

ЛИТЕРАТУРА

1. Смит. Достоверный контроль качества микропроцессоров путем функционального тестирования.— «Электроника», 1977, № 4, с. 56—62.
2. Чанг, Маккаски. Два новых способа упрощения тестовой проверки микропроцессоров.— «Электроника», 1976, № 2, с. 45—52.
3. Жорно С. Е., Маковоз Б. И., Плутонов В. К. Структура и принцип работы устройства для функционально-временного контроля микропроцессоров.— В кн.: Микропроцессоры. (Тезисы докладов 2-го Всесоюзного совещания). Т. I. Рига, «Зинатне», 1977, с. 178—180.
4. Зеленевская Е. А., Сабадаш Н. Г., Обуховский И. П., Темченко В. А., Плотников Е. И. Специфика организации контроля БИС микропроцессорных наборов в процессе производства и эксплуатации.— В кн.: Микропроцессоры. Т. I. Рига, «Зинатне», 1977, с. 175—177.
5. Сантони. Совершенствование тестеров для наладки микропроцессорных систем.— «Электроника», 1976, № 26, с. 26—39.
6. Мик, Скопмейер. Микропроцессор для отладки микропрограмм машин на основе процессорных секций.— «Электроника», 1977, № 9, с. 51—56.
7. Черепанов В. Г. Программирование в кодах ЭВМ «Электроника С-50». Красноярск, изд. КПИ, 1976.
8. Борде Б. И., Гурьевич А. С., Красников В. И., Романов В. Г., Черепанов В. Г. Автоматическая обработка результатов исследования распределения термо-ЭДС в полупроводниковых минералах.— «Автометрия», 1977, № 4, с. 25—30.
9. Борде Б. И., Иванов В. И. Структуры программируемых многофункциональных аналого-цифровых преобразователей и коммутаторов.— В кн.: Тезисы докладов Всесоюзной конференции по аналоговой и аналого-цифровой вычислительной технике. М., «Сов. радио», 1977, с. 35.
10. Аваев Н. А., Душний В. Н., Наумов Ю. Е. Большие интегральные схемы с инжекционным питанием. М., «Сов. радио», 1977.

*Поступила в редакцию 7 февраля 1978 г.;
окончательный вариант — 18 апреля 1978 г.*

УДК 681.142.4

В. И. ВИНОГРАДОВ, Т. Л. КАРАВИЧЕВА

(Москва)

ЯЗЫК ПРОМЕЖУТОЧНОГО УРОВНЯ SAMILA ДЛЯ ПРОГРАММИРОВАНИЯ SAMAC НА RDP-11 МАЛОЙ КОНФИГУРАЦИИ

1. **Введение.** Язык программирования промежуточного уровня [1] SAMILA (SAMAC—INTERMEDIATE—LANGUAGE FOR ASSEMBLER) совместно с ассемблером ACC-400A предназначен для программирования предэкспериментальных этапов и экспериментальных исследований, использующих оборудование в стандарте SAMAC и требующих, как правило, быстрой реакции системы в реальном масштабе времени. Трансляция языка промежуточного уровня на машинный осуществляется методом предкомпиляции (предпроцессор), обеспечивающим выделение операторов и замену их соответствующими макро, написанными на уровне ассемблера используемого языка. Таким образом, на следующем этапе производится обычное ассемблирование программы в машинные коды.

Настоящий текст содержит описание только языка SAMILA. Предполагается предварительное знакомство со стандартом SAMAC [2—4] и ассемблером ACC-400A. Язык SAMILA обеспечивает независимость написания предложений от конфигурации используемых аппаратных средств, позволяет описать разные процедуры, решаемые в рамках

стандарта САМАС, и обеспечивает возможность доступа ко всем элементам аппаратных средств на различных уровнях.

Язык САМІЛА написан для работы с машинами семейства PDP-11 и допускает минимальную конфигурацию машины. Длина операндов принята равной длине машинного слова (16 разрядов). Данная версия языка требует следующий номинальный состав оборудования вычислительной машины:

- 1) процессор М-400 с 8 К слов памяти (минимум 4 К);
- 2) терминал ("Consul-260");
- 3) фотосчитыватель;
- 4) перфоратор.

