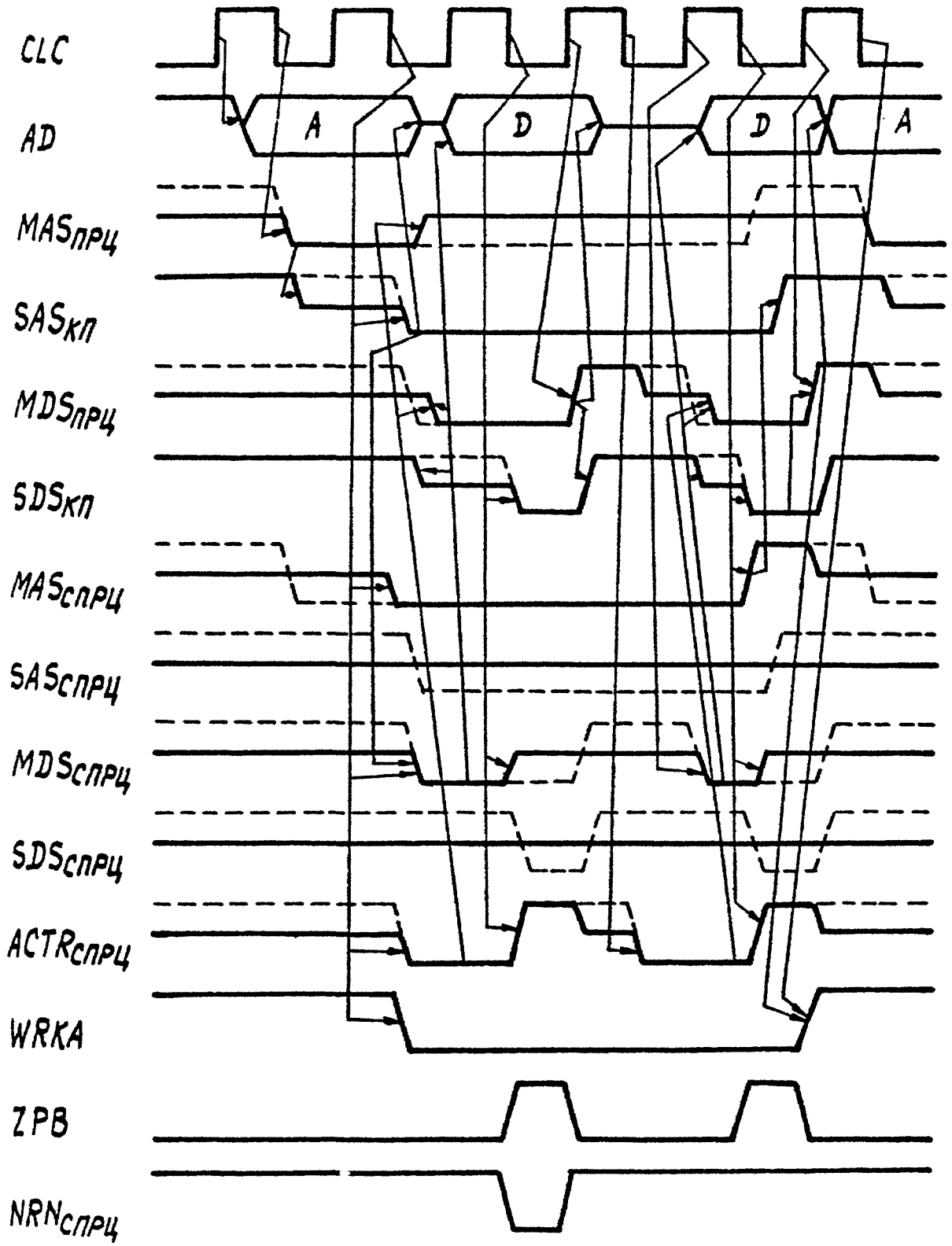


Рис. 15.

уровня сигнала CLC сопроцессор выдает низким уровнем сигнал готовности к приему старшей половины данных $MDS_{СПРЦ}$. При наличии сигналов $ACTR_{СПРЦ}$ и $MDS_{СПРЦ}$ центральный процессор выставляет низким уровнем свой сигнал $MDS_{ПРЦ}$. Наличие низкого уровня на цепи, объединяющей все выводы MDS микросхем, приводит к тому, что выходной элемент SDS контроллера памяти переводится в третье состояние, а выходные элементы AD контроллера памяти переводятся из третьего состояния в состояние выдачи данных. Высокий уровень на цепи, объединяющей выводы SDS микросхем, после того как вывод $SDS_{КП}$ перешел в третье состояние, поддерживается подачей на нее питания через резистор. В следующий полупериод тактовой частоты, т.е. в фазе низкого уровня сигнала CLC, сопроцессор переводит вывод $MDS_{СПРЦ}$ в третье состояние и снимает (устанавливает высоким уровнем) сигналы $ACTR_{СПРЦ}$ и $MAS_{СПРЦ}$. При этом сопроцессор активно поддерживает высокий уровень сигналов $MAS_{СПРЦ}$ и $ACTR_{СПРЦ}$ только лишь в течение полупериода тактовой частоты, после чего переводит их в третье состояние. Высокий уровень на цепях, объединяющих выводы микросхем MAS и ACTR, после того как выводы $MAS_{СПРЦ}$ и $ACTR_{СПРЦ}$ будут переведены в третье состояние, поддерживается подачей на них питания через резисторы. Как только на цепи, объединяющей выводы микросхем MAS устанавливается высокий уровень, контроллер памяти снимает сигнал $SAS_{КП}$ (устанавливает высоким уровнем). В этой же фазе низкого уровня сигнала CLC, если на магистрали AD выставлена старшая половина истинных данных, контроллер памяти выставляет низким уровнем сигнал $SDS_{КП}$, в противном случае сигнал $SDS_{КП}$ будет выставлен позже, по мере готовности данных, в фазе низкого уровня сигнала CLC. В этой же фазе низкого уровня сигнала CLC, в которой произошёл сброс сигналов $ACTR_{СПРЦ}$ и $MAS_{СПРЦ}$ при приеме старшей

половины операнда, сопроцессор вырабатывает импульс записи данных во входной буфер операндов. Если сигнал $SDS_{КП}$ появится в этой же фазе сигнала CLC, то интерфейсный блок сопроцессора формирует один импульс записи старшей половины операнда во входной буфер операндов, если сигнал $SDS_{КП}$ появится через период, относительно первого импульса записи, то интерфейсный блок сопроцессора формирует два импульса и т.д. Достоверные данные принимаются по последнему импульсу записи во входной буфер операндов, т.е. на фоне прихода сигнала $SDS_{КП}$.

Истинные данные на магистрали AD от контроллера памяти должны появиться не позднее окончания первой половины полупериода фазы низкого уровня сигнала CLC, в которой выдается сигнал $SDS_{КП}$, при работе на максимальной тактовой частоте (10 МГц). Если сигнал $SDS_{КП}$ при приеме старшей половины операнда успеет поступить на центральный процессор и будет принят в той же фазе низкого уровня сигнала CLC, что и выдавался сигнал $SDS_{КП}$, то центральный процессор в фазе высокого уровня сигнала CLC снимет (установит высоким уровнем) сигнал $MDS_{ПРЦ}$, в противном случае сигнал $MDS_{ПРЦ}$ снимется с задержкой в период CLC. Высокий уровень на выводе $MDS_{ПРЦ}$ будет поддерживаться процессором активно в течение полупериода тактовой частоты, после чего вывод $MDS_{ПРЦ}$ переводится в третье состояние, при этом высокий уровень на цепи, объединяющей выводы микросхем MDS, будет поддерживаться подачей на нее питания через резистор.

Как только на цепи, объединяющей выводы микросхем MDS, установится высокий уровень, так сразу контроллер памяти снимет (установит высоким уровнем) сигнал $SDS_{КП}$ и переведет свои выходные элементы AD в третье состояние, при этом выходные элементы AD процессора переводятся в активное состояние (вы-

дается следующий адрес или активно устанавливается высокий уровень). На этом выполнение данной процедуры заканчивается.

Если данный обмен является первым обменом команды, то через период CLC относительно выдачи сигнала готовности к приему младшей половины операнда $MDS_{СПРЦ}$, интерфейсный блок сопроцессора формирует полупериодный импульс сброса NRN.

4.7. Запись данных в формате байт, слово, двойное слово во входной буфер операндов СПРЦ по адресу нулевой ячейки системной памяти

Эта процедура используется, если данные в формате байт, слово или двойное слово требуется передать из процессора в сопроцессор. В этом случае интерфейсный блок сопроцессора выступает в роли контроллера ячейки системной памяти, имеющей нулевой адрес, в которую требуется записать данные. Временная диаграмма обмена представлена на рис.16.

В фазе высокого уровня сигнала CLC процессор выставляет на магистраль AD адрес записи в нулевую ячейку системной памяти (F2000000$_{16}$ – адрес записи операнда в формате байт, D2000000$_{16}$ – адрес записи операнда в формате слово, C2000000$_{16}$ – адрес записи операнда в формате двойное слово), после чего в фазе низкого уровня CLC выставляет сигнал сопровождения адреса $MAS_{ПРЦ}$.

Низкий уровень на цепи, объединяющей выводы MAS микросхем, вызывает переключение выходного элемента SAS контроллера памяти в третье состояние. Высокий уровень на цепи, объединяющей выводы SAS микросхем, после отключения выходного элемента $SAS_{КП}$, поддерживается подачей на нее питания через резистор. После появления сигнала $MAS_{ПРЦ}$ интерфейсный блок сопроцессора в течение периода тактовой частоты занимается дешифрацией адреса, в результате которой в фазе низкого уровня сигнала CLC выстав-

Временная диаграмма записи данных в формате байт, слово, двойное слово во входной буфер операндов СПРЦ

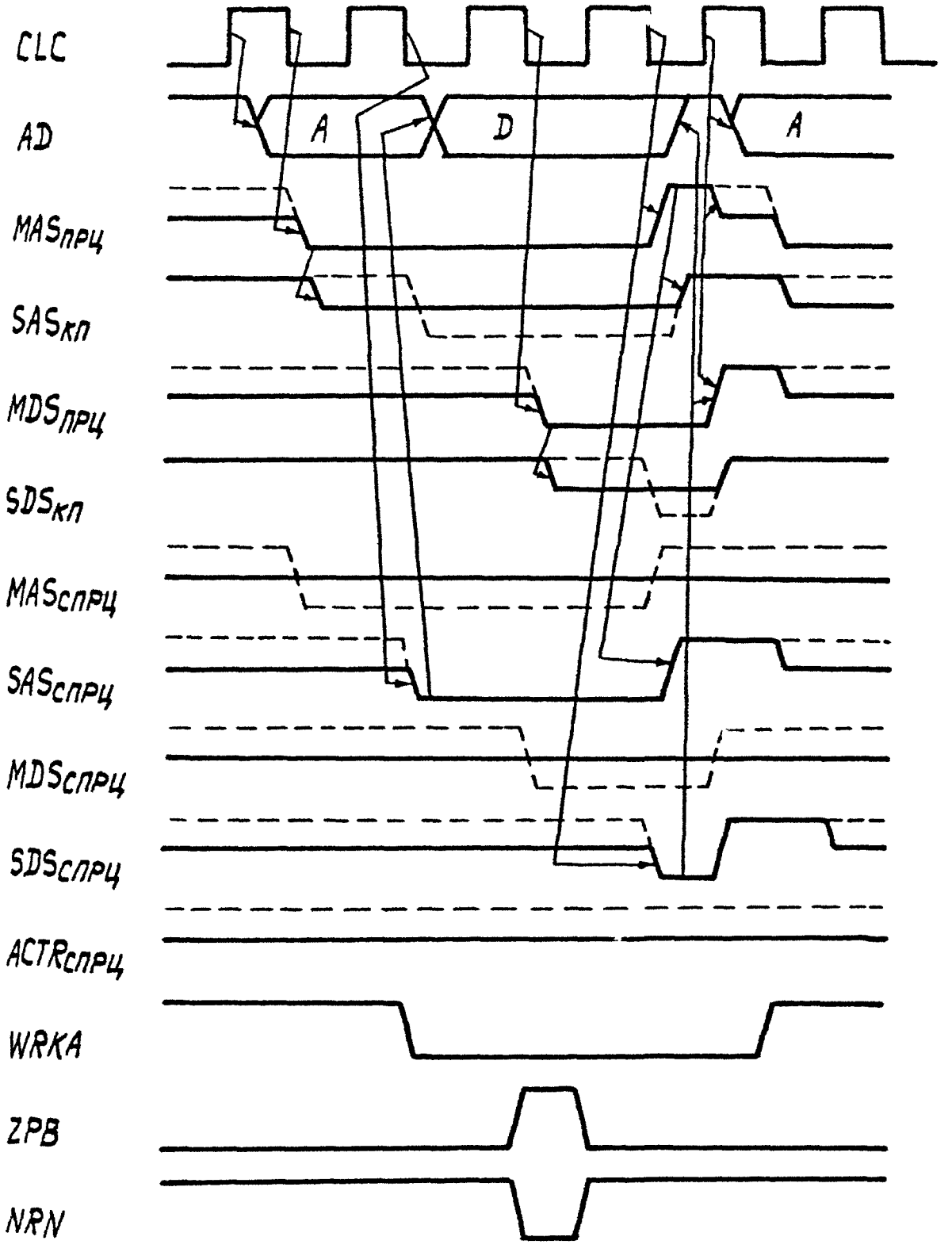


Рис. I6.

ляет низким уровнем сигнал подтверждения приема адреса $SAS_{СПРЦ}$. По сигналу $SAS_{СПРЦ}$ центральный процессор переключает свои выходные элементы AD с выдачи адреса на выдачу содержимого выходного буфера операндов процессора. Если к моменту прихода сигнала $SAS_{СПРЦ}$ процессор уже произвел запись операнда в свой выходной буфер операндов, то через период тактовой частоты относительно появления сигнала $SAS_{СПРЦ}$ процессор в фазе низкого уровня сигнала CLC низким уровнем выдаст сигнал сопровождения данных $MDS_{ПРЦ}$. В противном случае процессор выставит сигнал $MDS_{ПРЦ}$ в фазе низкого уровня CLC через полтора периода тактовой частоты относительно импульса записи в свой выходной буфер операндов. Истинные данные появятся на магистрали AD за период тактовой частоты до появления сигнала $MDS_{ПРЦ}$.

Наличие низкого уровня на цепи, объединяющей выводы MDS микросхем, приводит к тому, что выходной элемент SDS контроллера памяти переключается в третье состояние, при этом высокий уровень на цепи, объединяющей выводы SDS микросхем, поддерживается подачей на нее питания через резистор.

Через период CLC после выдачи сигнала $SAS_{СПРЦ}$ интерфейсный блок сопроцессора начинает вырабатывать импульсы записи данных во входной буфер операндов сопроцессора. Эти импульсы формируются в фазе низкого уровня сигнала CLC и кратны полупериоду тактовой частоты. Последний импульс формируется на фоне прихода сигнала $MDS_{ПРЦ}$. Т.е., если сигнал $MDS_{ПРЦ}$ появился через период CLC после выставления сигнала $SAS_{СПРЦ}$, то интерфейсный блок формирует один импульс записи во входной буфер операндов, если через два периода, то два импульса и т.д. Достоверные данные принимаются по последнему импульсу записи во входной буфер операндов.

Кроме этого, если данный обмен является первым обменом команды, то через период CLC относительно выдачи сигнала $SAS_{СПРЦ}$ интерфейсный блок сопроцессора формирует полупери - одный импульс сброса NRN .

Через период тактовой частоты относительно появления сигнала $MDS_{ПРЦ}$ в фазе низкого уровня сигнала CLC сопроцессор выдает низким уровнем сигнал подтверждения приема данных $SDS_{СПРЦ}$, а центральный процессор снимает(устанавливает высоким уровнем) сигнал $MAS_{ПРЦ}$. Процессор активно поддерживает высокий уровень сигнала $MAS_{ПРЦ}$ в течение полупериода тактовой частоты, после чего переключает выходной элемент в третье состояние. Высокий уровень на цепи , объединяющей выводы MAS микросхем, после переключения вывода $MAS_{ПРЦ}$ в третье состояние, поддерживается подачей на нее питания через резистор. Как только на цепи,объединяющей выводы микросхем MAS, устанавливается высокий уровень, контроллер памяти и сопроцессор. снимают (устанавливают высоким уровнем) свои сигналы $SAS_{КЛ}$ и $SAS_{СПРЦ}$, причем контроллер памяти будет держать активно высокий уровень на выводе $SAS_{КЛ}$ до появления сигнала MAS в следующем обмене, а сопроцессор переключит вывод $SAS_{СПРЦ}$ в третье состояние через полпериода тактовой частоты относительно снятия сигнала $MDS_{ПРЦ}$ текущего обмена. Если процессор успеет принять сигнал $SDS_{СПРЦ}$ в той же фазе низкого уровня CLC , в которой он и выдавался, то процессор в фазе высокого уровня CLC снимет (установит высоким уровнем) сигнал $MDS_{ПРЦ}$, в противном случае сигнал $MDS_{ПРЦ}$ будет снят с задержкой в период тактовой частоты.

Высокий уровень на выводе $MDS_{ПРЦ}$ будет поддерживаться процессором активно в течение полупериода тактовой частоты,после чего выходной элемент $MDS_{ПРЦ}$ переключается в третье со -

стояние. Высокий уровень на цепи, объединяющей выводы MDS микросхем, после того как вывод $MDS_{ПРЦ}$ переключится в третье состояние, поддерживается подачей на нее питания через резистор. Как только на цепи, объединяющей выводы микросхем MDS, устанавливается высокий уровень, сопроцессор и контроллер памяти снимают (устанавливают высоким уровнем) свои сигналы $SDS_{СПРЦ}$ и $SDS_{КП}$, причем контроллер памяти держит активно высокий уровень на выводе $SDS_{КП}$ до появления сигнала MDS в следующем обмене, а сопроцессор держит активно высокий уровень на выводе $SDS_{СПРЦ}$ в течение периода тактовой частоты. Кроме этого, процессор снимает со своих выходных элементов AD данные и выставляет либо адрес следующего обмена, либо активно поддерживает высокий уровень на магистрали AD. На этом выполнение данной процедуры заканчивается.

4.8. Чтение данных в формате байт, слово, двойное слово из выходного буфера операндов СПРЦ по адресу нулевой ячейки системной памяти

Эта процедура используется, если данные в формате байт, слово или двойное слово требуется передать из сопроцессора в процессор. В этом случае интерфейсный блок сопроцессора выступает в роли контроллера ячейки системной памяти, имеющей нулевой адрес, из которой требуется прочитать данные. Временная диаграмма обмена представлена на рис. I7.

В фазе высокого уровня сигнала CLC процессор выставляет на магистраль AD адрес чтения нулевой ячейки системной памяти ($F1000000_{I6}$ – адрес чтения операнда в формате байт, $D1000000_{I6}$ – – адрес чтения операнда в формате слово, $C1000000_{I6}$ – адрес чтения операнда в формате двойное слово), после чего в фазе низкого уровня сигнала CLC выставляет сигнал сопровождения адреса

Временная диаграмма чтения данных в формате байт, слово, двойное слово из выходного буфера операндов СПРЦ

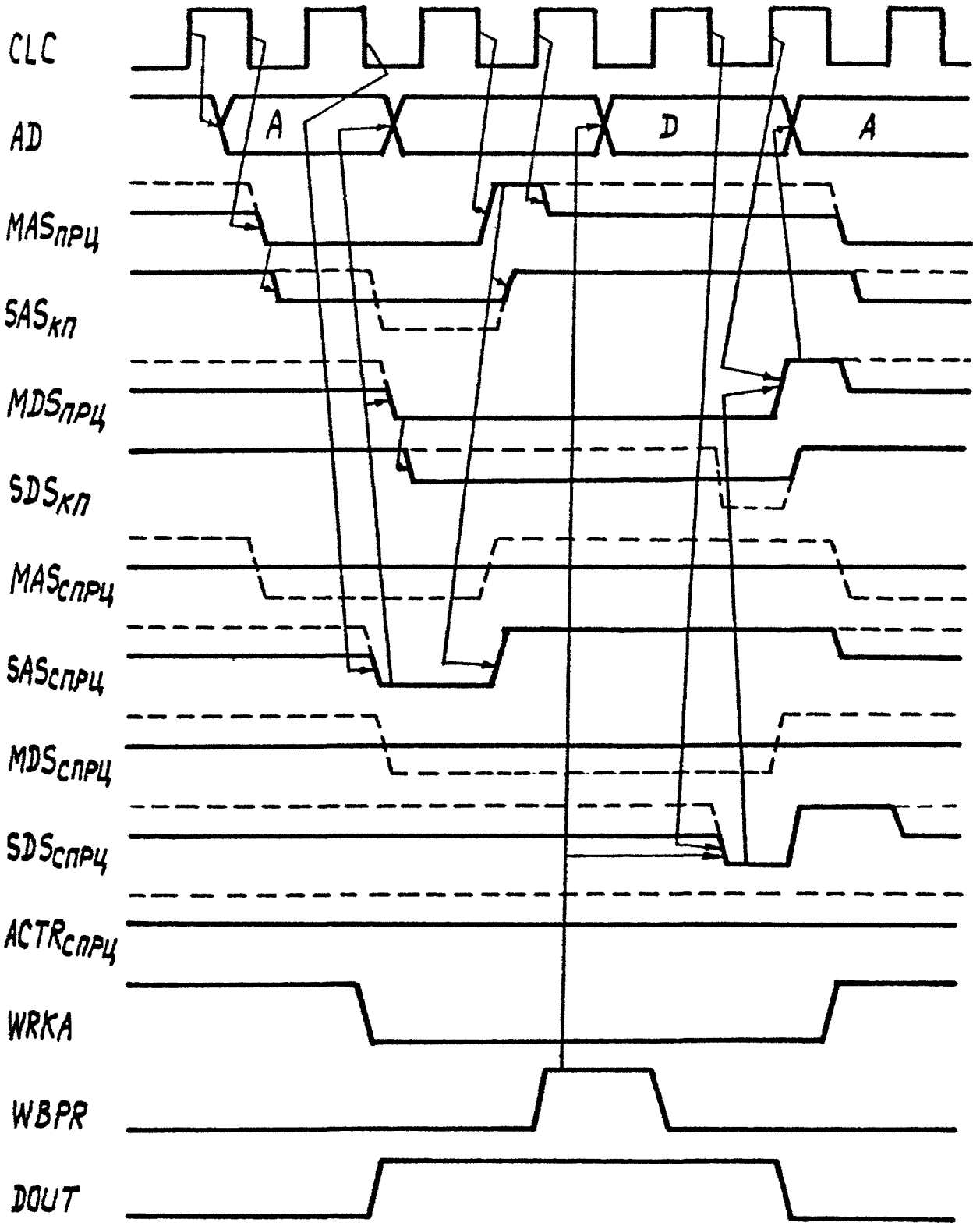


Рис. I7.

$MAS_{ПРЦ}$.

Низкий уровень на цепи, объединяющей выводы MAS микросхем, вызывает переключение выходного элемента SAS контроллера памяти в третье состояние. Высокий уровень на цепи, объединяющей выводы микросхем SAS, после переключения выходного элемента $SAS_{КП}$ в третье состояние, поддерживается подачей на нее питания через резистор.

После появления сигнала $MAS_{ПРЦ}$, интерфейсный блок сопроцессора в течение периода тактовой частоты занимается дешифрацией адреса, в результате которой в фазе низкого уровня сигнала CLC выставляет низким уровнем сигнал подтверждения приема адреса $SAS_{СПРЦ}$.

По сигналу $SAS_{СПРЦ}$ процессор переключает свои выходные элементы в третье состояние, сопроцессор же свои выходные элементы переключает из третьего состояния в состояние выдачи данных из выходного буфера операндов. Кроме этого, если процессор успеет принять сигнал $SAS_{СПРЦ}$ в той же фазе сигнала CLC, в которой он и выдавался, то процессор выставляет низким уровнем сигнал готовности к приему данных $MDS_{ПРЦ}$, в противном случае сигнал $MDS_{ПРЦ}$ будет выставлен с задержкой в один период тактовой частоты. Наличие низкого уровня на цепи, объединяющей выводы MDS микросхем, приводит к тому, что выходной элемент SDS контроллера памяти переключается в третье состояние. Высокий уровень на цепи, объединяющей выводы SDS микросхем, после того как выходной элемент $SDS_{КП}$ переключится в третье состояние, поддерживается подачей на нее питания через резистор. Если к моменту выдачи сигнала $MDS_{ПРЦ}$ данные уже записаны в выходной буфер операндов, то через период тактовой частоты относительно выдачи сигнала $MDS_{ПРЦ}$, сопроцессор в фазе низкого уровня сигнала CLC выдаст низким уровнем сигнал $SDS_{СПРЦ}$. В

противном случае сопроцессор выставит сигнал $SDS_{СПРЦ}$ в фазе низкого уровня сигнала CLC через полтора периода тактовой частоты относительно импульса записи результата в выходной буфер операндов сопроцессора.

