

МИНИСТЕРСТВО ВЫСШЕГО И СРЕДНЕГО СПЕЦИАЛЬНОГО ОБРАЗОВАНИЯ ЛИТОВСКОЙ ССР
ИНСТИТУТ ПОВЫШЕНИЯ КВАЛИФИКАЦИИ СПЕЦИАЛИСТОВ НАРОДНОГО ХОЗЯЙСТВА ЛИТОВСКОЙ ССР

Кафедра вычислительной техники

СИСТЕМА КОМАНД ВЫЧИСЛИТЕЛЬНОГО КОМПЛЕКСА СМ 1700

Материалы для практических занятий

Вильнюс 1983

Материалы для практических занятий подготовил канд. техн. наук П. В. Б а л ь б е р г

Обсуждены и одобрены на заседании кафедры вычислительной техники 28 декабря 1987 г.

В материалах для практических занятий приведен перечень команд вычислительного комплекса СМ 1700, указаны наименования, mnemonicические и шестнадцатиримчные коды, форматы, спецификаторы операндов, алгоритмы выполнения, значения кодов условий и краткие описания команд. Предназначены для слушателей Института, изучающих принципы работы и программирование ВМ СМ 1700.

Рецензент — зав. сектором СКБ вычислительных машин Пулокас Д.

ВВЕДЕНИЕ

Вычислительный комплекс СМ 1700 предназначен в основном для использования в системах автоматизации проектирования и управления производством и предприятиями. Комплекс представляет собой вычислительную машину СМ 2700 и широкий набор внешних устройств (запоминающих, отображения, ввода-вывода цифровой, алфавитной и графической информации).

Вычислительная машина СМ 2700 предназначена для выполнения арифметических и логических операций, реализующих систему команд комплекса, и обеспечивает работу операционной системы. Она также обеспечивает программную совместимость с комплексами СМ 1600, СМ 1420 и другими моделями СМ, построенными на базе процессора ВК СМ-4.

Система команд ориентирована на обработку двоичной, десятичной и символьной информации следующих форматов:

двоичные числа с фиксированной запятой — 8, 16, 32, 64 и 128 двоичных разрядов;

двоичные числа с плавающей запятой — 32, 64 и 128 двоичных разрядов;

десятичные числа — до 31 десятичного разряда плюс знак;

символьные строки — до 64х35 символов;

двоичные поля переменной длины — до 32 двоичных разрядов.

Описываемые команды комплекса сведены в таблицу и объединены в отдельные группы, близкие по свойствам. В таблице приводятся следующие сведения:

- 1) наименование команды;
- 2) мнемонический и шестнадцатиричный коды каждой команды;
- 3) спецификаторы операндов, представленные названием, типом и последовательностью появления, которые определяют формат команды;
- 4) алгоритм выполняемой функции (где это возможно);
- 5) коды условий, указывающие их значения после выполнения команды. Коды устанавливаются в единицу, если данное условие выполнено;
- 6) описание команды в сокращенном виде, не содержащее исключений и условий нарушения.

Название операнда соответствует его функциональному назначению в команде и указывается сокращенным словом английского языка:

src — источник,

dst — приемник,

add — слагаемое,

sum — сумма,

dif — разность,

min — уменьшаемое,

sub — вычитаемое,

divr — делитель,

divd — делимое,

quo — частное,

rem — остаток,

mulr — множитель,

muld — множимое,

prod — произведение,

mask — маска,

cnt — счетчик,

pos — позиция,

size — размер,

base — базовый адрес,

startpos — начальная позиция,

findpos — найденная позиция,

limit — граница,

index — индекс,

displ — смещение,

selector — селектор,

arglist — список аргументов,

numarg — число аргументов,

subatript — индексное значение,

low — нижний предел,

high — верхний предел,

indexin — входной индекс,

indexout — выходной индекс,

entry — элемент,

header — заголовок,

pred — предыдущий элемент,

addr — адрес,

frac — дробная часть,

int — целая часть,

mulrx — расширенный множитель,

arg — аргумент,

degree — степень,

tbladdr — адрес таблицы,

len - длина,
 srcaddr - адрес источника,
 arclen - длина источника,
 fill - заполнитель,
 char - символ,
 objlen - объектная строка,
 dstaddr - адрес приемника,
 dstlen - длина приемника,
 addlen - длина слагаемого,
 addaddr - адрес слагаемого,
 sumlen - длина суммы,
 sumaddr - адрес суммы,
 sublen - длина вычитаемого,

srcaddr - адрес вычитаемого,
 diflen - длина разности,
 difaddr - адрес разности,
 minlen - длина уменьшаемого,
 minaddr - адрес уменьшаемого,
 mulrlen - длина множителя,
 mulraddr - адрес множителя,
 muldlen - длина множимого,
 muldaddr - адрес множимого,
 prodlen - длина произведения,
 prodaddr - адрес произведения,
 divrlen - длина делителя,
 divraddr - адрес делителя,

divdlen - длина делимого,
 divdaddr - адрес делимого,
 quotlen - длина частного,
 quoadr - адрес частного,
 round - округление,
 pattern - строка маски,
 tbl - таблица,
 incrc - начальный СКС,
 strlen - длина потока данных,
 stream - поток данных,
 mode - режим,
 code - код,
 procreg - процессорный регистр,
 esc - выходной символ.

Тип доступа спецификатора операнда обозначается первой буквой поля названия операнда:

a - вычисляется адрес операнда в памяти;
 b - спецификатор операнда представляет собой смещение перехода;
 m - операнд может быть считан, модифицирован и записан;
 r - операнд может быть считан;
 v - вычисляется адрес операнда в памяти или Rn;
 w - операнд может быть записан.

Тип данных обозначается второй буквой поля названия операнда:

b - байт;
 d - плавающее слово формата D;
 f - плавающее слово формата F;
 g - плавающее слово формата G;
 h - плавающее слово формата H;
 l - длинное слово;
 o - октаслово;
 q - кватрасслово;
 w - слово;
 x - первый тип данных определяется командой;
 y - второй тип данных определяется командой.

1. ЦЕЛОЧИСЛЕННЫЕ АРИТМЕТИЧЕСКИЕ И ЛОГИЧЕСКИЕ КОМАНДЫ

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
1. Пересылка	90	MOVB	src.rx, dst.wx	$dst \leftarrow src$	$N \leftarrow dst < 0,$ $Z \leftarrow dst = 0,$ $V \leftarrow 0,$ $C \leftarrow C$	По адресу операнда (оп.)-приемника пересылается содержимое оп.-источника
	B0	MOVW				
	D0	MOVL				
	7D	MOVQ				
	7DFD	NOVO				
2. Пересылка с отрицанием	80	MNECB	src.rx, dst.wx	$dst \leftarrow -src$	$N \leftarrow dst < 0,$ $Z \leftarrow dst = 0,$ $V \leftarrow \text{переполн.},$ $C \leftarrow dst \neq 0$	По адресу оп.-приемника пересылается отрицательное содержимое оп.-источника
	A3	MNEGW				
	CE	MNEQL				
3. Пересылка с инверсией	92	MCOMB	src.rx, dst.wx	$dst \leftarrow \overline{src}$	$N \leftarrow dst < 0,$ $Z \leftarrow dst = 0,$ $V \leftarrow 0,$ $C \leftarrow C$	По адресу оп.-приемника пересылается инвертированное содержимое оп.-источника
	B2	MCOMW				
	D2	MCOML				
4. Пересылка с левыми нулями	9B	MOVZBW	src.rx, dst.wx	$dst \leftarrow \text{EXT}(src),$ где EXT обозначает дополнение ведущими нулями источника до размера приемника	$N \leftarrow 0,$ $Z \leftarrow dst = 0,$ $V \leftarrow 0,$ $C \leftarrow C$	Содержимое оп.-источника пересылается в младшие разряды оп.-приемника, а старшие разряды заполняются нулями
	9A	MOVZBL				
	3C	MOVZLL				
	94	CLRB	dst.w	$dst \leftarrow 0$	$N \leftarrow 0,$ $Z \leftarrow 1,$ $V \leftarrow 0,$ $C \leftarrow C$	По адресу оп.-приемника заносится ноль
	34	CLRW				
	34	CLRL				
	7C	CLRLQ				

