Настоящие методические указания к лабораторному практикуму предназначены для практического закрепления материала по дисциплинам "Информатика", “Архитектура ЭВМ и систем” и "Вычислительная техника". Лабораторные работы и домашние задания охватывают часть курсов, посвященную знакомству с принципами функционирования ЭВМ. Они разбиты на три раздела. В первый раздел включены четыре лабораторных работы и два домашних задания, предназначенные для ознакомления с учебой ЭВМ (базовой ЭВМ), на которой выполняются все лабораторные работы, и реализации с ее помощью простейших алгоритмов. Во втором разделе рассматривается организация ввода-вывода информации в базовой ЭВМ, а в третьем - реализация ее микропрограммного устройства управления. В приложениях приведена инструкция по работе с базовой ЭВМ и справочные таблицы.

# РАЗДЕЛ 1. БАЗОВАЯ ЭВМ

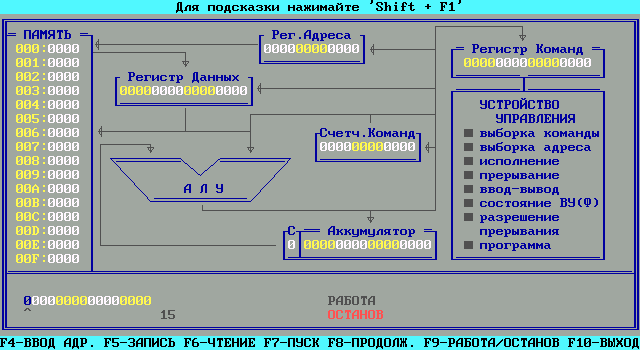
## 1.1 Назначение базовой ЭВМ

Базовая ЭВМ - это простая гипотетическая машина, обладающая типичными чертами многих конкретных ЭВМ. Знание принципов построения и функционирования этой ЭВМ будет хорошей базой для освоения ЭВМ любых типов и моделей. Естественно, что начинать изучение ЭВМ лучше всего с простых моделей, которые и были выбраны за прототип базовой ЭВМ.

## 1.2 Структура базовой ЭВМ

На рис. 1.1 приведена упрощенная структура базовой ЭВМ. Это одноадресная ЭВМ, работающая с 16-разрядными словами. В ней реализованы два вида адресации: прямая и косвенная.

Рассмотрим составные части базовой ЭВМ, не касаясь пока устройств ввода-вывода (УВВ) и пульта управления (ПУ).

**

*Рис. 1.1. Модель базовой ЭВМ*

**Память.** Состоит из 2048 ячеек (16-битовых) с адресами 0,1,...,2046,2047.Восемь ячеек памяти с адресами 008,...,00F несколько отличаются от остальных. Эти ячейки называются индексными и их лучше использовать в циклических программах (п. 1.5).

**Процессор**. Состоит из ряда регистров, арифметическо-логического устройства и устройства управления.

Счетчик команд (СК) служит для организации обращений к ячейкам памяти, в которых хранятся команды программы. После исполнения любой команды СК указывает адрес ячейки памяти, содержащей следующую команду программы. Так как команды могут размещаться в любой из 2048 = 211 ячеек памяти, то СК имеет 11 разрядов.

Регистр адреса (РА) 11-разрядный регистр, содержащий значение исполнительного адреса (адреса ячейки памяти, к которой обращается ЭВМ за командой или данными).

Регистр команд (РК). Этот 16-разрядный регистр используется для хранения кода команды, непосредственно выполняемой машиной.

Регистр данных (РД). Используется для временного хранения 16-рязрядных слов при обмене информацией между памятью и процессором.

Аккумулятор (А). 16-разрядный регистр, являющийся одним из главных элементов процессора. Машина может одновременно выполнять арифметические и логические операции только с одним или двумя операндами. Один из операндов находится в аккумуляторе, а второй (если их два) - в регистре данных. Результат помещается в А.

Регистр переноса (С) - это одноразрядный регистр, выступающий в качестве продолжения аккумулятора и заполняющийся при переполнении А. Этот регистр используется при выполнении сдвигов.

Арифметическо-логическое устройство (АЛУ) может выполнять такие арифметические операции, как сложение и сложение с учетом переноса, полученного в результате выполнения предыдущей операции. Кроме того, оно способно выполнять операции логического умножения, инвертирования, циклического сдвига.

## 1.3. Система команд базовой ЭВМ

Классификация команд. ЭВМ способна понимать и выполнять точно определенный набор команд. При составлении программы пользователь ограничен этими командами. В зависимости от того, к каким блокам базовой ЭВМ обращается команда или на какие блоки она ссылается, команды можно разделить на три группы:

* обращения к памяти (адресные команды);
* обращения к регистрам (регистровые или безадресные команды);
* команды ввода-вывода.

Команды обращения к памяти предписывают машине производить действия с содержимым ячейки памяти, адрес которой указан в адресной части команды.

Безадресные команды выполняют различные действия без ссылок на ячейку памяти. Например, команда CLA (табл. 1.1) предписывает ЭВМ очистить аккумулятор (записать в А код нуля). Это команда обработки операнда, расположенного в конкретном месте, "известном" машине. Другой пример безадресной команды - команда HLT.

Команды ввода-вывода осуществляют обмен данными между процессором и внешними устройствами ЭВМ.

Полный перечень команд базовой ЭВМ приведен в таблице 1.1.