Для сопряжения одной секции * САМАС с общей шиной ЭВМ М-400 используется специализированный контроллер секции КК М-400 [3]. Предкомпилятор практически не зависит от типа контроллера, подключаемого к вычислительной машине, модификация его под конкретный контроллер требует незначительных изменений в программе.

Для описания формата команд используется метаязык, синтаксис которого включает следующие специальные знаки:

- ::= — разделение левой и правой части определений, означающее «определяется как»;
- <> — соответственно открывание и закрывание последовательности знаков, содержащих символы метаязыка;
- / — разделение альтернативных синтаксических блоков в правой части определения;
- [] — соответственно открывание и закрывание возможного дополнительного набора синтаксических блоков;
- (), {} — соответственно открывание и закрывание группы элементов, рассматриваемых как синтаксический блок;
- ... означает, что предшествующий синтаксический блок может быть повторен один или более раз.

2. Основные определения. Синтаксис языка обеспечивает две формы предложений. Первая форма обеспечивает декларирование имен, соответствующих внешним и внутренним ссылкам, вторая — соответствует операторам действия, использующим декларированные имена или любую прямую комбинацию <BCNA>.

Программа на языке САМІЛА состоит из строк; в каждой строке может быть записана только одна команда, которая заключается в скобки и при печати обязательно должна заканчиваться нажатием клавиш CR (возврат каретки) и LF (перевод строки). У оператора действия за круглыми скобками ставится знак квалификатора (об использовании квалификатора будет сообщено ниже). Номер строки выполняет роль меток для реализации взаимных ссылок внутри языка.

Алфавит языка САМІЛА состоит из следующих символов:

буква ::= A/B/C/D/E/F/G/H/I/J/K/L/M/N/O/P/Q/R/S/T/U/V/W/X/Y/Z

цифра ::= 0/1/2/3/4/5/6/7/8/9/

символ ::= буква/цифра ; <> , ! ! \$ # / & / / = ? * ' @ []

Использовать пробел как символ нельзя, так как он игнорируется предкомпилятором!

Идентификатор в языке САМІЛА служит для того, чтобы назвать определенную константу, переменную, массив, буфер, данную комбинацию значений <BCNA> или группу комбинаций значений <BCNA> своим именем. Идентификатором является произвольная последовательность четырех символов, начинающаяся с буквы (первая буква в

* Терминология соответствует литературе [4].

идентификаторе определяет способ адресации, поэтому она не может быть выбрана). Следует помнить, что два одинаковых идентификатора не могут встретиться в одной программе.

Числа в языке SAMILA используются двух типов (для записи величины констант и переменных): десятичные и восьмеричные. Для указания типа числа служит предупредительная точка, которая ставится в конце десятичного числа. Во всех остальных случаях используются только десятичные числа (для большей наглядности), записанные в виде двух (четырёх) цифр без точки.

Номер оператора записывается в начале строки, при необходимости он может быть любым целым числом от 00 до 99 включительно (номер строки должен состоять из двух цифр).

Операторы языка могут быть двух типов. Декларативный оператор предписывает, какие действия следует выполнять предкомпилятору в момент его работы. Оператор действия предписывает, какие действия нужно выполнить в момент исполнения исходной программы.

3. Операторы языка. Полный набор операторов языка в версии 1 разбит на пять групп:

- 1) декларативные операторы (4 типа);
- 2) основные операторы однократного действия (4 типа);
- 3) системные операторы (21 тип);
- 4) операторы условных переходов (10 типов);
- 5) операторы блочного обмена (5 типов).

3.1. Декларативные операторы. Операторы декларации определяют свойства идентификатора отдельного оператора, которые должны быть известны оператору действия, ссылающемуся к идентификатору, чтобы генерировать правильный код [4].