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условия	Описание
	16-ричный	мнемонический				
	7CFD	CLRO				
6. Преобразование	99	CVTBW	src.rx, dst.wy	dst ← преобразованный src	N ← dst < 0, Z ← dst = 0, V ← переполн., C ← C	Производится преобразование формата оп.-источника в формат оп.-приемника
	98	CVTEB				
	33	CVTWB				
	32	CVTEW				
	76	CVTLB				
	F7	CVTLW				
7. Сравнение	91	CMFB	src1.rx, src2.rx	src1 - src2	N ← src1 < src2, Z ← src1 = src2, V ← 0, C ← src1 < src2	Операнды источников сравниваются и остаются неизменными
	B1	CMFV				
	D1	CMFL				
8. Сложение	80	ADDR2	add.rx, sum.wx	sum ← sum + add	N ← sum < 0, Z ← sum = 0, V ← переполн., C ← перенос из ст. разряда	Производится сложение двух операндов
	81	ADDR3	add1.rx, add2.rx, sum.wx	sum ← add1 + add2		
	A0	ADDW2				
	A1	ADDW3				
	CC	ADDL2				
	C1	ADDL3				
9. Сложение выравненных слов с блокировкой памяти	58	ADAW1	add.rx, sum.wx	sum ← sum + add	То же, что и в п. 6	Производится сложение выравненного оп.-суммы с оп.-слагаемым с одновременной блокировкой памяти

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условия	Описание
	16-ричный	мнемонический				
10. Сложение с переносом	D8	ADWC	add.rl, sum.ml	$sum \leftarrow sum + add + C$	То же, что и в п. 3	Производится сложение операндов с кодом условия C
11. Инкрементирование	96	INCB	sum.mx	$sum \leftarrow sum + 1$	То же, что и в п. 3	К содержимому оп.-суммы прибавляется единица
	B6	INCW				
	D6	INCL				
12. Вычитание	82	SUBB2	sub.rx, dif.mx	$dif \leftarrow dif - sub$	N $\leftarrow$ dif < 0, Z $\leftarrow$ dif = 0, V $\leftarrow$ переполн., C $\leftarrow$ перенос из ст. разряда	Производится вычитание содержимого оп.-вычитаемого из содержимого оп.-разности (2-операндный формат команды) или из содержимого оп.-уменьшаемого (3-операндный формат)
	83	SUBB3	sub.rx, min.rx, dif.wx	$dif \leftarrow min - sub$		
	A2	SUBW2				
	A3	SUBW3				
	C2	SUBL2				
	C3	SUBL3				
13. Вычитание с переносом	D9	SBWC	sub.rl, dif.ml	$dif \leftarrow dif - sub - C$	То же, что и в п. 12	Содержимое оп.-вычитаемого и значение кода условий C вычитается из содержимого оп.-разности
14. Декрементирование	97	DECB	dif.mx	$dif \leftarrow dif - 1$	То же, что и в п. 12	Из оп.-разности вычитается единица
	B7	DECW				
	D7	DECL				
15. Деление	86	DIVD2	divr.rx, quo.mx	$quo \leftarrow quo / divr$	N $\leftarrow$ quo < 0, Z $\leftarrow$ quo = 0, V $\leftarrow$ переполнение или divr=0,	Содержимое оп.-частного (в 2-операндном формате содержится делимое) или оп.-делимого (3-операндный формат)
	87	DIVD3	divr.rx, divd.rx, quo.wx	$quo \leftarrow divd / divr$		
	A6	DIVW2				

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условия	Описание
	16-ричный	мнемонический				
16. Расширенное деление	A7	DIVW2			C ← 0	Делится на содержимое оп.-делителя. Результат вычисляется по адресу оп.-частного
	06	DIVL2				
	07	DIVL3				
	7B	EDIV	divr.rl, divd.rq, quo.wl, rem.wl	quo ← divd/divr rem ← остаток	То же, что и в п. 16	Содержимое оп.-делимого делится на содержимое оп.-делителя. Частное запоминается по адресу оп.-частного, а остаток - по адресу оп.-остатка
17. Умножение	84	MULB2	mulr.rx, prod.wx	prod ← prod * mulr	N ← prod < 0, Z ← prod = 0, V ← переполн., C ← 0	Содержимое оп.-производимого (в 2-операндном формате содержится множимое) или оп.-множимого (3-операндный формат) умножается на содержимое оп.-множителя. Младшие разряды результата запоминаются по адресу оп.-произведения
	85	MULB3	mulr.rx, muld.rx, prod.wx	prod ← muld * mulr		
	A4	MULW2				
	A5	MULW3				
	C4	MULL2				
	C5	MULL3				
18. Расширенное умножение	7A	EMUL	mulr.rl, muld.rl, add.rl, prod.wq	prod ← (muld * mulr) + EXT(add), где EXT обозначает расширение знака оп.- слагаемого до размера оп.-произведения	N ← prod < 0, Z ← prod = 0, V ← 0, C ← 0	Содержимое оп.-множимого умножается на содержимое оп.-множителя. К произведению добавляется содержимое оп.-слагаемого
19. Проверка	95	TSTB	src.rx	src = 0	N ← src < 0, Z ← src = 0, V ← 0, C ← 0	Устанавливаются коды условия в соответствии с содержимым источника
	B5	TSTW				
	D5	TSTL				

наименование	код команды		спецификаторы	алгоритм	коды условий	описание
	16-ричный	мнемонический				
20. Сброс разрядов	8A	BICB2	mask.rx, dst.mx	$dst \leftarrow dst \wedge \overline{mask}$, $dst \leftarrow src \wedge \overline{mask}$	$N \leftarrow dst < 0$, $Z \leftarrow dst = 0$, $V \leftarrow 0$, $C \leftarrow C$	Содержимое оп.-приемника (2-операндный формат) или содержимое оп.-источника (3-операндный формат) логически умножается на инвертированное содержимое оп.-маски
	8B	BICB3	mask.rx, src.rx, dst.wx			
	8A	BICW2				
	8B	BICW3				
	8A	BICL2				
	8B	BICL3				
21. Установка разрядов	88	BISB2	mask.rx, dst.mx	$dst \leftarrow dst \vee \overline{mask}$, $dst \leftarrow src \vee \overline{mask}$	То же, что и в п. 20	Содержимое оп.-приемника (2-операндный формат) или содержимое оп.-источника (3-операндный формат) логически складывается с содержимым оп.-маски
	89	BISB3	mask.rx, src.rx, dst.wx			
	88	BISW2				
	89	BISW3				
	88	BISL2				
	89	BISL3				
22. Исключающее ИЛИ	8C	XORB2	mask.rx, dst.mx	$dst \leftarrow dst \oplus \overline{mask}$, $dst \leftarrow src \oplus \overline{mask}$	То же, что и в п. 20	Содержимое оп.-приемника (2-операндный формат) или содержимое оп.-источника (3-операндный формат) складывается по модулю 2 с содержимым оп.-маски
	8D	XORB3	mask.rx, src.rx, dst.wx			
	8C	XORW2				
	8D	XORW3				
	8C	XORL2				
	8D	XORL3				
23. Инвертирование разрядов	83	BITB	mask.rx, src.rx	$tmp \leftarrow src \wedge \overline{mask}$, где tmp — временный операнд	$N \leftarrow tmp < 0$, $Z \leftarrow tmp = 0$, $V \leftarrow 0$, $C \leftarrow 0$	Содержимое оп.-источника логически умножается на содержимое оп.-маски. Результатом операции являются коды условий
	83	BITW				
	83	BITL				