Кроме этого, через период тактовой частоты относительно появления сигнала $MDS_{ПРЦ}$ процессор снимает (устанавливает высоким уровнем) сигнал $MAS_{ПРЦ}$. Процессор активно поддерживает высокий уровень сигнала $MAS_{ПРЦ}$ в течение полупериода тактовой частоты, после чего переключает выходной элемент в третье состояние. Высокий уровень на цепи, объединяющей выводы микросхем MAS, после переключения вывода $MAS_{ПРЦ}$ в третье состояние, поддерживается подачей на нее питания через резистор. Как только на цепи, объединяющей выводы микросхем MAS, устанавливается высокий уровень, контроллер памяти и сопроцессор снимают (устанавливают высоким уровнем) свои сигналы $SAS_{КП}$ и $SAS_{СПРЦ}$, причем контроллер памяти будет держать активно высокий уровень на выводе $SAS_{КП}$ до появления сигнала MAS в следующем обмене, сопроцессор же переключит вывод $SAS_{СПРЦ}$ в третье состояние через полпериода тактовой частоты относительно снятия сигнала $MDS_{ПРЦ}$ текущего обмена.

Если процессор успеет принять сигнал $SDS_{СПРЦ}$ в той же фазе низкого уровня CLC, в которой он и выдавался, то процессор в фазе высокого уровня сигнала CLC снимет (установит высоким уровнем) сигнал $MDS_{ПРЦ}$. В противном случае сигнал $MDS_{ПРЦ}$ будет снят с задержкой в период тактовой частоты. Высокий уровень на выводе $MDS_{ПРЦ}$ поддерживается активно процессором в течение полупериода тактовой частоты, после чего выходной элемент $MDS_{ПРЦ}$ переключается в третье состояние. Высокий уровень на цепи, объединяющей выводы микросхем MDS, после того как вывод $MDS_{ПРЦ}$ переключается в третье состояние, поддерживается

подачей на нее питания через резистор. Как только на цепи, объединяющей выводы микросхем MDS, устанавливается высокий уровень, сопроцессор и контроллер памяти снимают (устанавливают высоким уровнем) свои сигналы $SDS_{СПРЦ}$ и $SDS_{КП}$, причем контроллер памяти держит активно высокий уровень на выводе $SDS_{СП}$ до появления сигнала MDS в следующем обмене, а сопроцессор держит активно высокий уровень на выводе $SDS_{СПРЦ}$ в течение периода тактовой частоты. Кроме этого, сопроцессор переключает свои выходные элементы AD в третье состояние, а процессор переключает свои выходные элементы AD из третьего состояния в состояние выдачи адреса следующего обмена, либо в состояние выдачи на магистраль AD высокого уровня. На этом выполнение данной процедуры заканчивается.

4.9. Запись данных в формате четверное слово во входной буфер операндов СПРЦ по адресу нулевой ячейки системной памяти

Эта процедура используется, если данные в "D", "G" плавающем формате или четверное слово целочисленного формата требуется передать из процессора в сопроцессор. В этом случае интерфейсный блок сопроцессора выступает в роли контроллера нулевой ячейки системной памяти. Временная диаграмма обмена представлена на рис. I8.

Выполнение первой половины данного обмена, а именно, запись во входной буфер операндов сопроцессора младших 32 разрядов 64-разрядного операнда, идентично выполнению процедуры записи данных в формате двойное слово во входной буфер операндов СПРЦ по адресу нулевой ячейки системной памяти. Исключение составляет то, что процессор выставляет адрес $E2000000_{16}$, а также то, что сигнали $MAS_{ПРЦ}$, а следовательно, и $SAS_{СПРЦ}$, не снимаются

Временная диаграмма записи данных в формате четверное слово
во входной буфер операндов СПРЦ

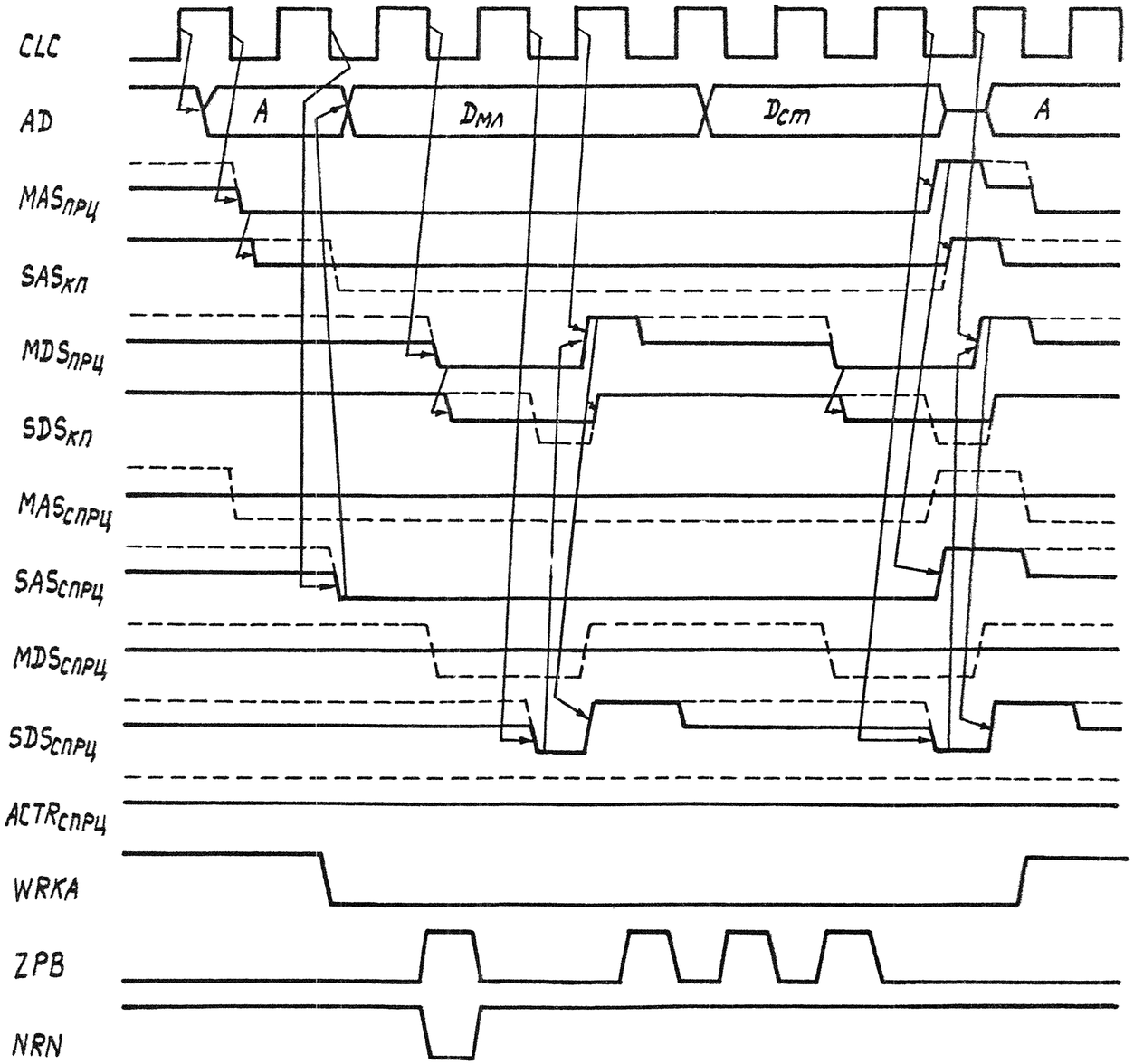


Рис. I8.

на фоне записи младшей половины операнда. Кроме этого, после снятия сигнала сопровождения младшей половины операнда $MDS_{ПРЦ}$ выходные элементы AD процессора и сопроцессора не меняют своих состояний (выходные элементы AD процессора находятся в состоянии выдачи содержимого выходного буфера операндов процессора на магистраль AD, а сопроцессора - в третьем состоянии).

После того как фаза записи младшей половины операнда во входной буфер операндов сопроцессора закончилась, интерфейсный блок процессора переходит в состояние ожидания импульса записи старшей половины операнда в выходной буфер операндов процессора. Через полтора периода тактовой частоты относительно начала импульса записи старшей половины операнда в выходной буфер операндов процессора, интерфейсный блок процессора в фазе низкого уровня сигнала CLC выдает низким уровнем сигнал сопровождения старшей половины операнда $MDS_{ПРЦ}$.

Истинные данные появляются на магистрали AD за период тактовой частоты до появления сигнала $MDS_{ПРЦ}$. Минимальное время между снятием сигнала сопровождения младшей половины данных $MDS_{ПРЦ}$ и выставлением сигнала сопровождения старшей половины данных $MDS_{ПРЦ}$ составляет два с половиной периода тактовой частоты. Наличие низкого уровня на цепи, объединяющей выводы MDS микросхем, приводит к тому, что выходной элемент $SDS_{КП}$ переключается в третье состояние, при этом высокий уровень на цепи, объединяющей выводы микросхем SDS, поддерживается подачей на нее питания через резистор. Через полпериода тактовой частоты относительно снятия сигнала сопровождения младшей половины данных $MDS_{ПРЦ}$ интерфейсный блок сопроцессора начинает вырабатывать импульсы записи старшей половины результата во входной буфер операндов. Эти импульсы фор-

мируются в фазе низкого уровня сигнала CLC и кратны полупериоду тактовой частоты. Последний импульс формируется на фоне прихода сигнала $MDS_{ПРЦ}$. Достоверные данные принимаются по последнему импульсу записи во входной буфер операндов.

Через период тактовой частоты относительно появления сигнала сопровождения старшей половины данных $MDS_{ПРЦ}$, в фазе низкого уровня сигнала CLC сопроцессор выдает низким уровнем сигнал подтверждения приема данных $SDS_{СПРЦ}$, а центральный процессор снимает (устанавливает высоким уровнем) сигнал $MAS_{ПРЦ}$. Процессор активно поддерживает высокий уровень сигнала $MAS_{ПРЦ}$ в течение полупериода тактовой частоты, после чего переключает выходной элемент в третье состояние. Высокий уровень на цепи, объединяющей выводы микросхем MAS, после того как вывод $MAS_{ПРЦ}$ переключается в третье состояние, поддерживается подачей на нее питания через резистор. Как только на цепи, объединяющей выводы микросхем MAS, устанавливается высокий уровень, контроллер памяти и сопроцессор снимают (устанавливают высоким уровнем) свои сигналы $SAS_{КП}$ и $SAS_{СПРЦ}$, причем контроллер памяти будет держать активно высокий уровень на выводе $SAS_{КП}$ до появления сигнала MAS в следующем обмене, а сопроцессор переключает вывод $SAS_{СПРЦ}$ в третье состояние через полпериода тактовой частоты относительно снятия сигнала сопровождения старшей половины данных $MDS_{ПРЦ}$ текущего обмена. Если процессор успеет принять сигнал $SDS_{СПРЦ}$ в той же фазе низкого уровня сигнала CLC, в которой он и выдавался, то процессор в фазе высокого уровня CLC снимет (установит высоким уровнем) сигнал $MDS_{ПРЦ}$. В противном случае сигнал $MDS_{ПРЦ}$ будет снят с задержкой в период тактовой частоты. Высокий уровень на выводе $MDS_{ПРЦ}$ будет поддерживаться процессором активно в течение полупериода тактовой частоты, после чего выходной элемент

$MDS_{ПРЦ}$ переключается в третье состояние.

Высокий уровень на цепи, объединяющей выводы MDS микросхем, после того как вывод $MDS_{ПРЦ}$ переключается в третье состояние, поддерживается подачей на нее питания через резистор. Как только на цепи, объединяющей выводы микросхем MDS, установится высокий уровень, сопроцессор и контроллер памяти снимают (устанавливают высоким уровнем) свои сигналы $SDS_{СПРЦ}$ и $SDS_{КП}$, причем контроллер памяти держит активно высокий уровень на выводе $SDS_{КП}$ до появления сигнала MDS в следующем обмене, а сопроцессор – в течение периода тактовой частоты. Кроме этого, процессор снимает со своих выходных элементов AD данные и выставляет либо адрес следующего обмена, либо активно поддерживает высокий уровень на магистрали AD. На этом выполнение данной команды заканчивается.

Если данный обмен является первым обменом команды, то интерфейсный блок сопроцессора через период тактовой частоты относительно выдачи сигнала $SAS_{СПРЦ}$ в фазе низкого уровня сигнала CLC формирует полупериодный импульс сброса NRN.

4.10. Чтение данных в формате четверное слово из выходного буфера операндов СПРЦ по адресу нулевой ячейки системной памяти

Эта процедура используется, если данные в "D", "G" плавающем формате или четверное слово целочисленного формата требуется передать из сопроцессора в процессор. В этом случае интерфейсный блок сопроцессора выступает в роли контроллера нулевой ячейки системной памяти. Временная диаграмма обмена представлена на рис. 19.

Выполнение первой половины данного обмена, а именно, чтение из выходного буфера операндов младших 32 разрядов 64-разрядного

Временная диаграмма чтения данных в формате четверное слово из выходного буфера операндов СПРЦ

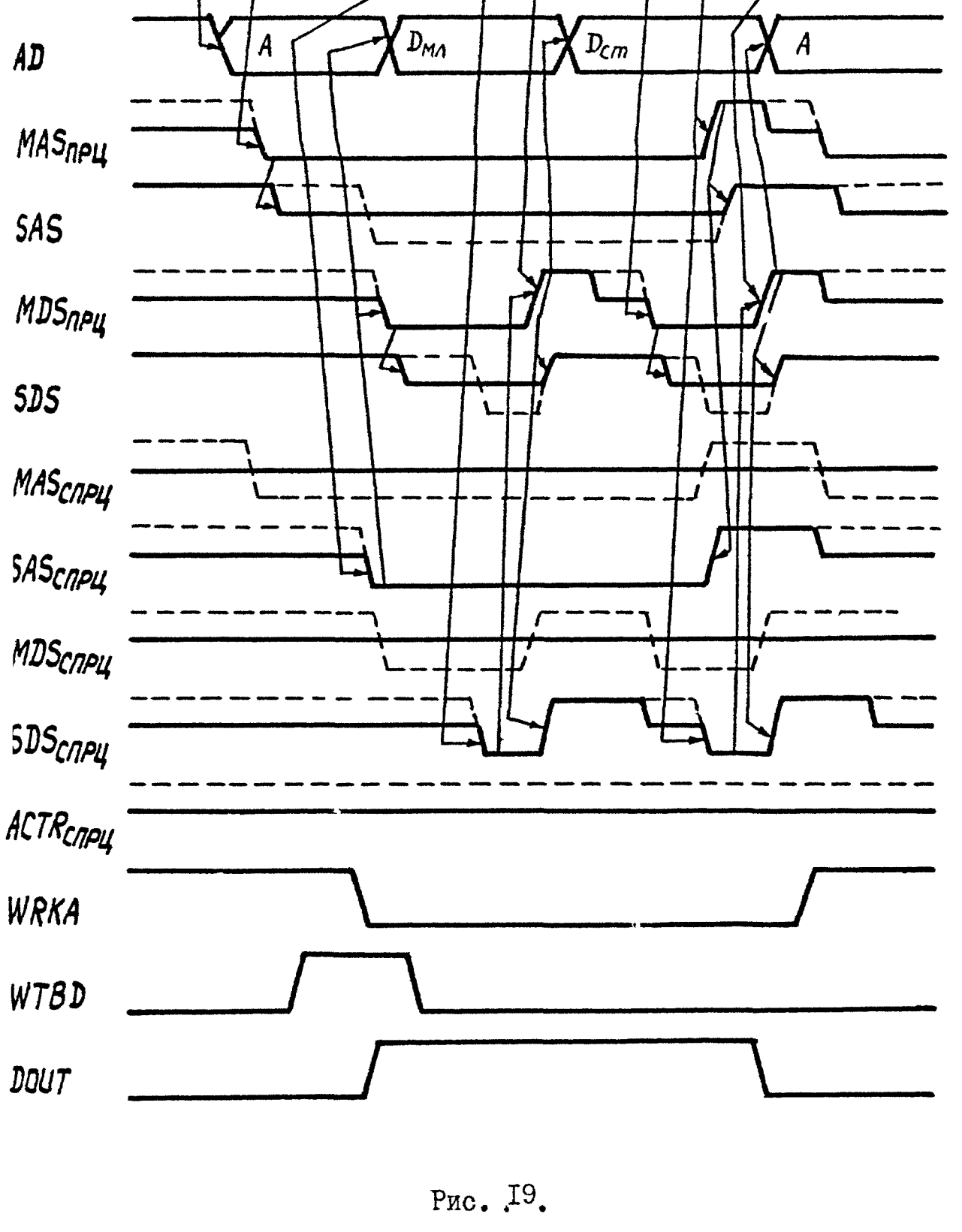


Рис. 19.

операнда, идентично выполнению процедуры чтения данных в формате двойное слово из выходного буфера операндов по адресу нулевой ячейки системной памяти. Исключение составляет то, что процессор выставляет адрес $EI000000_{I6}$, а также то, что сигналы $MAS_{ПРЦ}$, а следовательно и $SAS_{СПРЦ}$, не снимаются на фоне чтения младшей половины операнда. Кроме этого, после снятия сигнала готовности к приему младшей половины данных $MDS_{ПРЦ}$ выходные элементы AD процессора и сопроцессора не меняют своих состояний (выходные элементы AD процессора находятся в третьем состоянии, выходные элементы AD сопроцессора настраиваются на выдачу старшей половины операнда).

Через период тактовой частоты после снятия сигнала готовности к приему младшей половины операнда $MDS_{ПРЦ}$ процессор в фазе высокого уровня сигнала CLC выставляет низким уровнем сигнал готовности к приему старшей половины операнда $MDS_{ПРЦ}$. Истинные данные устанавливаются на магистрали AD за период до появления сигнала $MDS_{ПРЦ}$. Наличие низкого уровня на цепи, объединяющей выводы микросхем SDS, приводит к тому, что выходной элемент SDS контроллера памяти переключается в третье состояние. Высокий уровень на цепи, объединяющей выводы SDS микросхем, после того как вывод $SDS_{КП}$ переключился в третье состояние, поддерживается подачей на нее питания через резистор. Через полпериода тактовой частоты относительно выставления процессором сигнала $MDS_{ПРЦ}$, в фазе низкого уровня сигнала CLC процессор снимает (устанавливает высоким уровнем) сигнал $MAS_{ПРЦ}$, сопроцессор же выставляет низким уровнем сигнал подтверждения выдачи старшей половины операнда $SDS_{СПРЦ}$. Процессор активно поддерживает высокий уровень сигнала $MAS_{ПРЦ}$ в течение полупериода тактовой частоты, после чего переключает выходной элемент в третье состояние. Высокий уровень на цепи, объе-

диняющей выводы микросхем MAS , после того как вывод $MAS_{ПРЦ}$ был переключен в третье состояние, поддерживается подачей на нее питания через резистор. Как только на цепи, объединяющей выводы микросхем MAS, будет установлен высокий уровень, контроллер памяти и сопроцессор снимают (устанавливают высоким уровнем) свои сигналы $SAS_{КП}$ и $SAS_{СПРЦ}$, причем контроллер памяти будет держать активно высокий уровень на выводе $SAS_{КП}$ до появления сигнала MAS в следующем обмене, сопроцессор же переключит вывод $SAS_{СПРЦ}$ в третье состояние через полпериода тактовой частоты относительно снятия сигнала $MDS_{ПРЦ}$ текущего обмена. Если процессор успеет принять сигнал $SDS_{СПРЦ}$ в той же фазе низкого уровня сигнала CLC , в которой он выдавался, то процессор в фазе высокого уровня сигнала CLC снимет (установит высоким уровнем) сигнал $MDS_{ПРЦ}$. В противном случае сигнал $MDS_{ПРЦ}$ будет снят процессором с задержкой в период тактовой частоты. Высокий уровень на выводе $MDS_{ПРЦ}$ поддерживается активно процессором в течение полупериода тактовой частоты, после чего выходной элемент $MDS_{ПРЦ}$ переключается в третье состояние. Высокий уровень на цепи, объединяющей выводы микросхем MDS , после того как вывод $MDS_{ПРЦ}$ будет переключен в третье состояние, поддерживается подачей на нее питания через резистор. Как только на цепи,объединяющей выводы микросхем MDS, будет установлен высокий уровень, сопроцессор и контроллер памяти снимают (устанавливают высоким уровнем) свои сигналы $SDS_{СПРЦ}$ и $SDS_{КП}$, причем контроллер памяти держит активно высокий уровень на выводе $SDS_{КП}$ до появления сигнала MDS в следующем обмене, а сопроцессор – в течение периода тактовой частоты. Кроме этого, сопроцессор переключает свои выходные элементы AD в третье состояние, а процессор переключает свои выходные элементы AD из третьего состояния в состо-

яние выдачи адреса следующего обмена, либо в состояние выдачи высокого уровня на магистраль AD . На этом выполнение данной процедуры заканчивается.

4.II. Чтение регистра кода ошибок сопроцессора

Эта процедура используется в микропрограммах обработки прерываний или ошибок, возникающих в результате выполнения команд сопроцессором, для передачи кода ошибки из сопроцессора в процессор. Процессор читает регистр кода ошибок по адресу системной памяти, равному единице. Интерфейсный блок сопроцессора при чтении регистра кода ошибок выступает в роли контроллера ячейки системной памяти, адрес которой равен I. Временная диаграмма обмена представлена на рис.20,

Данная процедура выполняется идентично процедуре чтения данных в формате двойное слово из выходного буфера операндов сопроцессора. Исключение составляет то, что процессор выставляет адрес $CI000001_{I6}$, данные на магистраль AD выдаются из регистра кода ошибок сопроцессора, сопроцессор выставляет сигнал $SDS_{СПРЦ}$ через период тактовой частоты относительно выставляемого процессором сигнала $MDS_{ПРЦ}$.

4.I2. Чтение регистра кода ветвления сопроцессора

Это процедура используется в микропрограммах команд ACBF, ACBD и ACBG для передачи из сопроцессора в процессор кода ветвления. Процессор читает регистр кода ветвления по адресу системной памяти, равному 2. Интерфейсный блок сопроцессора при чтэнии регистра кода ветвления выступа-

Временная диаграмма чтения регистра кода ошибок СПРЦ

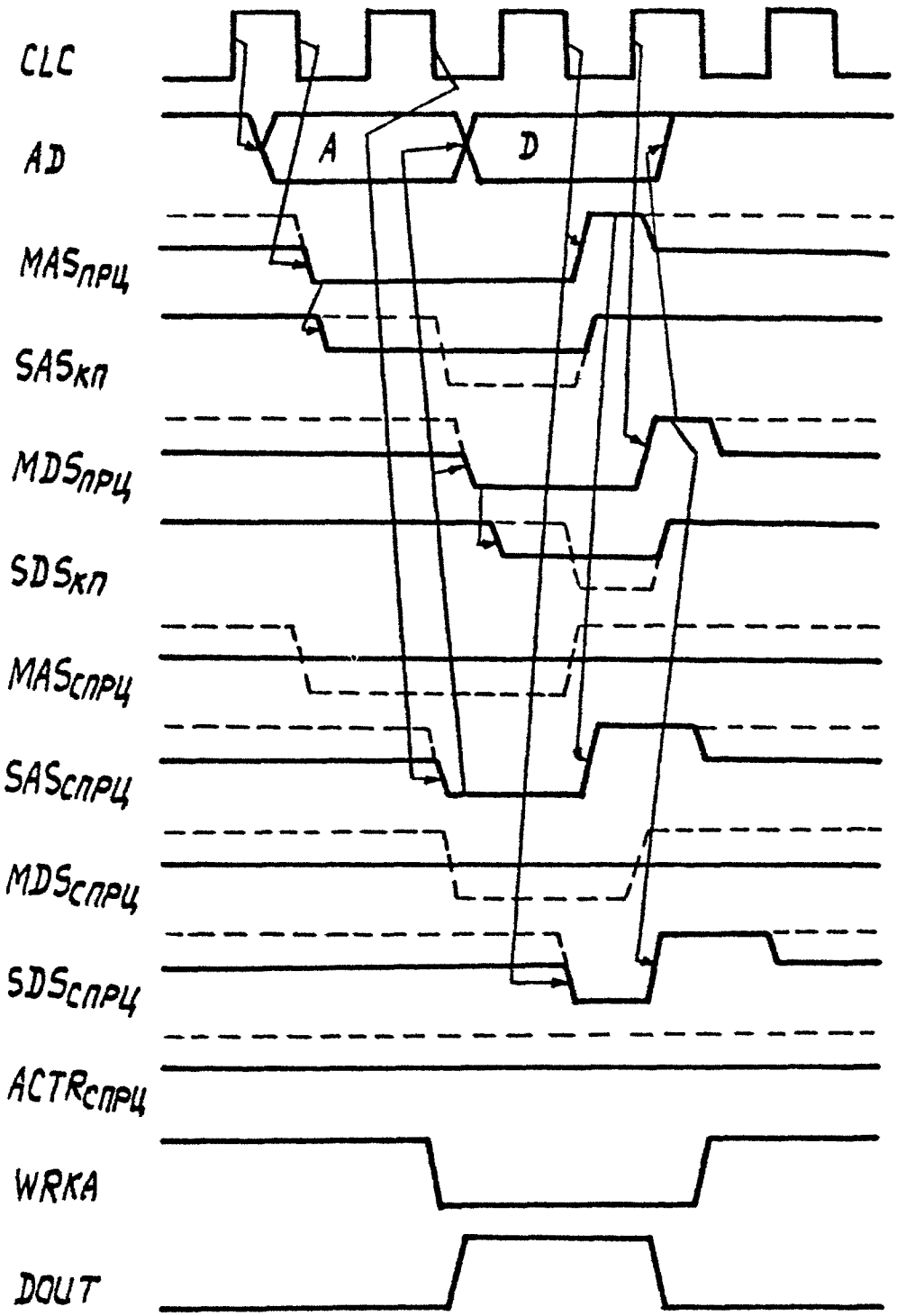


Рис. 20.

