

ГУАП

КАФЕДРА № 42

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ
ПРЕПОДАВАТЕЛЬ

старший преподаватель

должность, уч. степень, звание

подпись, дата

Т. В. Семененко

инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ № 5

ПРЕДСТАВЛЕНИЕ ДАННЫХ В ЭВМ ТИПА VAX-11. АРИФМЕТИКО-
ЛОГИЧЕСКИЕ ОПЕРАЦИИ

по курсу:

АРХИТЕКТУРА ЭВМ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ гр. № 4326

подпись, дата

Г. С. Томчук

инициалы, фамилия

Санкт-Петербург 2025

1 Цель работы

Цель работы: знакомство с архитектурой процессора VAX-11, изучение форматов команд и данных, типов адресации, системы арифметико-логических операций процессора VAX-11. Освоение симулятора процессора VAX-11 и выполнение программы арифметико-логической обработки целочисленных данных с использованием прямой адресации и различных способов косвенной адресации.

2 Задание

Практическая часть работы включает выполнение следующих действий:

- формирование числовых значений операндов в соответствии с индивидуальным заданием, перевод их в шестнадцатеричную систему счисления и определение минимального формата представления исходных данных;
- определение номеров РОН и адресов в памяти для размещения операндов;
- составление и выполнение программы работы с данными, хранящимися в РОН или в памяти с использованием прямой и различных способов косвенной адресации, по заданному алгоритму;
- тестирование программы путем ручной трассировки заданного алгоритма с последующим сравнением результатов работы программы с результатами ручной трассировки.

3 Ход выполнения

3.1 Определение исходных данных

Вариативные параметры: $NV=17$, $NGL=6$, $NGP=2$.

Начальный адрес размещения программы: $\text{Адр}(\text{PROGRAM})=17 * 20 + 6 * 100 = 940_{10} = 3AC_{16}$.

Промежуточные ячейки должны быть размещены с адреса:
 $\text{Адр}(\text{TEMP})=17*2+1000=1034_{10}=40A_{16}$

Номер алгоритма: $1 + (17+6+2) \bmod 9 = 8$.

Вариант комбинации способов адресации: $1 + 17 \bmod 20 = 18$.

Алгоритм № 8 (первые три действия):

$$X4 = 0$$

$$X9 = X7 * 2^{X3}$$

$$X7 = X8 - X9$$

Переменные:

$$X3 = (-1)^{17+6} * (17 + 2 + 6 + 21) = -46 = D2_{16} \text{ (byte)}$$

$$X4 = (-1)^{17} * (17 + 2 + 29)^2 = -2304 = F700_{16} \text{ (word)}$$

$$X5 = (-1)^{(17+1)} * (17 + 6 + 23)^2 = 2116 = 0844_{16} \text{ (word)}$$

$$X6 = (-1)^{(17+6)} * (17 + 2 + 6 + 79)^2 = -10816 = D5C0_{16} \text{ (word)}$$

$$X7 = X4^2 = -2304^2 = 5308416 = 00510000_{16} \text{ (long)}$$

$$X8 = -X5^2 = -2116^2 = -4477456 = FFBBADF0_{16} \text{ (long)}$$

$$X9 = (-1)^{(17+6)} * X6^2 = -116985856 = F906F000_{16} \text{ (long)}$$

Размещение в регистрах: X1 — R3, X3 — RE, X5 — RB.