Декларирование памяти осуществляет оператор LCM, определяющий константу, переменную, массив, индексруемый массив и буфер (т. е. внутреннюю ссылку). Константа, декларированная в языке SAMILA, может использоваться в ассемблере без предварительного ее определения. Переменная, массив и индексруемый массив могут использоваться только в данном языке. Буфер является связывающим звеном двух языков и доступен операторам обоих языков. В языке SAMILA длина слова не указывается, так как она принята всегда равной 16 битам.

Возможны следующие синтаксически определенные конструкции предложений, использующих оператор LCM:

LCM (<K—NAME>, <LITERAL>
/ (<V—NAME>)/(<A—NAME>(<ARRAY—LENGTH>))
/(<S—NAME>(<ARRAY—LENGTH>))/(<B—NAME>, <BUFFER—
LENGTH>)), где <I—NAME>— имя идентификатора;

I = K/V/A/B/S/

<LITERAL>— числовая константа, записанная в виде десятичного или восьмеричного числа;

<ARRAY—LENGTH>— числовое значение длины массива, записывается в виде числа от 00 до 99;

<BUFFER—LENGTH>— числовое значение длины буфера, записывается в десятичном или восьмеричном виде.

Имя идентификатора должно начинаться соответственно с буквы K, V, A, B, S (которая и определяет способ адресации) и иметь полную длину, равную четырем знакам, где K указывает, что декларируется константа, V — переменная, B — буфер, A и S — соответственно неиндексруемый и индексруемый массив. Внешняя форма конструкции для индексруемого и неиндексруемого массива одинакова, но при использовании индексруемого массива можно обращаться к любому элементу массива.

Для всех способов адресации, кроме буфера, язык SAMILA сам резервирует необходимое число ячеек памяти. При декларации буфера в языке SAMILA отводятся ячейки только для заголовка буфера (т. е. для хранения начального адреса буфера, текущего адреса и общего числа ячеек), а сам буфер должен резервироваться в основном языке (ассемблере). Начальному адресу буфера присваивается имя буфера. Контроль за правильностью организации буфера и его использованием возлагается на программиста.

Следует помнить, что организация буфера такова, что он имеет два указателя текущего значения. Один из них используется для записи информации в буфер, другой — для считывания информации. Когда содержимое двух указателей одинаково, происходит восстановление первоначального положения, т. е. указатели задают начальный адрес буфера. Возврат буфера в исходное состояние осуществляется при считывании из буфера информации.

Пример записи декларативного оператора:

```
(LCM BUFF, 10.)
```

Пример резервирования буфера:

```
BR. +24
```

BUFF: .WORD 0 — для буфера резервируются десять ячеек и имя его — BUFF

За программистом остается право выбора другого способа резервирования.

Декларирование устройств осуществляет оператор LCD, определяющий один адрес устройства, заданную неиндексируемую и индексируемую матрицу адресов устройств и матрицу сканирования. На способ адресации указывают соответственно буквы H, G, T и Q, с которых должно начинаться имя внешней ссылки. Возможны следующие варианты синтаксически верных конструкций использования этого оператора:

```
LCD (<H—NAME>, <BCNA>)/(<G—NAME> (<#—BCNA>)  
[, <BCNA>...])  
/(<Q—NAME> (LN), <BCNA>)/(<T—NAME> (<#—BCNA>)  
[<BCNA>...])
```

где <J—NAME> — имя идентификатора;

J=H/G/Q/T

<#—BCNA> — числовое значение общего числа декларируемых устройств.

В матрице сканирования <BCNA> указывает начальный адрес сканирования, а LN — числовое значение длины сканирования (LN принимает значение от 02 до 15). Внешняя форма конструкции для индексируемой и неиндексируемой матрицы адресов устройств одинакова, но индексируемая матрица адресов устройств дает возможность обратиться к любому элементу массива.

Замечание. Все правильные конструкции языка записываются в одну строку, исключение составляет оператор декларации заданного массива.