ет в роли контроллера ячейки системной памяти, адрес которой равен 2. Временная диаграмма обмена представлена на рис.21,

Данная процедура выполняется идентично процедуре чтения данных в формате двойное слово из выходного буфера операндов сопроцессора. Исключение составляет то, что процессор выставляет адрес $CI000002_{I6}$, данные на магистраль AD выдаются из регистра кода ветвления сопроцессора, сопроцессор выставляет сигнал $SDS_{СПРЦ}$ через два с половиной периода тактовой частоты относительно импульса записи данных в выходной буфер операндов.

Временная диаграмма чтения регистра кода ветвления СПРЦ

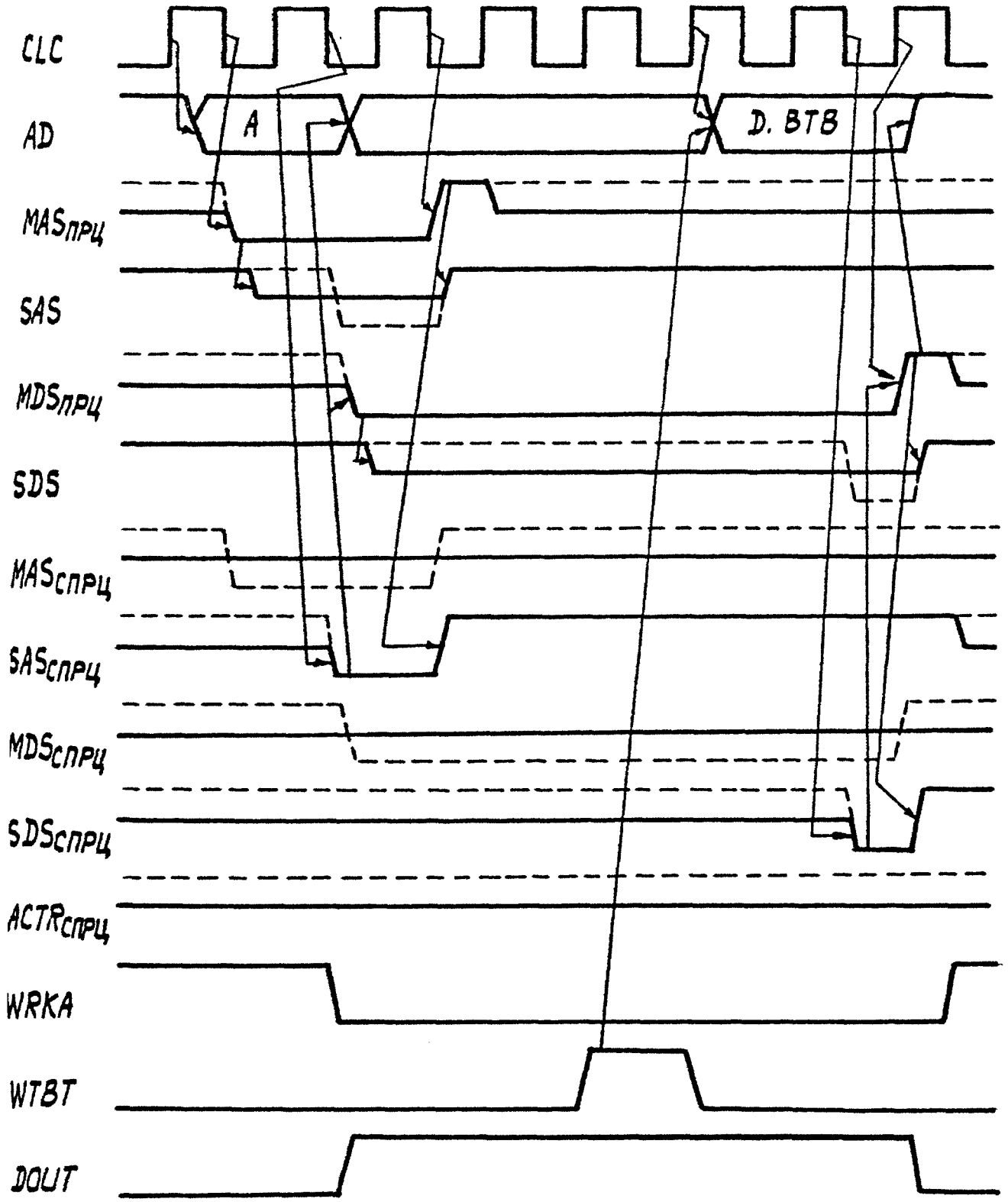
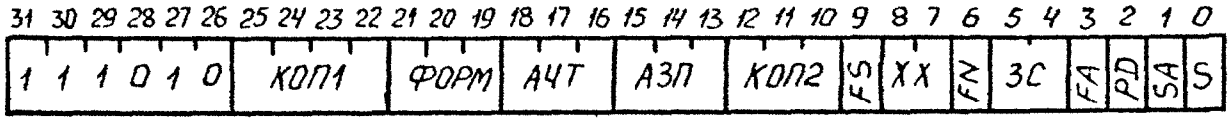


Рис. 21.

5. МИКРОПРОГРАММНОЕ УПРАВЛЕНИЕ

Формат микрокоманды



Разряды (3I-26) определяют формат микрокоманды сопроцессора.

Разряды (25-22) образуют КОПI, который управляет в основном подготовительными операциями при выполнении команды. Это различные пересылки из регистра по адресу АЧТ в регистр по адресу АЗП. Эти пересылки сопровождаются, как правило, некоторыми действиями. Далее будет описано,как выполняются все КОПI.

Разряды (2I-I9) образуют формат операндов.

Разряды (I8-I6) образуют адрес чтения.

Разряды (I5-I3) образуют адрес записи.

Разряды (5,4) образуют код записи состояния.

Разряды (I2-I0) образуют КОП2, который определяет основную операцию микрокоманды.

Разряды 9,6,3,2 и I - это различные вспомогательные признаки, используемые при выполнении команды. Их назначение будет описано ниже.

Разряды 8,7 и 0 не используются.

Код				Мнемоника	Наименование
25	24	23	22		
0	0	0	0	MOV	Пересылка
0	0	0	I	MNS	Пересылка с изменением знака, если знак П3 отрицательный
0	0	I	0	MNN	Пересылка с изменением знака, если знак целого отрицательный
0	0	I	I	RND	Пересылка с округлением
0	I	0	0	MSP	Пересылка с вычитанием порядков
0	I	0	I	MIF	Пересылка с образованием формата F
0	I	I	0	MID	Пересылка с образованием формата D
0	I	I	I	MIG	Пересылка с образованием формата G
I	0	0	0	MWP2	Пересылка с записью в регистр порядка II
I	0	0	I	MRB	Пересылка с образованием формата B
I	0	I	0	MRW	Пересылка с образованием формата W
I	0	I	I	MRL	Пересылка с образованием формата L
I	I	0	0	MWP1	Пересылка с записью регистра порядка I
I	I	0	I	MPA	Восстановление порядка и знака в команде POLY
I	I	I	0	MPS	Пересылка с вычитанием константы из порядка
I	I	I	I	MSD	Пересылка со сдвигом

Формат

Код			Мнемоника	Наименование
21	20	19		
0	0	0	B	Байт
0	0	1	W	Слово
0	1	0	L	Длинное слово
0	1	1	Q	Четверное слово
1	0	0	F	F-формат
1	0	1	R	Резервный формат
1	1	0	D	D-формат
1	1	1	G	G-формат

Адреса регистров

Код			Мнемоника	Наименование
18 15	17 14	16 13		
0	0	0	MA	Регистр мантиссы А
0	0	1	MB	Регистр мантиссы В
0	1	0	SDV	Сдвигатель
0	1	1	BPOR	Блок порядка
1	0	0	MNM	Множимое
1	0	1	MNT	Множитель
1	1	0	DLT	Делитель
1	1	1	BPR	Буфер приема/выдачи

Код			Мнемоника	Наименование
I2	II	I0		
0	0	0	NOP	Нет операции
0	0	I	NOR	Нормализация
0	I	0	SDV	Сдвиг
0	I	I	ADD	Сложение
I	0	0	SUB	Вычитание
I	0	I	MULS	Умножение знаковое
I	I	0	MULK	Умножение кодовое
I	I	I	DIV	Деление

Запись состояния

Код		Мнемоника	Наименование
05	04		
0	0		Нет записи
0	I	ZNZ	Запись кода ветвления
I	0	ZV	Запись V
I	I	ZS	Запись N и Z

Функции признака FS (МК 09)

MOV & FS – сдвиг вправо на 32 разряда

MSD & FS – сдвиг множимого вправо в командах MULB и MULW перед умножением

MRL & FS – выделение целой части

MPS & FS – выделение дробной части

MSP&FS – используется во второй части команды ACB для установки порядка и знака перед вычитанием

MWP2&FS – используется в команде EMOD для записи в блок умножения множителя после расширения.

Функции признака FN (МК 06)

FN всегда вызывает инверсию знака ПЗ.

MOV&FN – используется в команде EMOD для установки признака V по факту переполнения ПЗ при умножении. Кроме того устанавливается блокировка нормализации.

MWP2&FN – блокировка нормализации после умножения в команде POLY .

MPS&FN – вычитание константы из порядка делителя в командах DIVF , DIVD и DIVG .

MSP&FN – используется в команде POLY для установки порядка и знака перед сложением.

Прочие признаки:

FA – запись в аккумулятор.

PD – прием данных.

SA – выдача состояния в процессор.

Структурная схема блока микропрограммного управления представлена на рис.8.

Каждая микрокоманда, поступающая на центральный процессор (ПРЦ) вместе с сигналами сопровождения поступает и на сопроцессор. Сигналы сопровождения MMAS, MSDS поступают на схему синхронизации (SS) БМУ. Разряды микрокоманды MCI-MC25 поступают на регистр

микрокоманд один (RMC1), а разряды MC26-MC31 на дешифратор формата микрокоманды (DFM). Если формат микрокоманды относится к формату микрокоманды сопроцессора, то DFM вырабатывает управляющий сигнал, который при наличии сигналов сопровождения приема микрокоманды (MMAS, MSDS) запускает SS.

Схема синхронизации БМУ вырабатывает импульс записи микрокоманды в RMC1. Если микрокоманда не является последней микрокомандой микропрограммы СПРЦ, то одновременно с записью в RMC1, в центральном процессоре взведется признак торможения приема следующей микрокоманды (внешним признаком торможения приема микрокоманды является отсутствие снятия сигнала MMAS центральным процессором), который сбрасывается сигналом сопроцессора ACRA, сообщающим, что сопроцессор микрокоманду принял. Если к моменту записи микрокоманды регистр микрокоманд два (RMC2) свободен, то схема SS вырабатывает сигнал переписи микрокоманды из RMC1 в RMC2. Одновременно с переписью, если эта микрокоманда не является последней микрокомандой микропрограммы сопроцессора, СПРЦ выставляет сигнал ACRA. Сигнал ACRA имеет постоянную длительность, равную одному периоду тактовой частоты. Центральный процессор обязан принять этот синхронный сигнал, в противном случае поведение процессорного модуля непредсказуемо. Микрокоманда с выхода RMC2 поступает на дешифратор микрокоманды (DMC). Если к моменту записи микрокоманды в RMC2 один из исполнительных блоков СПРЦ занят выполнением предшествующей микрокоманды в соответствии с ее КОП1 или КОП2, или же текущая микрокоманда требует чтения входного буфера операндов, а он пуст, то дальнейшая работа SS будет заторможена одним из сигналов, поступающим по шине управления *INCONT*.

Независимо от этого торможения следующая микрокоманда будет принята на регистр *RMC1*, если это микроко-

манда СПРЦ. Однако, если занят RMC2 , то сигнал ACRA не вырабатывается, что в свою очередь затормозит прием последующей микрокоманды. Если торможения нет,или как только снимется торможение, схема синхронизации вырабатывает управляющие сигналы, которые обеспечивают выполнение первой фазы микрокоманды в соответствии с ее КОП1 и адресами чтения и записи, а именно загрузку соответствующих регистров ОБ, БУМ, БОП.

Схема SS вырабатывает управляющие сигналы на DMC, регистр хранения (RST), а также сигнал управления чтением/подбросом динамической магистралью MD (0-63). DMC вырабатывает управляющие сигналы на основании КОП1,КОП2, АЧТ, АЗП, формата данных, кода записи состояний, а также вспомогательные признаки. Как только заканчивается запись в регистр исполнительного устройства СПРЦ по адресу записи (заканчивается первая фаза выполнения микрокоманды), то в случае загруженного RMC1, разрешается перепись новой микрокоманды из RMC1 в RMC2 и вырабатывается внешний сигнал ACRA.

Если RMC1 не загружен, то запись в RMC2 произойдет по мере прихода новой микрокоманды. Если КОП1 требует сдвига операнда или КОП2 отличен от NOP (нет операции), то исполнительная часть СПРЦ приступает к выполнению второй фазы микрокоманды.Однако RMC2 загружен уже новой микрокомандой. Поэтому необходим регистр хранения (RST), на котором определенные признаки текущей микрокоманды хранятся до конца выполнения текущей микрокоманды. Если для выполнения микрокоманды требуется только одна первая фаза, то после записи операнда по АЗП выполнение микрокоманды заканчивается и разрешается выполнение следующей микрокоманды. Если присутствует вторая фаза выполнения, то начало выполнения следующей микрокоманды тормозится до конца выполнения текущей микрокоманды. При поступлении первой микрокоманды микропрограммы сопроцессора на

ПРЦ и СПРЦ, вывод АСС процессора из отключенного состояния переключается в активное состояние, т.е. по нему передается признак разрешения плавающего переполнения FU , а вывод АСС СПРЦ из активного состояния переключается в отключенное состояние, т.е. переключается на прием FU.

По приходе последней микрокоманды (признак SA) микропрограммы СПРЦ, после того как процессор получит сигнал ACRA от предшествующей микрокоманды, в процессоре устанавливается разрешение на работу схемы приема состояний, а вывод АСС переключается на прием бита С от СПРЦ. При этом торможение на прием следующей микрокоманды не устанавливается. Сигнал ACRA от последней микрокоманды выдается лишь с началом выполнения первой фазы этой микрокоманды. Одновременно с этим SS вырабатывает сигнал переключения вывода АСС СПРЦ из пассивного состояния в состояние передачи бита С. С приходом сигнала ACRA на ПРЦ от последней микрокоманды микропрограммы сопроцессора, в ПРЦ запускается схема приема состояний и формирования кода ошибки.

Внешний сигнал ACR - сигнал фатальной ошибки в процессорном модуле. Если на входе ACR установился низкий уровень, то SS вырабатывает сигнал управления, по которому происходит начальная установка сопроцессора. При этом, если сопроцессор был занят выполнением команды, то с появлением сигнала ACR выполнение команды искусственно прекращается. Кроме этого, ACR объединен по схеме ИЛИ с сигналом NRI интерфейсного блока, формируя сигнал сброса SS БМУ.

В случае возникновения фатальной ошибки, работа SS БМУ прекращается сигналом ACR. Однако, если по окончании выполнения очередной команды процессор уходит на обработку прерывания, а следующая команда была командой сопроцессора, то

прежде чем будет абортирован поток микрокоманд, вызванный следующей командой, сопроцессор может принять одну или две микрокоманды. Если в такой ситуации не сбросить SS БМУ, то первая же команда сопроцессора вызовет непредсказуемое поведение процессорного модуля.

Любая микропрограмма обработки прерывания включает в себя чтение архитектурных регистров (например, регистр SCBB), которые физически расположены в областях системной памяти. Интерфейсный блок содержит дешифратор адреса системной памяти, и как только на магистрали AD появится один из адресов системной памяти CI0001XX, CI0002XX, ..., CI000FXX, то интерфейсный блок вырабатывает импульс NRI, по которому происходит сброс SS БМУ.

На рис 22 приведена временная диаграмма работ БМУ при выполнении команды MULB2, где:

CLC – сигнал тактовой частоты;

AD – 32-разрядная магистраль адрес/данные;

MAS, SAS, MDS, SDS – сигналы сопровождения обмена по магистрали AD;

A_{MC} – магистраль адреса микрокоманды;

MC – магистраль микрокоманд;

MMAS, MSDS – сигналы сопровождения приема микрокоманды;

WRMC1, WRMC2 – сигналы записи в регистры БМУ СПРЦ RMC1 и RMC2;

RD, WT – сигналы сопровождения (чтение и запись) при выполнении первой фазы микрокоманды СПРЦ;

TORM – обобщенный сигнал торможения начала выполнения микрокоманды СПРЦ;

SACC СПРЦ и SACC ПРЦ – сигналы управления приемом и передачей по выводу ACC.

Временная диаграмма выполнения команды MULB2 (RN)+

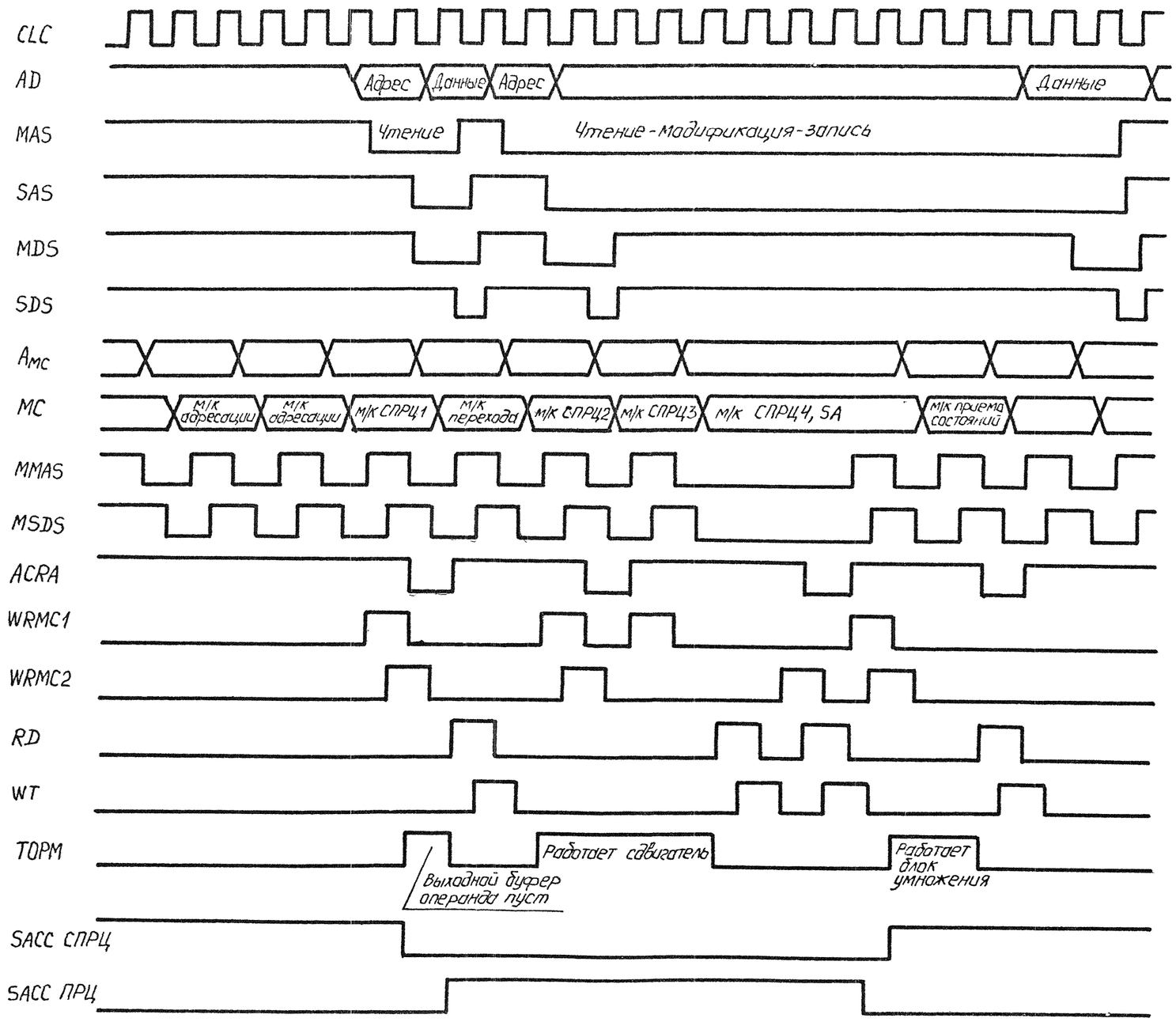


Рис.22

6. ВЫПОЛНЕНИЕ СОПРОЦЕССОРОМ ПЕРВОЙ ФАЗЫ МИКРОКОМАНДЫ, ОПРЕДЕЛЯЕМОЙ ПОЛЕМ КОП1

В данной главе рассматриваются операции над операндами, не акцентируя внимания на обработку знака. Обработка знака будет рассмотрена в отдельной главе.

Выполнение сопроцессором второй фазы микрокоманды, определяемой полем КОП2, рассматривается при описании алгоритмов выполнения команд.

6.1. *MOV* без признаков

Данные читаются на магистраль *MD* по адресу АЧТ и записываются по адресу АЗП без всяких преобразований.

6.2. *MOV* с признаком *FS*

Данные, записанные на регистр *REGS*, сдвигаются на 32 разряда вправо. Сдвиг начинается после импульса записи по адресу АЗП.

REGS (32-63) ⟶ REGR (0-31)

0 ⟶ REGR (33-63)

X (неопределенно) ⟶ REGR (32)

REGR ⟶ RMA

6.3. *MOV* с признаком *FN*

Данные читаются на магистраль *MD* по адресу АЧТ и записываются по адресу АЗП без всяких преобразований. Кроме этого по КОП1 *MOV* и признаку *FN* вводится флаг *EMOD*, который в командах *EMODF*, *EMODD*, *EMODG* обеспечивает блокировку округления после умножения, расширение на один разряд мантиссы остатка, а также в случае возникновения плавающего переполнения при умножении устанавливает признак целого переполнения.

6.4. MNS

Микрокоманды сопроцессора, КОП1 которых равен MNS , используются в микропрограммах преобразования чисел из формата ПЗ в целочисленный формат.

Данные читаются на магистраль MD по адресу АЧТ и записываются по адресу АЗП. Если число в формате ПЗ было положительным (признак ZNAK=0), то на этом выполнение операции, определяемой КОП1, заканчивается. Если число в формате ПЗ было отрицательным (признак ZNAK=1), то после импульса записи по адресу АЗП запускается ALU ОБ, на котором формируется дополнительный код целого числа, хранящегося в регистре RMA.

$$(\overline{RMA}+1) \rightarrow REGS \rightarrow REGR \rightarrow RMA$$

6.5. MNN

Микрокоманды сопроцессора, КОП1 которых равен MNN , используются в микропрограммах преобразования чисел из целого формата в формат ПЗ .

Данные читаются на магистраль MD по адресу АЧТ и записываются по адресу АЗП. Если целое число, читаемое на магистраль MD является положительным числом (MD63=0), то на этом выполнение операции, определяемой КОП1, заканчивается. Если целое число, читаемое на магистраль MD, является отрицательным числом (MD63=1), то после импульса записи по адресу АЗП запускается ALU ОБ, на котором формируется прямой код отрицательного числа, хранящегося в регистре RMA .

$$(\overline{RMA}+1) \rightarrow REGS \rightarrow REGR \rightarrow RMA$$

6.6. RND

Микрокоманды сопроцессора, КОП1 которых равен RND , используются в микропрограммах, где требуется округление чисел.

Данные читаются на магистраль MD по адресу АЧТ и записываются по адресу АЗП. Если разряды операндов, по которым определяют необходимость округления равны нулю, то на этом выполнение операции, определяемой КОП1, заканчивается. В противном случае после импульса записи по адресу АЗП запускается ALU ОБ, на котором происходит сложение операнда, хранящегося в регистре RMA с константой округления.

$$RMA + 00000000800000000_{I6} \longrightarrow REGS \longrightarrow REGR \longrightarrow RMA$$

– округление целого числа в формате двойное слово.