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
24. Арифметический сдвиг	78 79	ASHL ASHQ	cnt.rb, src.rh, dst.wx	$dst \leftarrow src$, сдвинутый на значение cnt	$N \leftarrow dst < 0$, $Z \leftarrow dst = 0$, $V \leftarrow$ переполн., $C \leftarrow 0$	Содержимое оп.-источника арифметически сдвигается влево или вправо на число разрядов, определенное значением оп.-счетчика
25. Циклический сдвиг длинного слова	9C	ROTL	cnt.rb, src.rl, dst.wl	$dst \leftarrow src$, сдвинутый на значение cnt	$N \leftarrow dst < 0$, $Z \leftarrow dst = 0$, $V \leftarrow 0$, $C \leftarrow 0$	Содержимое оп.-источника циклически сдвигается влево или вправо на число разрядов, определенное значением оп.-счетчика
26. Загрузка длинного слова в стек	DD	FUSHL	src.rl	$-(SP) \leftarrow src$	$N \leftarrow src < 0$, $Z \leftarrow src = 0$, $V \leftarrow 0$, $C \leftarrow 0$	Длинное слово, содержащееся по адресу оп.-источника, помещается в стек

11. АДРЕСНЫЕ КОМАНДЫ

Наименование	КОД КОМАНДЫ		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	Мнемонический				
27. Пересылка адреса	9E	MOVAB	src.ax, dst.wl	$dst \leftarrow src$	$X \leftarrow dst < 0,$ $Z \leftarrow dst = 0,$ $V \leftarrow 0,$ $C \leftarrow 0$	Адрес оп.-источника пересылается по адресу оп.-приемника
	3E	MOVAW				
	DE	MOVAB				
		MOVAF				
	7E	MOVAB				
		MOVAD				
		MOVAC				
28. Загрузка в стек адреса	7EFD	MOVANI	src.ax	$-(SP) \leftarrow src$	$X \leftarrow src < 0,$ $Z \leftarrow src = 0,$ $V \leftarrow C,$ $C \leftarrow C$	Адрес оп.-источника записывается в стек
		MOVAO				
	9F	PUSHAD				
	3F	PUSHAW				
	DF	PUSHAL				
		PUSHAH				
	7F	PUSHAQ				
		PUSHAD				
		PUSHAG				
	7FFD	PUSHAC				
		PUSHAH				

III. КОМАНДЫ ДЛЯ ПОЛЕЙ С ПЕРЕМЕРНЫМ КОЛИЧЕСТВОМ РАЗРЯДОВ

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
29. Сравнение поля	BC	CMRV	pos.rl, size.rb, base.vb, arg.rl	ЗЕХТ ПОЛЕ (pos, size, base) ← arg,	N ← ПОЛЕ < arg, Z ← ПОЛЕ = arg, V ← 0, C ← ПОЛЕ < arg	Содержимое поля, определяемое операндами позиции, размера и базового адреса, сравнивается с содержанием оп.-источника. Результатом операции является установка кодов условий
	BD	CMRZV		ЗЕХТ ПОЛЕ (pos, size, base) ← arg		
30. Чтение поля	BE	EXTV	pos.rl, size rb, base.vb, dst.wl	dst ← ЗЕХТ ПОЛЕ (pos, size, base),	N ← dst < 0, Z ← dst = 0, V ← 0, C ← C	Содержимое поля, определяемое операндами позиции, размера и базового адреса, передается по адресу оп.-приемника
	BF	EXTZV		dst ← ЗЕХТ ПОЛЕ (pos, size, base)		
31. Поиск первого сброшенного (установленного) разряда	EB	FFC	startpos.rl, size.rt, base.vb, findpos.wl	findpos ← startpos + количество разрядов до найденного	N ← 0, Z ← разряд не найден, V ← 0, C ← 0	В поле, определяемом операндами начальной позиции, размера и базового адреса, производится поиск первого разряда, состояние которого указано в команде, и его позиция заносится по адресу оп.-назначенной позиции
	EA	FFZ				
32. Запись поля	FC	INZV	arg.rl, pos.rl, size.rb, base.vb	ПОЛЕ (pos, size, base) ← arg [(size-1):0]	Не изменяются	Содержимое разрядов [(size - 1):0] оп.-источника заносится в поле, определяемое операндами позиции, размера и базового адреса

19. КОМАНДЫ УПРАВЛЕНИЯ

Наименование	Код команды		Спецификаторы	Алгоритмы	Коды условий	Описание
	16-ричный	мнемонический				
33. Сложение, сравнение и переход	9D	ASPB	limit.rx, add.rx, index.mx, displ.bw	1) $index \leftarrow index + add$; 2) $index - limit$; 3) $PC \leftarrow PC + SEXT$ (displ), если $((add \geq 0) \wedge (index \leq limit)) \vee ((add < 0) \wedge (index \geq limit))$	N $\leftarrow$ index < 0, Z $\leftarrow$ index = 0, V $\leftarrow$ переполн., C $\leftarrow$ C	Содержимое оп.-слагаемого складывается с содержимым оп.-индекса. Если слагаемое больше или равно нулю, а результат сравнения меньше или равен нулю или если слагаемое отрицательное, а результат сравнения больше или равен нулю, то происходит переход по адресу $PC + SEXT(displ)$
	3D	ASBW				
	F1	ASBL				
	4F	ASBF				
	6F	ASBD				
	4FFD	ASBG				
	6FFD	ASBH				
34. Прибавить единицу и перейти, если "меньше или равно"	F3	AOBLEQ	limit.rl, index.ml, displ.bb	1) $index \leftarrow index + 1$; 2) $PC \leftarrow PC + SEXT$ (displ), если $index \leq limit$	N $\leftarrow$ index < 0, Z $\leftarrow$ index = 0, V $\leftarrow$ переполн., C $\leftarrow$ C	К содержимому оп.-индекса прибавляется единица. Если после сравнения сумма меньше или равна содержимому оп.-границы, то происходит переход по адресу $PC + SEXT(displ)$
35. Прибавить единицу и перейти, если "меньше"	F2	AOBLSS	limit.rl, index.ml, displ.bb	1) $index \leftarrow index + 1$; 2) $PC \leftarrow PC + SEXT$ (displ), если $index < limit$	То же, что и в п. 34	К содержимому оп.-индекса прибавляется единица. Если после сравнения сумма меньше содержимому оп.-границы, то происходит переход по адресу $PC + SEXT(displ)$

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условия	Описание
	16-ричная	мнемонический				
36. Переход по условию			dlpr1.bb	$PC \leftarrow PC + \text{next}(dlpr1)$, если условие выполнено	Не изменяется	Происходит переход по адресу $PC + \text{next}(dlpr1)$, если условие выполнено
Знаковый переход по JOLLS	14	BGTR				X или $Z = 0$
Знаковый переход по JALLS или RAJLO	15	BLBQ				X или $Z = 1$
Знаковый переход по JALJLSTIV	12	BNEQ				$Z = 0$
Беззнаковый переход по JALJLSTIV	12	BNEQU				$Z = 0$
Знаковый переход по RAJLJLSTIV	13	BZQL				$Z = 1$
Беззнаковый переход по RAJLJLSTIV	13	BZQLU				$Z = 1$
Знаковый переход по JALLS или RAJLO	10	ECBQ				$X = 0$
Знаковый переход по JALLS	19	BLSS				$X = 1$
Беззнаковый переход по JALLS	1A	BGTRU				C или $Z = 0$
Беззнаковый переход по JALLS или RAJLO	1B	BLBQU				C или $Z = 1$
Переход по отсутст- вию перемещения	1C	BVC				$V = 0$
Переход по наличие перемещения	1D	BVS				$V = 1$
Беззнаковый переход по JOLLS или RAJLO	1E	BGBQU				$C = 0$
Переход по отсутст- вию переноса	1F	BCC				$C = 0$
Беззнаковый переход по JALLS	1F	BLSSU				$C = 1$
Переход по наличие переноса	1F	BCS				$C = 1$