Форматы команд и способы адресации. Разработчики базовой ЭВМ выбрали три формата 16-битовых (однословных) команд с 4-битовым кодом операции (рис. 1.2).

*Рис 1.2. Форматы команд: а - адресных, б - безадресных, в - команд ввода-вывода*

3

1

0

2

15

13

12

14

11

9

8

10

7

5

4

6

**1**

**1**

**1**

**0**

Приказ

на

ввод-вывод

КОП

Адрес устройства

ввода-вывода

в)

КОП

Расширение кода операции

1

0

2

15

13

12

14

11

9

8

10

7

5

4

6

**1**

**1**

**1**

**1**

3

б)

Код

операции

(КОП)

Адрес

Бит вида адресации

Адресная часть команды

3

1

0

2

15

13

12

14

11

9

8

10

7

5

4

6

а)

3

1

0

2

15

13

12

14

11

9

8

10

7

5

4

6

**1**

**1**

**1**

**0**

Приказ

на

ввод-вывод

КОП

Адрес устройства

ввода-вывода

в)

КОП

Расширение кода операции

1

0

2

15

13

12

14

11

9

8

10

7

5

4

6

**1**

**1**

**1**

**1**

3

б)

Код

операции

(КОП)

Адрес

Бит вида адресации

Адресная часть команды

3

1

0

2

15

13

12

14

11

9

8

10

7

5

4

6

а)

В командах обращения к памяти на адрес отведено 11 бит. Следовательно, можно прямо адресоваться к 211 = 2048 ячейкам памяти, т.е. ко всей памяти базовой ЭВМ (прямая адресация). В этом случае бит вида адресации должен содержать 0. Если же в этом же бите установлена 1, то адрес, размещенный в адресной части команды, указывает на ячейку, в которой находится адрес операнда (косвенная адресация).

Отметим, что при мнемонической записи команд указание косвенной адресации производится путем заключения адреса в скобки. Например, команда ADD (25) -сложить содержимое А с содержимым ячейки, адрес которой хранится в ячейке 25 (косвенная адресация).

## 1.4 Арифметические операции

В этом разделе рассматриваются основные способы записи чисел в базовой ЭВМ, арифметические операции, выполняемые с этими числами, и команды, инициирующие арифметические операции.

Целые двоичные числа без знака можно использовать для представления нуля и целых положительных чисел. При размещении таких чисел в одном 16-разрядном слове они могут изменяться от (0000 0000 0000 0000)2 = (0000)16 = 0 до (1111 1111 1111 1111)2 = (1FFF)16 = 215 - 1 = 65535.

Подобные числа (так же как и рассмотренные ниже двоичные числа со знаком) относятся к числам с фиксированной запятой, разделяющей целую и дробную части числа. В числах, используемых в базовой ЭВМ, положение запятой строго фиксировано после младшего бита слова.

Целые двоичные числа со знаком используются тогда, когда необходимо различать положительные и отрицательные числа. В них старший бит используется для кодирования знака: 0 - для положительных чисел и 1 - для отрицательных чисел. Отрицательные числа представлены в дополнительном коде (табл. 1.2). Это упрощает конструкцию ЭВМ, так как при сложении двух таких чисел, имеющих разные знаки, не требуется переходить к операциям вычитания меньшего (по модулю) числа из большего и присвоения результату знака большего числа.

*Таблица 1.1*

*Система команд базовой ЭВМ*

|  |  |  |  |
| --- | --- | --- | --- |
| **Наименование** | **Мнемоника** | **Код** | **Описание** |
| ***Адресные команды*** | | | |
| Логическое умножение  Пересылка  Сложение  Сложение с переносом  Вычитание | AND M  MOV M  ADD M  ADC M  SUB M | 1XXX  3XXX  4XXX  5XXX  6XXX | (M) & (A)  A  (A)  M  (M) + (A)  A  (M) + (A) + (C)  A  (A) – (M)  A |
| Переход, если перенос  Переход, если плюс  Переход, если минус  Переход, если ноль  Безусловный переход  Приращение и пропуск | BCS M  BPL M  BMI M  BEQ M  BR M  ISZ M | 8XXX  9XXX  AXXX  BXXX  CXXX  0XXX | Если (C) = 1, то M  CK  Если (A) >= 0, то M  CK  Если (A) < 0, то M  CK  Если (A)и(C) = 0, то M  CK  M  CK   1. + 1  M, если (M) >= 0,   то (CK) + 1  CK |
| Обращение к подпрограмме | JSR M | 2XXX | (CK)  M, M + 1  CK |
| ***Безадресные команды*** | | | |
| Очистка аккумулятора  Очистка рег. переноса  Инверсия аккумулятора  Инверсия рег. переноса  Циклический сдвиг  влево на 1 разряд  Циклический сдвиг  вправо на 1 разряд  Инкремент аккумулятора  Декремент аккумулятора | CLA  CLC  CMA  CMC  ROL  ROR  INC  DEC | F200  F300  F400  F500  F600  F700  F800  F900 | 0  A  0  C  (!A)  A  (!C)  C  Содержимое A и C сдвигается влево, А(15)  C, C  A(0)  Содержимое A и C сдвигается вправо, A(0)  C, C  A(15)  (A) + 1  A  (A) – 1  A |
| Останов  Нет операции  Разрешение прерывания  Запрещение прерывания | HLT  NOP  EI  DI | F000  F100  FA00  FB00 |  |
| ***Команды ввода-вывода*** | | | |
| Очистка флага  Опрос флага  Ввод  Вывод | CLF B  TSF B  IN  OUT | E0XX  E1XX  E2XX  E3XX | 0  флаг устр. B  Если (флаг устр. B) = 1,  то (CK) + 1  CK  (B)  A  (A)  B |
| ***Примечания:***  (M), (A), (CK), (C), (B) – содержимое ячейки с адресом M, аккумулятора, счетчика команд, регистра переноса и регистра данных устройства ввода-вывода с адресом B.  XXX – адрес ячейки памяти.  XX – адрес устройства ввода-вывода. | | | |