Данная операция используется в микропрограммах команд преобразования чисел из формата ПЗ в целое число формата двойное слово с округлением (CVTRFL, CVTRDL, CVTRGL)

$$RMA + 0000004000000000_{I6} \longrightarrow REGS \longrightarrow REGR \longrightarrow RMA$$

– округление мантиссы числа плавающего формата "F".

Данная операция используется в микропрограммах команд преобразования чисел из " D ", " G " плавающего формата в плавающий формат " F " (CVTDF , CVTGF).

6.7. MSP без признаков

Микрокоманды сопроцессора, КОП1 которых равен MSP , используются в микропрограммах, где требуется сложение или вычитание чисел в формате ПЗ .

Для нормального выполнения данной операции необходимо, чтобы перед этой микрокомандой мантисса первого операнда была записана в регистр ОБ RMA , а порядок – в регистры БОП RPI и RPOR.

Мантисса второго операнда по адресу АЧТ, равному 111_2, считывается с входного буфера операндов на магистраль MD и по адресу АЗП, равному 001_2, записывается в регистры ОБ RMB,

REGS, REGR . По КОП1 MSP при отсутствии признаков FS или FN порядок второго операнда из входного буфера операндов переписывается в регистр БОП RP2, после чего аппаратно выполняется операция выравнивания порядков.

$$RP1 - RP2 \longrightarrow RPSD \longrightarrow RP1$$

Если число в RPSD имеет положительный знак, то:

$$REGS \xrightarrow[\text{на } RPSD]{\text{сдвиг вправо}} REGR \longrightarrow REGS, RMB$$

Если число в RPSD имеет отрицательный знак, то:

$$RP2 \longrightarrow RPOR$$
$$RMA \longrightarrow REGS \longrightarrow REGR$$
$$(\overline{RP1} + 1) \longrightarrow RPSD \longrightarrow RP1$$
$$REGS \xrightarrow[\text{на } RPSD]{\text{сдвиг вправо}} REGR \longrightarrow REGS, RMA$$

По окончании сдвига производится пересылка

$$RPOR \longrightarrow RP1$$

В результате выполнения операции, определяемой КОП1 MSP, мантисса первого операнда будет расположена на регистре RMA мантисса второго операнда – на регистре RMB , наибольший из двух порядков – на регистрах RP1 и RPOR.

6.8. MSP с признаком FS

Выполняются те же операции, что и в случае MSP без признаков, за исключением того, что мантисса второго операнда в соответствии с адресом АЧТ, равным 101_2, считывается из регистра ОБ RAK, а порядок – из регистра БОП RAP.

6.9. MSP с признаком FN

Выполняются те же операции, что и в случае MSP без признаков, за исключением того, что на фоне записи порядка второго операнда из входного буфера операндов в регистр RP2 , в регистр RP1 происходит запись порядка из регистра RPOR.

6.10. *MIF* , *MID* , *MIG* с целочисленным форматом

Микрокоманды сопроцессора, КОП которых равен *MIF* , *MID*, *MIG*, формат операнда которых – целое число, используются в микропрограммах команд преобразования целых чисел в числа ПЗ .

Данные, как целое 64-разрядное число со знаком, читаются на магистраль *MD* из входного буфера операндов в соответствии с АЧТ, равным 111_2, и записываются по АЗП, равным 000_2, в регистр RMA.

В регистр БОП RPI записывается константа.

Величины констант следующие:

КОП	Формат	Константа
MIF, MID	*B*	087_{I6}
MIF, MID	*W*	$08F_{I6}$
MIF, MID	*L*	$09F_{I6}$
MIG	*B*	407_{I6}
MIG	*W*	$40F_{I6}$
MIG	*L*	$41F_{I6}$

6.11. *MIF* с форматом ПЗ " *G* " и *MIG* с форматом ПЗ "F"

Микрокоманды, с указанными полями, используются в микропрограммах команд преобразования *CVTFG* и *CVTGF*

Пересылка мантисс в данных микрокомандах имеет символическое значение (пересылка содержимого регистра самого в себя).

В БОП выполняются следующие действия:

RPI – хранит порядок исходного операнда без смещения, который был получен в предшествующей микрокоманде;

const ⟶ RP2

RP1 + RP2 ⟶ RPOR, RPSD ⟶ RP1

const = 80_{16} - для MIG+F

const = 400_{16} - для MIF+G

6.I2. MWP2 без признаков

Мантисса числа в формате ПЗ по адресу АЧТ, равному III_2, считывается с входного буфера операндов на магистраль MD и записывается в регистр по адресу АЗП. Порядок числа переписывается из входного буфера операндов в регистр БОП RP2.

6.I3. MWP2 с признаком FS

Выполняются те же операции, что и в случае MWP2 без признаков, за исключением того, что запись порядка в RP2 не производится.

6.I4. MWP2 с признаком FN

Выполняются те же операции, что и в с учае MWP2 без признаков. Кроме этого, по КОПI MWP2 и признаку FN набрасывается флаг блокировки нормализации, который используется во второй фазе выполнения микрокоманды, определяемой полем КОП2.

6.I5. MRB , MRW, MRL без признаков

Микрокоманды сопроцессора, КОПI которых равен MRB , MRW, MRL без признаков, используются в микропрограммах команд преобразования чисел в формате ПЗ в числа целочисленного формата.

Мантисса числа в формате ПЗ по адресу АЧТ, равному III_2, считывается с входного буфера операндов на магистраль MD и в соответствии с адресом АЗП, равному 000_2, записывается в регистр ОБ РМА. Порядок числа переписывается из входного буфера операндов в регистр БОП RPI. В регистр RP2 записывается константа согласно табл.5.

Таблица 5

КОП I	Формат	Константа
MRB	F, D	087_{I6}
MRB	G	407_{I6}
MRW	F, D	$08F_{I6}$
MRW	G	$40F_{I6}$
MRL	F, D	$09F_{I6}$
MRL	G	$41F_{I6}$

6.I6. MRL с признаком FS

Микрокоманда сопроцессора, КОП I которой равен MRL , а также присутствует признак FS , используется в микропрограммах команд EMODF , EMODD , EMODG для выделения целой части из расширенного произведения.

Мантисса произведения по адресу АЧТ считывается на магистраль MD и записывается по адресу АЗП. Запись в регистр порядка RP1 блокируется признаком FS . В регистр RP2 записывается константа. Значения констант для КОП I MRL указаны в табл. 5.

6.I7. MWP1

Мантисса числа в формате ПЗ по адресу АЧТ, равному III_2, считывается с входного буфера операндов на магистраль MD и записывается в регистр в соответствии с адресом АЗП. Порядок числа переписывается из входного буфера операндов в регистр БОП RPI.

6.I8. MPA

Мантисса числа по адресу АЧТ, равному OIO_2, считывается

с регистра *REGR* на магистраль *MD* и записывается по адресу АЗП. Причем, при считывании мантиссы с регистра *REGR*, те. разряды мантиссы, которые выходят за разрядную сетку соответствующего формата, обнуляются. Кроме того, в регистр БОП RPI переписывается порядок из регистра *RAP*, в регистр БОП RP2 переписывается порядок из регистра *RPOR*.

6.I9. *MPS* без признаков

Мантисса числа по адресу АЧТ, равному III_2, считывается с входного буфера операндов на магистраль *AD* и записывается в регистр в соответствии с адресом АЗП. Порядок числа переписывается из входного буфера операндов в регистр БОП RPI. В регистр *RP2* записывается константа 080_{I6} - для *F*- и *D*- формата или 400_{I6} - для *G*- формата. После этого из порядка вычитается смещение *RP1 - RP2 ⟶ RPSD, RPOR ⟶ RP1*

6.20. *MPS* с признаком *FS*

Микрокоманды с такими полями используются в микропрограммах выполнения команд *EMODF*, *EMODD*, *EMODG* при выделении дробной части.

Мантисса результата умножения по адресу АЧТ, равному OOI_2, считывается с регистра *RMB* на магистраль *MD* и в соответствии с адресом АЗП, равному OIO_2, записывается в регистры *REGR* и *REGS*. В регистр БОП RPI переписывается из регистра *RPOR* порядок произведения, в регистр *RP2* записывается константа 080_{I6} - для *F*- и *D*- формата, 400_{I6} - для *G*- формата.

После этого выполняется вычитание

RP1 - RP2 ⟶ RPSD ⟶ RP1

Если результат вычитания - отрицательное число, то целая часть произведения равна нулю. В этом случае бло-

кируется операция сдвига, которая должна быть выполнена в этой микрокоманде по КОП2, а в регистре *RP1* восстанавливается исходный порядок.

$$RPOR \rightarrow RP1$$

Если результат вычитания – положительное число, то в соответствии с КОП2 этой микрокоманды, выполняется сдвиг влево мантиссы произведения на величину содержимого *RPSD* (выделение дробной части результата умножения).

По окончании сдвига в регистр *RP1* записывается константа 080_{I6} для *F*- и *D*- формата или 400_{I6} – для *G*- формата.

6.21. *MPS* с признаком *FN*

Микрокоманды с такими полями используют в микропрограммах команд деления чисел в формате П3 для загрузки делителя. Выполнение этих микрокоманд отличается от выполнения микрокоманд с полями *MPS* без признаков тем, что на БОП выполняется операция *RP1-RP2-1* вместо операции *RP1-RP2*.

6.22. *MPS* с признаком *FA*

Микрокоманды с такими полями используются в микропрограммах команд вычисления полинома для загрузки аргумента. Выполнение этих микрокоманд отличается от выполнения микрокоманд с полями *MPS* без признаков тем, что результат операции БОП *RP1-RP2* записывается не только в регистры *RPSD*, *RPOR* и *RP1*, но и в регистр *RAP*. Кроме этого, мантисса аргумента помимо записи в регистр согласно адресу АЗП по признаку *FA*, записывается в регистр ОБ RAK.

6.23. *MSD* с форматами *B*, *W*, *L* и признаком *FA*

Микрокоманды с такими полями используются в микропрограммах команд деления *DIVB*, *DIVW*, *DIVL* для загрузки и сдвига делимого. Целочисленное делимое по адресу АЧТ, равному III_2, считывает-

ся из входного буфера операндов на магистраль MD и по адресу АЗП, равному 000_2, записывается в регистры REGS, REGR, RMA. В регистр RMB записывается ноль. Кроме этого, по признаку FA делимое записывается в регистр RAK с размножением знака (с 32 по 63 разряды).

В регистр БОП RPI из регистра RPOR переписывается параметр нормализации делителя, полученный в предшествующей микрокоманде. В регистр RP2 записывается константа 008_{I6} для формата B, 010_{I6} для формата W и 020_{I6} для формата L. После этого БОП выполняет операцию

$$RP2 - RP1 \longrightarrow RPSD \longrightarrow RP1,$$

а ОБ выполняет сдвиг вправо делимого на величину результата вычитания, находящегося в регистре RPSD.

6.24. MSD с форматом Q и признаком FA

Микрокоманда с такими полями используется в микропрограмме команды EDIV для загрузки и сдвига делимого. Целочисленное делимое по адресу АЧТ, равному 111_2, считывается из входного буфера операндов на магистраль MD и в соответствии с адресом записи АЗП, равным 000_2, записывается в регистры REGS, REGR, RMA. В регистр RMB записывается ноль. Кроме этого, младшие 32 разряда делимого записываются в регистр RAK с размножением знака (с 32 по 63 разряды). В регистр БОП RPI из регистра RPOR переписывается параметр нормализации делителя, полученный в предшествующей микрокоманде.

В регистр RP2 записывается константа ноль. После этого БОП выполняет операцию вычитания

$$RP1 - RP2 \longrightarrow RPSD \longrightarrow RP1,$$

на основании которой ОБ выполняет сдвиг делимого влево на величину содержимого RPSD

6.25. *MSD* с форматом *R*

Микрокоманда с такими полями используется в микропрограмме команды *EDIV* для сдвига остатка. Пересылка остатка в данной микрокоманде имеет символическое значение (остаток из регистра *REGR* переписывается в регистр множимого).

В регистр БОП RPI из регистра *RPOR* переписывается параметр нормализации делителя. В регистр *RP2* записывается константа ноль. После этого БОП выполняет операцию вычитания

$$RP1 - RP2 \longrightarrow RPSD \longrightarrow RP1,$$

на основании которой ОБ выполняет сдвиг остатка вправо на величину содержимого *RPSD*.

6.26. *MSD* с признаком *FS*

Микрокоманды с такими полями используются в микропрограммах команд целочисленного умножения *MULB*, *MULW* для загрузки и сдвига множителя.

Целочисленный множитель по адресу АЧТ, равному 111_2, считывается с входного буфера операндов и в соответствии с адресом АЗП, равным 010_2, записывается в регистры *REGS*, *REGR*. В регистр БОП *RP2* загружается константа 018_{16} для формата байт или 010_{16} для формата слово. После этого БОП выполняет операцию пересылки

$$RP2 \longrightarrow RPSD \longrightarrow RP1,$$

на основании которой ОБ выполняет сдвиг множителя вправо на величину содержимого *RPSD*.

7. АЛГОРИТМЫ ФОРМИРОВАНИЯ ЗНАКА

Схема формирования знака входит в схему управления работой ALU ОБ. На схеме формирования знака формируются следующие признаки операндов в формате ПЗ.

ZN1 – знак первого операнда;

ZN2 – знак второго операнда;

ZNADD – знак слагаемого;

ZNRAK – знак операнда, хранящегося на аккумуляторе;

ZNREZ – знак результата операции;

ZNAK – знак результата команды.

Признаки ZN1, ZN2, ZNADD, ZNRAK, ZNREZ формируются на D-триггерах, логические функции для входов C и D которых имеют следующий вид:

$$CZN1 = (MPA + MPS + MWP1 + MRB + MRW + MRL)\,\&\,\overline{FS} + MPA + ((MSP\,\&\,(FS + FN))$$

$$DZN1 = \overline{MPA}\,\&\,\overline{((MSP\,\&\,(FS + FN))}\,\&\,ZNBX + MPA\,\&\,ZNRAK + ZNREZ\,\&\,((MSP\,\&\,(FS + FN))$$

$$CZN2 = (MWP2 + MSP)\,\&\,\overline{FS} + MPA + MSP\,\&\,FS$$

$$DZN2 = \overline{MPA}\,\&\,\overline{(MSP\,\&\,FS)}\,\&\,ZNBX + MPA\,\&\,ZNREZ + MSP\,\&\,FS\,\&\,ZNRAK$$

$$CZNADD = ADD$$

$$DZNADD = ZN1$$

$$CZNRAK = ZPAK$$

$$DZNRAK = ZNBX$$

$$CZNREZ = (MIF + MID + MIG)\,\&\,\overline{F467} + (MRB + MRW + MRL)\,\&\,\overline{FS} + MWP1 + MSP\,\&\,\overline{FS} + ((FUMN + FDEL)\,\&\,SSP2) + ((FADD + FSUB)\,\&\,SSUM)$$

$$DZNREZ = ((MIF + MID + MIG)\,\&\,\overline{F467}\,\&\,MD63) + ((((MRB + MRW + MRL)\,\&\,\overline{FS}) + MWP1 + (MPS\,\&\,\overline{FS}))\,\&\,ZNBX) + (((FUNN + FDEL)\,\&\,SSP2)\,\&\,(ZN1 \oplus ZN2)) + (((FADD + FSUB)\,\&\,A63\,\&\,\overline{VM}) + (FADD\,\&\,ZN1\,\&\,ZN2) + (FSUB\,\&\,\overline{ZN1}\,\&\,ZN2))$$

Триггеры ZN1 и ZN2 сбрасываются импульсом NRN

$$ZNAK = ZNREZ \oplus FN$$

MPA, MPS, MWP1, MRB, MRW, MRL, MSP, MWP2, MIF, MID, MIG – сигналы БМУ, вырабатываемые в соответствии с КОП1 микрокоманды, длительностью в один период тактовой частоты.

ADD – сигнал БМУ, вырабатываемый в соответствии с КОП2, *ADD* микрокоманды, длительностью в один период тактовой частоты.

ZPAK – сигнал БМУ, вырабатываемый в соответствии с признаком FA микрокоманды, длительностью в один период тактовой частоты.

Все перечисленные сигналы формируются точно так же, как формируется сигнал *WT*, показанный на рис. 22.

FS, FN – выходы регистра *RMC2* БМУ, соответствующие признакам *FS* и *FN* микрокоманды.

ZNBX – выход разряда входного буфера операндов, соответствующего знаку числа в формате ПЗ.

MD63 – старший разряд магистрали *MD*, соответствующий знаку числа.

F467 – выход *DMC* БМУ, соответствующего плавающему формату данных в микрокоманде (формат *F* или *D*, или *G*).

FADD, FSUB – сигналы схемы управления работой ALU ОБ, вырабатываемые в соответствии с КОП2 *ADD*, *SUB* микрокоманды. Данные сигналы устанавливаются импульсами *ADD*, *SUB* БМУ, а снимаются по окончании работы ALU.

A63 – старший (знаковый) разряд регистра *REGS*,

VM – признак арифметического переполнения. Формируется на выходе ALU ОБ.

SSUM – сигнал записи результата операции, выполненной ALU ОБ из *REGS* в *REGR*.

FUMN, FDEL – сигналы схем управления работой блока умножения и блока деления, вырабатываемые в соответствии с КОП2 *MUL* и *DIV* микрокоманды. Данные сигналы устанав-

ливаются импульсами MUL , DIV БМУ , а снимаются по окончании работы умножителя или делителя соответственно.

SSP2 – сигнал запуска ALUP БОП.

7.I. Формирование знака результата

В командах пересылок MOV(F,D,G) и MNEG (F,D,G),тестирования TST(F,D,G) и преобразования плавающего формата CVTFD и CVTDF знак формируется в результате записи в триггер ZNREZ сигнала ZNBX по импульсу MWP1 . При этом для команд MOV (F,D,G), TST(F,D,G), CVTFD и CVTDF ZNAK=ZNREZ, а для команд MNEG(F,D,G) ZNAK=$\overline{ZNREZ}$, так как в микрокомандах выдачи результата в выходной буфер операндов микропрограмм этих команд присутствует признак FN .

В командах преобразования целых чисел в числа с плавающей запятой CVTB(F,D,G) , CVTW(F,D,G) и CVTL (F,D,G) знак формируется в результате записи в триггер ZNREZ сигнала MD63 по импульсу (MIF, MID, MIG)&$\overline{F468}$. Знак результата команд ZNAK=ZNREZ.

В командах преобразования чисел в формате П3 в целочисленный формат CVTF (B,W,L), CVTRFL , CVTD (B,W,L), CVTRDL, CVTG (B,W,L) и CVTRGL знак целого числа результата определяется по значению признака ZNAK . Сигнал ZNBX по импульсу (MIF , MID , MIG)&$\overline{FS}$ записывается в триггер ZNREZ . Так как в микропрограммах этих команд нет признака FN , то ZNAK= = ZNREZ . Если ZNAK = I, то в последующей микрокоманде,поле КОПI которой MNS , произойдет преобразование положительного числа в отрицательное число в дополнительном коде.

В командах преобразования чисел в формате П3 CVTFG и CVTGF знак формируется в результате записи в триггер ZNREZ

сигнала ZNBX по импульсу MPS& $\overline{FS}$. Знак результата команд ZNAK=ZNREZ.

При выполнении команд сложения/вычитания чисел в формате ПЗ ADD (F,D,G), SUB (F,D,G) и CMP(F,D,G) знак первого операнда ZNBX по импульсу MWP1& $\overline{FS}$ записывается в триггер ZN1 , а знак второго операнда ZNBX по импульсу MSP& $\overline{FS}$ записывается в триггер ZN2 . После приема операндов и выравнивания порядков на ALU ОБ выполняется операция сложения/вычитания. Если старший разряд результата операции сложения/вычитания A63 = I и не было переполнения, или была операция сложения двух отрицательных чисел или операция вычитания из отрицательного числа положительного, то по импульсу записи (FADD + + FSUB)& SSUM на триггере взведется признак ZREZ=1 . При этом для команд сложения и вычитания ZNAK=ZREZ , а для команд сравнения ZNAK= $\overline{ZREZ}$, так как в микрокомандах выдачи результата в выходной буфер операндов в микропрограммах команд сравнения присутствует признак FN .

При выполнении команд умножения и деления чисел в формате ПЗ MUL(F,D,G), DIV(F,D,G) знак первого операнда ZNBX по импульсу MPS& $\overline{FS}$ записывается в триггер ZN1 , а знак второго операнда ZNBX по импульсу MWP2& $\overline{FS}$ записывается в триггер ZN2 . После приема операндов в соответствии с КОП2 микрокоманды запускаются блок умножения или блок деления. Кроме этого по импульсу SSP2 запускается ALUP БОП, на котором происходит сложение (умножение) или вычитание (деление) порядков операндов. Одновременно с этим по импульсу (FUMN + FDEL)& SSP2 в триггер ZNREZ записывается знак результата операции ZN1 ⊕ ZN2 . Знак результата для этих команд ZNAK = ZNREZ .

При выполнении команд расширенного умножения чисел в формате ПЗ с выделением из результата умножения целой и дробной

части EMOD (F,D,G) мантисса операнда множителя пересылается в регистр REGR, а знак ZNBX по импульсу MWP2&$\overline{FS}$ записывается в триггер ZN2. После этого мантисса операнда множителя пересылается в ПРЦ, где к ней добавляется операнд расширения мантиссы. Расширенная мантисса операнда множителя пересылается из ПРЦ в СПРЦ. Мантисса операнда множимого записывается в регистр RMNTL, а знак ZNBX по импульсу MPS&$\overline{FS}$ записывается в триггер ZN1. Расширенная мантисса множителя записывается в регистр RMNM, после чего в соответствии с КОП2 MUL следующей микрокоманды выполняется операция умножения. Точно так же как и в командах MUL(F,D,G) в триггер ZNREZ по импульсу (FUMN+FDEL)&SSP2 записывается знак операции ZN1 ⊕ ZN2. Знак результата для этих команд ZNAK=ZNREZ.

Если ZNAK=1, то после выделения целой части результата в микрокоманде, поле КОП1 которой MNS, произойдет преобразование положительного числа в отрицательное число в дополнительном коде. Знак дробной части результата будет равен ZNAK.

В командах, работающих с целочисленными операндами, MUL(B,W,L), EMUL, DIV(B,W,L), EDIV и ASHQ знак результата получается автоматически, так как исполнительные блоки сопроцессора ALU, SDV, BDEL, BUM работают с дополнительными кодами чисел. В этом случае знак представляет собой старший разряд результата.

7.2. Запись и восстановление знаков

В командах, описанных в предшествующем п.7.1, есть только однократная запись знака операнда и знака результата.

В команде ACB(F,D,G) знак операнда-предел ZNBX по импульсу ZPAK записывается в триггер ZNRAK, знак операнда-слагаемого ZNBX по импульсу MWP1&$\overline{FS}$ записывается в триггер ZN1, знак операнда-индекс ZNBX по импульсу MSP&$\overline{FS}$ записы-

вается в триггер ZN2 . Кроме этого, в микрокоманде приема операнда индекс по КОП2 ADD формируется импульс записи знака операнда слагаемое ZN1 в триггер ZNADD . После приема операндов и выравнивания порядков на ALU ОБ выполняется операция сложения операнда-индекс с операндом-слагаемое, при этом признаки ZNREZ и ZNAK формируются точно так же, как при выполнении команд сложения чисел в формате ПЗ .

В следующей за микрокомандой сложения микрокоманде, результат сложения выдается на выходной буфер операндов с записью состояний. После этого выполняется микрокоманда вычитания из операнда-предел результата предшествующего сложения. По импульсу записи MSP&FS в триггер ZN1 переписывается знак результата сложения ZNREZ , а в триггер ZN2 переписывается знак операнда-предел ZNRAK . Мантисса операнда-предел считывается с регистра ОБ RAK и записывается в регистр RMB (результат предшествующего сложения расположен на регистре RMA), после чего выполняются операции выравнивания порядков и вычитание. Признаки ZNREZ и ZNAK формируются точно так же, как и в командах вычитания чисел в формате ПЗ . В последней микрокоманде результат вычитания выдается в выходной буфер операндов, а по признакам ZNAK , ZNADD и признаку равенства нулю результата вычитания формируется код ветвления.

Выполнение команд вычисления полинома POLY(F,D,G) начинается с пересылки операнда аргумент из ПРЦ в СПРЦ и приема на сопроцессор из памяти коэффициента $C[\alpha]$, где α - показатель степени.

После этого ПРЦ анализирует операнд-показатель степени на равенство нулю. Если операнд показатель степени равен нулю, то в качестве результата выполнения команды вычисления полинома будет выступать коэффициент $C[0]$. В этом случае входной

буфер операндов СПРЦ освобождается от операнда аргумент , после чего СПРЦ выполняет те же самые операции, что и при выполнении команд MOV(F,D,G).