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
37. Переход по состоянию разряда	B0	BBS	pc.r1, base.vb, displ.bb	PC ← PC + SEXT(displ), если разряд = 1 если разряд = 0	Не изменяются	Проверяется разряд, определяемый операндами позиции и базового адреса. Если разряд соответствует указанному в команде, происходит переход по адресу PC + SEXT(displ)
	B1	BBC				
38. Переход по состоянию разряда и его модификации (без блокировки памяти)	B2	BBSZ	pc.r1, base.vb, displ.bb	1) PC ← PC + SEXT(displ) разряд = 1 или 0 в соответствии с командой; 2) разряд ← 1 или 0 в соответствии с командой	Не изменяются	Переход по установленному разряду и его установка
	B3	BBCZ				Переход по сброшенному разряду и его установка
	B4	BBSC				Переход по установленному разряду и его сброс
	B5	BBCS				Переход по сброшенному разряду и его сброс
	B6	BBSZI				Команды аналогичны командам BBSZ и BBCS, только проверка и установка разряда производится с блокировкой памяти
B7	BBCZI	То же, что и в п.38	Не изменяются			
39. Переход по состоянию разряда (с блокировкой памяти)	B6	BBSZI		pc.r1, base.vb, displ.bb	То же, что и в п.38	Не изменяются
	B7	BBCZI				
40. Переход по состоянию младшего разряда	B8	BBSI	src.r1, displ.bb	1) проверка src [0] = 0 или 1; 2) PC ← PC + SEXT(displ)	Не изменяются	Проверяется состояние младшего разряда сп.-источника. Производится переход, если разряд установлен
	B8	BBSI				

Наименование	Код команды		Спецификаторы	Алгоритмы	Коды условий	Описание
	16-ричный	мнемонический				
41. Безусловный переход	B9	BLPC				Производится переход, если разряд сброшен
	11	BRE	displ.bx	$PC \leftarrow PC + \text{SEXT}(\text{displ})$	не изменяются	К счетчику команд прибавляется смещение перехода
	31	BW				
42. Переход к подпрограмме	10	BZBB	displ.bx	1) $\text{ZF} \leftarrow PC$ 2) $PC \leftarrow PC + \text{SEXT}(\text{displ})$	не изменяются	Содержимое счетчика команд заносится в стек и осуществляется переход
	30	BZBW				
43. Выбор	8F	CASEB	selector.rx, base.rx, limit.rx, displ[0].bw, ..., displ[limit].bw	1) $\text{tmp} \leftarrow \text{selector} - \text{base}$; 2) $\text{tmp} = \text{limit}$; 3) $PC \leftarrow PC + \text{SEXT}(\text{displ}[\text{tmp}])$, если $\text{tmp} \leq \text{limit}$; $PC \leftarrow PC + 2(1 + \text{SEXT}(\text{limit}))$, если $\text{tmp} > \text{limit}$	$N \leftarrow \text{tmp} < \text{limit}$, $Z \leftarrow \text{tmp} = \text{limit}$, $V \leftarrow 0$, $C \leftarrow \text{tmp} < / \text{limit}$	Содержимое оп.-базы вычитается из содержимого оп.-селектора. Если этот промежуточный результат меньше или равен содержимому оп.-граничн., то к PC прибавляется смещение, выбираемое по значению tmp. В противном случае производится переход на массив, определяемый оп.-границей
	AF	CASEW				
	CF	CASEL				
44. Универсальный переход	17	JLP	dst.ab	$PC \leftarrow \text{dst}$	не изменяются	В счетчик команд заносится содержимое оп.-приемника
45. Универсальный переход к подпрограмме	16	JSB	dst.ob	1) $\text{ZF} \leftarrow PC$; 2) $PC \leftarrow \text{dst}$	не изменяются	Содержимое счетчика команд заносится в стек, а затем в счетчик команд - содержимое оп.-приемника

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
46. возврат из подпрограммы	05	RSB		$PC \leftarrow (SP) +$	Не изменяются	В счетчик команд заносится длинное слово из стека
47. вычесть единицу и переход, если БОЛЬШЕ или РАВНО	F4	SOBGRQ	index.ml, displ.bb	1) $index \leftarrow index - 1$; 2) $PC \leftarrow PC + SEXT(displ)$, если $index \geq 0$	$N \leftarrow index < 0$, $Z \leftarrow index = 0$, $V \leftarrow \text{переполн.}$, $C \leftarrow C$	Из содержимого оп.-индекса вычитается единица. Если результат больше или равен нулю, то происходит переход по адресу $PC + SEXT(displ)$
48. вычесть единицу и переход по БОЛЬШЕ	F5	SOBGR	index.ml, displ.bb	1) $index \leftarrow index - 1$; 2) $PC \leftarrow PC + SEXT(displ)$, если $index > 0$	То же, что и в п. 47	После вычитания единицы из содержимого оп.-индекса выполняется переход, если результат вычитания больше нуля

У. КОМАНДЫ ВЫЗОВА ПРОЦЕДУРЫ

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
49. Вызов процедур общего вида	FA	CALLG	arglist.ab, dat.ab		Обнуляются	Производится вызов процедуры со списком аргументов, расположенным в произвольном месте памяти. Оп.-списка аргументов определяет адрес списка аргументов, а оп.-приемника - адрес процедуры. Формируется кадр вызова в стеке для сохранения содержимых слова состояния и необходимых регистров
50. Вызов процедур стекового вида	FB	CALLS	numarg. rl, dat.ab		Обнуляются	Производится вызов процедуры со списком аргументов, расположенным в стеке. Содержимое оп.-числа аргументов содержит числа аргументов в списке, а оп.-приемника определяет адрес процедуры. Формируется кадр вызова
51. Возврат из процедуры	04	RST			Восстанавливаются предшествовавшие вызову процедуры	Восстанавливаются состояние процессора и содержимое регистров, существовавшие до вызова процедуры

У1. КОМАНДЫ РАЗЛИЧНОГО НАЗНАЧЕНИЯ

Наименование	Код команды		Операнды	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
02. Сброс разрядов в слове состояния процессора	B9	BICPSW	mask.rw	$PSW \leftarrow PSW \wedge \overline{mask}$	$N \leftarrow N \wedge \overline{mask}[3],$ $Z \leftarrow Z \wedge \overline{mask}[2],$ $V \leftarrow V \wedge \overline{mask}[1],$ $C \leftarrow C \wedge \overline{mask}[0]$	Содержимое слова состояния процессора логически умножается на инвертированное значение оп.-маски
03. Установка разрядов в слове состояния процессора	B8	BISPSW	mask.rw	$PSW \leftarrow PSW \vee mask$	$N \leftarrow N \vee mask[3],$ $Z \leftarrow Z \vee mask[2],$ $V \leftarrow V \vee mask[1],$ $C \leftarrow C \vee mask[0]$	Содержимое слова состояния процессора логически складывается с содержимым оп.-маски
04. Ложушка отладчика	03	BFT		1) $PSL[TP] \leftarrow 0;$ 2) прерывание программы	Обнуляются	Происходит прерывание трассирования, если в начале команды установлен $PSL[T]=1$
00. Останов	00	HALT			При останове процессора коды условий сохраняются. При прерывании программы они сбрасываются	Производится останов процессора. Если $PSL[CUR MOD] \neq 0$ (т.е. установлен режим, отличный от ядра), то произойдет прерывание программы по ошибке привилегированной команды
06. Вычисление индекса	0A	INDEX	substript.rl, low.rl, high.rl, size.rl, indexin.rl, indexout.wl	$indexout \leftarrow (indexin + substript) * size$	$N \leftarrow indexout < 0,$ $Z \leftarrow indexout = 0,$ $V \leftarrow 0,$ $C \leftarrow 0$	Содержимое оп.-входного индекса складывается с содержимым оп.-индексного значения. Полученная сумма умножается на содержимое оп.-размера и заносится по адресу оп.-выходного индекса