Пример: (LCD HARD, .B01C02.N03.A04)

```
(LCD GIVN(2), .B01.C02.N03.A04, .B01.C02.N03.A05)
```

Декларирование LAM-запросов осуществляет оператор LCL, определяющий устройство, рассматриваемое в программе как источник LAM-запросов. Правильная конструкция оператора LCL:

```
(LCL<L—NAME>, <BCNA>)
```

<L—NAME> — имя идентификатора, которое должно начинаться с буквы

<BCNA> — САМАС-адрес.

Пример: (LCL LAM4, .B01.C02.N04.A05)

Декларирование канала осуществляет оператор LCC, описывающий канал, используемый для организации блочных операций. Этот оператор имеет формат:

(LCC<CH—NAME>, <REPEAT>, <TALLY>, <STATUS>)
 <REPEAT> ::= <K—NAME>/<V—NAME> — число повторений исполнения операций однократного действия в блочных операциях;

<TALLY> — адрес ячейки памяти, выполняющей роль счетчика числа повторений операций однократного действия;
 <STATUS> — адрес ячейки статуса канала.

Декларирование канала сводится к резервированию блока ячеек памяти, состоящего из четырех адресов для канала типа <K—NAME> и трех адресов для канала типа <V—NAME>. Ссылки к TALLY и STATUS осуществляются по их символическим именам, определяемым в основном языке, а REPEAT определяется в языке CAMILA.

Резервирование памяти для канала:

<K—NAME>

BR.+10.

.WORD DATA (K—NAME)

CH—NAME: .WORD.—2

<V—NAME>

BR.+8.

CH—NAME: .WORD ADR (V—NAME)

.WORD ADR (TALLY)

.WORD ADR (STATUS)

.WORD ADR (TALLY)

.WORD ADR (STATUS)

Пример: (LCC CAN1, KNST, TAL1, STAT)

3.2. **Операторы действия** (табл. 1) осуществляют основные операции однократного действия по указанному адресу. Правильные конструкции:

(<F—READ> <DEV—1>, <MEM—1>)1

(<F—WRITE> <DEV—1>, <MEM—2>)1

(<F—OPERATE> <DEV—1>)1

(<F—STATUS> <DEV—1>, <V—NAME>)1

где <F—READ>, <F—WRITE>, <F—OPERATE>, <F—STATUS> — операция чтения, записи, операция без передачи данных, статусные операции, которые могут быть записаны в виде кода операции или соответствующей мнемоники;

Т а б л и ц а 1

Основные операции однократного действия

Операция	CAMILA	Время исполнения	Операция	CAMILA	Время исполнения*
Чтение регистра 1-й группы G1	F00/RD1	21 мкс	Перезапись в регистр G1	F16/WT1	20 мкс
Чтение регистра 2-й группы G2	F01/RD2		Перезапись в регистр G2	F17/WT2	
Чтение и сброс регистра G1	F02/RC1		Селективная установка регистра G1	F18/SS1	
Чтение обратного кода регистра G1	F03		Селективная установка регистра G2	F19/SS2	
Нестандартная Резервная	F04		Нестандартная Селективный сброс регистра G1	F20	
Нестандартная Резервная	F05		Нестандартная Селективный сброс регистра G2	F21/SC1	
Нестандартная Резервная	F06		Нестандартная Селективный сброс регистра G2	F22	
Проверка запроса	F08/TLM	21 мкс		F23/SC2	
Сброс регистра G1	F09/CL1				
Сброс запроса	F10/CLM	18 мкс	Запрещение	F24/DIS	18 мкс
Сброс регистра G2	F11/CL2		Исполнение	F25/XEQ	
Нестандартная Резервная	F12		Разрешение	F26/ENB	
Нестандартная Резервная	F13		Проверка статуса	F27/TST	
Нестандартная Резервная	F14		Нестандартная Резервная	F28	
Нестандартная Резервная	F15	Нестандартная Резервная	F29	21 мкс	
			Нестандартная Резервная	F30	18 мкс
			Нестандартная Резервная	F31	

* Оценки приведены для процессора PDP 11/40.