Если операнд показатель степени не равен нулю, то процессор организует прием на сопроцессор из памяти следующего коэффициента C[α-1], после чего на сопроцессоре выполняются операции умножения аргумента на коэффициент C[α] и сложения результата умножения с коэффициентом C[α-1].

Знак операнда аргумент ZNBX по импульсам MPS&$\overline{FS}$ и ZPAK записывается в триггера ZN1 и ZNRAK, знак коэффициента C[α] ZNBX по импульсу MWP2&$\overline{FS}$ записывается в тригтер ZN2 . После приема операндов сопроцессор в соответствии с КОП2 MUL микрокоманды выполняет операцию умножения, при этом признаки ZNREZ и ZNAK формируются точно так же как при выполнении команд умножения чисел в формате ПЗ . Знак результата умноже - ния ZNREZ по импульсу MSP&FN записывается в триггер ZN1, знак коэффициента C[α-1] ZNBX по импульсу MSP&$\overline{FS}$ записывается в триггер ZN2 . После приема коэффициента C[α-1] и выравнивания порядков на ALU ОБ в соответствии с КОП2 ADD микрокоманды выполняется операция сложения результата предшествующего умножения с коэффициентом C[α-1] , при этом признаки ZNREZ и ZNAK формируются точно так же, как при выполнении команд сложения чисел в формате ПЗ . На этом заканчивается первый цикл выполнения команд вычисления полинома.

По окончании первого цикла процессор анализирует оставшийся показатель степени (после первого цикла α-1) на равенство его нулю. Если он равен нулю, то сопроцессор выдает результат вычисления команды на выходной буфер операндов, в противном случае на сопроцессоре выполняется второй цикл выполнения команды, а именно прием на сопроцессор следующего коэффициента C[α-2],

умножение результата предшествующего цикла на операнд аргумент и сложение результата умножения с коэффициентом C[d-2]. Знак операнда аргумент ZNRAK по импульсу МРА записывается в триггер ZN1. Знак результата предшествующего цикла ZNREZ по импульсу МРА записывается в триггер ZN2. После этого в соответствии с КОП2 MUL микрокоманды сопроцессор выполняет операцию умножения, при этом признаки ZNREZ и ZNAK формируются точно так же как при выполнении команд умножения чисел в формате ПЗ.

Знак результата умножения ZNREZ по импульсу MSP& FN записывается в триггер ZN1, знак коэффициента C[d-2] ZNBX по импульсу MSP& $\overline{FS}$ записывается в триггер ZN2. После приема коэффициента C[d-2] и выравнивания порядков на ALU ОБ в соответствии с КОП2 ADD микрокоманды выполняется операция сложения результата предшествующего умножения с коэффициентом C[d-2], при этом признаки ZNREZ и ZNAK формируются точно так же, как при выполнении команд сложения чисел в формате ПЗ. На этом заканчивается второй цикл выполнения команд вычисления полинома. Второй цикл выполнения команд вычисления полинома повторяется до тех пор, пока в соответствии со значением операнда показатель степени не будут задействованы в вычислениях все коэффициенты, т.е. $d-n=0$ (n – номер цикла).

8. АЛГОРИТМЫ ВЫПОЛНЕНИЯ КОМАНД

8.1. Команда сдвига ASHQ

Первая микрокоманда – микрокоманда приема параметра сдвига.

Операнд – параметр сдвига загружается в RPI с расширением знака

;MOV :B/R(7) ⟶ R(3)_NOP; PDA

Вторая микрокоманда – микрокоманда приема сдвигаемого и сдвиг.

Сдвигаемое число в формате четверное слово из входного буфера операндов загружается в регистры REGS и REGR.

;MOV :Q/R(7) ⟶ R(2)_SDV; PDA,ZV

По импульсу SDV, формируемому на основе КОП2 SDV микрокоманды, запускается схема управления работой сдвигателя.

На сумматоре блока обработки порядка устанавливается режим пересылки RP1 ⟶ RPSD.

После выполнения пересылки анализируется знак параметра сдвига. Если он отрицательный, то на сумматоре блока обработки порядка устанавливается режим изменения знака. RPSD пересылается в RP1 и выполняется пересылка с изменением знака $\overline{RP1} \rightarrow RPSD$. Если изменения знака не было, то устанавливается признак сдвига влево, если было – то вправо. Далее выполняется сдвиг $REGS \xrightarrow{сдвиг} REGR \rightarrow REGS$. Если параметр сдвига больше I5, устанавливается код сдвига (CSD), равный I5. Во время сдвига CSD записывается в RP2, а параметр сдвига –в RPI. Между циклами сдвига выполняется вычитание

RP1 - RP2 ⟶ RPSD ⟶ RP1.

Так продолжается, пока параметр сдвига не станет меньше I5. Это говорит о том, что следующий цикл сдвига будет последним. По

окончании сдвига выполнение микрокоманды заканчивается. Выполнение заканчивается и в том случае, если на каком-то цикле сдвигаемое число становится равным нулю. В результате выполнения микрокоманды на REGR оказывается операнд, сдвинутый на заданное количество разрядов.

Третья микрокоманда - микрокоманда пересылки REGR в выходной буфер операндов

;MOV :Q/R(2) ⟶ R(7)_NOP; ZS,SA

Кроме пересылки, в этой микрокоманде выполняется запись состояний.

8.2. Команды преобразования действительных в целые числа

Все команды этого класса выполняются по одному алгоритму. Первая микрокоманда это MRB , MRW или MRL в зависимости от того, в какой формат происходит преобразование. В поле формата этой микрокоманды указывается из какого формата происходит преобразование

MRB(W,L) :F(D,G)/R(7) ⟶ R(0)_SDV;PDA,ZV

По этой микрокоманде мантисса записывается в RMA , REGS и REGR , а порядок - в RP1 . В RP2 записывается константа, которая равна порядку байта, слова или длинного слова, если их представить в формате П3. Величина констант приведена в п.6.I5. По импульсу SDV , определяемому КОП2 SDV микрокоманды, запускается схема управления работой сдвигателя.

На ALUP БОП выполняется операция вычитания

RP1-RP2 ⟶ RPSD ⟶ RP1.

Если параметр сдвига получился отрицательный, то устанавливается режим сдвига вправо и выполняется цикл изменения знака параметра сдвига. Если параметр сдвига положительный, то устанав-

ливается режим сдвига влево. Далее выполняется сдвиг точно также, как в команде сдвига. После выполнения сдвига, выполнение микрокоманды заканчивается. В результате в *REGR* сформировано число, которое для байта занимает 8 старших разрядов, для слова - I6 и для длинного слова - 32 разряда. Для целых форматов, кроме *Q*, результат должен находиться в младших 32 разрядах *REGR*. Поэтому следующая микрокоманда - это сдвиг вправо на 32 разряда.

MOV :B(W,L)/R(2) ⟶ R(2)_NOP;FS

Если знак исходного числа был отрицательный, то знак полученного целого числа тоже должен быть отрицательным. Поэтому следующая микрокоманда - микрокоманда изменения знака, если знак ПЗ отрицательный.

MNS :B(W,L)/R(2) ⟶ R(0)_NOP;

Последняя микрокоманда - микрокоманда пересылки из *REGR* в выходной буфер с записью состояний

MOV :B(W,L)/R(2) ⟶ R(7)_NOP; ZS,SA

Для команд преобразования с округлением *CVTRFL*, *CVTRDL* и *CVTRGL* перед микрокомандой сдвига на 32 разряда добавляется еще микрокоманда округления

RND :L/R(2) ⟶ R(0)_NOP;

По этой микрокоманде 64-разрядный операнд, старшие 32 разряда которого представляют целую часть результата преобразования, а младшие - дробную часть, из *REGR* переписывается в *RMA*. Если дробная часть $< \frac{1}{2}$, то на этом выполнение микрокоманды заканчивается. Если дробная часть $\geq \frac{1}{2}$, то запускается ALU операционного блока, на котором к дробной части прибавляется I/2

RMA + 0000000080000000 ⟶ REGS ⟶ RECR.

В результате такого сложения к целой части добавится I.

8.3. Команды преобразования целых чисел в действительные

Во всех командах этого класса первая микрокоманда – это микрокоманда *MIF*, *MID* или *MIG*, в зависимости от того, в какой формат преобразуется исходное целое число. Формат целого числа указывается в поле формата микрокоманды

MIF (D,G) ;B(W,L)/R(7) → R(0)_NOP;PDA

По этой микрокоманде целое число (64-разрядное) переписывается из входного буфера в *RMA*, *REGS*, *REGR*. Младшие разряды заполняются нулями. Таким образом формируется мантисса числа. В *RP1* записывается константа, равная порядку байта, в который происходит преобразование. Величины констант приведены в п.6.10.

Поскольку преобразуемое число может быть отрицательным, то следующая микрокоманда – это *MNN* – изменить знак числа, если оно отрицательное

MNN ;L/R(2) → R(0)_NOP;

Далее идет микрокоманда, которая устанавливает режим нормализации. В поле формата указывается формат, в который преобразуется исходное целое число

MOV ;F(D,G)/R(2) → R(2)_NOR;ZV

В режиме нормализации параметром сдвига служит количество нулей в старшей части мантиссы, которое определяется специальной схемой в ОБ. Эта схема может определять наличие в старшей части мантиссы (не считая старшего 64 бита) от одного до 16 нулей. Код количества нулей передается на сдвигатель и выполняется сдвиг влево. Код сдвига перед циклом сдвига записывается в *RP2* и во время цикла выполняется операция

RP1-RP2 → RPSD → RP1

После сдвига определяется новое количество нулей и цикл повторяется. Так продолжается, пока количество нулей перед очеред-

ным циклом не станет меньше I6. В этом случае следующий цикл будет последним. В нем происходит запись очередного вычитания RP1-RP2 кроме указанных выше регистров еще в RPOR, т.е. фиксируется порядок результата. Знак результата записывается в триггер знака результата в первой микрокоманде.

Если целое число представляет собой максимальное отрицательное число соответствующего формата, то после микрокоманды изменения знака 64 бит мантиссы будет равен I. В этом случае формируется признак нормализации вправо. По этому признаку формируется код сдвига, равный I, сдвигатель настраивается на сдвиг вправо, ALUP блока обработки порядка - на сложение

$$RP1+RP2 \longrightarrow RPSD \longrightarrow RP1$$

Последняя микрокоманда - это запись результата в выходной буфер

$$;MOV \quad :F(D,G)/R(2) \longrightarrow R(7)\_NOP;ZS,SA$$

В ней формируется ПЗ формат и записывается состояние.

8.4. Команды преобразования CVTFG и CVTGF

Первая микрокоманда - это микрокоманда пересылки с вычитанием константы из порядка

$$MPS \;:F(G)/R(7) \longrightarrow R(0)\_NOP;PDA$$

По этой микрокоманде мантисса переслается из входного буфера в RMA, REGS, REGR, а порядок - в RP1. В RP2 загружается константа 080_{I6} для F-формата или 400_{I6} для G-формата. Далее выполняется вычитание

$$RP1-RP2 \longrightarrow RPSD, RPOR \longrightarrow RP1$$

Вторая микрокоманда-это MIF с форматом G для CVTFG и MIG с форматом F для CVTGF

$$MIF \;:G/R(2) \longrightarrow R(2)\_NOP;ZV$$

или

$$MIG \;:F/R(2) \longrightarrow R(2)\_NOP;ZV$$

По этой микрокоманде выполняется только преобразование порядка. В RP2 загружается константа 400_{I6} для MIF или 080_{I6}

для MIG . Затем производится сложение

RP1+RP2 → RPOR, RPSD → RP1

Таким образом в RPOR формируется порядок преобразованного слова. При преобразовании CVTG-F требуется округление. Поэтому в этой команде есть микрокоманда округления

RND ;F/R(2) → R(2)_NOP; ZV

Если при округлении получается переполнение, то автоматически выполняется нормализация вправо. Мантисса сдвигается на один разряд вправо и выполняется прибавление I к порядку.

RP1+1 → RPOR

Последняя микрокоманда такая же, как во всех командах

MOV ;G(F)/R(2) → R(7)_NOP; ZS, SA

8.5. Команды преобразования CVTFD и CVTDF

Первая микрокоманда - это микрокоманда пересылки с расформированием ПЗ формата.

MWP1 ;F(D)/R(7) → R(0)_NOP; PDA

По этой микрокоманде мантисса пересылается из входного буфера операндов в регистры RMA , REGS , REGR , а порядок - в регистры блока обработки порядка RP1 и RPOR . На этом выполнение первой микрокоманды заканчивается.

Разрядность порядков в F-и D- форматах одинакова, следовательно, преобразовывать порядок не требуется. Мантисса числа в F-формате на регистре REGR представлена следующим образом: старшие 25 разрядов - мантисса числа, младшие 39 разрядов - нули. Так как мантисса числа в D - формате должна занимать 57 старших разрядов, то в случае преобразования числа из F- в D - формат дополнительного преобразования мантиссы не требуется.

Наоборот, если требуется преобразовать число из D - формата в F - формат, необходимо выполнить "усечение" мантиссы числа

в D-формате. Для чего требуется дополнительная микрокоманда округления

RND :F/R(2) → R(2)_NOP ; ZV

Если 39 бит мантиссы равен I, то выполняется сложение

RMA + 0000004000000000 ——→ REGS ——→ RECR

Если при округлении получается переполнение, то автоматически выполняется нормализация вправо. Мантисса сдвигается на один разряд вправо и выполняется прибавление I к порядку

RP1+1 → RPSD, RPOR

Последняя микрокоманда - микрокоманда выдачи результата в выходной буфер. MOV :F(D)/R(2) → R(2)_NOP; ZS, SA

8.6. Команды сложения и вычитания в формате ПЗ

Команды сложения и вычитания выполняются одинаково, за исключением управления ALU , которое устанавливается по КОП2 микрокоманды (ADD/SUB) и знаком операндов.

Первая микрокоманда - это микрокоманда пересылки с расформированием формата ПЗ , согласно поля формата (F, D, G) в микрокоманде

;MWP1 :F(D,G)/R(7) → R(0)_NOP; PDA

Мантисса первого операнда записывается в регистр RMA , REGS, REGR, порядок в регистр RP1 и RPOR.

Вторая микрокоманда:

;MSP :F(D,G)/R(7) → R(1)_ADD(SUB); PDA, ZV

Мантисса второго операнда, согласно поля формата (F, D, G) в микрокоманде, записывается в регистры RMB , REGS , REGR, порядок - в регистр RP2.

Затем производится вычитание

RP1-RP2 → RPSD → RP1

Если результат вычитания положительный, то мантисса второго операнда сдвигается вправо на величину RPSD и записывается в RMB.

$$REGS \xrightarrow[RPSD]{\text{сдвиг на}} REGR \longrightarrow RMB$$

Если результат вычитания отрицательный, то в REGS из RMA переписывается мантисса первого операнда, порядок второго операнда из регистра RP2 переписывается в регистр RPOR, знак параметра сдвига меняется на положительный. Затем мантисса первого операнда сдвигается на величину RPSD и записывается в RMA.

$$RP2 \longrightarrow RPOR$$

$$RMA \longrightarrow REGS, REGR$$

$$(\overline{RP1}+1) \longrightarrow RPSD$$

$$REGS \xrightarrow[\text{на } RPSD]{\text{сдвиг}} REGR \longrightarrow RMA$$

По окончании сдвига содержимое RPOR пересылается в регистр RP1.

$$RPOR \longrightarrow RP1$$

В результате имеем: мантисса первого операнда на RMA, мантисса второго операнда на RMB, наибольший из двух порядков на RPOR и RP1.

Следующим шагом при выполнении этой микрокоманды является сложение/вычитание мантисс.

$$RMB \pm RMA \longrightarrow REGS \longrightarrow REGR \longrightarrow RMA$$

Если в результате сложения/вычитания получился отрицательный результат, то устанавливается признак ZNAK, а мантисса преобразуется в прямой код.

Если требуется нормализация, то выполняется нормализация

$$CSD \longrightarrow RP2$$

$$RP1 \pm RP2 \longrightarrow RPSD \longrightarrow RPOR$$

Если требуется округление, то выполняется округление.

Если после округления требуется снова нормализация, то выполняется нормализация.

Последняя микрокоманда – это микрокоманда записи результата в выходной буфер

;MOV :F(D,G)/R(2) → R(7)_NOP ;ZS, SA

В ней формируется плавающий формат и записывается состояние.

8.7. Команда сложения с последующим сравнением и ветвлением

Первая микрокоманда – микрокоманда пересылки с расформированием формата ПЗ , в соответствии с полем формата (F, D, G) в микрокоманде

;MWP1 ;F(D,G)/R(7) → R(0)_NOP; PDA, FA

Мантисса операнда – предел по признаку FA записывается в регистр-аккумулятор. Кроме этого, стандартно мантисса записывается в RMA, REGS, REGR . Порядок операнда – предел записывается в регистры RP1, RPOR, а по признаку записи в регистр-аккумулятор записывается и в регистр RAP . Знак операнда-предел по признаку записи в RAK запоминается на специальном триггере ZNRAK.

Следующие две микрокоманды идентичны первым двум микрокомандам команды сложения чисел в формате ПЗ . Выполнение этих двух микрокоманд в составе данной микропрограммы не имеет никаких особенностей по сравнению с их выполнением в команде сложения.

;MWP1 :F(D,G)/R(7) → R(0)_NOP; PDA
;MSP ;F(D,G)/R(7) → R(1)_ADD ;PDA, ZV

В результате выполнения этих двух микрокоманд мантисса результата будет находиться на регистрах RMA, REGS , REGR , порядок результата – на регистре RPOR , знак результата – на

триггере, хранящем признак ZNREZ, знак операнда-слагаемое – на триггере ZNADD.

Четвертая микрокоманда – микрокоманда выдачи результата сложения и формирования состояния.

;MOV :F(D,G)/R(2) → R(7)_NOP; ZS

Пятая микрокоманда – микрокоманда пересылка результата сложения из REGR в RMA с обнулением младших разрядов мантиссы, выходящих за разрядную сетку соответствующих форматов

MPA :F(D,G)/R(2) → R(0)_NOP;

Шестая микрокоманда – микрокоманда пересылки с последующим вычитанием

;MSP :F(D,G)/R(5) → R(1)_SUB; FS

По этой микрокоманде знак результата операции сложения ZNREZ переписывается в триггер ZN1, знак операнда-предел ZNRAK переписывается в триггер ZN2 знака первого операнда. Порядок операнда-предел переписывается из регистра RAP в регистр RP2 (порядок результата сложения расположен в RP1). Мантисса операнда-предел считывается с регистра RAK на шину MD и записывается в регистр RMB. После этого выполняется стандартная процедура вычитания чисел в формате ПЗ RMA-RMB, в результате чего мантисса результата вычитания запишется в регистры RMA, REGS, REGR, порядок результата – в регистр RPOR, знак – на триггере, хранящем признак ZNREZ.

Последняя микрокоманда – микрокоманда записи результата вычитания в выходной буфер данных и записи признака ВТВ в триггер ветвления

;MOV :F(D,G)/R(2) → R(7)_NOP; ZNZ, SA

Запись в триггер ветвления происходит по признаку ZNZ

BTB = (ZNADD ⊕ ZNAK) + ZREZ

ZREZ – признак равенства нулю результата вычитания.

В конце выполнения команды процессор читает из сопроцессора признак "ВТВ" и в зависимости от его состояния определяется необходимость выполнения операции ветвления.

8.8. Команды сравнения двух чисел в формате ПЗ

Команды сравнения чисел в формате ПЗ выполняются так же, как и команды вычитания в формате ПЗ . Исключение составляет последняя микрокоманда, в которой добавляется признак *FN*.

;MOV :F(D,G)/R(2) → R(7)_ NOP ; FZ , ZS, SA

По признаку *FN* меняется знак результата. Изменение знака связано с тем, что при обычном вычитании выполняется операция *RMB-RMA*, а команда сравнения требует операции *RMA-RMB*.

8.9. Команды пересылки и команды тестирования чисел в формате ПЗ

Команды пересылки и команды тестирования чисел в формате ПЗ выполняются одинаково. Исключение составляет лишь то, что результат команды, записанный в выходной буфер в командах тестирования, не выдается на внешнюю магистраль адрес/данные, так как в этих командах нет адресации приемника.

Первая микрокоманда:

MWP1 :F(D,G)/R(7) → R(2)_ NOP; PDA

Мантисса операнда приемника, согласно полю формата микрокоманды записывается в *REGR*, порядок - в *RPOR*.

Вторая микрокоманда:

;MOV :F(D,G)/R(2) → R(7)_ NOP ; ZS, SA

В этой микрокоманде на базе содержимого *REGR, RPOR* и признака *ZNAK* в выходном буфере формируется плавающий формат, а также происходит запись состояния.

8.10. Команды пересылки чисел в формате ПЗ с изменением знака

Команды пересылки чисел в формате ПЗ с изменением знака выполняются также, как команды пересылки чисел в формате ПЗ . Исключение составляет последняя микрокоманда, в которой добавляется признак *FN*.

;MOV :F(D,G)/R(2) → R(7)_ NOP; FN, ZS, SA

По признаку *FN* меняется знак результата.

8.11. Команды целочисленного умножения

По системе команд, в командах умножения первый операнд, принимаемый на сопроцессор, называется множитель, второй операнд – множимое. В соответствии с этим и регистры, на которые помещаются эти операнды, соответственно называются регистр множителя и регистр множимого. Однако в силу того, что операция умножения симметрична, то умножитель сопроцессора использует содержимое регистра множителя в качестве множимого, а регистра множимого в качестве множителя.

Первая микрокоманда:

;MSD :B(W)/R(7) → R(2)_ NOP; PDA, FS

Целочисленный множитель в соответствии с форматом считывается с входного буфера операндов и записывается в *REGS* и *REGR* . По признаку *MSD+FS* выполняется сдвиг вправо множителя (байт на 24 разряда, слово на 16 разрядов). Сдвиг необходим для того, чтобы байт/слово результата занимали 31–24/31–16 разряда 64-разрядного выходного операнда умножителя. Именно эти разряды при выдаче результата *B/W* переписываются из *REGR* в выходной буфер операндов. Сдвиг выполняется за две итерации. Сначала сдвиг на 15 разрядов (максимальный сдвиг сдвигателя за один цикл), затем на 9/1 разрядов.

$18_{16}/10_{16} \rightarrow RP2$

$RP2 \rightarrow RPSD \rightarrow RP1$

$REGS \xrightarrow[RPSD]{\text{сдвиг на}} REGR \rightarrow REGS$

$CSD \rightarrow RP2$

$RP1-RP2 \rightarrow RPSD \rightarrow RP1$

$REGS \xrightarrow[\text{на } RPSD]{\text{сдвиг}} REGR \rightarrow REGS$

В случае целочисленного умножения операндов в формате "двойное слово" эта микрокоманда отсутствует. Это связано с тем, что в качестве результата умножения операндов в формате "двойное слово" система команд использует с 0 по 31 разряды 64-разрядного выходного операнда умножителя, а именно эти разряды при выдачи результата в формате L переписываются из REGR в выходной буфер операндов.

Вторая микрокоманда:

;MOV :L/R(2) → R(5)_NOP;

;MOV :L/R(7) → R(5)_NOP; PDA

Сдвинутый множитель из регистра REGR, в случае целочисленного умножения в форматах B/W, или множитель из входного буфера операндов в случае целочисленного умножения в формате L, записывается в регистр множителя. Разряды с 0 по 31 регистра множителя подаются на сумматор, на выходе которого формируется младшая половина утроенного множителя (без учета знака). Разряды с 32 по 63 регистра множителя подаются на другой сумматор, на выходе которого формируется старшая половина утроенного множителя (с учетом алгебраического знака).

Третья микрокоманда:

;MOV :B(W,L)/R(7) → R(4)_MULS; PDA, ZV

В соответствии с форматом множимое из входного буфера операндов записывается в регистр множимого. Одновременно с записью в регистр множимого информация с выхода сумматоров записывается в регистр утроенного множителя.

Через мультиплексоры информации со старших 32 разрядов регистров множителя и множимого, а также со старших 34 разрядов регистра утроенного множителя подается на множитель MNTL. Множитель выполняет операцию умножения и коррекцию результата (умножение с учетом алгебраических знаков). Результат умножения выдается на шину MD и записывается в регистр REGR.

Последняя микрокоманда – это микрокоманда выдачи данных и формирование состояния

;MOV :B(W,L)/R(2) → R(7)_NOP; ZS, SA

8.12. Команда расширенного умножения

В команде расширенного умножения происходит перемножение двух операндов в формате L , а затем к 64-разрядному произведению прибавляется знакорасширенный операнд-слагаемое. Так как первая часть выполнения команды включает в себя перемножение двух операндов в формате L , то первые две микрокоманды выполнения этой команды точно такие же, как и в команде целочисленного умножения операндов в формате L.