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
57. Пересылка из PSL	DC	MOVPSL	dat wl	dat ← PSL	не изменяются	Содержимое длинного слова состояния процессора пересылается по адресу оп.-приемника
58. Отсутствие операции	CI	NOP			не изменяются	не производится никаких операций
59. Загрузка общих регистров в стек	EB	PUSHR	mask.rw		не изменяются	Содержимые регистров, номера которых соответствуют единицам оп.-маски, заносятся в стек
60. Извлечение общих регистров из стека	BA	POPR	mask.rw		не изменяются	Длинные слова из стека заносятся в регистры, номера которых соответствуют единицам оп.-маски
61. Вызов специальной функции	FC	XFC			не изменяются	Команда производит прерывание программы. Она может вызвать функцию, заданную пользователем

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
62. Занесение элемента в начало очереди	5C	INSQHI	entry.ab, header.aq	Вслед за header $\leftarrow$ entry	Если занесен лишь первый элемент очереди, то $K \leftarrow 0$, $Z \leftarrow (\text{entry}) =$ $= (\text{entry} + 4)$, $V \leftarrow 0$, $C \leftarrow 0$. В противном случае $K \leftarrow 0$, $Z \leftarrow 0$, $V \leftarrow 0$, $C \leftarrow 1$	Элемент, определяемый оп.-элементом, заносится в очередь вслед за заголовком
63. Занесение элемента в конец очереди	5D	INSQTI	entry.ab, header.aq	В конец очереди $\leftarrow$ entry	То же, что и в п. 62	Элемент, определяемый оп.-элементом, заносится в конец очереди
64. Занесение элемента в очередь	0E	INSQUL	entry.ab, pred.ab	1) (entry) $\leftarrow$ (pred); 2) (entry + 4) $\leftarrow$ (pred); 3) (pred + 4) $\leftarrow$ entry; 4) (pred) $\leftarrow$ entry	Если заносится лишь первый элемент очереди, то $K = (\text{entry}) < (\text{entry} + 4)$, $Z = (\text{entry}) = (\text{entry} + 4)$, $V \leftarrow 0$, $C = \text{entry} < \text{entry} + 4 $. В противном случае $C \leftarrow 0$	Элемент, определяемый оп.-элементом, заносится в очередь вслед за элементом, определяемым оп.-предыдущего элемента

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условия	Описание
	16-ричный	мнемонический				
65. Удаление элемента из начала очереди	5E	REMQHI	header.aq, addr.wl	1) $tmp \leftarrow (header);$ 2) запись оп.-адреса; 3) чтение содержимого по адресу $header + tmp;$ 4) запись по адресу $header + tmp +$ $+ (header + tmp);$ 5) удаление элемента из очереди	$N \leftarrow 0,$ $Z \leftarrow 0,$ $V \leftarrow 1,$ $C \leftarrow 1.$ Если очередь была пуста, тогда $N \leftarrow 0,$ $Z \leftarrow (header) = 0,$ $V \leftarrow tmp = 0,$ $C \leftarrow 0$	Элемент очереди, следующий за заголовком, удаляется из очереди. Содержимое оп.-адреса заменяется адресом удаленного элемента
66. Удаление элемента из конца очереди	5F	REMQTI	header.aq, addr.wl	1) $tmp \leftarrow (header);$ 2) запись оп.-адреса; 3) чтение содержимого по адресу $header + (header +$ $+ 4);$ 4) запись по адресу $header + (header +$ $+ 4) + (header + 4 +$ $+ (header + 4));$ 5) удаление элемента из очереди	То же, что и в п. 65	Элемент очереди, предшествующий заголовку, удаляется из очереди. Содержимое оп.-адреса заменяется адресом удаленного элемента
67. Удаление элемента из очереди	0F	REMQUE	entry.ab, addr.wl	1) $(entry + 4) \leftarrow$ $\leftarrow (entry);$ 2) $((entry) + 4) \leftarrow$ $\leftarrow (entry + 4);$ 3) $addr \leftarrow entry$	$N \leftarrow (entry) <$ $< (entry + 4),$ $Z \leftarrow (entry) =$ $= (entry + 4);$ если очередь пуста, $V \leftarrow entry = (entry + 4);$ если нет удаляемого элемента, $C \leftarrow ((entry) <$ $< (entry + 4))$	Элемент, определяемый оп.-элементом, удаляется из очереди. Содержимое оп.-адреса заменяется адресом удаленного элемента

VIII. КОМАНДЫ ПЛАВАЮЩЕЙ ЗАПЯТОЙ

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
66. Пересылка	50	MOVF	src.rx, dst.wx	$dst \leftarrow src$	$N \leftarrow dst < 0,$ $Z \leftarrow dst = 0,$ $V \leftarrow 0,$ $C \leftarrow C$	Содержимое оп.-источника пересылается по адресу оп.-приемника
	70	MOVD				
	50FD	MOVG				
	70FD	MOVH				
67. Пересылка с отрицанием	52	MNEGf	src.rx, dst.wx	$dst \leftarrow -src$	$N \leftarrow dst < 0,$ $Z \leftarrow dst = 0,$ $V \leftarrow 0,$ $C \leftarrow 0$	Содержимое оп.-источника пересылается по адресу оп.-приемника с противоположным знаком
	72	MNEGD				
	52FD	MNEGg				
	72FD	MNEGH				
70. Очистка	D4	CLRF	dst.wx	$dst \leftarrow 0$	$N \leftarrow C,$ $Z \leftarrow 1,$ $V \leftarrow 0,$ $C \leftarrow C$	По адресу оп.-приемника заносится нуль
	7C	CLRD				
		CLRG				
	7CFD	CLRH				
71. Сравнение	51	CMFF	src1.rx, src2.rx	$src1 - src2$	$N \leftarrow src1 < src2,$ $Z \leftarrow src1 = src2,$ $V \leftarrow 0,$ $C \leftarrow 0$	Операнды источников сравниваются и остаются неизменными
	71	CMFD				
	51FD	CMFG				
	71FD	CMFH				
72. Преобразование	4C	CVTBF	src.rx, dst.wy	$dst \leftarrow \text{преобразованный } src$	$N \leftarrow dst < 0,$ $Z \leftarrow dst = 0,$ $V \leftarrow \text{переполн.},$ $C \leftarrow 0$	Производится преобразование формата оп.-источника в формат оп.-приемника
	6C	CVTBD				
	4CFD	CVTBG				
	6CFD	CVTBH				
	4D	CVTWF				

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
	6D	CVTWD				
	4DFD	CVTWG				
	6DFD	CVTWH				
	43	CVTLF				
	6E	CVTLD				
	4EFD	CVTLQ				
	6EFD	CVTLN				
	48	CVTFB				
	68	CVTDB				
	48FD	CVTCB				
	68FD	CVTCB				
	49	CVTFW				
	69	CVTDW				
	49FD	CVTGW				
	69FD	CVTHW				
	4A	CVTFL				
	6A	CVTDL				
	4AFD	CVTGL				
	6AFD	CVTHL				
	4B	CVTRFL				
	6B	CVTRDL				
	4BFD	CVTRGL				
	6BFD	CVTRHL				
	56	CVTFD				
	99FD	CVTFH				
	76	CVTDF				
	33FD	CVTCF				
	56FD	CVTCH				
	F6FD	CVTHF				
	F7FD	CVTHD				
	76FD	CVTHG				