Размещение в памяти:

$$1. \text{ Адр}(X2) = 17 * 6 + 2 = 104_{10} = 68_{16}$$

$$2. \text{ Адр}(X4) = 17 * 6 + 2 + 10 = 114_{10} = 72_{16}$$

$$3. \text{ Адр}(X6) = 17 * 6 + 2 + 20 = 124_{10} = 7C_{16}$$

$$4. \text{ Адр}(X7) = 17 * 6 + 2 + 30 = 134_{10} = 86_{16}$$

$$5. \text{ Адр}(X8) = 17 * 6 + 2 + 40 = 144_{10} = 90_{16}$$

$$6. \text{ Адр}(X9) = 17 * 6 + 2 + 50 = 154_{10} = 9A_{16}$$

Способы адресации (вариант № 18):

$$\text{Оп1} = 0$$

$$\text{Оп4} = \text{Оп3} * 2^{\text{Оп2}}$$

$$\text{Оп7} = \text{Оп6} - \text{Оп5}$$

1. Оп1 — 8 (автоинкрементная)

2. Оп2 — 7 (автодекрементная)

3. Оп3 — 7 (автодекрементная)

4. Оп4 — 7 (автодекрементная)

5. Оп5 — 9 (косвенная автоинкрементная)

6. Оп6 — 6 (косвенная регистровая)

7. Оп7 — 6 (косвенная регистровая)

3.2 Карта распределения памяти под команды и данные

Сперва необходимо составить карту выделения памяти под нужные данные для работы программы. Данная карта представлена в таблице 1.

Таблица 1 — Карта распределения памяти

Объект	Формат	Шестнадцатеричный код	Адрес загрузки
X4	W	00 F7	72
X6	W	C0 D5	7C
X7	L	00 00 51 00	86
X8	L	F0 AD BB FF	90
X9	L	00 F0 06 F9	9A
X3'	B	FA	40A
Адр(X9)	L	9A 00 00 00	40B
Текст программы		Текст программы	3AC–3B5

3.3 Текст программы в мнемонических и машинных кодах

По составленному алгоритму была написана программа в шестнадцатеричных кодах. Программа представлена в таблице 2.

Таблица 2 — Текст программы

Оператор	Адрес	Шестнадцатеричный код	Мнемонический код	Комментарий
1	3AC	B4 80	CLRW (R0)+	X4 = 0
2	3AE	78 71 72 74	ASHL -(R1), -(R2), -(R4)	$X9 = X7 \times 2^{X3'}$
3	3B2	C3 96 67 68	SUBL3 @(R6)+, (R7), (R8)	$X7 = X8 - X9$

3.4 Таблица трассировки программы

Также была проведена ручная трассировка алгоритма, результаты которой представлены в таблице 3.

Таблица 3 — Результаты трассировки

Оператор	Объект	До операции	После операции
1	72 (X4)	F700	0000
	R0	00000072	00000074
2	40A (X3')	FA	FA
	R1	0000040B	0000040A
	86 (X7)	00510000	00510000
	R2	0000008A	00000086
3	9A (X9)	F906F000	00014400
	R4	0000009E	0000009A
	40B (Адр(X9))	0000009A	0000009A
	R6	0000040B	0000040F
	9A (X9)	00014400	00014400
	90 (X8)	FFBBADF0	FFBBADF0
	R7	00000090	00000090
	86 (X7)	00510000	FFBA69F0
	R8	00000086	00000086

3.1 Результаты работы программы

На рисунках 1–4 представлены результаты каждой операции программы.