<DEV—1> — имя внешней ссылки;
 <MEM—1>, <MEM—2> — имя внутренних ссылок:
 <F—READ> ::= (F00/RD1)/(F01/RD2)/(F02/RC1)/(F03/RCM)/F04/
 /F05/F06/F07
 <F—WRITE> ::= (F16/WT1)/(F17/WT2)/(F18/SS1)/(F19/SS2)/F20/
 /(F21/SC1)/F22/(F23/SC2)
 <F—OPERATE> ::= (F09/CL1)/(F10/CLM)/(F11/CL2)/F12/F14/F15/
 /(F24/DIS)/(F25/XEQ)/(F26/ENB)/F28/F29/F30/F31
 <F—STATUS> ::= (F08/TLM)/(F27/TST)
 <DEV—1> ::= <H—NAME>/<DEV—S>/<BCNA>
 <DEV—S> ::= <T—NAME>, <SUBSCRIP>

где

<SUBSCRIP> — числовое значение индекса, записанного в восьмеричном или десятичном виде (десятичное число следует отмечать точкой);

<MEM—1> ::= <B—NAME>/<V—NAME>/<MEM—S>

<MEM—2> ::= <K—NAME>/<MEM—1>

<MEM—S> ::= <S—NAME>, <SUBSCRIP>

Единица, стоящая за круглыми скобками, указывает, что операция, мнемоника или код, упомянутые внутри команды, используются один раз. В языке имеется возможность использовать недеklarированное устройство, для этого вместо <DEV—1> надо писать <BCNA>.

Порядок адресных аргументов одного операнда произвольный, один или несколько аргументов могут опускаться, при этом недостающие аргументы сохраняются из последней исполняемой команды.

Пример: (F01 HARD, VAR1)1
 (RD1 .C02.N03.A04, VAR1)1
 (RD2 .A02)1

Последняя команда чтения выполняется в секции 2 станции 3 по субадресу 2.

Системные операторы (табл. 2) подразделяются на 3 подгруппы, каждая из которых характеризуется уровнем адресации (субадрес, секция и ветвь). В данной версии языка включены все перечисленные уровни адресации.

А. Операции на уровне субадреса реализуются следующими операторами:

ENL, DIL осуществляют соответственно разрешение или запрещение запроса в модуле. Формат использования операторов: (ENL/DIL<L—NAME>)1

Пример: (ENL LAM1)1

CLL производит сброс запроса в модуле. Формат использования: (CLL<L—NAME>)1

Пример: (CLL LPEN)1

RDL производит чтение запроса в модуле. Формат команды: (RDL<L—NAME>, <MEM—1>)1

Пример: (RDL LAM2, VARB)1

RLS осуществляет операцию чтения статусного регистра LAM-запросов. Формат команды: (RLS <BCN>, <MEM—1>)1 где <BCN>—САМАС-адрес без указания субадреса.

Пример: (RLS .B01.C02.N04, VAR1)1

RLR предписывает выполнение операций чтения регистра запросов LAM. Формат команды: (RLR <BCN>, <MEM—1>)1

Пример: (RLR .B01.C02.N04, VAR2)1

RLM выполняет операцию чтения регистра маски запросов LAM. Формат команды: (RLM <BCN>, <MEM—1>)1

Пример: (RLM .B01.C02.N03, VAR3)1

WLM осуществляет запись маски запросов LAM в регистр маски. Формат команды: (WLM <BCN>, <MEM—2>)1

Таблица 2

Системные операции

Операция	CAMILA	Время исполнения
Разрешение L-запросов в модулях Запрещение L-запросов в модулях Сброс L-запросов в модулях Чтение запроса LAM Чтение регистра статуса LAM Чтение регистра запросов LAM Чтение регистра маски запросов LAM Запись маски запросов LAM	FNL DIL CLL	18 мкс
	RDL RLS RLR	21 мкс
	RLM WLM	20 мкс
Запуск адресуемой секции Сброс адресуемой секции Установка запрета в секции Сброс запрета в секции Разрешение запроса в секции Запрещение запроса в секции Чтение запроса секции Чтение статуса секции	CRZ CRC SCI CCI	5,1 мкс
	ECD DCD RCG RCS	14,2 мкс
Запуск ветви Разрешение запроса ветви Запрещение запроса ветви Чтение запросов системы ветви Чтение Q последней операции	BRZ EBD DBD	5,1 мкс
	RBG	21 мкс
	RBQ	7,4 мкс