Преобразование из округленно-го формата с плавающей запятой в длинное слово

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условия	Описание
	16-ричный	мнемонический				
73. Сложение	40	ADDF2	add.rx, sum.mx	$sum \leftarrow sum + add$	$N \leftarrow sum < 0,$	Производится сложение двух операндов
	41	ADDF3	add1.rx, add2.rx, sum.wx	$sum \leftarrow add1 + add2$	$Z \leftarrow sum = 0,$	
	60	ADD2			$V \leftarrow \text{переполн.},$	
	61	ADD3			$C \leftarrow C$	
	40FD	ADDG2				
	41FD	ADDG3				
	60FD	ADDH2				
	61FD	ADDH3				
74. Вычитание	42	SUBF2	sub.rx, dif.mx	$dif \leftarrow dif - sub$	$N \leftarrow dif < 0,$	Производится вычитание содержимого оп.-вычитаемого из содержимого оп.-разности (2-операндный формат) или из содержимого оп.-уменьшаемого (3-операндный формат)
	43	SUBF3	sub.rx, min.rx, dif.wx	$dif \leftarrow min - sub$	$Z \leftarrow dif = 0,$	
	62	SUB2			$V \leftarrow \text{переполн.},$	
	63	SUB3			$C \leftarrow C$	
	42FD	SUBG2				
	43FD	SUBG3				
	62FD	SUBH2				
	63FD	SUBH3				
75. Деление	46	DIVF2	divr.rx, quo.mx	$quo \leftarrow quo / divr$	$N \leftarrow quo < 0,$	Содержимое оп.-частного (в 2-операндном формате содержится делимого) или оп.-делимого (3-операндный формат) делится на содержимое оп.-делителя. Результат запоминается по адресу оп.-частного
	47	DIVF3	divr.rx, divd.rx, quo.wx	$quo \leftarrow divd / divr$	$Z \leftarrow quo = 0,$	
	66	DIV2			$V \leftarrow \text{переполн. или деление на нуль},$	
	67	DIV3			$C \leftarrow C$	
	46FD	DIVG2				
	47FD	DIVG3				
	66FD	DIVH2				
	67FD	DIVH3				

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
76. Умножение	44	MULF2	mulr.rx, prod.mx	$prod \leftarrow prod * mulr$ $prod \leftarrow muld * mulr$	$N \leftarrow prod < 0,$ $Z \leftarrow prod = 0,$ $V \leftarrow \text{переполн.},$ $C \leftarrow 0$	Содержимое оп.-произведения (в 2-операндном формате содержится максимум) или оп.-множимого (3-операндный формат) умножается на содержимое оп.-множителя. Округленный результат запоминается по адресу оп.-произведения
	45	MULF3	mulr.rx, muld.rx, prod.wx			
	64	MULD2				
	65	MULD3				
	44FD	MULC2				
	45FD	MULC3				
	64FD	MULH2				
77. Расширенное умножение с выделением целой части	54	EMODF	mulr.rx, mulrx.rb, muld.rx, int.wl, fract.wx	1) $int \leftarrow \text{целая часть } muld * (mulr * mulrx);$ 2) $fract \leftarrow \text{дробная часть } muld * (mulr * mulrx)$	$N \leftarrow fract < 0,$ $Z \leftarrow fract = 0,$ $V \leftarrow \text{переполн.},$ $C \leftarrow 0$	Содержимое оп.-множимого умножается на содержимое оп.-множителя, расширенное присоединением 6(EMODF, EMODD), 11(EMODI) или 15(EMODH) яд. разрядов в дробной части от оп.-расширения. Целая и дробная части результата запоминаются в соответствующих операндах
	74	EMODD				
	54FD	EMODG				
	74FD	EMODH				
78. Проверка	53	TSTF	src.rx	$src = 0$	$N \leftarrow src < 0,$ $Z \leftarrow src = 0,$ $V \leftarrow 0,$ $C \leftarrow 0$	Устанавливаются коды условий в соответствии с содержанием источника
	73	TSTD				
	53FD	TSTG				
	73FD	TSTH				

Наименование	Код команды		Спецификаторы	Алгоритм	Код условия	Описание
	16-ричная	мнемонический				
79. Вычисление полинома	55	POLXF	arg.rx, degree.rv, tbladdr.ab		$X \leftarrow E_0 < 0,$ $Z \leftarrow R_0 = 0,$ $Y \leftarrow \text{переполн.},$ $C \leftarrow 0$	Осуществляется вычисление полинома методом Хорнера. Оп.-таблица указывает адрес коэффициентов полинома. Тип данных коэффициентов определяет оп.-аргумента. Оп.-степени определяет номер ст. коэффициента, участвующего в подсчете. Результат помещается в R_0 (для POLXF) или R_1, R_0 (для POLXD и POLXG) или R_1, R_2, R_0 (для POLXE)
	79	POLXD				
	55FD	POLXG				
	75FD	POLXE				

IX. КОМАНДЫ ОБРАБОТКИ СИМВОЛИЧЕСКИХ СТРОК

Код 2

Длина строки 04-FFFF

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
60. Сравнение двух строк	29	CMPC3	len.rw, src1addr.ab, src2addr.ab	src1 - src2	$N \leftarrow src1 < src2,$ $Z \leftarrow src1 = src2,$ $V \leftarrow 0,$ $C \leftarrow src1 < src2 $	Производится сравнение двух строк побайтно, начиная с мал. адреса. Операция завершается при обнаружении несовпадения или истощения длины. В пяти-операционном формате короткая строка дополняется содержимым оп.-выполнителя.
	2D	CMPC5	src1len.rw, src1addr.ab, fill.rb, src2len.rw, src2addr.ab			оп.-операнд.
61. Локализация	3A	LOCC	char.rb, len.rw, addr.ab		$N \leftarrow 0,$ $Z \leftarrow 1,$ не обнаружен совпадающий байт, $V \leftarrow 0,$ $C \leftarrow 0$	Производится поиск байта в строке, равного оп.-символу. При обнаружении такого байта операция завершается.
62. Пропуск символа	3B	SKPC	char.rb, len.rw, addr.ab		$N \leftarrow 0,$ $Z \leftarrow 1,$ не обнаружен несовпадающий байт, $V \leftarrow 0,$ $C \leftarrow 0$	Производится поиск байта в строке, не равного оп.-символу. При обнаружении такого байта операция завершается.
63. Поиск подстроки символов	39	MATCHC	objlen.rw, objaddr.ab, src1len.rw, src1addr.ab		$N \leftarrow 0,$ $Z \leftarrow 1,$ подстрока найдена, $V \leftarrow 0,$ $C \leftarrow 0$	Производится поиск подстроки в строке, равной объектной строке. При обнаружении такой подстроки или, если после просмотра всей строки искомой подстроки не обнаружено, операция завершается.