Рассмотрим простое правило для получения дополнительного кода двоичного числа (для примера взято двоичное число, эквивалентное числу 709):

1. Получить инверсию заданного числа (все его 0 заменить на 1, а все 1 - на 0):

0 000 0010 1100 0101 Число

1 111 1101 0011 1010 Инверсия числа

2. Образовать дополнительный код заданного числа путем добавления 1 к инверсии этого числа:

1 111 1101 0011 1010 Инверсия числа

+

1 Слагаемое 1

————————————————————

1 111 1101 0011 1011 Дополнительный код числа

Проверим правильность вычисления дополнения путем сложения заданного числа и его дополнения:

1 1 111 1111 1111 1111 Переносы

+

0 000 0010 1100 0101 Число

1 111 1101 0011 1011 Дополнительный код числа

——————————————————————

1 0 000 0000 0000 0000 0

Так как перенос из старшего разряда выпадает за пределы разрядной сетки, то он не учитывается. Оставшаяся же 16-разрядная сумма равна нулю, что подтверждает правильность преобразования.

*Таблица 1.2.*

*Десятичные эквиваленты 16-битовых двоичных чисел в дополнительном коде.*

|  |  |  |  |
| --- | --- | --- | --- |
| Двоичное число в  прямом коде | Десятичное число | Двоичное число в  дополнительном коде | Десятичное число |
| 0 000 0000 0000 0000  0 000 0000 0000 0001  0 000 0000 0000 0010  0 000 0000 0000 0011  ...  0 111 1111 1111 1110  0 111 1111 1111 1111  —— | 0  +1  +2  +3  ...  +32766  +32767  —— | ——  1 111 1111 1111 1111  1 111 1111 1111 1110  1 111 1111 1111 1101  ...  1 000 0000 0000 0010  1 000 0000 0000 0001  1 000 0000 0000 0000 | ——  -1  -2  -3  ...  -32766  -32767  -32768 |

Сложение целых двоичных чисел со знаком и без знака выполняется в базовой ЭВМ с помощью команды ADD.

Увеличение на 1 (INCREMENT) и уменьшение на 1 (DECREMENT). По команде INC к содержимому аккумулятора прибавляется единица, а по команде DEC - единица вычитается. Если при этом возникает перенос из старшего разряда А, то в регистр переноса заносится 1, в противном случае в него заносится 0.

Вычитание (X-Y) может выполняться путем сложения уменьшаемого X и дополнительного кода вычитаемого Y. Однако это требует записи и выполнения нескольких команд (CLA, ADD Y, CMA, INC, ADD X). Для сокращения программ и времени выполнения вычитания в базовой ЭВМ предусмотрена команда SUB Y (CLA, ADD X, SUB Y), которая реализует те же действия за меньшее время.

Умножение и деление. В базовой ЭВМ нет команд для выполнения этих действий (АЛУ не выполняет таких операций).Поэтому произведение и частное приходится получать программным путем.

## 1.5 Управление вычислительным процессом, сдвиги и логические операции

Задача управления вычислительным процессом, т.е. требуемой последовательностью выполнения команд, решается в базовой ЭВМ при помощи команд переходов (BCS, BPL, BMI, BEQ, BR), команд "Приращение и пропуск" (ISZ) и "Останов" (HLT). Все эти команды (кроме HLT) являются адресными, т.е. в них указывается адрес той ячейки памяти, из которой должна быть выбрана следующая команда программы при выполнении того или иного условия. Если же условия не выполняются, то должна исполняться команда, расположенная вслед за данной командой управления. Как и в других адресных командах, здесь можно использовать косвенную адресацию. Команды переходов не изменяют состояния аккумулятора и регистра переноса. Они могут лишь изменить содержимое счетчика команд, поместив в него адрес, определяемый адресной частью команды.

BCS M (Переход, если перенос). Переход к команде, расположенной в ячейке с адресом M, если содержимое регистра переноса равно 1.

BPL M (Переход, если плюс). Переход к команде, расположенной в ячейке с адресом M, если содержимое аккумулятора больше или равно нулю, т.е. в его старшем разряде (знаковом разряде) содержится 0.

BMI M (Переход, если минус). Переход к команде, расположенной в ячейке с адресом M, если содержимое аккумулятора меньше нуля, т.е. в его старшем (знаковом) разряде содержится 1.

BEQ M (Переход, если нуль). Переход к команде, расположенной в ячейке с адресом M, если содержимое аккумулятора равно нулю.

BR M (Переход безусловный). Переход к команде, расположенной в ячейке с адресом M, осуществляемый при любых значениях А и С или других регистров базовой ЭВМ.

Команды переходов широко применяются для организации циклических программ, которые используются в тех случаях, когда требуется несколько раз выполнить набор одинаковых действий с различными наборами данных. Базовая ЭВМ обладает рядом средств

для упрощения циклических программ. Целесообразность введения этих средств удобнее рассмотреть на примерах.