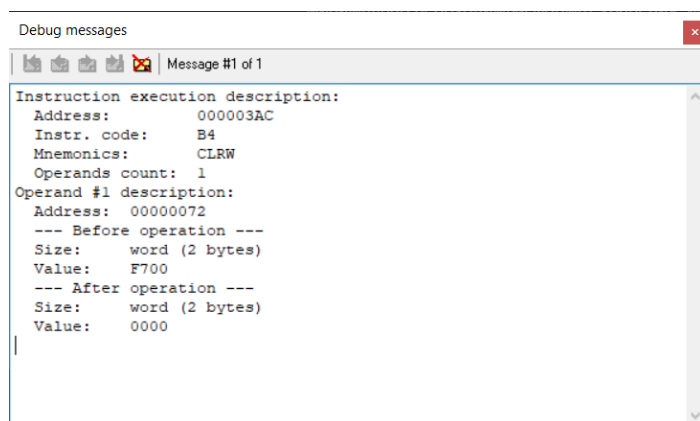


Рисунок 1 — X4 = 0

```

Debug messages
Message #1 of 1

Instruction execution description:
Address:      000003AE
Instr. code:  78
Mnemonics:    ASHL
Operands count: 3
Operand #1 description:
Address:      0000040A
Size:         byte
Value:        FA
Operand #2 description:
Address:      00000086
Size:         longword (4 bytes)
Value:        00510000
Operand #3 description:
Address:      0000009A
--- Before operation ---
Size:         longword (4 bytes)
Value:        F906F000
--- After operation ---
Size:         longword (4 bytes)
Value:        00014400

```

Рисунок 2 — $X9 = X7 \times 2^{X3}$

```

Debug messages
Message #1 of 1

Instruction execution description:
Address:      000003B2
Instr. code:  C3
Mnemonics:    SUBL3
Operands count: 3
Operand #1 description:
Address:      0000009A
Size:         longword (4 bytes)
Value:        00014400
Operand #2 description:
Address:      00000090
Size:         longword (4 bytes)
Value:        FFBBADF0
Operand #3 description:
Address:      00000086
--- Before operation ---
Size:         longword (4 bytes)
Value:        00510000
--- After operation ---
Size:         longword (4 bytes)
Value:        FFBA69F0

```

Рисунок 3 — $X7 = X8 - X9$

```

Debug messages
Message #1 of 1

Instruction execution description:
Address:      000003B6
Instr. code:  00
Mnemonics:    HALT
Operands count: 0

```

Рисунок 4 — Остановка

Как можно заметить, результаты ручной трассировки и результаты выполнения программы в симуляторе полностью совпадают.

4 Вывод

В ходе выполнения лабораторной работы было проведено знакомство с архитектурой процессора VAX-11 и его основными особенностями. Изучены

форматы представления данных различной разрядности: байт (B), слово (W) и длинное слово (L). Для каждой переменной индивидуального задания был определён минимально необходимый формат хранения, исходя из диапазона допустимых значений. Числовые значения операндов были переведены в шестнадцатеричную систему счисления с учётом представления отрицательных чисел в дополнительном коде и порядка следования байтов little-endian, характерного для архитектуры VAX-11.

В процессе работы были освоены различные способы адресации операндов: регистровая, автоинкрементная, автодекрементная, косвенная регистровая и косвенная автоинкрементная. Каждый способ адресации имеет свои особенности применения и кодирования в машинном коде. Автоинкрементная и автодекрементная адресации позволяют автоматически модифицировать содержимое регистра на величину, равную размеру операнда, что удобно при последовательной обработке данных. Косвенные способы адресации обеспечивают дополнительный уровень обращения к памяти через указатели.

Была составлена программа арифметико-логической обработки данных, реализующая заданный алгоритм с использованием команд CLRW (очистка слова), ASHL (арифметический сдвиг) и SUBL3 (вычитание длинных слов). Программа размещена в памяти начиная с адреса 3AC и занимает 11 байт. Для корректной работы программы были подготовлены промежуточные ячейки памяти, содержащие значение операнда сдвига X3' и указатель на переменную X9.

Тестирование программы выполнено методом ручной трассировки с пошаговым отслеживанием изменений содержимого регистров и ячеек памяти. Результаты ручной трассировки полностью совпали с результатами выполнения программы в симуляторе VAX-11: переменная X4 обнулена, переменная X9 получила значение 82944 (результат арифметического сдвига X7 вправо на 6 разрядов), переменная X7 получила значение -4560400 (результат вычитания X9 из X8). Совпадение результатов подтверждает

корректность составленной программы и правильность понимания принципов работы процессора VAX-11.

Таким образом, цель лабораторной работы достигнута: получены практические навыки работы с симулятором процессора VAX-11, освоены принципы формирования машинного кода команд, изучены различные способы адресации операндов и особенности выполнения арифметико-логических операций над целочисленными данными различных форматов.