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
04. Пересылка символов	20	MOV13	len.rw, srcaddr.ab, dstaddr.ab		Для MOV13 $N \leftarrow 0$, $Z \leftarrow 1$, $V \leftarrow 0$, $C \leftarrow 0$.	Производится пересылка строки-источника по адресу строки-приемника. В пятиоперандном формате, если строка приемника длиннее строки-источника, мл. байты приемника заполняются заданным оп.-заполнителем
	2C	MOV15	srcLen.rw, srcaddr.ab, fill.rb, dstLen.rw, dstaddr.ab		Для MOV15 $N \leftarrow srcLen < dstLen$, $Z \leftarrow srcLen = dstLen$, $V \leftarrow 0$, $C \leftarrow srcLen < dstLen $	
05. Пересылка символов с трансляцией	2A	MOVTC	srcLen.rw, srcaddr.ab, fill.rb, tbladdr.ab, dstLen.rw, dstaddr.ab		$N \leftarrow srcLen < dstLen$, $Z \leftarrow srcLen = dstLen$, $V \leftarrow 0$, $C \leftarrow srcLen < dstLen $	Производится трансляция содержимого оп.-источника согласно таблице. Адрес байта таблицы определяется суммой адреса таблицы и содержимого байта источника. Определенный таким образом байт таблицы размещается последовательно по адресу оп.-приемника. Если строка приемника длиннее строки-источника, мл. байты приемника заполняются символом-заполнителем
06. Пересылка с трансляцией до выходного символа	2F	MOVTCU	srcLen.rw, srcaddr.ab, enc.rb, tbladdr.ab, dstLen.rw, dstaddr.ab		$N \leftarrow srcLen < dstLen$, $Z \leftarrow srcLen = dstLen$, $V \leftarrow$ завершение операции по выходному символу, $C \leftarrow srcLen < dstLen $	Операция аналогична MOVTC, только завершение ее происходит в следующих случаях: 1) если обнаружен байт таблицы, равный выходному символу; 2) если не обнаружен такой байт таблицы, но исчерпана строка источника либо приемника

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
57. Сканирование	2A	SCANS	len.rw, addr.ab, tbl addr.ab, mask.rb	1) $tmp \leftarrow (tbladdr +$ $+ ZEXT(addr));$ 2) $tmp \leftarrow tmp \wedge mask$	$N \leftarrow 0,$ $Z \leftarrow 1,$ не обнаружен байт, вызвавший не- нулевой результат умножения, $V \leftarrow 0,$ $C \leftarrow 0$	Выбранный из таблицы байт, аналогично команде SCOUTS, логически умножается на со- держимое оп.-маски. Операция завершается, если получен ненулевой результат умноже- ния ($tmp \neq 0$) или если все байты строки исследованы
58. Поиск совпадения	2B	SPANC	len.rw, addr.ab, tbladdr.ab, mask.rb	То же, что и в п. 57	$N \leftarrow 0,$ $Z \leftarrow 1,$ не обнаружен байт, вызвавший ну- левой результат ум- ножения, $V \leftarrow 0,$ $C \leftarrow 0$	Команда аналогична SCANS, только операция завершается, если получен нулевой резуль- тат умножения ($tmp = 0$) или если все байты строки иссле- дованы

X. КОМАНДЫ ФОРМАТОА АРИТЕТИКИ

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
09. Пересылка в упакованном формате	34	MOVF	len.rw, srcaddr.ab, dstaddr.ab	$dst \leftarrow src$	$N \leftarrow dst < 0,$ $Z \leftarrow dst = 0,$ $V \leftarrow 0,$ $C \leftarrow 0$	Строка-источник пересылается по адресу строки-приемника
10. Сравнение в упакованном формате	35	CMPP3	len.rw, src1addr.ab, src2addr.ab	$src1 - src2$	$N \leftarrow src1 < src2,$ $Z \leftarrow src1 = src2,$ $V \leftarrow 0,$ $C \leftarrow 0$	Арифметически сравниваются строки источников, результатом операции являются коды условий
	37	CMPP4	src1len.rw, src1addr.ab, src2len.rw, src2addr.ab			
11. Сложение в упакованном формате	20	ADDP4	addlen.rw, add1addr.ab, sumlen.rw, sumaddr.ab	$sum \leftarrow sum + add$	$N \leftarrow sum < 0,$ $Z \leftarrow sum = 0,$ $V \leftarrow \text{переполн.},$ $C \leftarrow 0$	В 4-операндном формате строка-слагаемое складывается со строкой-суммой, в которой размещается результат. В 6-операндном формате складываются строки слагаемых, и результат размещается по адресу строки-суммы
	21	ADDP6	add1len.rw, add1addr.ab, add2len.rw, add2addr.ab, sumlen.rw, sumaddr.ab	$sum \leftarrow add1 + add2$		
12. Вычитание в упакованном формате	22	SUBP4	sublen.rw, subaddr.ab, diflen.rw, difaddr.ab	$dif \leftarrow dif - sub$	$N \leftarrow dif < 0,$ $Z \leftarrow dif = 0,$ $V \leftarrow \text{переполн.},$ $C \leftarrow 0$	В 4-операндном формате строка вычитаемого вычитается из строки разности, в которой размещается результат. В 6-операндном формате строка вычитаемого вычитается из строки уменьшаемого. Разность
	23	SUBP6	sublen.rw, subaddr.ab, minlen.rw, minaddr.ab, diflen.rw, difaddr.ab	$dif \leftarrow min - sub$		

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	Ю-рич-ный	мнемонический				
93. Умножение в упакованном формате	25	MULP	mulrlen.rw, mulraddr.ab, mulrlen.rw, muldaddr.ab, prodlen.rw, prodaddr.ab	$prod \leftarrow mald * mair$	$N \leftarrow prod < 0,$ $Z \leftarrow prod = 0,$ $V \leftarrow \text{переполн.},$ $C \leftarrow 0$	помещается по адресу строки разности Строка множимого умножается на строку множителя. Произведение помещается по адресу строки-произведения
94. Деление в упакованном формате	27	DIVP	divrlen.rw, divraddr.ab, divrlen.rw, divdaddr.ab, quotlen.rw, quodaddr.ab	$quo \leftarrow divd/divr$	$N \leftarrow quo < 0,$ $Z \leftarrow quo = 0,$ $V \leftarrow \text{переполн.},$ $C \leftarrow 0$	Строка-делимое делится на строку-делитель. Частное помещается по адресу строки-частного
95. Арифметический сдвиг с округлением в упакованном формате	F8	ASHP	cnt.rb, srclen.rw, srcaddr.ab, round.rb, dstlen.rw, dstaddr.ab	$dst \leftarrow (src + \text{round}[3:0] * 10^{(-cnt-1)}) * 10^{cnt}$	$N \leftarrow dst < 0,$ $Z \leftarrow dst = 0,$ $V \leftarrow \text{переполн.},$ $C \leftarrow 0$	Строка-источник сдвигается на количество десятичных разрядов вправо или влево в зависимости от величины и знака оп.-счетчика. При сдвиге вправо ($cnt < 0$) производится округление ст.значадей не отбрасываемых цифр с распространением переноса. Результат помещается по адресу строки-приемника

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	Ю-рич-ный	мнемонический				
56. Преобразование длинного слова в упакованный формат	79	CVTLP	src.r1, dstlen.rw, dstaddr.ab	$dst_{10} \leftarrow$ преобразованный src_2	$K \leftarrow dst < 0,$ $Z \leftarrow dst = 0,$ $V \leftarrow$ переполн., $C \leftarrow 0$	Строка-источник, содержащая длинное слово, преобразуется в упакованную десятичную строку, которая размещается по адресу строки-приемника
57. Преобразование упакованной строки в длинное слово	36	CVTFL	srclen.rw, srcaddr.ab, dst.w1	$dst_{10} \leftarrow$ преобразованный src_{10}	$K \leftarrow dst < 0,$ $Z \leftarrow dst = 0,$ $V \leftarrow$ переполн., $C \leftarrow 0$	Источник, представляющий собой упакованную десятичную строку, преобразуется в длинное слово и размещается по адресу оп.-приемника
58. Преобразование упакованной строки в числовую с выделенным ведущим знаком	08	CVTPS	srclen.rw, srcaddr.ab, dstlen.rw, dstaddr.ab	dst (числовая строка с выделенным ведущим знаком) $\leftarrow$ преобразованный src_{10}	$K \leftarrow src < 0,$ $Z \leftarrow src = 0,$ $V \leftarrow$ переполн., $C \leftarrow 0$	Десятичная строка упакованного формата преобразуется в числовую с выделенным ведущим знаком и помещается по адресу строки-приемника. В первом байте (имеющем наименьший адрес) помещается знак
59. Преобразование упакованной строки в числовую строку зонного (или перфо) формата	24	CVTPT	srclen.rw, srcaddr.ab, tbladdr.ab, dstlen.rw, dstaddr.ab	dst (зонный формат) $\leftarrow$ преобразованный src_{10}	$K \leftarrow src < 0,$ $Z \leftarrow src = 0,$ $V \leftarrow$ переполн., $C \leftarrow 0$	Десятичная строка упакованного формата преобразуется в числовую строку с любым кодированием цифр (согласно таблице) и размещением знака в мл. разряде (ст. адресуемому байте). Адрес байта таб-