Пример: (WLM.B01.C02.N05, KNST) 1

Б. Операции на уровне секции осуществляют следующие операторы:
CRZ и **CRC** выполняют соответственно запуск и сброс адресуемой секции. Формат команды: (CRZ/CRC<BC>)1

SCI и **CCI** осуществляют соответственно операции установки и сброса запрета в секции. Формат команды: (SCI/CCI<BC>)1
где <BC> — CAMAC-адрес секции (без субадреса и номера станции).

Пример: (SC1.B01.C02) 1

ECD и **DCD** осуществляют операции разрешения и запрещения запроса в секции. Формат команды: (ECD/DCD<BC>)1

Пример: (ECD.B01.C01) 1

RCG выполняет операцию чтения упорядоченных запросов секции. Формат команды: (RCG<BC>, <K—NAME>/<V—NAME>, <MEM—1>)1
где <K—NAME>/<V—NAME> — маска регистра запросов.

Пример: (RCG.B01.C02, KMSC, VAR1) 1

RCS выполняет операцию чтения статуса секции, т. е. чтение регистра запросов без маскирования. Формат команды: (RCS<BC>, <MEM—1>)1

Пример: (PCS.B01.C02, VARN) 1

В. Операции на уровне ветви реализуют следующие операторы:
BRZ осуществляет запуск адресуемой ветви. Формат команды: (BRZ<BRANCH>)1

где <BRANCH> — CAMAC-адрес ветви.

Пример: (BRZ.B01) 1

EBR и **DBR** выполняют соответственно операцию разрешения и запрещения запроса ветви. Формат команды: (EBR/DBR<BRANCH>)1

Пример: (EBR.B01) 1

RBG выполняет операцию чтения запросов ветви. Формат команды: (RBG<BRANCH>, <MEM—1>)1

Операции условных переходов

Операция	CAMILA	Время исполнения
Переход по запросу в модулях Переход при отсутствии запроса в модуле Переход по статусу запросов в модулях Переход при отсутствии статуса запросов	YLA NLA YSA NSA	17,5 мкс
Переход по запрету в секции Переход при отсутствии запрета в секции Переход по запросу в секции Переход при отсутствии запроса в секции	YCI NCI YCD NCD	17,5 мкс
Переход по BQ=1 в ветви Переход по BQ=0 в ветви	YBQ NBQ	17,5 мкс

Пример: (RBG .B01, VAR1)1

RBQ осуществляют операцию чтения Q последней операции. Формат команды: (RBQ<BRANCH>, <V—NAME>))1

Пример: (RBQ .B01, VAR2)1

Для специализированного контроллера секции ветвь вырождается в магистраль ЭВМ (т. е. B00), поэтому в указании CAMAC-адреса в операторе действия опускаем аргумент ветви.

Замечание. Для более быстрого запоминания мнемоники системных операторов следует помнить, что буквы мнемоники определяют название операции (латинский алфавит), указывают уровень адресации и уточняют смысл выполняемой операции.

Например, операция установки запрета в секции имеет мнемонику SCI. Это операция установки (SET) в секции (CREATE) запрета (INHIBIT).

Условные переходы (табл. 3) осуществляют следующие операторы.

A. YLA, NLA производят соответственно переход по наличию или отсутствию запроса в модуле. Формат команды: (YLA/NLA<L—NAME>, <LABEL>))1

<LABEL> — номер строки, к которой осуществляется переход при выполнении условия перехода.