Пример 1.1 Получить произведение Z = Y \* 50 .

Так как в системе команд базовой ЭВМ нет команды умножения, то воспользуемся простейшим способом: будем 50 раз складывать значение Y, используя программу, приведенную в табл. 1.3.

Так как в этой программе аккумулятор используется не только для накопления произведения, но еще для изменения количества выполненных циклов и сравнения их со значением множителя, то промежуточные результаты Z и С пришлось сохранить в памяти ЭВМ. Очевидно, что обсуждаемую программу можно существенно упростить, при наличии такого средства учета числа выполненных циклов и проверки условия завершения циклической программы, которое не затрагивает содержимого аккумулятора. Таким средством является команда ISZ (Приращение и пропуск). При каждом выполнении команды ISZ M, расположенной по адресу А, к содержимому ячейки с адресом М добавляется 1 и если результат меньше нуля, то выполняется команда, следующая за ISZ M (команда, расположенная по адресу А+1), в противном случае эта команда \*попускается, т.е. выполняется команда, расположенная по адресу А+2. Программа с использованием команды ISZ приведена в табл. 1.4.

*Таблица 1.3*

*Первый вариант программы для получения Z = Y \* 50*

|  |  |  |  |
| --- | --- | --- | --- |
| Ад-рес | Содержимое | | Комментарии |
|  | Код | Мнемо-ника |  |
| 5  6  7  8  . . .  10  11  12  13  14  15  16  17  18  19  1A | 0078  0000  0032  0000  . . .  F200  4006  4005  3006  F200  4008  F800  3008  6007  A010  F000 | Y  Z  M  C  . . .  CLA  ADD 6  ADD 5  MOV 6  CLA  ADD 8  INC  MOV 8  SUB 7  BMI 10  HLT | Множимое (здесь – десятичное значение 120)  Ячейка, отведенная для накопления результата. В ней поочередно будут храниться значения Y, 2\*Y, … После 50 суммирований в этой ячейке будет содержаться искомый результат – 50 \* Y  Множитель 50 = (32)16  Ячейка, используемая для накопления числа выполненных циклов, - счетчик циклов  К промежуточному результату, находящемуся в ячейке 6,  добавляется еще одно значение множимого Y    Содержимое счетчика циклов увеличивается на 1,  а его копия пока сохраняется в аккумуляторе      Если содержимое счетчика циклов меньше значения  множителя, то выполняется переход к командам,  осуществляющим новое суммирование Y с промежу-  точным значением Z  Останов. В ячейке 6 хранится искомый результат |

*Таблица 1.4*

*Второй вариант программы для получения Z = Y* \* 50

|  |  |  |  |
| --- | --- | --- | --- |
| Ад-рес | Содержимое | | Комментарии |
|  | Код | Мнемо-ника |  |
| 5  6  7  . . .  10  11  12  13  14  15 | 0078  0000  FFCE  . . .  F200  4005  0007  C011  3006  F000 | Y  Z  M  CLA  ADD 5  ISZ 7  BR 11  MOV 6  HLT | Множимое  Ячейка, отведенная для накопления результата.  Отрицательное значение множителя (-50)  Очистка аккумулятора  К содержимому аккумулятора добавляется значение Y  Содержимое M наращивается на 1 и, если оно еще  меньше нуля, то выполняется команда BR 11. При  M = 0 команда BR 11 пропускается  Результат 50 сложений Y заносится в ячейку 6  Останов ЭВМ |

Пример 1.2. Получить в ячейке 005 сумму 32 элементов массива, элементы которого размещены в ячейках памяти с 010 по 02F.

В отличие от предыдущей задачи, где многократно суммировалось содержимое одной ячейки (Y), здесь надо суммировать содержимое разных ячеек. Если бы команды базовой ЭВМ позволяли лишь прямо адресовать ячейки памяти, то в программе решения поставленной задачи пришлось бы либо использовать 32 команды сложения (4010, 4011,...,402E, 402F), либо применять модификацию адресной части команды сложения. Последнее реализовано в программе табл. 1.5.

*Таблица 1.5*

*Первый вариант программы суммирования элементов массива*

|  |  |  |  |
| --- | --- | --- | --- |
| Ад-рес | Содержимое | | Комментарии |
|  | Код | Мнемо-ника |  |
| 5  6  . . .  10  .  .  .  2F  30  31  32  33  34  35  36  37  38  39  3A | 0000  FFE0  F200  4005  4010  3005  F200  4032  F800  3032  0006  C030  F000 | CLA  ADD 5  ?  MOV 5  CLA  ADD 32  INC  MOV 32  ISZ 6  BR 30  HLT | Ячейка, отведенная для накопления результата  Отрицательное число элементов массива  Численные значения элементов массива  Промежуточный результат (ячейка 5) суммируется  с содержимым элемента массива, адрес которого  расположен в адресной части команды, находящейся  в ячейке 32 (сначала этот адрес равен 10, а затем он  увеличивается при каждом прохождении цикла на 1  командами с 34 по 37)  Пересылка в аккумулятор команды, расположенной  в ячейке 32, добавление к ее содержимому 1 и запись  модифицированной команды на старое место  (в ячейку 32)  Наращивание на 1 содержимого счетчика элементов  массива и переход к команде 30 пока оно < 0  Останов ЭВМ |

Модификация команд практически не используется в современных ЭВМ. Для сближения языка команд с алгоритмическими языками и для обеспечения возможности работы с программами, записанными в постоянные запоминающие устройства (откуда можно лишь читать команды), разработаны специальные средства адресации, одним из которых является косвенная адресация.

