**Лабораторная работа 2**

УСЛОВНЫЕ И БЕЗУСЛОВНЫЕ ПЕРЕХОДЫ. ОБРАБОТКА МАССИВОВ.

Цель работы: изучение особенностей адресации в командах условного

─────────── и безусловного переходов, изучение адресации с ин-

дексированием, изучение организации циклических вы-

числений, выполнение программы арифметико-логической

обработки массивов данных, хранящихся в памяти.

**Методические указания**

В данной лабораторной работе исследуются команды условного и безусловного перехода, а также возможности ЭВМ типа VAX-11 по организации циклических вычислений и обработке массивов.

В архитектуре VAX-11 используются две независимые системы адресации: одна используется в командах преобразования-тестирования данных, а также в команде безусловного перехода JMP и команде обращения к подпрограмме JSB, а другая - в командах условного перехода, в команде безусловного перехода BR и командах организации циклов типа команды ACB. Введение для команд перехода отдельной системы адресации связано с тем, что чаще всего переходы делаются внутри тела программы. Для того, чтобы программа была свободно перемещаемая в памяти, необходимо адреса перехода определять как смещения относительно некоторого адреса в программе. Естественным указателем на программу является счетчик команд (R15), поэтому адрес перехода лучше всего определять как относительный по счетчику команд. Тогда, если адреса перехода определяются как относительные по счетчику команд, способ адресации и номер регистра-указателя определяются по коду операции (по умолчанию), а в команде указывается только смещение.

В командах условного перехода величина смещения определяется в формате байта, и в этих командах за байтом кода операции следует байт, в котором указывается смещение от начала следующей команды до начала той команды, на которую делается переход. Величина этого смещения лежит в интервале от +127(10) до -128(10). В командах безусловного перехода типа BR и обращения к подпрограмме типа BSB величина смещения определяется или в формате байта (команды BRB и BSBB), или в формате слова (команды BRW и BSBW). В командах организации цикла типа команды ACB величина смещения всегда определяется в формате слова (W).

В тех случаях, когда адрес перехода отличается от текущего значения счетчика команд PC на величину, превышающую 65536 (65К) байт, используются команды JMP и JSB. В этих командах адрес перехода определяется по полной схеме в соответствии с указанным в байте-спецификаторе способом адресации и значением в регистре-указателе. При этом, если по способу адресации требуется определение формата данных (например, при модификации указателя или при адресации с индексированием), то, так как команды представляются в побайтовой форме, принимается, что данные представлены в байтовом формате.

При выполнении вычислений на массивах для выборки элементов массива (ов) как правило используются или различные варианты косвенной адресации, или адресации с индексированием. (Заметим, что адресация с индексированием может рассматриваться как один из вариантов адресации по смещению.

При адресации по смещению одно из слагаемых исполнительного адреса является постоянной величиной, не изменяемой в процессе выполнения программы; а при адресации с индексированием оба слагаемых исполнительного адреса могут модифицироваться в ходе выполнения программы. Кроме того в адресации с индексированием дополнительно учитывается формат данных.)

При использовании косвенной адресации в регистре находится адрес элемента массива (т.е. содержимое регистра как бы указывает на нужное слово, и поэтому довольно-таки часто, как содержимое регистра, так и сам регистр, содержащий адрес слова, называют "указателем"). Если к каждому элементу массива происходит однократное обращение, то используется косвенная адресация с модификацией указателя (инкрементная или декрементная адресация). При этом значение указателя автоматически модифицируется на длину слова, т.е. указатель переустанавливается на следующий элемент массива. При многократных обращениях к каждому злементу массива цепочка последовательных обращений к одному элементу делается с использованием простой косвенной адресации и только последнее обращение (при прямом переборе злементов массива) или первое обращение (при обратном переборе) делается с модификацией указателя.

При использовании адресации с индексированием в индексном регистре, как правило, но не обязательно, находится номер текущего элемента массива, т.е. индексный регистр одновременно может использоваться как счетчик элементов (длины) массива. При этом переход к следующему элементу массива происходит при изменении счетчика элементов массива. Таким образом, при любом числе обращений к одному элементу массива не нужно заботиться об определении исполнительного адреса следующего элемента массива.

Окончание массива определяется или по достижении предельного значения счетчика (инкрементного или декрементного), или по достижении одного из граничных адресов массива.

Для выполнения циклических вычислений в ЭВМ VAX-11 предусмотрены специальные команды организации циклов. По каждой такой команде в машине выполняется сразу несколько действий: во-первых, модифицируется некоторое значение (например, некоторый счетчик или адрес элемента), затем это модифицированное значение сравнивается с контрольным значенеим (например, предельным значением счетчика или граничным адресом) и по результатам этого сравнения выполняется условный переход. Самой универсальной из этой группы команд является команда ACB.

**Практическая часть.**

Практическая часть работы включает выполнение следующих действий:

а) в соответствии с индивидуальным заданием составление двух программ обработки массивов, содержащих не менее 10 целых чисел; одна программа для обращения к элементам массивов должна использовать косвенные способы адресации, а другая - адресацию с индексированием; во второй программе для организации цикла необходимо использовать команду ACB, а в первой программе использование команды организации цикла ACB запрещено;

б) формирование и занесение в память исходных значений массивов, определение и занесение в память и РОНы необходимых вспомогательных данных;

в) запись программ обработки массивов данных, хранящихся в памяти;

г) выполнение программ;

д) контроль результатов работы программ.

Правильность разработки и выполнения программ арифметико-логической обработки данных контролируется путем ручной трассировки заданных алгоритмов с последующим сравнением результатов работы программ с результатами ручной трассировки.