Пример: (YLA LAM1, 10)1

YSA, NSA выполняют переход соответственно по наличию запросов в модуле или отсутствию статуса. Формат команды: (YSA/NSA<L—NAME>, <LABEL>))1

Пример: (NSA LAM3, 20)1

Б. Проверка условий перехода на уровне секции осуществляется следующими операторами:

YCI, NCI осуществляют соответственно переход по наличию или отсутствию запрета в секции. Формат команды: (YCI/NCI<BC>, <LABEL>))1

Пример: (YCI .B01.C02, 19)1

YCD, NCD выполняют соответственно переход по наличию или отсутствию запроса в секции. Формат команды: (YCD/NCD<BC>, <LABEL>))1

Пример: (YCD .B01.C02, 19)1

В. Проверка условий перехода на уровне ветви производится следующими операторами:

YBQ осуществляет переход по BQ=1. Формат команды: (YBQ<BRANCH>, <LABEL>))1

Операции блочного обмена

Операция	SAMILA	Время исполнения
Метод останова, синхронизация по LAM Метод останова, синхронизация контроллером Метод повторения, синхронизация по Q	(COMMAND)S (COMMAND)C (COMMAND)R	31+p8,7 мкс
Обмен данными со многими устройствами Мультиадресное сканирование.	(COMMAND)P (COMMAND)Q	31+p8,7 мкс

Пример: (YBQ .B01, 12)1

NBQ осуществляет переход по BQ=0 в ветви. Формат команды: (NBQ<BRANCH>, <LABEL>)1

Пример: (NBQ .B01, 12)1

Примечание. Мнемоника этих операторов начинается с Y(YES) и N(NO). Вторая и третья буквы несут такое же смысловое значение, как соответствующие буквы в мнемонике системных операций.

Многократные действия и операции блочного обмена (табл. 4) разделяются на две подгруппы: одноадресного и многоадресного обмена. Блочные операции одноадресного обмена предназначены для передачи слов SAMAC между одним регистром и памятью ЭВМ. Многоадресные операции блочного обмена определяют однократное действие по каждому адресу из последовательности адресов SAMAC.

Операции блочного одноадресного обмена различаются по методу организации процедур обмена. Для организации блочного обмена команда заключается в круглые скобки и за скобками вместо числа повторений ставят один из квалификаторов S(STOP), C(CONTROLLER), R(REPEAT).

А. Блочный обмен, использующий метод останова. Формат команды: (<F—READ>)/<F—WRITE><DEV—1>, <MEM—3>, [<L—NAME>], <CH—NAME>)S <MEM—3>::=<A—NAME>/<B—NAME>/<MEM—S> <L—NAME> используется для синхронизации по LAM

Пример: (RD1 HS12, BUF2, LAM8, CHN2)S

Б. Блочный обмен с синхронизацией контроллером:

(<F—READ>)/<F—WRITE><DEV—1>, <MEM—3>, <CH—NAME>)C

Пример: (RD1 HNAM, VNAI, CAN1)C

В. Блочный обмен с повторением при Q=0 (метод повторения):

(<F—READ>)/<F—WRITE><DEV—1>, <MEM—3>, <CH—NAME>)R

Пример: (RD1 HARD, BUF1, CAN1)R

Операции блочного многоадресного обмена включают операции с заданными модулями и сканерный режим.

А. Обмен данными со многими устройствами. Формат команды: (<F—READ>)/<F—WRITE><DEV—2>, <MEM—3>, <CH—NAME>)P

Пример: (RD1 GIV1, BUF2, CHN1)P

Б. Мультиадресное сканирование. Правильные конструкции этой операции: (<F—READ>)/<F—WRITE><DEV—3>, <CH—NAME>)Q или (<F—OPERATE><DEV—3>, <CH—NAME>)Q

<DEV—3>::=<Q—NAME> <DEV—2>::=<G—NAME>/<DEV—S>

Пример: (RD1 QNAM, BUF2, CHN2)Q

Рассмотрим примеры реализации программ на языке SAMILA.