;MOV :L/R(7) → R(5)_NOP; PDA

;MOV :L/R(7) → R(4)_MULS; PDA

В результате выполнения этих двух микрокоманд 64-разрядное произведение будет записано в REGR и RMB . Третья микрокоманда – это микрокоманда формирования знакорасширенного операнда слагаемого

;MOV :L/R(7) → R(0)_NOP; PDA, FA

По этой микрокоманде операнд-слагаемое как целое число в формате L считывается на шину MD (разряды 63–32) и записывается в регистр RMA . Кроме этого, по признаку FA и формату L операнд-слагаемое с шины DA (0–31) записывается в разряды 31–0 регистра-аккумулятора. При этом в разряды 63–32 за-

писывается знак, т.е. DA31.

В следующей микрокоманде знакорасширенный операнд-слагаемое считывается с регистра RAK на магистраль MD и записывается в регистр RMA.

;MOV :L/R(5) ⟶ R(0)_ADD;

Адрес R(5) в поле чтения указывает на то, что на шину MD считывается RAK. После записи операнда-слагаемого в RMA происходит его сложение с произведением

RMA + RMB ⟶ REGR

Последняя микрокоманда – микрокоманда выдачи результата выполнения команды из REGR в выходной буфер операндов и формирование состояния.

;MOV :Q/R(2) ⟶ R(7)_NOP;SA,ZS

8.I3. Команды умножения двух чисел, представленных в формате с плавающей запятой

Так как блок умножения за один цикл выполняет умножение двух 32-разрядных чисел, то для выполнения умножения чисел в формате F требуется один цикл умножения, а для чисел в формате D и G – четыре цикла.

8.I3.I. Команда умножения чисел, представленных в формате F

Первая микрокоманда – микрокоманда пересылки с расформированием формата ПЗ

;MPS :F/R(7) ⟶ R(5)_NOP;PDA

Также как в случае целочисленного умножения, мантисса множителя записывается в RMNT. Кроме того, в RP1 записывается порядок, в RP2 – константа 080_{16}. Далее выполняется вычитание:

RP1 - RP2 ⟶ RPSD, RPOR ⟶ RP1

Вторая микрокоманда:

;MWP2 :F/R(7) ⟶ R(4)_MULS; PDA, ZV

Порядок множимого записывается в RP2, после чего происходит суммирование порядков

RP1+RP2 ⟶ RPSD, RPOR ⟶ RP1

Мантисса множимого со сдвигом на один разряд влево (кодовое умножение) заносится в RMNM , утроенный множитель записывается в регистр утроенного множителя, после чего БУМ выполняет операцию умножения.

64-разрядное произведение считывается на магистраль MD и записывается в регистры RMA, RMB, REGS, REGR.

Информация с REGS поступает на схему нормализации. Схема нормализации по I7 старшим разрядам регистра REGS формирует пятиразрядный двоичный код FA параметра нормализации. Параметр нормализации, пройдя мультиплексор (второй вход мультиплексора RPSD), формирует 4-разрядный двоичный код сдвига CSD . Если CSD не равен нулю (в данном случае он может быть либо 0, либо I), то происходит нормализация мантиссы.

REGS $\xrightarrow{\text{сдвиг влево}}$ REGR ⟶ REGS, RMA

CSD ⟶ RP2

RP1-RP2 ⟶ RPSD, RPOR ⟶ RP1

По значению 38 разряда 64-разрядной нормализованной мантиссы произведения определяется необходимость округления. Если 38 разряд (счет начинается с нулевого разряда) регистра REGR равен единице, то выполняется округление

RMA + 0000004000000000 ⟶ REGS , RECR ⟶ RMA

Если после операции округления в знаковом разряде мантиссы установилась единица, то необходимо снова произвести нормализацию. Единица в 63 разряде регистра REGS устанавливает код сдвига CSD, равный I.

REGS $\xrightarrow[\text{вправо}]{\text{сдвиг}}$ REGR $\longrightarrow$ REGS, RMA

CSD $\longrightarrow$ RP2

RP1+RP2 $\longrightarrow$ RPSD, RPOR $\longrightarrow$ RP1

Последняя микрокоманда – микрокоманда выдачи признака ZNAK, мантиссы (из REGR), порядка (из RPOR) в выходной буфер операндов и формирование состояния.

;MOV :F/R(2) $\longrightarrow$ R(7)_ NOP; ZS, SA

8.I3.2. Команда умножения чисел представленных в формате D и G

Первая микрокоманда – микрокоманда пересылки с расформированием формата П3

;MPS :D(G)/R(7) $\longrightarrow$ R(5)_ NOP; PDA

64-разрядная мантисса множителя записывается в регистр RMNT. В регистр RP1 записывается порядок, в RP2 – константа 080_{I6} для D – формата, 400_{I6} для G – формата. Выполняется вычитание

RP1-RP2 $\longrightarrow$ RPSD, RPOR $\longrightarrow$ RP1

Вторая микрокоманда:

;MWP2 :D(G)/R(7) $\longrightarrow$ R(4)_ MULK; PDA, ZV

Порядок множимого записывается в RP2, после чего происходит сложение порядков

RP1+RP2 $\longrightarrow$ RPSD, RPOR $\longrightarrow$ RP1

Мантисса множимого со сдвигом на один разряд влево (кодовое умножение) записывается в регистр RMNM, утроенный множитель записывается в регистр утроенного множителя, после чего блок умножения совместно с операционным блоком за четыре итерации выполняет операцию умножения.

Первая итерация: младшая половина множимого умножается на младшую половину множителя. Результат умножения через шину MD записывается в RMA, REGS, REGR, RMB, после чего – сдвиг ре –

зультата на 32 разряда вправо

$$REGS \xrightarrow[вправо]{сдвиг} REGR \longrightarrow REGS, RMA$$

Вторая итерация: старшая половина множимого умножается на младшую половину множителя. Результат умножения через шину MD записывается в RMB, после чего производится сложение частичных произведений

$$RMA + RMB \longrightarrow REGS \longrightarrow REGR, RMA$$

Третья итерация: младшая половина множимого умножается на старшую половину множителя. Результат умножения через шину MD записывается в RMB, после чего производится сложение частичных произведений

$$RMA + RMB \longrightarrow REGS \longrightarrow REGR, RMA$$

Частичное произведение сдвигается на 32 разряда вправо

$$REGS \xrightarrow[вправо]{сдвиг} REGR \longrightarrow REGS, RMA$$

Четвертая итерация: старшая половина множимого умножается на старшую половину множителя. Результат умножения через шину MD записывается в RMB

$$RMA + RMB \longrightarrow REGS \longrightarrow REGR, RMA$$

После выполнения сложения на $REGS$, $REGR$ располагается мантисса окончательного произведения.

Смена подачи младших и старших частей сомножителей на умножитель осуществляется мультиплексорами блока умножения. Информация с $REGS$ поступает на схему нормализации. Выработанный схемой код параметра нормализации формирует код сдвига CSD. Если CSD не равен нулю (в данном случае он может быть либо 0, либо I), то происходит нормализация мантиссы

$$REGS \xrightarrow[влево]{сдвиг} REGR \longrightarrow REGS, RMA$$

$$CSD \longrightarrow RP2$$

$$RP1 - RP2 \longrightarrow RPSD, RPOR \longrightarrow RP1$$

По значению 6 разряда (для *D* -формата) или 9 разряда (для *G* -формата) 64-разрядной нормализованной мантиссы произведения определяется необходимость округления. Если данный разряд равен I, то выполняется операция округления

RMA + 0000000000000040 ⟶ REGS ⟶ REGR ⟶ RMA (для *D* - формата)

RMA + 0000000000000200 ⟶ REGS ⟶ REGR ⟶ RMA (для G - формата)

Если после операции округления в знаковом разряде мантиссы установилась единица, то необходимо снова нормализовать мантиссу. Единица в 63 разряде регистра *REGS* устанавливает код сдвига *CSD* равным I.

REGS $\xrightarrow[вправо]{сдвиг}$ *REGR* ⟶ *REGS, RMA*

CSD ⟶ *RP2*

RP1+RP2 ⟶ *RPSD, RPOR* ⟶ *RP1*

Последняя микрокоманда - микрокоманда выдачи признака *ZNAK*, мантиссы (из *REGR*), порядка (из *RPOR*) в выходной буфер операндов и формирование состояния.

;MOV :D(G)/R(2) ⟶ R(7)_NOP ;ZS, SA

8.I4. Команда расширенного умножения чисел в формате ПЗ с выделением целой части

Первая микрокоманда - микрокоманда пересылки с расформированием формата ПЗ в соответствии с полем формата (*F, D, G*) в микрокоманде

;MWP2 :F(D,G)/R(7) ⟶ R(2)_NOP; PDA, FA

Мантисса множителя записывается в *REGS*, *REGR*, а по признаку *FA* - в *RAK*.

Порядок множителя записывается в *RP2*, а по признаку *FA* в *RAP* .

Следующая микрокоманда - микрокоманда выдачи мантиссы мно-

жителя как целого числа из REGR в выходной буфер операндов с последующей передачей ее в процессор.

;MOV :Q/R(2) → R(7)_NOP;

В процессоре к мантиссе множителя добавляется операнд-расширитель разрядностью 8 (F-и D-формат) или II (G-формат)бит. Расширенная и сдвинутая на один разряд влево мантисса множителя из процессора передается в сопроцессор.

Третья микрокоманда - микрокоманда пересылки с расформированием формата ПЗ

;MPS :F(D,G)/R(7) →R(5)_NOP ; PDA

Мантисса множимого записывается в регистр множителя. Порядок множимого записывается в регистр RP1 . В регистр RP2 заносится константа 080_{I6} для F- и D- формата, 400_{I6} - для G-формата. Выполняется вычитание:

RP1-RP2 ⟶ RPSD, RPOR ⟶ RP1

Четвертая микрокоманда - микрокоманда пересылки мантиссы расширенного множителя в регистр множимого, а также записи утроенного множимого в регистр утроенного множителя.

; MWP2 :R(Q)/R(7) ⟶ R(4)_NOP; PDA, FS

По признаку записи в регистр множимого и наличии формата R или Q мультиплексор на входе регистра RP2 настраивается на соединение выхода регистра RAP со входом регистра RP2. Таким образом, в регистре RP2 будет восстановлен порядок множителя.

Признак FS в данной микрокоманде служит для того, чтобы заблокировать запись в триггер, хранящий знак множителя ZN2 в схеме управления ALU. Состояние этого триггера было определено в первой микрокоманде.

Пятая микрокоманда - микрокоманда умножения

;MOV : F(D,G)/R(0) → R(0)_ MULS(MULK); 2V,FN

В данной микрокоманде выполняется стандартная процедура умножения двух чисел в формате ПЗ .

По признакам MOV и FN взводится триггер, формирующий сигнал EMOD . Этот сигнал вносит ряд особенностей в дальнейшее выполнение команды. Во-первых, если выполняется умножение двух чисел в формате F и в конце умножения есть нормализация, то взводится признак EMDF , если выполняется умножение двух чисел в формате D или G , то на специальном триггере запоминается самый старший разряд мантиссы результата, ушедший за пределы 64-разрядной сетки справа. Во-вторых, этим сигналом блокируется выполнение операции округления в конце микрокоманды умножения.

По окончании процедуры умножения мантисса результата будет размещена на регистрах REGS, REGR, RMA, RMB. Кроме того, при умножении операндов в форматах D или G на специальном триггере хранится дополнительный разряд мантиссы. Выход этого триггера соединен со входом младшего разряда сдвигателя, осуществляющего сдвиг влево. По требованию системы команд, непосредственно после выполнения операции умножения в дальнейших вычислениях должны участвовать для F - формата 63-31 разряды мантиссы результата умножения, для D- и G -форматов 63-0 разряды мантиссы результата умножения плюс бит, хранящийся на специальном триггере. Если после умножения произойдет нормализация мантиссы, то в регистре RMB будет расположена не нормализованная мантисса, а в регистрах RMA, REGS , REGR- нормализованная. При этом, если формат числа, которое нормализуют F , то взводится признак EMDF , свидетельствующий о том, что в дальнейших вычислениях должны участвовать разряды мантиссы с 63 по 32, если формат D или G , то в результате нормализации допол-

нительный бит со специального триггера перепишется в нулевой разряд мантиссы, после чего данный триггер сбрасывается, т.е. в дальнейших вычислениях должны участвовать разряды мантиссы с 63 по 0.

После выполнения операции умножения порядок результата размещается на регистрах RP1 , RPSD , RPOR , знак результата – на триггере, хранящем признак ZNREZ.

Если после выполнения умножения требуется нормализация, то

CSD ⟶ RP2

RP1-RP2 ⟶ RPSD, RPOR ⟶ RP1

Шестая микрокоманда – микрокоманда выделения целой части.

;MRL :F/R(1) ⟶ R(1)_ SDV; ZV, FS , если формат F.

;MRL :D(G)/R(2) ⟶ R(1)_ SDV; ZV, FS , если формат D и G.

Если мантисса представлена в F -формате, то в зависимости от того, была или не была в конце предыдущей микрокоманды нормализация на шину MD читается либо регистр RMB , либо REGR Если признак EMDF не установлен, то на шину MD читается регистр RMB , т.е. разряды с 63 по 31 представляют собой мантиссу, разряды с 30 по 0 искусственно обнуляются. Если признак EMDF установлен, то на шину MD читается регистр REGR , т.е. разряды с 63 по 32 представляют собой мантиссу, разряды с 31 по 0 искусственно обнуляются. Информация с шины MD по признаку записи в регистр RMB записывается в RMB, REGS , REGR .

RMB или REGR ⟶ RMB, REGS, REGR

Если мантисса представлена в D- или G- форматах, то на шину MD читается регистр REGR , при этом, если в предыдущей микрокоманде была нормализация, то в регистр RMB запишется нормализованная мантисса

REGR ⟶ RMB, REGS, REGR

После того, как регистр будет прочитан на магистраль MD сигналом SDV , будут сброшены признаки EMDD и EMDF , при этом, если в предшествующей микрокоманде получения произведения двух операндов в D- или G-формате не было нормализации, то бит расширения мантиссы, хранящийся на специальном триггере, будет храниться.

По признаку MRL и формату в регистр блока обработки порядка RP2 записывается константа:

для F- и D- формата - $09F_{16}$,

для G-формата - $41F_{16}$.

Признак FS в этой микрокоманде блокирует запись в регистр RP1 по признаку MRL . После этого выполняется

$$RP1 - RP2 \longrightarrow RPSD \longrightarrow RP1$$

Если параметр сдвига получился отрицательным, то устанавливается режим сдвига вправо и выполняется цикл изменения знака параметра сдвига. Если параметр сдвига получился положительным, то устанавливается режим сдвига влево. Далее выполняется сдвиг точно так же, как в команде сдвига. Исключение составляет лишь то, что, если мантисса представлена в формате D или G , бит расширения, хранящийся на специальном триггере, не сброшен в предшествующей микрокоманде и осуществляется сдвиг влево, то этот бит расширения будет участвовать в сдвиге в первом цикле (т.е. он попадет в 64-разрядную сетку).

В результате выполнения этой микрокоманды в REGR будет сформировано число, старшие 32 разряда которого представляют собой целую часть результата умножения двух операндов. Поэтому следующая микрокоманда - микрокоманда сдвига вправо на 32 разряда содержимого REGR.

```
;MOV  :L/R(2) --> R(2)_NOP;FS
REGS  --(сдвиг вправо на 32 разряда)--> REGR
```

Если знак результата отрицательный, то знак полученного целого числа должен быть тоже отрицательный. Поэтому следующая микрокоманда – микрокоманда изменения знака, если знак результата отрицательный

;MNS :L/R(2) → R(0)_NOP

REGR → RMA

Если же знак отрицательный, то добавляется цикл суммирования:

$\overline{RMA}$ +1 → REGS → REGR → RMA

Девятая микрокоманда – микрокоманда выдачи целой части результата умножения в выходной буфер операндов

;MOV :L/R(2) → R(7)_NOP;

Десятая микрокоманда – микрокоманда выделения дробной части.

;MPS :F(D,G)/R(1) → R(2)_SDV;FS

По этой микрокоманде мантисса результата умножения переписывается из RMB в REGR

RMB → REGS, REGR

По признаку MPS+FS в регистр блока обработки порядка RP1 из RPOR переписывается порядок произведения. По признаку MPS в RP2 записывается константа 080_{I6} – для F-и D-формата, 400_{I6} – для G-формата. Кроме этого, по сигналу MPS+FS взводится схема анализа на равенство целой части нулю.

После этого выполняется вычитание

RP1 - RP2 → RPSD → RP1

Если результат вычитания получился отрицательный, то запускается схема анализа на равенство нулю целой части, которая блокирует сдвиг мантиссы и вырабатывает сигнал записи, восстанавливающий в RP1 исходный порядок RPOR → RP1

Если результат вычитания положительный, то выполняется процедура сдвига мантиссы влево (удаление целой части) на величину, определяемую содержимое регистра *RPSD*. Если разрядность целой части больше I5 разрядов, то потребуется более одного цикла сдвига. Если после выполнения операции умножения над числами в *D*- или *G*- формате не было нормализации, то в первом цикле сдвига влево будет участвовать бит расширения, хранящийся на специальном триггере, т.е. бит расширения мантиссы попадет в 64-разрядную сетку

$$REGS \xrightarrow[на\ RPSD]{сдвиг\ влево} REGR \rightarrow REGS, RMA$$

$$CSD \rightarrow RP2$$

$$RP1-RP2 \rightarrow RPSD \rightarrow RP1$$

$$REGS \xrightarrow[на\ RPSD]{сдвиг\ влево} REGR \rightarrow REGS, RMA$$

и т.д.

Как только содержимое *RPSD* станет меньше или равно I5, то это значит, что выполняется последний цикл сдвига, когда сбрасывается схема анализа на равенство нулю целой части. Кроме того, этим же сигналом сброса, если первая разность *RP1-RP2* не была отрицательной, в регистр *RP1* записывается константа (порядок) 080_{I6} для *F*- и *D*- формата или 400_{I6} для *G*-формата.

На этом выполнение этой микрокоманды заканчивается. Порядок мантиссы остатка расположен на регистре *RP1*, мантисса остатка - на регистрах *RMA, REGS, REGR.*

Если в результате сдвига мантисса станет равной нулю, то взведется признак *ZREZ* (признак равенства нулю результата). По этому признаку выходы регистра *RPOR*, поступающие на соответствующие входы выходного буфера операндов, искусственно обнуляются.

Одиннадцатая микрокоманда – микрокоманда нормализации мантиссы остатка

;MOV :F(D,G)/R(2) ⟶ R(2)_NOR;ZV

По признаку NOR сбрасывается специальный триггер, хранящий бит расширения мантиссы при умножении в D- и G-формате. Это связано с тем, чтобы избежать его повторного появления в 64-разрядной сетке мантиссы при выполнении нормализации.

По окончании нормализации, если это необходимо, выполняется цикл округления.

При округлении может возникнуть переполнение, в этом случае выполняется нормализация вправо.

Последняя микрокоманда – микрокоманда выдачи признака ZNAK, мантиссы (из REGR), порядка (из RPOR) в выходной буфер операндов, формирования и выдачи состояний.

;MOV :F(D,G)/R(2) ⟶ R(7)_NOP; ZS, SA

8.15. Команды вычисления полинома

В соответствии с микропрограммой адресации аргумент, показатель степени и адрес таблицы коэффициентов принимаются в центральный процессор. Аргумент пересылается из центрального процессора в сопроцессор. Показатель степени и адрес таблицы коэффициентов остаются в центральном процессоре, где в соответствии со значением показателя степени организуется подкачка коэффициентов в сопроцессор.

В зависимости от начального значения показателя степени микропрограмма выполнения команды вычисления полинома распадается на две ветви.

Показатель степени равен нулю. В этом случае, в соответствии с требованием системы команд, в качестве результата выполнения команды берется коэффициент C (0).

Первая микрокоманда:

;MWP1 :F(D,G)/R(7) → R(2)_ NOP; PDA

В соответствии с форматом мантисса аргумента из входного буфера пересылается в регистр REGS и REGR . Порядок аргумента записывается в регистр блока обработки порядка RP1 и RPOR. Эта микрокоманда необходима для того, чтобы освободить входной буфер операндов и сделать анализ аргумента на резервный операнд. В дальнейших вычислениях аргумент в этой ветви микро - программы не используется и поэтому в следующей микрокоманде его значение теряется.

Вторая микрокоманда - микрокоманда пересылки коэффициента С (0).

;MWP1 :F(D,G)/R(7) → R(2)_ NOP; PDA

Мантисса коэффициента из входного буфера операндов запи - шется в регистры REGS и REGR. Порядок коэффициента запишется в регистры блока обработки порядка RP1 и RPOR.

Последняя микрокоманда - микрокоманда выдачи коэффициента в выходной буфер операндов с последующим приемом его и состояний в центральный процессор

;MOV :F(D,G)/R(2) → R(7)_ NOP; ZS, SA

Показатель степени не равен нулю. В этом случае результат вычисления должен выглядеть следующим образом:

$$REZULT = C[0] + X \cdot (C[1] + X \cdot (C[2] + \dots X \cdot C[\alpha])),$$

где α - степень;

X - аргумент.

Первая микрокоманда:

;MPS :F(D,G)/R(7) → R(5)_ NOP; PDA, FA

В соответствии с форматом мантисса аргумента из входного буфера операндов запишется в регистр множителя по

признаку FA и в регистр-аккумулятор. По признаку MPS порядок аргумента запишется в регистр блока обработки порядка RP1 , а константа 080_{I6} (для F- и D-формата) или 400_{I6} (для G-формата) запишется в регистр RP2. Кроме этого, по признаку записи в регистр-аккумулятор знак аргумента запишется на специальный триггер ZNRAK, хранящий знак числа размещенного в аккумуляторе. По сигналу записи порядка в RP1 произойдет запись знака в триггер знака первого операнда ZN1 .

Далее происходит вычитание из порядка смещения, с записью результата в регистр RPOR, а по признаку FA – и в регистр-аккумулятор порядка

$$RP1-RP2 \longrightarrow RPSD, RPOR, RAP \longrightarrow RP1.$$

Таким образом, порядок аргумента хранится на регистре RAP, мантисса – на регистре множителя, регистре утроенного множителя, знак – на специальном триггере во время выполнения команды.

Вторая микрокоманда:

;MWP2 :F(D,G)/R(7) →R(4)_MULS(_MULK);PDA,FN

Мантисса коэффициента из входного буфера операндов переписывается в регистр множимого, одновременно происходит запись утроенного множителя в регистр утроенного множителя. Порядок коэффициента записывается в регистр RP2 . Знак коэффициента записывается на триггер знака второго операнда ZN2 . После этого выполняется стандартная процедура умножения за исключением того, что по признаку (MWP2+MPA) FN взведется признак блокировки нормализации и округления произведения.

После выполнения этой микрокоманды порядок произведения располагается на регистрах RP1, RPSD, RPOR, мантисса произведения для F- формата – на регистрах RMA, RMB, REGS, REGR , для D- и G - форматов – на регистрах RMA, REGS, REGR. Знак произведения ZM⊕ZN2 запишется на триггер зна -

ка результата.

По требованию системы команд в последующем суммировании мантисса произведения должна иметь 32 разряда для F-формата и 64 разряда для D- и G-форматов. Поэтому следующая микрокоманда - микрокоманда усечения мантиссы.

;MOV :F/R(2) ⟶R(0)_NOP;

По этой микрокоманде из REGR на шину MD считывается 32 старших разряда мантиссы, младшие 32 разряда искусственно обнуляются.

$$REGR \xrightarrow[\text{младших 32 разрядов}]{\text{с обнулением}} RMA$$

В случае D- или G-формата эта микрокоманда отсутствует, т.к. в RMA по окончании умножения записывается 64-разрядная мантисса.

Четвертая микрокоманда - микрокоманда сложения произведения с очередным коэффициентом

;MSP :F(D,G)/R(7) ⟶R(1)_ADD;PDA,FN,ZV

Мантисса коэффициента из входного буфера операндов перепишется в регистр RMB, порядок коэффициента запишется в регистр RP2, знак коэффициента запишется в триггер знака второго операнда ZN2. По признаку MSP&FN знак произведения из триггера ZNREZ перепишется в триггер знака первого операнда ZN1.

После этого происходит описанная уже процедура сложения RMA+RMB, характеризующаяся выравниванием порядков, сложением и если требуется нормализацией и округлением.

В конце выполнения микрокоманды мантисса результата сложения располагается на регистрах RMA, REGS, REGR, порядок - на регистрах RPOR, RPSD, RP1, знак, определяемый знаком мантиссы - в триггере ZNREZ.

На фоне выполнения этой микрокоманды центральный процес - сор занимается анализом показателя степени. Из показателя степени вычитается единица и результат анализируется на равенство нулю.

Если результат равен нулю, то это значит, что вычисление полинома закончилось. Сопроцессор выполняет последнюю микрокоманду - микрокоманду выдачи, результата и состояний

;MOV :F(D,G)/R(2) ⟶ R(7)_NOP; ZS, SA

Результат вычисленного полинома поступает на центральный процессор, где в соответствии с форматом размещается на регистрах R0 или R0 и R1.

