МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Череповецкий государственный университет»

**Лабораторная работа № 3**

**«Архитектура ЭВМ и система команд»**

**Выполнил:**

**студент гр.** 1ИВТпб-01-31оп

Климов А.Г.  
**Проверил: преподаватель**

Виноградова Л.Н.  
**Отметка о зачете:**

Череповец

2017 год

**Общие положения**

Для решения с помощью ЭВМ некоторой задачи должна быть разработана программа. Программа на языке ЭВМ представляет собой последовательность команд. Код каждой команды определяет выполняемую операцию, тип адресации и адрес. Выполнение программы, записанной в памяти ЭВМ, осуществляется последовательно по командам в порядке возрастания адресов команд или в порядке, определяемом командами передачи управления.

Для того чтобы получить результат выполнения программы, пользователь должен:

* ввести программу в память ЭВМ;
* определить, если это необходимо, содержимое ячеек ОЗУ и РОН, содержащих исходные данные, а также регистров IR и BR;
* установить в PC стартовый адрес программы;
* перевести модель в режим Работа.

Каждое из этих действий выполняется посредством интерфейса модели, описанного в главе 8. Ввод программы может осуществляться как в машинных кодах непосредственно в память модели, так и в мнемокодах в окно Текст программы с последующим ассемблированием.

Цель настоящей лабораторной работы — знакомство с интерфейсом модели ЭВМ, методами ввода и отладки программы, действиями основных классов команд и способов адресации. Для этого необходимо ввести в память ЭВМ и выполнить в режиме Шаг некоторую последовательность команд (определенную вариантом задания) и зафиксировать все изменения на уровне программно-доступных объектов ЭВМ, происходящие при выполнении этих команд.

Команды в память учебной ЭВМ вводятся в виде шестиразрядных десятичных чисел (см. форматы команд на рис. 8.3, коды команд и способов адресации в табл. 8.2—8.4).

В данной лабораторной работе будем программировать ЭВМ в машинных кодах.

**Формулировка варианта задания**

1. Ознакомиться с архитектурой ЭВМ.

2. Записать в ОЗУ "программу", состоящую из пяти команд — варианты задания в табл. 1. Команды разместить в последовательных ячейках памяти.

3. При необходимости установить начальное значение в устройство ввода IR.

4. Определить те программно-доступные объекты ЭВМ, которые будут изменяться при выполнении этих команд.

5. Выполнить в режиме **Шаг** введенную последовательность команд, фиксируя изменения значений объектов, определенных в п. 4, в таблице (см. форму табл. 10.3).

6. Если в программе образуется цикл, необходимо просмотреть не более двух повторений каждой команды, входящей в тело цикла.

Таблица 1. Вариант задания

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **IR** | **Команда 1** | **Команда 2** | **Команда 3** | **Команда 4** | **Команда 5** |
| 5 | 000016 | IN | WR 8 | DIV #4 | WR @8 | JMP 002 |

Таблица 2. Команды и коды

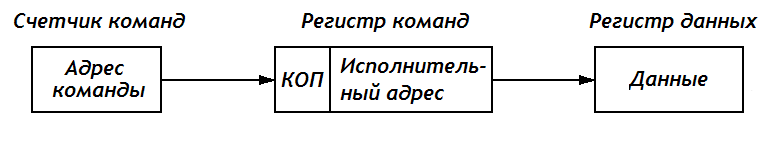
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Последовательность** | **Значения** | | | | |
| **Команды** | IN | WR 8 | DIV #4 | WR @8 | JMP 002 |
| **Коды** | 010000 | 220008 | 261004 | 222008 | 100002 |

Таблица 3. Типы адресации, их коды и обозначение

|  |  |  |  |
| --- | --- | --- | --- |
| **Обозначение** | **Код** | **Тип адресации** | **Пример команды** |
|  | 0 | Прямая (регистровая) | ADD 23 (ADD R3) |
| # | 1 | Непосредственная | ADD #33 |
| @ | 2 | Косвенная | ADD @33 |
| [ ] | 3 | Относительная | ADD [33] |
| @R | 4 | Косвенно-регистровая | ADD @R3 |
| @R+ | 5 | Индексная с постинкрементом | ADD @R3+ |
| –@R | 6 | Индексная с преддекрементом | ADD –@R3 |

**Прямая адресация**

Адрес указывается непосредственно в виде некоторого значения, все ячейки располагаются на одной странице. Преимущество этого способа в том, что он самый простой, а недостаток — в том, что разрядность регистров общего назначения процессора должна быть не меньше разрядности шины адреса процессора.