При использовании косвенной адресации нужно выбрать в памяти ЭВМ какую-либо ячейку (например, 007), записать в нее адрес первого элемента суммируемого массива (адрес 010), заменить в программе табл. 1.5 команду 4010 на команду 4807 (ячейка 32) и заменить команды модификации командами вычисления текущего адреса суммируемого элемента массива. Если же вычисление текущего адреса суммируемого элемента выполнять с помощью команды ISZ 7 (не затрагивая содержимого аккумулятора), то можно получить достаточно компактную программу, приведенную в табл. 1.6.

*Таблица 1.6*

Второй вариант программы суммирования элементов массива

|  |  |  |  |
| --- | --- | --- | --- |
| Ад-рес | Содержимое | | Комментарии |
|  | Код | Мнемо-ника |  |
| 5  6  7  . . .  10  .  .  .  2F  30  31  32  33  34  35  36  37 | 0000  FFE0  0010  F200  4807  0007  F100  0006  C031  3005  F000 | CLA  ADD (7)  ISZ 7  NOP  ISZ 6  BR 31  MOV 5  HLT | Ячейка, отведенная для накопления результата  Отрицательное число элементов массива  Текущий адрес элемента массива  Численные значения элементов массива  Очистка аккумулятора  Суммирование очередного элемента массива  Текущий адрес элемента массива наращивается на 1  Команда “Нет операции”  Наращивание на 1 содержимого счетчика элементов  массива и переход к команде 31, пока оно < 0  Запись результата в ячейку 5  Останов ЭВМ |

*Таблица 1.7*

*Третий вариант программы суммирования элементов массива*

|  |  |  |  |
| --- | --- | --- | --- |
| Ад-рес | Содержимое | | Комментарии |
|  | Код | Мнемо-ника |  |
| 5  6  . . .  F  10  .  .  .  2F  30  31  32  33  34  35 | 0000  FFE0  0010  F200  480F  0006  C031  3005  F000 | CLA  ADD (F)  ISZ 6  BR 31  MOV 5  HLT | Ячейка, отведенная для накопления результата  Отрицательное число элементов массива  Текущий адрес элемента массива  Численные значения элементов массива  Очистка аккумулятора  Суммирование очередного элемента массива. Так как сначала в индексную ячейку F помещен адрес первого элемента массива (10), то после первого выполнения данной команды содержимое ячейки F увеличится на 1 и будет указывать на второй элемент массива, после второго выполнения команды – на третий элемент массива и т.д.  Наращивание на 1 содержимого счетчика элементов  массива и переход к команде 31, пока оно < 0  Запись результата в ячейку 5  Останов ЭВМ |

Так как по команде ISZ 7 (табл. 1.6) производится наращивание положительной величины (адреса), то после ее выполнения счетчик команд будет указывать на команду 34 (команда по адресу 33 будет пропущена). Поэтому в ячейку 33 помещена команда "Нет операции", но можно было бы поместить даже число.

Наконец, рассмотрим еще одно средство: позволяющее упростить циклические программы базовой ЭВМ, - индексные ячейки (ячейки с адресами от 008 до 00F). Если произвести косвенное адресование какой-либо из этих ячеек, то сначала ее содержимое будет использовано в качестве адреса операнда, а затем оно автоматически увеличится на единицу. При прямом адресовании индексные ячейки (их содержимое может измениться лишь в случае записи информации в ячейку). Указанное свойство индексных ячеек позволяет составить оптимальную программу для суммирования элементов массива (табл. 1.7).

Побитовая обработка данных обеспечивается базовой ЭВМ командами логического умножения, циклических сдвигов, а также командами инвертирования и очистки регистра переноса.

Команда AND M (Логическое умножение) выполняет над каждым разрядом содержимого аккумулятора и содержимым ячейки М булеву операцию "&" ("И").

Результат выполнения команды для каждой пары битов операндов равен единице только тогда, когда оба бита равны единице, а в остальных случаях бит результата равен нулю, т.е. команда позволяет выделять или очищать определенные биты слова.

Команды ROL (циклический сдвиг влево на один разряд) и ROR (циклический сдвиг вправо на один разряд) замыкают аккумулятор и регистр переноса в кольцо и сдвигают все биты кольца на один разряд влево или вправо (рис.1.3). Сдвигами числа влево или вправо

|  |  |  |
| --- | --- | --- |
|  | Регистр |  |
|  | переноса | Аккумулятор |
| До сдвига | 0 | 1011100000101011 |
| а) После сдвига | 1 | 0111000001010110 |
|  |  |  |
| До сдвига | 0 | 1011100000101011 |
| б) После сдвига | 1 | 0101110000010101 |

*Рис. 1.3. Циклические сдвиги: а - влево, б - вправо*

можно реализовать операции умножения или деления на два (один сдвиг), на четыре (два сдвига), на восемь (три сдвига) и т.д.

## 1.6 Подпрограммы

Достаточно часто встречаются ситуации, когда отдельные части программы должны выполнить одни и те же действия по обработке данных (например, вычисление тригонометрической функции). В подобных случаях повторяющиеся части программы выделяют в подпрограмму, а в соответствующие места программы заносят лишь команды обращения к этой подпрограмме. В базовой ЭВМ для этой цели используется команда JSR (Обращение к подпрограмме). Ниже показана часть основной программы, содержащая две команды JSR 300, с помощью которых осуществляется переход к выполнению команд подпрограммы.