Если после вычитания показатель степени не равен нулю, то центральный процессор организует подкачку очередного коэффициента на сопроцессор, который в свою очередь продолжает вычисление полинома.

Пятая микрокоманда - микрокоманда умножения аргумента на результат последнего сложения.

;MPA :F(D,G) ⟶ R(4)_MULS; FN

По признаку MPA и формату операнда 64-разрядная мантисса результата сложения "усекается" до соответствующего разряда. Содержимое REGR читается на шину MD , при этом разряды, вышедшие за разрядную сетку соответствующего формата, обнуляются. С шины MD операнд записывается в регистр множимого. Мантисса аргумента исходно расположена в регистре множителя и регистре утроенного множителя.

По признаку MPA в регистр RP1 переписывается из регистра RAP порядок аргумента, в регистр RP2 переписывается из регистра RPOR порядок результата предшествующего сложения. Кроме этого, по признаку MPA в тригтер знака первого операнда ZN1 из триг - гера , хранящего знак аргумента ZNRAK переписывается знак ар-

гумента, признак ZNREZ переписывает знак результата в триггер знака второго операнда ZN2 . После этого выполняется умножение аналогичным образом, как при описании выполнения умножения во второй микрокоманде.

Шестая микрокоманда точно такая же, как третья микрокоманда

;MOV :F/R(2) → R(0)_ NOP;

Седьмая микрокоманда точно такая же, как четвертая микро - команда

;MSP :F(D,G)/R(7) →R(1)_ ADD; PDA, FN, ZV

На фоне выполнения седьмой микрокоманды процессор снова приступает к анализу показателя степени. Этот цикл будет продолжаться до тех пор, пока показатель степени не станет равным нулю.

8.16. Команды деления

В СПРЦ использован метод деления, основанный на том, что для сокращения времени деления во всех определенных остатках, кроме конечного, не производится полное приведение переносов. Каждый очередной остаток получается в двухрядном коде, состоящем из кодов поразрядной суммы и поразрядных переносов.

Для определения знака остатка полное приведение переносов осуществляется только в старших его разрядах. В зависимости от знаков приведенной части остатка и делителя при определении очередной цифры частного делитель прибавляется к остатку или вычитается из него. При этом на входы сумматора подаются три слагаемых: сдвинутые коды поразрядной суммы и поразрядных переносов и соответствующий действию код делителя.

В ходе деления формируется не истинный код частного , а два условных кода, которые обозначим А и В, где А - положительная составляющая частного, а В - отрицательная. Для этих кодов

справедливо равенство

$$D = (A+B)\cdot d + r,$$

где D – делимое, d – делитель и r – остаток.

Частное P определяется в конце деления как

$$P = A - B$$

Однорядный код остатка r получается лишь в конце деления путем полного приведения поразрядных переносов. Если необходимо, то он предварительно корректируется (восстанавливается), что сопровождается коррекцией частного.

Способ деления с полным приведением переносов лишь в нескольких старших разрядах остатков применим только в случае, если делитель нормализован. Тогда, если соотношение между величинами делимого и делителя удовлетворяет условиям корректности (для одинаковых знаков $\frac{|D|}{|d|} < 2^{31}$, для разных знаков $\frac{|D|}{|d|} < 2^{31}+1$), то любой очередной остаток по абсолютной величине будет меньше делителя. Следовательно, ни в одном из остатков не может быть обнаружено переполнение. Это, в свою очередь, дает возможность введя дополнительный знаковый разряд, определять знак остатка путем полного приведения переносов, например, лишь в четырех старших разрядах его модифицированного кода.

При исключении случаев переполнения в четырех приведенных разрядах, включающих два знаковых, возможно появление следующих девяти комбинаций.

Значение четырех старших приведенных разрядов остатка	Знак остатка
00.00 00.01 00.10 00.11	Положительный

10.11 11.00 11.01 11.10	Отрицательный
11.11	Не может быть определен без знания переноса из остальной неприведенной части остатка

Первые четыре комбинации соответствуют положительному остатку, вторые – отрицательному. Комбинации с 01... и 10... в знаковых разрядах, кроме приведенной выше, встретиться не могут, так как соответствуют числам, по абсолютной величине большим единицы. Комбинация 10.11 является допустимой, так как возможен перенос из неприведенной части, при учете которого получится код 11.00.

Девятая комбинация в зависимости от наличия или отсутствия переноса из неприведенной части остатка может превратиться соответственно либо в 00.00..., либо остаться такой, как есть: 11.11... Поэтому в данном случае без знания переноса из неприведенной части невозможно определить знак остатка. Однако, если учесть, что как при комбинации 00.00, так и при 11.11 по абсолютной величине остаток не превосходит 2^{61}, можно провести следующие рассуждения.

Пусть полученный остаток z_i удовлетворяет условию

$$-2^{61} \leq z_i < 2^{61},$$

тогда удвоение его даст

$$-2^{62} \leq 2 \cdot z_i < 2^{62}.$$

В зависимости от знаков остатка и делителя, делитель либо будет прибавляться к старшей половине удвоенного остатка, либо вычитаться из нее, т.е. следующим действием будет

$$z_{i+1} = 2 \cdot z_i \pm 2^{32} d$$

Поскольку величина $2r_i$ удовлетворяет неравенству $-2^{62} \leq 2r_i < 2^{62}$, то при нормализованном делителе знак r_{i+1} окажется противоположным знаку r_i. Следовательно, для определения r_{i+2} придется выполнить действие, обратное предыдущему, т.е.

$$r_{i+2} = 2(2r_i \pm 2^{32}\alpha) \mp 2^{32}\alpha,$$

что равносильно

$$r_{i+2} = 2 \cdot 2 \cdot r_i \pm 2^{32}\alpha$$

или

$$r_{i+2} = 2\, r_{i+1} \pm 2^{32}\alpha$$

Таким образом, при появлении комбинации II.II... можно не определять очередную разность, ограничившись лишь сдвигом остатка на разряд влево. После сдвига величину $2 \cdot r_i$ можно рассматривать как новый несдвинутый двухрядный остаток r_{i+1}.

Для того, чтобы определить характер следующего действия, надо привести и проанализировать его старшие четыре разряда. При каждом вычислении разности между полученным остатком и делителем формируются цифры положительной А и отрицательной В составляющих частного. Правило формирования этих цифр сводится к следующему.

Если при определении очередной разности на вход сумматора подается код делителя, обратный исходному, то в разряд положительной составляющей частного заносится единица, а соответствующий разряд отрицательной составляющей остается в нулевом состоянии. Если же делитель подается на вход сумматора, не изменяясь в том коде, в каком он поступил на деление, то единица наоборот, заносится в разряд отрицательной составляющей частного. В случае обнаружения комбинации, не определяющей знак остатка, со-

ответствующие шагу деления разряда положительной и отрицательной составляющих частного имеют нулевое значение.

Процесс деления начинается с нормализации делителя, затем на столько же разрядов влево сдвигается делимое. На первом шаге деления делимое сдвигается еще на разряд влево и в зависи - мости от знаков операндов, делитель вычитается из него или прибавляется к нему.

Циклически повторяющийся процесс деления продолжается до тех пор,пока не будут определены значения всех разрядов частного.

По окончании процесса деления вычисляется частное. Если последний остаток имеет тот же знак, что и делимое, или равен нулю, то частное определяется просто как разность. Если последний остаток не равен нулю и знак его противоположен знаку делимого, то по окончании процесса деления производится восстановление остатка. При этом делитель складывается с ним или вычитается из него согласно следующей схеме:

Знак			Действие восстановления
делимого	остатка	делителя	
+	−	+	сложение
+	−	−	вычитание
−	+	+	вычитание
−	+	−	сложение

Восстановление последнего остатка влечет за собой необходимость коррекции окончательно вычисляемого частного. Если при восстановлении код делителя прибавляется к остатку, то частное уменьшается на единицу. В случае же вычитания кода делителя частное,наоборот,увеличивается на единицу. Если перед началом

деления имела место нормализация делителя, то результирующий остаток получается увеличенным. Поэтому при наличии предварительной нормализации после восстановления остатка и определения частного, производится сдвиг окончательного остатка вправо на количество разрядов, на которое сдвигался делитель при нормализации.

В соответствии с вышеизложенным алгоритмом построен делитель СПРЦ, блок-схема которого представлена на рис. 23,

Делитель считывается на магистраль MD с входного буфера операндов и через мультиплексор делителя (MUX RDLT) записывается на регистр делителя (RDLT). В случае целочисленного делителя по окончании нормализации, нормализованный делитель с регистра REGR через MUXRDLT (с арифметическим сдвигом вправо на один разряд) записывается на регистр RDLT. Делитель с выхода A(0-63) RDLT подается на сумматоры (SUMDS , SUMDM) блока деления. Кроме этого, RDLT со сдвигом на один разряд влево может читаться на магистраль MD .

Делимое с входного буфера операндов считывается на магистраль MD и через мультиплексор регистра RMA (MUXA) записывается в регистр RMA . В случае плавающего формата делимого мантисса делимого записывается на RMA со сдвигом арифметическим вправо на один разряд. Так как для чисел в формате ПЗ не существует ограничений на абсолютные величины делимого и делителя, и кроме того делимое и делитель являются нормализованными числами, то для исключения случаев переполнения в старших приведенных разрядах необходимо сдвинуть делимое как минимум на один разряд вправо.

Если делимое является целым числом, то после сдвига делимого, связанного с нормализацией делителя, сдвинутое делимое с регистра REGR через MUXA записывается в регистр RMA.

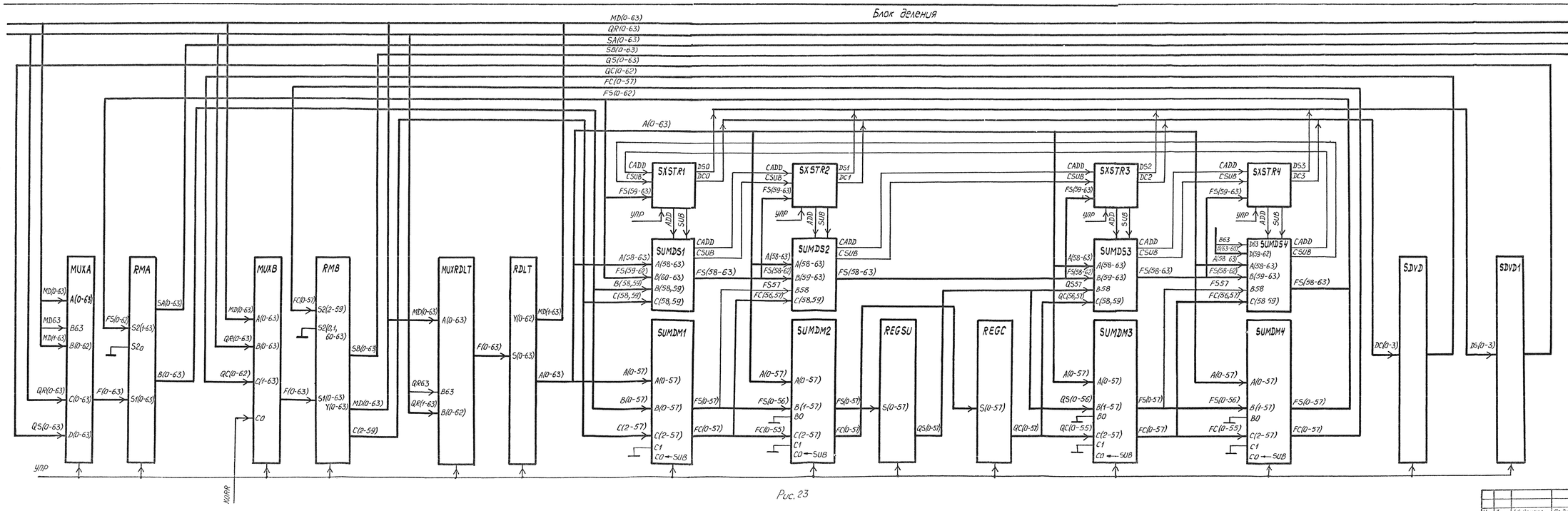


Рис. 23

MUXA – входной мультиплексор регистра RMA
MUXB – входной мультиплексор регистра RMB
RMA – регистр мантиссы A
RMB – регистр мантиссы B
MUXRDLT – мультиплексор делителя
RDLT – регистр делителя
SXSTR1...SXSTR4 – схемы управления сумматорами
SUMDS1...SUMDS4 – шестиразрядный полный сумматор
SUMDM1...SUMDM4 – трехвходовые полусумматоры
REGSU, REGC – 58-разрядные регистры
SDVD – регистр переносов частного
SDVD1 – регистр суммы частного

По признаку записи делимого в RMA в регистр RMB через MUXB переписывается содержимое регистра переносов частного (SDVD). По началу выполнения содержательной микропрограммы любой команды, касающейся СПРЦ, регистр сумм частного (SDVD1) устанавливается в единичное состояние во всех разрядах, а регистр переносов частного (SDVD) обнуляется. Следовательно, в RMB запишется ноль. В процессе деления регистр RMA используется, как регистр сумм остатка, а регистр RMB – как регистр переносов остатка. Выходы регистра RMA B(0–63) и RMB C(2–59) поступают на вход первого сумматора блока деления.

Блок деления содержит четыре блока суммирования.

Блок суммирования состоит из пятидесятивосьмиразрядного блока трехвходовых полусумматоров (SUMDM), шестиразрядного полного сумматора (SUMDS) и схемы управления сумматорами (SXSTR).

Схема управления сумматорами по пяти приведенным старшим разрядам остатка и признакам сложения (CADD) с остатком или вычитания (CSUB) из остатка делителя, получаемым с предшествующего шестиразрядного сумматора, формирует сигналы управления (ADD, SUB) сумматорами, а также двухбитовый код цифры частного. Код цифры частного должен быть представлен битом положительной составляющей и битом отрицательной составляющей частного. Однако данная схема формирует цифру частного как бит суммы (DS) и бит переноса (DC) от сложения бита положительной составляющей с обратным кодом бита отрицательной составляющей частного.

В случае, если пять старших приведенных разрядов остатка равны 00000 или IIIII, то схема управления формирует DS = I, DC = 0, ADD = 0, SUB = 0. Если пять старших разрядов остатка отличаются от этих комбинаций, сигналы CSUB = I, CADD = 0,

то схема управления формирует DS = 0, DC = I, ADD = 0, SUB = I. Если CSUB = 0, CADD = I, то схема управления формирует DS=0, DC = 0, ADD = I, SUB = 0.

Пятидесятивосьмиразрядный блок трехвходовых полусумматоров на основе пятидесяти пяти разрядов входных переносов, пятидесяти восьми разрядов входных сумм и пятидесяти восьми разрядов делителя вычисляет сумму или разность между остатком и делителем в виде пятидесятивосьмиразрядных кодов сумм и переносов.

Если схема управления формирует ADD=0 , SUB=0 , то на SUMDM происходит сложение входных сумм и переносов. Если ADD=1 , SUB=0 , то на SUMDM происходит сложение входных сумм и переносов остатка с делителем. Если ADD = 0, SUB = I, то на SUMDM происходит сложение входных сумм и переносов остатка с инверсией делителя, при этом в младший разряд входных переносов добавляется единица. Выходы SUMDM сумм со сдвигом на один разряд влево и переносов со сдвигом на два разряда влево поступают на следующий блок суммирования.

Шестиразрядный полный сумматор (SUMDS) на основе двух разрядов переносов, одного разряда суммы, пяти приведенных разрядов остатка с выхода предшествующего блока суммирования, а также шести старших разрядов делителя в соответствии с сигналами управления ADD, SUB , формирует шесть старших приведенных разряда очередного остатка. Кроме этого, на основании знака полученного остатка и знака делителя, SUMDS формирует сигналы CADD , CSUB , которые поступают на SXSTR последующего блока суммирования. Если сигналы ADD и SUB равны нулю, то на выходе SUMDS формируется шестиразрядная сумма от сложения входных пяти приведенных разрядов остатка и одного старшего разряда суммы с двумя старшими разрядами переноса. Если

сигнал ADD=1 , а сигнал SUB=0 , то на выходе SUMDS формируется шестиразрядная сумма от сложения входных пяти приведенных разрядов остатка и одного старшего разряда суммы с двумя старшими разрядами переноса плюс шесть старших разрядов делителя. Если сигнал ADD=0, а сигнал SUB=1 , то на выходе SUMDS формируется шестиразрядная сумма от сложения входных пяти приведенных разрядов остатка и одного старшего разряда суммы с двумя старшими разрядами переноса плюс инверсия шести старших разрядов делителя. Выходы SUMDS со сдвигом на один разряд влево поступают на вход SUMDS , а пять старших разрядов – на вход SXSTR , последующего блока суммирования.

В состав блока деления входят также два пятидесятивосьмиразрядных регистра (REGSU и REGC). Эти регистры предназначены для временного хранения сумм и переносов.

Кроме этих регистров в состав блока деления входят еще два регистра: регистр сумм частного (SDVD1) и регистр переносов частного (SDVD).

Схема управления блоком деления построена по такому принципу, что за один период тактовой частоты должны быть получены две цифры частного. В соответствии с этим схема управления вырабатывает поочередно импульсы запуска пар блоков суммирования длительностью кратной периоду тактовой частоты. За это время должна сработать пара последовательных блоков суммирования. При этом параметры очередного остатка записываются в промежуточные регистры REGSU , REGC или RMA , RMB , а параметры частного – в регистры SDVD1 , SDVD . Блоки SUMDS и SXSTR построены на основе квазистатических схем с парафазным управлением, за счет чего имеют триггерные защелки, и не нуждаются в регистрах для хранения промежуточных результатов.

Регистры SDVD и SDVD1 представляют собой 64-разрядные регистры с предварительной установкой. Запись в каждый из них

характеризуется тем, что за один цикл записи записываются два бита, при этом следующие два бита будут записаны в следующую правую пару разрядов регистра. В случае формата ПЗ запись в эти регистры начинается с 63 разряда, в случае целочисленного формата - с 31 разряда.

После загрузки делимого, а если требуется (для целочисленных форматов), то после сдвига делимого начинается процесс деления. SUMDS4 имеет некоторую особенность по сравнению с остальными тремя шестиразрядными полными сумматорами. А именно, в самом начале работы делителя схема управления вырабатывает импульс на SUMDS4, который по входу D формирует пятиразрядный выходной код FS (59-63), а также по состоянгям входов D63 и A63 формирует признаки CADD и CSUB. Выход FS 58=0.

В результате срабатывания схемы SDMD4 на вход SXSTR1 в зависимости от комбинации знаков делимого и делителя поступают сигналы управления CADD и CSUB, а также сдвинутый с размножением знака на один разряд вправо пятиразрядный код делимого FS(59-63). На вход B(60-63) SUMDS1 этот код поступает со сдвигом на один разряд влево FS(59-62), на два младших входа B(58,59) подаются выходы регистра RMA B(58,59). После срабатывания SUMDS4 схема управления блоком деления вырабатывает импульс запуска первой пары блоков суммирования. Сначала срабатывает SXSTR1, которая формирует старший разряд частного в виде двухрядного кода суммы и переноса, а также сигналы управления сумматорами SUMDS1 и SUMDM1, ADD и SUB. В зависимости от значения сигналов управления на выходе сумматоров первого блока суммирования формируется либо делимое (в последующих циклах остаток), либо сумма делимого (в последующих циклах остатка) и делителя, либо разность между делимым (в последующих циклах остатком) и делителем, при

этом младшие 58 разрядов представлены в виде двухрядного кода сумм FS(0–57) и переносов FC(0–57), старшие шесть разрядов – в виде полной приведенной суммы FS(58–63). Кроме этого, SUMDS1 по знакам остатка и делителя формирует сигналы управления CADD, CSUB следующей итерацией деления. Выходы первого блока суммирования FS(0–63) со сдвигом на один разряд влево, а FC(0–57) со сдвигом на два разряда влево поступают на SUMDS2, SUMDM2 второго блока суммирования. Кроме этого сигналы CADD, CSUB и выходы FS(59–63) с SUMDS1 поступают на SXSTR2. После этого запускается SXSTR2. Второй блок суммирования срабатывает аналогично первому. Выходы второго блока суммирования FS(0–57), FC(0–57) записываются в регистры REGSU и REGC соответственно. Выходы FS(58–63) и сигналы CADD, CSUB имеют триггерные защелки. Две цифры частного в виде двухрядного кода DS0, DS1; DC0, DC1 записываются в регистры SDVD1 и SDVD2 соответственно. Выходы регистра сумм QS(0–57) и выходы полной приведенной суммы с SUMDS2 FS(58–63) со сдвигом на один разряд влево, а выходы регистра переносов QC(0–57) со сдвигом на два разряда поступают на SUMDS3, SUMDM3 третьего блока суммирования. Кроме этого сигналы CADD, CSUB и выходы FS(59–63) с SUMDS2 поступают на SXSTR3. На этом работа первой пары блоков суммирования в данном цикле заканчивается.

После этого схема управления работой делителя вырабатывает импульс запуска второй пары блоков суммирования. Вторая пара срабатывает аналогично первой паре блоков суммирования. Выходы четвертого блока суммирования FS(0–63) со сдвигом на один разряд влево поступают на запись в регистр RMA, FC(0–57) со сдвигом на два разряда влево поступают на запись в регистр RMB. Разряды остатка с SUMDS4 FS(59–62) поступают на

вход B(60–63) SUMDS1 . На вход SUMDS1 B(58–59) подаются разряды остатка из регистра RMA B(58–59). Разряды переносов остатка из регистра RMB C(2–59) без сдвига подаются на SUMDS1 , SUMDM2 первого блока суммирования. Выходы SUMDS4 CADD , CSUB , FS(59–63) поступают на вход SXSTR1 . Следующие две цифры частного в виде двухрядного кода DS2 , DS3 и DC2 , DC3 записываются в регистры SDVD1 и SDVD2 , соответственно. На этом работа второй пары блоков суммирования в данном цикле заканчивается.

Таким образом,за один цикл работы делителя, равному двум периодам тактовой частоты, получается четыре цифры частного в виде двухрядного кода. В соответствии с этим при делении чисел в формате бай**т требуется два цикла делителя, слово – четыре цикла делителя, двойное слово или деление четверного слова на слово или деление чисел в формате F – восемь циклов, в формате D и G – шестнадцать циклов.

По окончании работы делителя частное в виде двухрядного кода находится в регистрах SDVD1 и SDVD ,остаток – в регистрах RMA и RMB .

В случае целочисленного деления разрядность вычисленного частного точно соответствует разрядности формата. Поэтому необходимо вычислить точный остаток для того, чтобы определить необходимость проведения коррекции частного. В этом случае сразу по окончании деления происходит вычисление остатка

$$RMA + RMB \longrightarrow REGS, REGR \longrightarrow RMA$$

После этого определяется необходимость коррекции остатка. Если коррекция остатка не требуется, то на этом микрокоманда целочисленного деления заканчивается. Если требуется коррекция остатка, то делитель со сдвигс : на один разряд влево (сдвиг связан с тем, что остаток,записанный в RMA и RMB был со сдви-

гом) считывается с регистра RDLT на магистраль MD и через MUXB записывается в регистр RMB.

RMA ±RMB ⟶ REGS, REGR ⟶ RMA

Набрасываются флаги коррекции частного, после чего выполнение микрокоманды деления заканчивается.

В случае деления чисел в формате П3 разрядность вычисленного частного превышает разрядность формата. В силу этого отпадает необходимость в вычислении остатка с целью определения коррекции частного.

Поэтому сразу по окончании деления происходит перепись регистра SDVD1 в регистр RMA , регистра SDVD со сдвигом на один разряд вправо в регистр RMB . Так как SDVD1 и SDVD представляют собой двухрядный код сумм и переносов от вычитания положительной и отрицательной составляющих, то в нулевой разряд регистра RMB сигналом KORR записывается I. После чего происходит вычисление частного

RMA + RMB ⟶ REGS, REGR ⟶ RMA

Если требуется, то после этого выполняются нормализация и округление. На этом выполнение микрокоманды деления чисел в формате П3 заканчивается.

8.I6.I. Команды целочисленного деления

Первая микрокоманда – микрокоманда приема делителя

;MOV :B(W,L)/R(7) ⟶ R(6)_NOR;PDA

По этой микрокоманде операнд-делитель в соответствии с форматом считывается с входного буфера операндов на магистраль MD (байт – старшие 8 разрядов, слово – старшие I6 разрядов, двойное слово – старшие 32 разряда 64-разрядной шины MD , остальные разряды равны нулю) и записывается в регистры REGS , REGR, а из регистра REGR с арифметическим сдвигом на один разряд вправо – в регистр делителя.

Информация с REGS поступает на схему формирования кода нормализации, которая по I7 старшим разрядам регистра REGS формирует пятиразрядный двоичный код параметра нормализации. Код параметра нормализации, пройдя мультиплексор (второй вход мультиплексора выхода регистра RPSD) поступает на схему формирования кода сдвига. Эта схема формирует 4-разрядный двоичный код сдвига CSD.