А. Требуется обеспечить считывание данных со счетчика, который находится на станции N=15 и содержит закодированную величину временного интервала. Считывание инициируется по запросу от блока управления. По данным, считываемым со счетчика, строится и накапливается в памяти гистограмма требуемой размерности. Далее приведена упрощенная программа с соответствующими комментариями:

(LCM BUFF, 1)	— декларирование единичного буфера;
(LCD HAR1, .C00.N15.A00)	— декларирование адреса счетчика;
(F00 HAR1, BUFF) 1	— чтение данных из счетчика в буфер;
MOV#BUFF, R0	— построение гистограммы по данным,
MOV#BUF1, R1	читанным из счетчика;
ASL (R0)	
ADD (R0), R1	
INC (R1)	
RTI	
BUFF: .WORD 0	
BUF1: .WORD 0	— резервирование ячеек для буфера.
.=.+1022.	

Б. Требуется обеспечить вывод информации на экран осциллографа со световым карандашом [5]. Для управления осциллографом используется интерфейс, расположенный на станции N=2. Вывод информации на осциллограф может производиться в двух режимах работы интерфейса — последовательном и произвольном. Принцип управления карандашом основан на использовании режима Q-останова. Вывод информации на экран осциллографа осуществляется с повторением при Q=1. Выход из этой операции производится по Q=0 при наличии сигнала от карандаша или по условию окончания блочной операции. В данном примере используется последовательный режим. При каждой записи числа, выводимого на экран, содержимое X увеличивается на 1. Ниже приведен простой вариант программы:

(LCM BUF2, 1024.)	— декларирование буфера;
(LCD HAR2, .C00.N02.A00)	— декларирование адреса интерфейса.
(LCC CAN1, KREP, TAL1)	— декларирование канала для блочной операции;
STAT)	
(LCM KREP, 1024.)	— декларирование константы;
(LCM KONS, 1)	— декларирование константы,
TAL1=100	
STAT=102	
(F17 HAR2, KONS) 1	— установка режима работы интерфейса;
K1:MOV #100, 177560	
(F16 HAR2, BUF2, CAN1) S	— блочная операция вывода информации на экран;
CMP#KREP, TAL1	— определение условия выхода из блочной операции;
BNE K2	Q=0
JMP K1	Q=1
K2: RTI	
BUF2: .WORD 0	— резервирование ячеек памяти для буфера.
.=.+1022.	

4. **Загрузка и контроль работы предпроцессора.** Программа «Предкомпилятор», отперфорированная в абсолютном двоичном формате, загружается в ОЗУ программой АЗАГР. Для запуска программы необходимо поместить исходную ленту (написанную на языке SAMILA) в перфосчитыватель и запустить программу. Если лента не начала свое движение через считывающее устройство, значит, программа «Предкомпилятор» загружена неправильно и требуется новая ее загрузка. Результат работы программы «Предкомпилятор» — вывод символьной ленты, готовой для ассемблирования. Если в программе обнаружена ошибка, то выводится текст соответствующего оператора с указанием кода ошибки.

ЛИТЕРАТУРА

1. Kubitz M., Kind R. Macro-IML manual for DEC PDP-11 computer with controller DEC CAMILA. BORER type 1533A, H — MI — B168. Berlin-Wannsee, 1975.
2. CAMAC. A modular instrumentation system for data handling. Description and specification EURATOM EUR 4100 e, March, 1969.— Revised CAMAC Specification EUR 4100 e, 1972.
3. Елизаров О. И., Жуков Г. П., Мячев А. А. Контроллер каркаса в стандарте КАМАК для связи с ЭВМ М-400. Препринт № 11—8396. Дубна, изд. ОИЯИ, 1974.
4. Виноградов В. И. Дискретные информационные системы в научных исследованиях. М., Атомиздат, 1976.
5. Петров А. Г., Сидоров В. Т., Синаев А. Н. Использование в физическом эксперименте осциллографа со световым карандашом, подключенного через интерфейс в стандарте КАМАК. Препринт № 10—11015. Дубна, изд. ОИЯИ, 1977.

*Поступила в редакцию 10 января 1978 г.;
окончательный вариант — 11 апреля 1978 г.*