По команде JSR 300, расположенной в ячейке 25, выполняется запись числа 25 + 1 = 26 (текущего значения счетчика команд) в ячейку с адресом 300 и запись числа 300 + 1 = 301 в счетчик команд (адрес первой команды подпрограммы). Таким образом осуществляется переход к выполнению команд подпрограммы. Далее начинается процесс выполнения команд подпрограммы, который завершается командой BR (300), расположенной в ячейке 326. Эта команда безусловного перехода с косвенной адресацией предписывает ЭВМ выполнить переход к команде, расположенной по адресу, сохраняемому в ячейке 300. Так как в эту ячейку ранее было записано число 26, то будет исполняться команда, находящаяся в ячейке 26, т.е. следующая за обращением к подпрограмме. Аналогично выполняется команда JSR 300, расположенная в ячейке 72 ( после выполнения команд подпрограммы будет выполнен переход к ячейке 73).

Таким образом, при оформлении подпрограммы перед ее первой командой следует разместить ячейку, в которую будет пересылаться адрес возврата из подпрограммы. В команде обращения к подпрограмме указывается адрес именно этой ячейки (например, адрес М в команде JSR M). Последней командой подпрограммы должна быть команда выхода (команда BR (M) для подпрограммы, размещенной начиная с ячейки М). По ней осуществляется переход к команде, адрес которой сохраняется в первой ячейке подпрограммы.

Обращение к

подпрограмме,

расположенной в

ячейках памяти

с 300 по 326

24

25

26

71

72

73

JSR 300

. . .

JSR 300

ОСНОВНАЯ ПРОГРАММА

Первая команда подпрограммы

Команда выхода из подпрограммы

Ячейка, используемая для размещения

адреса возврата из подпрограммы

300

301

326

. . .

BR (300)

ПОДПРОГРАММА

Обращение к

подпрограмме,

расположенной в

ячейках памяти

с 300 по 326

24

25

26

71

72

73

JSR 300

. . .

JSR 300

ОСНОВНАЯ ПРОГРАММА

Первая команда подпрограммы

Команда выхода из подпрограммы

Ячейка, используемая для размещения

адреса возврата из подпрограммы

300

301

326

. . .

BR (300)

ПОДПРОГРАММА

## 1.7 Выполнение машинных команд

В процессе исполнения команд устройство управления ЭВМ производит анализ и пересылку команд, отдельных ее частей (кода операции, признака адресации и адреса) или операнда из одного регистра ЭВМ в другой ее регистр, АЛУ, память или устройство ввода-вывода. Эти действия (микрооперации) протекают в определенной временной последовательности и скоординированы между собой. Для обеспечения такой последовательности в ЭВМ используется генератор тактовых импульсов.

Цикл команды. Для реализации одной команды требуется выполнить определенное количество микрокоманд, каждая из которых инициируется одним тактовым импульсом. Общее число тактовых импульсов, требуемых для выполнения команды, определяет время ее выполнения, называемое *циклом команды*. Цикл команды обычно включает один или несколько машинных циклов. Устройство управления базовой ЭВМ может находиться в четырех возможных состояниях: выборки команды, выборки адреса, исполнения и прерывания. Длительность каждого из этих четырех состояний определяет время выполнения соответствующего машинного цикла. Основные действия, выполняемые ЭВМ во время каждого из машинных циклов, описаны ниже и проиллюстрированы на рис. 1.4.

Выборка команд. В данном машинном цикле выполняется чтение команды из памяти и ее частичное декодирование.

1. Содержание ячейки памяти, на которую указывает счетчик команд, читается из памяти в регистр данных (рис. 1.4., а, б).
2. Содержимое счетчика команд увеличивается на 1 (рис. 1.4, б, в).
3. Содержимое регистра данных пересылается в регистр команд, код операции команды частично декодируется для выявления типа команды (адресная, безадресная или ввода-вывода), анализируя бит признака адресации и происходит подготовка цепей, необходимых для выполнения команды (рис. 1.4, г).

*Безадресные команды и команды ввода-вывода* окончательно исполняются в этом же цикле, т.е. это одноцикловые команды.

1. Выполняются действия по завершению одноцикловой команды. Выборка адреса. Этот машинный цикл следует за циклом выборки команды *для адресных команд с косвенной адресацией* (бит вида адресации равен 1). Цикл используется для чтения из памяти адреса операнда, результата или перехода и состоит из следующих шагов.
2. Адресная часть команды пересылается из регистра данных, где пока еще сохраняется копия команды, в регистр адреса.
3. Содержимое ячейки памяти, указываемой регистром адреса, читается в регистр данных. Теперь в этом регистре находится либо адрес операнда, либо адрес результата, либо адрес перехода, который будет использоваться в цикле исполнения команды. Если косвенно адресуется одна из индексных ячеек (адреса 8...F), то цикл выборки адреса операнда (результата) продолжается.
4. Содержимое регистра данных увеличивается на единицу.
5. Измененное содержимое регистра данных пересылается в ячейку памяти по адресу, указанному регистром адреса.
6. Содержимое регистра данных уменьшается на единицу.

После этой операции в регистре данных восстанавливается значение адреса, находившегося в индексной ячейке до выполнения шага 3. Содержимое же индексной ячейки увеличилось на 1 и при следующем обращении к ней будет выбран новый адрес операнда (результата).

Исполнение. Последовательность действий, выполняемых в этом цикле, определяется типом выполняемой адресной команды.