Признаком микрокоманды - NOR запускается схема управления работой сдвигателя, после чего происходит сдвиг влево делителя в соответствии с кодом сдвига

$$REGS \xrightarrow[влево]{сдвиг} REGR \longrightarrow REGS, RMA, RDLT$$

При этом запись в RDLT происходит с арифметическим сдвигом вправо на один разряд

$$0 \longrightarrow RP1$$

$$CSD \longrightarrow RP2$$

$$RP1+RP2 \longrightarrow RPOR, RPSD \longrightarrow RP1$$

Так как максимальное число, на которое может быть сдвинут операнд сдвигателем равен 15_{IO}, а для делителя, представленного в формате двойное слово, параметр нормализации может быть больше, то в данном случае потребуется еще один цикл сдвига. По новому (сдвинутому) коду с REGS формируется новый 4-разрядный двоичный код сдвига CSD , на основании которого сдвигатель выполняет следующий сдвиг

$$REGS \xrightarrow[влево]{сдвиг} REGR \longrightarrow REGS, RMA, RDLT$$

Запись в RDLT выполняется опять же со сдвигом

$$CSD \longrightarrow RP2$$

$$RP1+RP2 \longrightarrow RPOR, RPSD \longrightarrow RP1$$

Таким образом,по окончании этой микрокоманды нормализованный делитель находится на регистре RDLT , а код сдвига - на регистрах RP1 , RPSD , RPOR .

Следующая микрокоманда - микрокоманда загрузки делимого и деления.

;MSD : B(W,L)/R(7) ⟶ R(0)_DIV; PDA,FA

По признаку FA и целочисленному формату операнд-делимое (операнд-делимое в формате байт занимает старшие восемь разрядов 32-разрядного регистра входного буфера операндов, в формате слово занимает старшие I6 разрядов 32-разрядного ре - гистра входного буфера операндов, в формате двойное слово записывает все 32 разряда 32-разрядного регистра входного буфера операндов) из 32-разрядного регистра входного буфера операндов переписывается в младшие 32 разряда регистра-аккумулятора. Старшие 32 разряда RAK заполняются знаком делимого.

Кроме этого, операнд -делимое в соответствии с форматом считывается с входного буфера операндов на магистраль MD и записывается в регистры RMA, REGS , REGR . По признаку записи - в регистр RMA делимого, в регистр RMB из регистра SDVD переписывается ноль.

В соответствии с алгоритмом целочисленного деления делимое должно иметь в два раза большую разрядность, чем разрядность делителя.

В связи с этим исходное делимое необходимо сдвинуть вправо с размножением знака на количество разрядов , равное формату делимого (делимое в формате байт - на 8 разрядов, в формате слово - на I6 разрядов, в формате двойное слово - на 32 разряда).

Полученное таким образом делимое необходимо сдвинуть влево на величину сдвига делителя при нормализации. Анализируя вышесказанное, можно сделать вывод, что делимое необходимо сдвинуть вправо с размножением знака на величину, определяемую разностью между константой формата делимого и параметром нормализации делителя.

По признаку MSD плюс формат (байт или слово, или двойное слово) в регистр блока обработки порядка RP2 записывается константа (байт - 8_{10}, слово - 16_{10}, двойное слово - 32_{10}), а в схеме управления сдвигателем устанавливается признак сдвига вправо. По признаку MSD запускается схема управления работой сдвигателя, после чего происходит сдвиг вправо делимого, на величину, определяемую кодом регистра RPSD.

$$RP2-RP1 \longrightarrow RPSD \longrightarrow RP1$$
$$CSD \longrightarrow RP2$$
$$REGS \xrightarrow[вправо]{сдвиг} REGR \longrightarrow REGS, RMA$$

Если формат делимого "двойное слово" или "слово", то в случае превышения кодом, хранящемся в регистре RPSD максимального параметра сдвига сдвигателя, потребуется второй цикл сдвига.

$$RP1-RP2 \longrightarrow RPSD \longrightarrow RP1$$
$$CSD \longrightarrow RP2$$
$$REGS \xrightarrow[вправо]{сдвиг} REGR \longrightarrow REGS, RMA$$

Для делимого в формате двойное слово может потребоваться третий цикл сдвига.

Как только в регистре RMA будет получен окончательный код делимого, запускается блок деления. По окончании работы блока деления в регистрах SDVD1 и SDVD будет храниться двухрядный код частного, а в регистрах RMA и RMB - двухрядный код остатка.

Далее следует окончательное вычисление остатка

$$RMA+RMB \longrightarrow REGS \longrightarrow REGR \longrightarrow RMA$$

Если знак остатка не равен знаку делимого и при этом остаток не равен нулю, то необходимо выполнить коррекцию остатка.

Делитель аппаратно считывается на шину MD и записывается в регистр RMB. В зависимости от соотношения знаков остатка и делителя происходит сложение остатка и делителя, либо вычитание из остатка делителя

RMA ± RMB ⟶ REGS ⟶ REGR ⟶ RMA

Если коррекции остатка не требовалось или коррекция потребовала вычитание из кода остатка кода делителя, то взводится флаг (FLAG1) добавления единицы к младшей цифре частного. Кроме этого, в случае, если коррекция потребовала вычитания из кода остатка кода делителя, взведется еще один флаг (FLAG2) добавление единицы к младшей цифре частного.

На этом выполнение данной микрокоманды заканчивается.

Третья микрокоманда - микрокоманда вычисления частного.

;MOV :Q/R(2) ⟶ R(6)_NOP;

По признаку записи в регистр делителя в формате четверное слово происходит перепись содержимого регистров SDVD1 и SDVD в регистры RMA и RMB соответственно. Этим же признаком блокируется запись в регистр RDLT и формируется импульс запуска арифметико-логического устройства, на котором осуществляется сложение сумм и переносов частного.

RMA + RMB ⟶ REGS ⟶ REGR ⟶ RMA

Если в предшествующей микрокоманде был взведен флаг FLAG1 то для операндов в формате"байт"единица замещает ноль в 24-ом разряде переносов,в формате"слово"- в I6-ом разряде переносов,в формате"двойное слово"- в нулевом разряде переносов, хранящихся на регистре RMB.

В нулевом разряде замена происходит с помощью сигнала KORR, в I6-ом и 24-ом разрядах переносов замена происходит на входном мультиплексоре ALU .

Если в предшествующей микрокоманде был взведен флаг FLAG2, то на вход ALU поступает сигнал C_0 . Для операндов в формате двойное слово при взведенном флаге FLAG2 добавление единицы к частному будет осуществляться через сигнал переноса C_0.

Для операндов в формате слово при взведенном флаге FLAG2 до-

бавление единицы к частному будет осуществляться через сигнал переноса C_{16} (младшие 16 разрядов RMA равны единице в каждом разряде, младшие 16 разрядов RMB равны нулю, плюс сигнал C_0). Для операндов в формате байт при взведенном флаге FLAG2 добавление единицы к частному будет осуществляться через сигнал переноса C_{24} (младшие 24 разряда RMA равны единице в каждом разряде, младшие 24 разряда RMB равны нулю, плюс сигнал C_0).

Последняя микрокоманда – микрокоманда выдачи данных и состояний

;MOV :B(W,L)/R(2) ⟶ R(7)_NOP; ZS, SA

Если по ходу выполнения микропрограммы было либо деление на ноль, либо целое переполнение, то на шину MD вместо регистра REGR будет считываться регистр-аккумулятор. В этом случае вместо частного будет выдаваться делимое.

8.16.2. Команда расширенного деления

Первая микрокоманда – микрокоманда приема делителя

;MOV :L/R(7) ⟶ R(6)_NOR; PDA

Выполнение этой микрокоманды подробно рассмотрено при описании работы первой микрокоманды команд целочисленного деления.

Вторая микрокоманда – микрокоманда загрузки делимого и деление

;MSD :Q/R(7) ⟶ R(0)_DIV; PDA, FA, ZV

По признаку FA и целочисленному формату младшие 32 разряда операнда-делимого переписываются из входного буфера операндов в младшие 32 разряда регистра-аккумулятора . Старшие 32 разряда RAK заполняются старшим разрядом младшей половины делимого. Кроме этого, операнд-делимое считывается с входного буфера операндов на магистраль MD и записывается в регистры RMA, REGS, REGR . По признаку записи делимого-в регистр RMA, в регистр RMB из регистра SDVD переписывается ноль. Так как исходно

формат делимого имеет в два раза большую разрядность, чем формат делителя, то не требуется сдвигать делимое вправо, но сдвиг влево на величину сдвига делителя при нормализации необходим.

По признаку MSD и формату четверное слово в регистр RP2 блока обработки порядка записывается ноль, в схеме управления сдвигателем устанавливается признак сдвига влево. По признаку MSD в регистр RP1 блока обработки порядка переписывается код, на который был сдвинут делитель, из регистра RPOR , а также запускается схема управления сдвигателем

$$RP1 - RP2 \longrightarrow RPSD \longrightarrow RP1$$

По коду RPSD формируется 4-разрядный код сдвига CSD, на величину которого сдвигатель осуществляет сдвиг.

$$REGS \xrightarrow[влево]{сдвиг} REGR \longrightarrow REGS, RMA$$
$$CSD \longrightarrow RP2$$

Если параметр сдвига , хранящийся в регистре RPSD, превышает максимально возможный параметр сдвига сдвигателя, то потребуется еще цикл (циклы) сдвига

$$RP1 - RP2 \longrightarrow RPSD \longrightarrow RP1$$
$$REGS \xrightarrow[влево]{сдвиг} REGR \longrightarrow REGS, RMA$$
$$CSD \longrightarrow RP2$$

В процессе сдвига делимого может произойти арифметическое переполнение. В этом случае взведется признак арифметического переполнения.

Как только в регистре RMA будет получен окончательный код делимого, запускается блок деления. По окончании работы блока деления в регистрах SDVD1 и SDVD будет храниться двухрядный код частного, а в регистрах RMA и RMB - двухрядный код остатка. Далее следует окончательное вычисление остатка

$$RMA + RMB \longrightarrow REGS \longrightarrow REGR \longrightarrow RMA$$

Если знак остатка не равен знаку делимого и при этом оста-

ток не равен нулю, то необходимо выполнить коррекцию остатка. Делитель аппаратно считывается на шину *MD* и записывается в регистр *RMB* . В зависимости от соотношения знаков остатка и делителя происходит сложение остатка и делителя, либо вычитание из остатка делителя.

$$RMA \pm RMB \longrightarrow REGS \longrightarrow REGR \longrightarrow RMA$$

Если коррекции остатка не требовалось, то взводится флаг (*FLAG1*) добавления единицы к младшей цифре частного. Если коррекция потребовала вычитание из кода остатка кода делителя, то взводятся флаги (*FLAG1*) и (*FLAG2*) добавления двойки к частному.

На этом выполнение данной микрокоманды заканчивается.

Третья микрокоманда - микрокоманда сдвига и сохранение остатка

;MSD :R/R(2) ⟶ R(4)_NOP;

По признаку *MSD* и формату *R* в регистр *RP2* блока обработки порядка записывается ноль, а в схеме управления сдвигателем устанавливается признак сдвига вправо.

По признаку *MSD* в регистр *RP1* блока обработки порядка из регистра *RPOR* переписывается код сдвига делителя при нормализации, а также запускается схема управления работой сдвигателя.

$$RP1-RP2 \longrightarrow RPSD \longrightarrow RP1$$

По коду *RPSD* формируется 4-разрядный код сдвига *CSD* , на величину которого сдвигатель осуществляет сдвиг остатка вправо

$$REGS \xrightarrow[вправо]{сдвиг} REGR \longrightarrow REGS, RMA$$

$$CSD \longrightarrow RP2$$

Если параметр сдвига, хранящийся в регистре *RPSD* , превышает максимально возможный параметр сдвига сдвигателя, то потребуется еще цикл (циклы) сдвига.

$$RP1-RP2 \longrightarrow RPSD \longrightarrow RP1$$
$$REGS \xrightarrow[\text{вправо}]{\text{сдвиг}} REGR \longrightarrow REGS, RMA$$
$$CSD \longrightarrow RP2$$

По окончании работы сдвигателя остаток с арифметическим сдвигом вправо на один разряд переписывается из регистра *REGR* в регистр *RDLT*.

Четвертая микрокоманда – микрокоманда вычисления частного

;MOV :Q/R(2) ⟶ R(6)_NOP;

По признаку записи в регистр делителя в формате четверное слово происходит перепись содержимого регистров *SDVD1* и *SDVD* в регистры *RMA* и *RMB* соответственно. Этим же признаком блокируется запись с шины *MD* содержимого регистра *REGR* в регистр *RDLT*, а также формируется импульс запуска *ALU*, на котором осуществляется сложение суммы и переносов частного

$$RMA+RMB \longrightarrow REGS \longrightarrow REGR \longrightarrow RMA$$

Если во второй микрокоманде был взведен флаг *FLAG1*, то с помощью сигнала *KORR* в нулевой разряд *RMB* будет записана I. Если был взведен флаг *FLAG2*, то на вход переноса в нулевой разряд *ALU* будет подан сигнал C_0.

Пятая микрокоманда – микрокоманда выдачи частного и состояний

;MOV :L/R(2) ⟶ R(7)_NOP;ZS

Если по ходу выполнения микропрограммы было либо деление на ноль, либо целое переполнение, то на шину *MD* вместо регистра *REGR* будет читаться регистр-аккумулятор. В этом случае вместо частного будет выдаваться младшая половина делимого.

Шестая микрокоманда – микрокоманда восстановления остатка

;MOV :L/R(6) ⟶ R(2)_NOP;FS

Остаток со сдвигом на один разряд влево читается на магистраль *MD* и записывается в регистр *REGS*, *REGR*, после

чего по признаку FS сдвигается на 32 разряда вправо.

$$REGS \xrightarrow[на\ 32]{сдвиг\ вправо} REGR, REGS$$

Последняя микрокоманда - микрокоманда выдачи остатка

$$;MOV:L/R(2) \longrightarrow R(7)\_NOP; SA$$

Если по ходу выполнения микропрограммы было либо деление на ноль, либо целое переполнение, то вместо остатка на внешнюю шину AD будет выдаваться ноль.

8.I6.3. Команды деления чисел, представленных в формате ПЗ

Первая микрокоманда - микрокоманда загрузки делителя

$$;MPS:F(D,G)/R(7) \longrightarrow R(6)\_NOP; PDA, FN$$

Мантисса операнда-делителя из входного буфера операндов считывается на шину MD и записывается в регистр делителя.

Порядок из входного буфера операндов записывается в регистр блока обработки порядка RP1. В регистр RP2 записывается константа. Для делителя в формате F или D в RP2 записывается константа 080_{I6}, в формате G - 400_{I6}. После этого из порядка делителя вычитается смещение. Признак FN в данной микрокоманде блокирует сигнал переноса в нулевой разряд арифметико-логического устройства блока обработки порядка. Таким образом в блоке обработки порядка будет выполнена следующая операция:

$$RP1-RP2-1 \longrightarrow RPOR, RPSD \longrightarrow RP1$$

Вычитание единицы из порядка делителя связано с тем, что в следующей микрокоманде мантисса делимого должна быть сдвинута на один разряд вправо, вследствии чего порядок делимого должен быть на I. После этого из увеличенного на единицу порядка делимого вычитается порядок делителя, а это равнозначно вычитанию из порядка делимого, уменьшенного на единицу порядка делителя.

Вторая микрокоманда - микрокоманда загрузки делимого и деления

;MWP2 :F(D,G)/R(7) ⟶ R(0)_DIV; PDA, ZV

Мантисса делимого с входного буфера операндов считывается на шину MD и со сдвигом на один разряд вправо записывается в регистр RMA . По признаку записи делимого в регистр RMA, в регистр RMB переписывается ноль из регистра SDVD.

Порядок делимого из входного буфера операндов переписывается в регистр блока обработки порядка RP2 . После этого запускается блок деления. Параллельно с работой блока деления блок обработки порядка выполняет вычитание из порядка делимого порядок делителя

RP2-RP1 ⟶ RPOR, RPSD ⟶ RP1

По окончании работы блока деления двухрядный код мантиссы частного переписывается из регистров SDVD1 и SDVD в регистры RMA и RMB соответственно. В нулевой разряд RMB запишется сигнал KORR=1 . Затем запускается ALU операционного блока, на котором вычисляется 32 (F -формат) или 64 (D- и G-формат) мантиссы частного

RMA+RMB ⟶ REGS ⟶ REGR ⟶ RMA

После того как получена мантисса частного в однорядном коде, если требуется, то выполняется или нормализация или округление,или нормализация и округление мантиссы частного.

Последняя микрокоманда – микрокоманда выдачи частного и состояний

;MOV :F(D,G)/R(2) ⟶ R(7)_NOP; ZS, SA

9. СОЕДИНЕНИЕ СБИС СОПРОЦЕССОРА СО СБИС ЦЕНТРАЛЬНОГО ПРОЦЕССОРА В ВЫЧИСЛИТЕЛЬНОМ УСТРОЙСТВЕ

Схема соединения СБИС СПРЦ со СБИС ЦПР представлена на рис. 24 . На схеме изображены следующие устройства и элементы:

СБИС ЦПР (микросхема CPU типа ЛІ839ВМІ);

СБИС СПРЦ (микросхема FPU типа ЛІ839ВМ2);

ПЗУ накопителя микропрограмм ROM емкостью І6Кх32 разряда;

узел формирования сигналов CE для ПЗУ и MSDS для ЦПР и СПРЦ;

генератор тактовой частоты F_{CLCI} до І0 МГц;

резисторы хранения уровня электрической "І" R1-R10;

цепи коммутации.

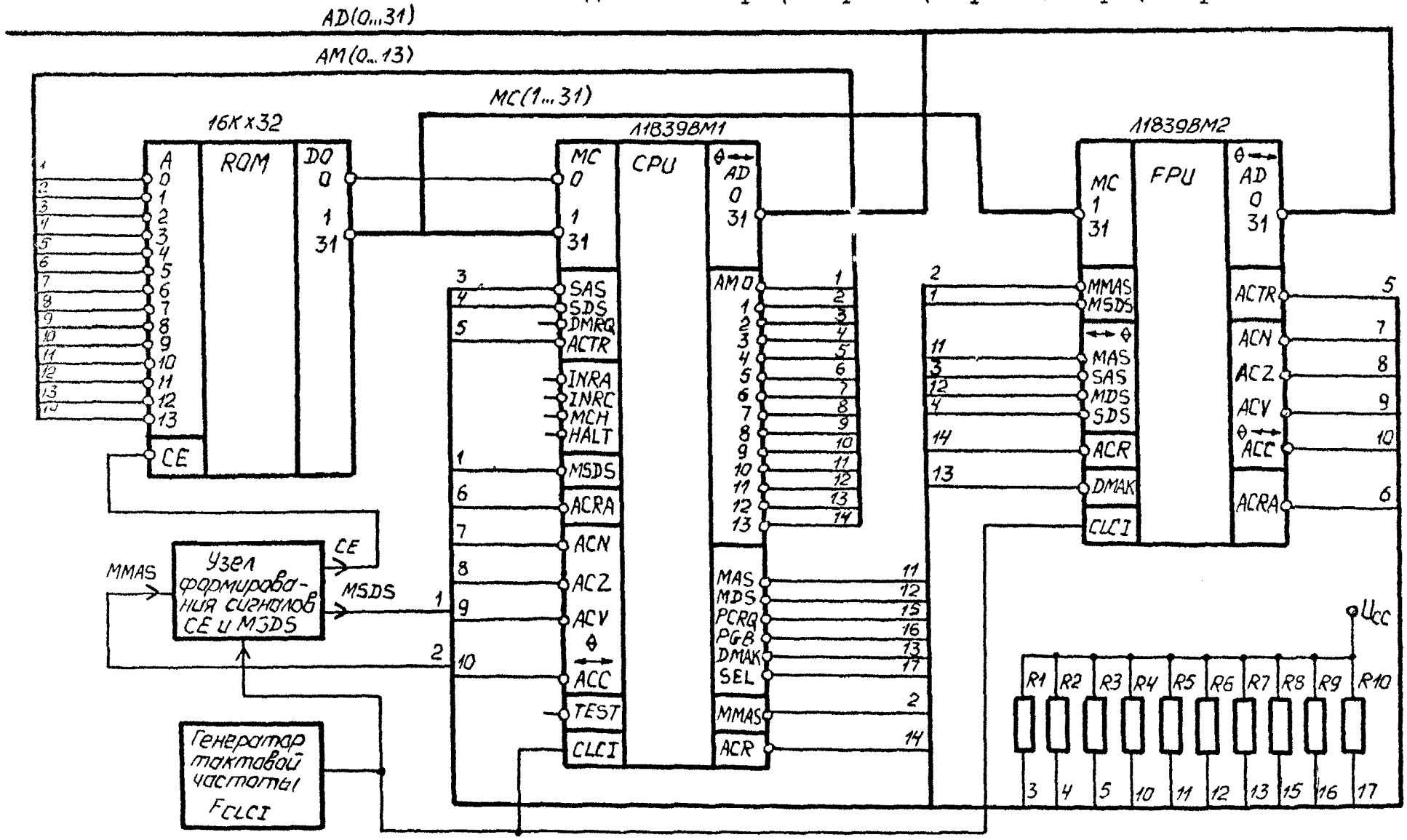
ЦПР и СПРЦ имеют общую микропрограммную магистраль от ПЗУ микропрограмм в разрядах MC(1-31). Разряд MCO поступает только на ЦПР. ПЗУ может быть выполнено на одной или нескольких микросхемах с общей организацией памяти І6Кх32, временем выборки по сигналу $\overline{CE}$ не более І00 нс и временем цикла не более 200 нс при работе на максимальной тактовой частоте.

ЦПР и СПРЦ имеют также общую магистраль адресов и данных AD (0-3І).

Для обеспечения функционирования СПРЦ с ЦПР соединяются одноименные выводы MAS, MDS, SAS, SDS, MMAS, MSDS, ACTR, ACRA, DMAK, ACN, ACZ, ACV, ACC, CLCI.

На выходах ЦПР и СПРЦ с 3-м состоянием подключаются резисторы к цепи источника питания для обеспечения уровня электрической "І" в моменты отключения выходов.

Схема соединения сопроцессора с центральным процессором



R1...R10 – резисторы 0,125-2,7 кОм ± 10%.

Примечание. Не задействованные сигналы ПМР, а также сигналы MAS, SAS, MDS, SDS, DMAK, CLCI используются для связи с другими микросхемами МПК и при построении микро-ЭВМ.

Рис 24

IO. ПРОИЗВОДИТЕЛЬНОСТЬ ВЫЧИСЛИТЕЛЯ, ПОСТРОЕННОГО НА БАЗЕ МИКРОСХЕМ Л1839ВМ1 И Л1839ВМ2

IO.I. Производительность при выполнении команд в формате "память-память" с минимальным временем обращения к памяти:

MULB2,3 - 500 тыс.оп/с (2 мкс)

MULW2,3 - 500 тыс.оп/с (2 мкс)

MULL2 - 660 тыс.оп/с (I,5 мкс)

MULL3 - 625 тыс.оп/с (I,6 мкс)

ASHQ - 625 тыс.оп/с (I,6 мкс)

- 330 тыс.оп/с (3 мкс)

EMUL - 385 тыс.оп/с (2,6 мкс)

DIVB2,3 - 300 тыс.оп/с (3,2 мкс)

DIVW2,3 - 300 тыс.оп/с (3,2 мкс)

DIVL2,3 - 238 тыс.оп/с (4,2 мкс)

EDIV - I50 тыс.оп/с (6,5 мкс) *min*

- I30 тыс.оп/с (7,5 мкс) *max*

ADDF2, SUBF2 - 660 тыс.оп/с (I,5 мкс) *min*

- 385 тыс.оп/с (2,6 мкс) *max*

ADDF3, SUBF3 - 588 тыс.оп/с (I,7 мкс) *min*

- 357 тыс.оп/с (2,8 мкс) *max*

MULF2 - 660 тыс.оп/с (I,5 мкс) *min*

- 454 тыс.оп/с (2,2 мкс) *max*

MULF3 - 625 тыс.оп/с (I,6 мкс) *min*

- 435 тыс.оп/с (2,3 мкс) *max*

DIVF2,3 - 320 тыс.оп/с (3,I мкс) *min*

- 285 тыс.оп/с (3,5 мкс) *max*

ADDD2,3
SUBD2,3 - 435 тыс.оп/с (2,3 мкс) *min*

ADDG2,3
SUBG2,3 - 243 тыс.оп/с (4,2 мкс) *max*

DIVD2,3	176 тыс.оп/с (5,6 мкс)	*min*
DIVG2,3	166 тыс.оп/с (6 мкс)	*max*
MULD2,3	300 тыс.оп/с (3,3 мкс)	*min*
MULG2,3	250 тыс.оп/с (4 мкс)	*max*

Лист регистрации изменений

Изм.	Номера листов (страниц)				Всего листов (страниц) в докум.	№ докум.	Входящий № сопроводительного документа и дата	Подп.	Дат.
	Измененных	Замененных	новых	аннулированных					