**Непосредственная адресация**

В команде содержится не адрес операнда, а непосредственно сам операнд. При непосредственной адресации не требуется обращения к памяти для выборки операнда и ячейки памяти для его хранения. Это способствует уменьшению времени выполнения программы и занимаемого ею объёма памяти. Непосредственная адресация удобна для хранения различного рода констант.

**Косвенная адресация**

Адресный код команды в этом случае указывает адрес ячейки памяти, в которой находится адрес операнда или команды. Косвенная адресация широко используется в малых и микроЭВМ, имеющих короткое машинное слово, для преодоления ограничений короткого формата команды (совместно используются регистровая и косвенная адресация).

**Относительная (базовая) адресация**

При этом способе адресации исполнительный адрес определяется как сумма адресного кода команды и базового адреса, как правило хранящегося в специальном регистре — регистре базы.

Относительная адресация позволяет при меньшей длине адресного кода команды обеспечить доступ к любой ячейке памяти. Для этого число разрядов в базовом регистре выбирают таким, чтобы можно было адресовать любую ячейку оперативной памяти, а адресный код команды используют для представления лишь сравнительно короткого «смещения». Смещение определяет положение операнда относительно начала массива, задаваемого базовым адресом.

**Индексная адресация**

Для реализуемых на ЭВМ методов решения математических задач и обработки данных характерна цикличность вычислительных процессов, когда одни и те же процедуры выполняются над различными операндами, упорядоченно расположенными в памяти. Поскольку операнды, обрабатываемые при повторениях цикла, имеют разные адреса, без использования индексации требовалось бы для каждого повторения составлять свою последовательность команд, отличающихся адресными частями.

Программирование циклов существенно упрощается, если после каждого выполнения цикла обеспечено автоматическое изменение в соответствующих командах их адресных частей согласно расположению в памяти обрабатываемых операндов. Такой процесс называется модификацией команд, и основан на возможности выполнения над кодами команд арифметических и логических операций.

**Пояснения к табл. 4:**

 DD — данные, формируемые командой в качестве (второго) операнда: прямо или косвенно адресуемая ячейка памяти или трехразрядный непосредственный операнд;

R\* — содержимое регистра или косвенно адресуемая через регистр ячейка памяти;

ADR\* — два младших разряда ADR поля регистра CR;

V — адрес памяти, соответствующий вектору прерывания;

М(\*) — ячейка памяти, прямо или косвенно адресуемая в команде;

I — пятиразрядный непосредственный операнд со знаком.

**Набор команд:**

IN

WR 8

DIV #4

WR @8

JMP 002

Таблица 4. Система команд учебной ЭВМ

|  |  |  |  |
| --- | --- | --- | --- |
| **КОП** | **Мнемокод** | **Название** | **Название** |
| 01 | IN | Ввод | Acc ← IR |
| 32 | WR | Запись | R\* ← Acc |
| 36 | DIV | Деление | Acc ← Acc/R\* |
| 10 | JMP | Безусловный переход | PC ← CR[ADR] |

Таблица 5. Содержимое регистров

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **PC** | **Acc** | **M(30)** | **M(20)** | **PC** | **Acc** | **M(30)** | **M(20)** |
| 000 | 000000 | 000000 | 000000 | 004 |  |  |  |
| 001 | 000016 |  |  | 002 |  |  |  |
| 002 |  |  |  | 003 | 000001 |  |  |
| 003 | 000004 |  |  | 004 |  |  |  |

RD #20

WR 30

ADD #5

WR @30

JNZ 002

Программно-доступные регистры и флаги:

• Асс — аккумулятор;

• PC — счетчик адреса команды, содержащий адрес текущей команды;

• SP — указатель стека, содержащий адрес верхушки стека;

• RB — регистр базового адреса, содержащий базовый адрес;

• RA — регистр адреса, содержащий исполнительный адрес при косвен- ной адресации;

• IR — входной регистр;

• OR — выходной регистр;

• I — флаг разрешения прерываний.

Системные регистры и флаги:

• DR — регистр данных АЛУ, содержащий второй операнд;

• MDR — регистр данных ОЗУ;

• MAR — регистр адреса ОЗУ;

RDR — регистр данных блока РОН;

• RAR — регистр адреса блока РОН;

• CR — регистр команд, содержащий поля:

◊ СОР — код операции;

◊ ТА — тип адресации;

◊ ADR — адрес или непосредственный операнд;

• Z — флаг нулевого значения Асс;

• S — флаг отрицательного значения Асс;

• OV — флаг переполнения.