1. Для команд, при выполнении которых требуется выборка операнда из памяти ЭВМ (AND, ADD, ADC, SUB, ISZ), цикл исполнения используется для чтения операнда в регистр данных и выполнения операции, указываемой кодом операции команды. Пример цикла исполнения команды ADD 21 приведен на рис. 1.4, д, е, ж, з.
2. По команде пересылки (MOV) в этом машинном цикле производится запись содержимого аккумулятора в ячейку памяти с адресом, расположенным в регистре данных. Для этого содержимое регистра данных пересылается в регистр адреса, а содержимое аккумулятора - в регистр данных и далее в ячейку памяти, указываемую регистром адреса.
3. При исполнении команд переходов (BCS, BPL, BMI, BEQ) производится проверка соответствующего условия (1 - в регистре переноса, 0 - в знаковом разряде аккумулятора и т.п.) и пересылка адреса из регистра данных в счетчик команд при выполнении этого условия. Иначе будет выбрана команда, расположенная вслед за командой перехода. При исполнении команды безусловного перехода (BR) пересылка адреса перехода в счетчик команд выполняется без какой-либо проверки.
4. Для команды обращения к подпрограмме (JSR) во время этого машинного цикла осуществляется пересылка содержимого счетчика команд в ячейку памяти, адрес которой содержится в регистре данных, и занесение в счетчик команд увеличенного на единицу содержимого регистра данных.

АЛУ

21

д

20

22

21

23

24

25

26

***Память***

106

0

CLA

ADD 20

ADD 21

MOV22

53

РА

21

РД

ADD 21

СК

26

53

А

С

РК

ADD 21

УУ

***Процессор***

е

20

22

21

23

24

25

26

***Память***

106

0

CLA

ADD 20

ADD 21

MOV22

53

РА

21

РД

106

СК

26

53

А

С

АЛУ

РК

ADD 21

УУ

***Процессор***

20

22

21

23

24

25

26

***Память***

106

0

CLA

ADD 20

ADD 21

MOV22

53

ж

РА

21

РД

106

СК

26

53

А

С

АЛУ

53+106=159

РК

ADD 21

УУ

***Процессор***

20

22

21

23

24

25

26

***Память***

106

0

CLA

ADD 20

ADD 21

MOV22

53

з

РА

21

РД

106

СК

26

159

А

С

АЛУ

РК

ADD 21

УУ

***Процессор***

а

АЛУ

25

20

22

21

23

24

25

26

***Память***

106

0

CLA

ADD 20

ADD 21

MOV22

53

РА

53

РД

53

СК

25

53

А

С

РК

ADD 20

УУ

***Процессор***

б

20

22

21

23

24

25

26

***Память***

106

0

CLA

ADD 20

ADD 21

MOV22

53

РА

25

РД

ADD 21

СК

25

53

А

С

АЛУ

25+1=26

РК

ADD 20

УУ

***Процессор***

20

22

21

23

24

25

26

***Память***

106

0

CLA

ADD 20

ADD 21

MOV22

53

в

РА

25

РД

ADD 21

СК

26

53

А

С

АЛУ

26

РК

ADD 20

УУ

***Процессор***

20

22

21

23

24

25

26

***Память***

106

0

CLA

ADD 20

ADD 21

MOV22

53

г

РА

25

РД

ADD 21

СК

26

53

А

С

АЛУ

РК

ADD 21

УУ

***Процессор***

а

АЛУ

25

20

22

21

23

24

25

26

***Память***

106

0

CLA

ADD 20

ADD 21

MOV22

53

РА

53

РД

53

СК

25

53

А

С

РК

ADD 20

УУ

***Процессор***

б

20

22

21

23

24

25

26

***Память***

106

0

CLA

ADD 20

ADD 21

MOV22

53

РА

25

РД

ADD 21

СК

25

53

А

С

АЛУ

25+1=26

РК

ADD 20

УУ

***Процессор***

20

22

21

23

24

25

26

***Память***

106

0

CLA

ADD 20

ADD 21

MOV22

53

в

РА

25

РД

ADD 21

СК

26

53

А

С

АЛУ

26

РК

ADD 20

УУ

***Процессор***

20

22

21

23

24

25

26

***Память***

106

0

CLA

ADD 20

ADD 21

MOV22

53

г

РА

25

РД

ADD 21

СК

26

53

А

С

АЛУ

РК

ADD 21

УУ

***Процессор***

### **Домашнее задание № 1**

***Выполнение арифметических операций с двоичными числами.***

Цель задания - овладеть простейшими навыками перевода чисел в различные системы счисления и выявить ошибки, возникающие из-за их ограниченной разрядности.