**Варианты заданий.**

Тексты заданий приведены в табл.2.7.

Допустимые способы проверки конца массива (организации цикла) приведены в табл.2.8.

Способы адресации, используемые для вычисления базового адреса при адресации с индексированием, приведены в табл.2.9.

Размеры элементов массива определены в табл.2.10..

Исходные данные должны располагаться в памяти, начиная с адреса, определяемого выражением:

Aдр = NВ \* NГ

Начальный адрес размещения программ определяется выражением:

Aдр = ( NВ \* NГ ) + 60

Вспомогательные значения, необходимые для выполнения программ, должны быть расположены в памяти, начиная с адреса, определяемого выражением:

Aдр = ( NВ \* NГ ) + 300

**Порядок выполнения работы.**

А. В процессе самостоятельной работы

1. Выбрать исходные данные в соответствии с номером варианта.

2. Составить алгоритмы программ для решения поставленной задачи.

3. Составить программы вычислений в мнемонических и машинных кодах.

4. Составить карту распределения памяти под команды и данные.

5. Произвести ручную трассировку программ с использованием заданных исходных данных, при этом в таблице трассировки должны быть отражены значения информации в ячейках памяти и используемых регистрах.

6. Оформить отчет по лабораторной работе.

Б. В учебной лаборатории

1. Записать в память и РОН исходные данные, программы и дополнительную информацию, необходимую для реализации заданных алгоритмов и различных способов адресации.

2. Выполнить программы, занесенные в память.

3. Проверить результаты выполнения программ, сравнивая их с

результатами ручной трассировки алгоритма.

**Содержание отчета**

1. Титульный лист.

2. Текст задания.

3. Перевод исходных данных в шестнадцатеричную систему счисления.

4. Схемы алгоритмов программ.

5. Тексты программ в мнемонических и машинных кодах.

6. Карта распределения памяти под команды и данные.

7. Таблицы трассировки программ.

Реализуемые алгоритмы Таблица 2.7

───────┬──────────────────────────────────────────────────────────

Номер │ Задание

───────┼──────────────────────────────────────────────────────────

1 │ Найти максимальный элемент массива (его номер и значение)

2 │ Найти минимальный элемент массива (его номер и значение)

3 │ Заменить элементы массива:

│ X(i) := 0 , если X(i) < 0

│ X(i) := FF , если X(i) = 0

4 │ Найти сумму положительных элементов массива

5 │ Найти сумму отрицательных элементов массива

6 │ Найти логическую сумму элементов массива , содержащих 0 в

│ младшем разряде

7 │ Найти логическое произведение элементов массива, содержа-

│ щих 1 в старшем разряде

8 │ Найти сумму модулей элементов массива

9 │ Осуществить сдвиг влево положительных элементов массива

│ на 4 разряда

10 │ Найти арифметическую сумму элементов массива, имеющих

│ четный номер

11 │ Построить ряд из чисел Фибоначчи { F(1) = F(2) = 1 ;

│ F(i+1) = F(i) + F(I-1) }

12 │ Заменить элементы массива:

│ X(i) := -X(i) , если X(i) - четное

│ X(i) := 0 , если X(i) - нечетное

13 │ Найти арифметическую сумму элементов массива, значения

│ которых лежат в интервале -10 < X(i) < 20

14 │ Найти логическую сумму элементов массива, по абсолютной

│ величине не превышающих 40

15 │ Найти минимальный положительный элемент массива (его но-

│ мер и значение)

16 │ Найти элемент массива, имеющий максимальное абсолютное

│ значение (его номер и значение)

17 │ Найти количество элементов массива, имеющих отрицательное

│ значение и четный номер

18 │ Найти отрицательный элемент массива, имеющий максимальное

│ абсолютное значение (его номер и значение)

19 │ Найти количество элементов массива, значения которых ле-

│ жат в интервале -20 < X(i) < 50

20 │ Найти количество положительных, нулевых и отрицательных

│ элементов массива

Варианты организации цикла Таблица 2.8

────────┬─────────────────────────────────────────────────────────

Номер │ Способ проверки конца массива

────────┼─────────────────────────────────────────────────────────

1 │ По инкрементному счетчику (по достижению максимального

│ значения номера массива)

2 │ По декрементному счетчику (по достижению минимального

│ значения номера массива)

3 │ По достижению максимального адреса в массиве

4 │ По достижению минимального адреса в массиве

Варианты вычисления базового адреса массива Таблица 2.9

──────────┬───┬───┬───┬───┬───┬───┬───┬───┬───┬───┬───┬───┬───┬───

Вариант │ 1 │ 2 │ 3 │ 4 │ 5 │ 6 │ 7 │ 8 │ 9 │ 10│ 11│ 12│ 13│ 14

──────────┼───┼───┼───┼───┼───┼───┼───┼───┼───┼───┼───┼───┼───┼───

Способ │ │ │ │ │ │ │ │ │ │ │ │ │ │

адресации │ 6x│ 9F│ Ax│ Bx│ Cx│ Dx│ Ex│ Fx│ AF│ BF│ CF│ DF│ EF│ FF

Формат элементов массива Таблица 2.10

────────┬──────────┬───────────┬───────────┬───────────┬──────────

Вариант │ 1 │ 2 │ 3 │ 4 │ 5

────────┼──────────┼───────────┼───────────┼───────────┼──────────

Элемент │ Байт (B) │ Слово (W) │ Двойное │ Слово (W) │ Двойное

массива │ │ │ слово (L) │ │ слово (L)