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
100. Преобразование числовой строки зонного (или перфо) формата в упакованную	26	CVTTP	srcLen.rw, srcAddr.ab, tblAddr.ab, dstLen.rw, dstAddr.ab	dst ₁₀ ← преобразованный зонный формат	N ← dst < 0, Z ← dst = 0, V ← пересыл., C ← 0	лишь определяется суммированием адреса таблицы со значением послубайта десятичной строки Десятичная строка зонного или перфоформата преобразуется в упакованную строку и помещается по адресу строки-приемника

trailing string
нахвостовый строка

Перевод из упакованного в любой формат и из любого формата в упакованный

XI. ПРОЧИЕ КОМАНДЫ

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
101. Редактирование упакованной строки в символьную	38	EDITPS	srcLen.rw, srcaddr.ab, pattern.ab, dstaddr.ab		$X \leftarrow src < 0,$ $Z \leftarrow src = 0,$ $V \leftarrow \text{переполн.},$ $C \leftarrow \text{значимость (наличие значащих цифр в исходной строке)}$	Исходная десятичная строка упакованного формата преобразуется в символьную строку, пригодную для вывода. Преобразование выполняется согласно строке-маске, которая обеспечивает: - эспозицию ведущих нулей; - запрет ведущих нулей; - вставку и распространение необходимых знаков; - преобразование упакованных десятичных цифр в символьные в коде Юни-0
102. Циклический контроль	08	CRC	tbl.ab, incre.rl, strlen.rw, stream.ab		$N \leftarrow RC < 0,$ $Z \leftarrow RC = 0,$ $V \leftarrow 0,$ $C \leftarrow 0$	Вычисляется СКО для потока данных, определяемых операндами длины и адреса. Начальное СКО задается операндом incre и обычно равно 0 или -1, если только СКО не вычисляется в несколько этапов. Результат отбрасывается в 40. Если полином меньше, чем обычно (-32), то результат должен быть выведен из полученного числа

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условия	Описание
	Ю-рич-ный	мнемонический				
103. Запрос на сообщение об ошибках	YERR RDEF	BUGW BUGL			Не изменяется	Происходит прерывание программы и выдается сообщение об обнаруженных программных ошибках в виде слова или длинного слова в соответствии с кодом команды
104. Проверка доступности	OC OD	PROBER PROBWR	mode.rb, len.rw, base.ab		N ← 0, Z ← 0, если оба байта доступны, V ← 0, C ← 0	Команда проверяет доступность чтения или записи первого и последнего байтов области, определяемой операндами базового адреса и длины. Защита проверяется согласно большему коду режима, указанному в разрядах [1:0] оп.-режима и поди предыдущего режима P31.
105. Смена режима	BC BD BE BF	CHDK CHME CHMS CHMU	code.rw		Обнуляются	Производится прерывание программы, изменение режима работы процессора в соответствии с кодом команды, запоминание содержимого оп.-кода, PC и P31 в стеке, извлекается вектор данного прерывания

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мнемонический				
106. Возврат из нарушения или прерывания	02	REI			Заполняются из сохраненного PSL	Из стека переписываются длинные слова в PC и PSL. Восстанавливается содержимое других регистров. Процессор приступает к дальнейшему выполнению прерванной программы
107. Сохранение контекста процессора	07	SVSTX			не изменяются	Содержимое регистров KSP, LSP, SSP, USI, PC+PIL, AF и PC переписывается в блок управления процессора PCB. Значения PC и PSL переписываются в PCB из стека
108. Загрузка контекста процессора	06	LOPSTX			не изменяются	Контекст блока управления процессора PCB переписывается в соответствующие регистры процессора. Содержимые PC и PSL переписываются в стек

Наименование	Код команды		Спецификаторы	Алгоритм	Коды условий	Описание
	16-ричный	мизмони- ческий				
I09. Пересылка в процессорный регистр	DA	MTPR	src.r1, procreg.r1	$(procreg) \leftarrow src$	$N \leftarrow src < 0,$ $Z \leftarrow src = 0,$ $V \leftarrow 0$ (кроме регистра ТВСНК), $C \leftarrow C$	Содержимое оп.-источника пересылается в процессорный регистр, номер которого указан в оп.-процессорного регистра
I10. Пересылка из процессорного регистра	DB	MTPR	procreg.r1, dst.al	$dst \leftarrow (procreg)$	$N \leftarrow dst[0] = 0,$ $Z \leftarrow dst = 0,$ $V \leftarrow 0,$ $C \leftarrow C$	Содержимое процессорного регистра, номер которого указан в соответствующем операнде, пересылается по адресу оп.-приемника

ЛИТЕРАТУРА

Бальберс П. В., Беляцкий Б. Б. Вычислительный комплекс СМ 1700: Учебное пособие. - Вильнюс: МПСО ЛитССР, 1987.
Операционная система МОС ВП. Архитектура и система машинных инструкций. Справочный материал. ОО152-О1 97 О1-1.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	3
I. ЦЕЛОЧИСЛЕННЫЕ АРИФМЕТИЧЕСКИЕ И ЛОГИЧЕСКИЕ КОМАНДЫ.....	5
II. АДРЕСНЫЕ КОМАНДЫ.....	11
III. КОМАНДЫ ДЛЯ ПОЛЕЙ С ПЕРЕМЕННЫМ КОЛИЧЕСТВОМ РАЗРЯДОВ.....	12
IV. КОМАНДЫ УПРАВЛЕНИЯ.....	13
V. КОМАНДЫ ВЫЗОВА ПРОЦЕДУРЫ.....	18
VI. КОМАНДЫ РАЗЛИЧНОГО НАЗНАЧЕНИЯ.....	19
VII. КОМАНДЫ ДЛЯ РАБОТЫ С ОЧЕРЕДЬМИ.....	21
VIII. КОМАНДЫ ПЛАВАЮЩЕЙ ЗАПЯТОЙ.....	23
IX. КОМАНДЫ ОБРАБОТКИ СИМВОЛЬНЫХ СТРОК.....	28
X. КОМАНДЫ ДЕСЯТИЧНОЙ АРИФМЕТИКИ.....	31
XI. ПРОЧИЕ КОМАНДЫ.....	35
ЛИТЕРАТУРА.....	39

Кафедра вычислительной техники

СИСТЕМА КОМАНД ВЫЧИСЛИТЕЛЬНОГО КОМПЛЕКСА СИ 1700

Материалы для практических занятий

Подготовил канд. техн. наук Б а л б е р г е Пётр Владиславович

Вильнюс

Редактор О. Петраускайте

Подписано в печать 0.04.69. 60x84/16. Бумага писчая. Печ.л. 2,5, усл.печ.л. 2,32, учётн.-изд.л. 1,82, авт.л. 1,78. Тираж 1000 экз. Бесплатно. Заказ № 176.

Отпечатано на ротационной машине ЛитССР

232600 Вильнюс, Коммунистический район, 4

Scanned for lib.gva.org.ua

Special thanks to Oleksandr Rybak
for providing the material!

(c) 2025 gva.org.ua