1. По заданному варианту исходных данных получить набор десятичных чисел: Х1=А, Х2=С, Х3=А+С, Х4=А+С+С, Х5=С-А, Х6=65536-Х4, Х7= -Х1, Х8= -Х2, Х9= -Х3, Х10= -Х4, Х11= - Х5, Х12= -Х6. Выполнить перевод десятичных чисел Х1,...,Х12 в двоичную систему счисления, получив их двоичные эквиваленты В1,...,В12 соответственно. Для представления двоичных чисел В1,...,В12 использовать 16-разрядный двоичный формат со знаком. Для контроля правильности перевода выполнить обратный перевод двоичных чисел в десятичные и подробно проиллюстрировать последовательность прямого и обратного перевода для чисел Х1, В1, Х7 и В7.
2. Выполнить следующие сложения двоичных чисел: В1+В2, В2+В3, В7+В8, В8+В9, В2+В7, В1+В8. Для представления слагаемых и результатов сложения использовать 16-разрядный двоичный формат со знаком. Результаты сложения перевести в десятичную систему счисления, сравнить с соответствующими десятичными числами. Дать подробные комментарии полученным результатам.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Операнд | Номер варианта | | | | | | |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| А  С | 2006  15452 | 6390  14940 | 4186  15772 | 1818  16924 | 5238  15900 | 2262  16028 | 6582  17436 |
| Операнд | Номер варианта | | | | | | |
|  | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| А  С | 4154  16162 | 2902  18006 | 1722  16988 | 2774  15388 | 5302  14972 | 2294  16064 | 1978  15516 |
| Операнд | Номер варианта | | | | | | |
|  | 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| А  С | 2998  16288 | 6518  15260 | 2678  16160 | 5238  14932 | 4314  15420 | 2422  17500 | 1754  17820 |

### **Домашнее задание № 2**

***Программирование циклических алгоритмов***

Написать комплекс программ, состоящий из программы и подпрограммы и обеспечивающий подсчет количества требуемых элементов массива данных. Программа должна выявлять требуемые элементы, а их подсчет должен производиться в подпрограмме.

Варианты задания: подсчитать количество

1. неотрицательных элементов из CEBA, 0848, 3476, AE05, B0BA;
2. отрицательных элементов из 71BC, ABBA, 63CE, 5826, C748;
3. нулевых элементов из 0000, 0707, 0000, C0AE, 0000;
4. ненулевых элементов из 0000, CBAE, 0707, 000, BACE;
5. положительных элементов из 0000, 0707, BACE, 0000, AE01;
6. отрицательных элементов из 0000, CCCE, 90BA, 0000, EEBB.

### **Лабораторная работа № 1**

***Исследование работы ЭВМ при выполнении линейных программ.***

Цель работы - изучение приемов работы на базовой ЭВМ и исследование порядка выполнения арифметических команд и команд пересылки.

Порядок выполнения работ. Познакомиться с инструкцией по работе с моделью базовой ЭВМ (см. приложение №1), занести в память базовой ЭВМ заданный вариант программы и, выполняя ее по командам, заполнить таблицу трассировки выполненной программы.

*Таблица 1.8*

*Форма таблицы трассировки.*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Выполняемая команда | | Содержимое регистров процессора после выполнения команды. | | | | | | Ячейка, содержим. которой изменилось после вып. Программы | |
| Адрес | Код | СК | РА | РК | РД | А | С | Адрес | Новый код |
| ххх | хххх | хххх | хххх | хххх | хххх | хххх | х | ххх | хххх |

Содержание отчета по работе.

1. Текст исходной программы по следующей форме:

|  |  |  |  |
| --- | --- | --- | --- |
| "Адрес" | "Код команды" | "Мнемоника" | "Комментарии" |
| 21 | 4015 | ADD 15 | (A)+(15)A |

2. Таблица трассировки

3. Описание программы:

- назначение программы и реализуемые ею функции (формулы);

- область представления данных и результатов;

- расположение в памяти ЭВМ программы, исходных данных и результатов;

- адреса первой и последней выполняемой команд программы;

4. Вариант программы с меньшим числом команд.

Варианты программ (первая команда программы помечена знаком "+").

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Адрес | Варианты программ | | | | | |
|  | 1 | 2 | 3 | 4 | 5 | 6 |
| 017  018  019  01A  01B  01C  01D  01E  01F  020  021  022  023  024 | 0000  F1AA  7C89  2A5A  0000  + F200  4018  501A  301B  F200  4019  101B  301B  F000 | 0000  + F200  4021  6022  3024  F200  4023  1024  3024  F000  1377  2295  7C90  301A | + F200  4022  4021  3020  F200  4023  1020  3020  F000  0000  7C89  01AA  A299  0000 | 0000  4017  2009  00F4  + F200  4024  6018  301A  F200  401A  1019  301A  F000  C000 | 0000  4015  4019  + F200  4018  6024  3017  F200  4019  1023  3017  F000  0001  0255 | 0000  0018  + F200  4023  6024  3018  F200  4022  1018  3018  F000  21AA  0255  FC00 |

### **Лабораторная работа № 2**

***Исследование работы ЭВМ при выполнении разветвляющихся программ.***

Цель работы - изучение команд переходов, способов организации разветвляющихся программ и исследование порядка функционирования ЭВМ при выполнении таких программ.

Подготовка к выполнению работы.

1. Восстановить текст заданного варианта программы (см. п.1 лабораторной работы № 1).
2. Заполнить таблицу трассировки, выполняя за базовую ЭВМ заданный вариант программы (теоретическая таблица).
3. Составить описание программы (см. п.3 лабораторной работы №1).

Порядок выполнения работы. Занести в память базовой ЭВМ заданный вариант программы и заполнить таблицу трассировки, выполняя эту программу по командам (экспериментальная таблица).

Содержание отчета по работе. Текст программы с комментариями, две таблицы трассировки ("теоретическая" и "экспериментальная"); описание программы; вариант программы с меньшим числом команд.

Варианты программ (первая команда программы помечена знаком "+").

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Адрес | Варианты программ | | | | | |
|  | 1 | 2 | 3 | 4 | 5 | 6 |
| 016  017  018  019  01A  01B  01C  01D  01E  01F  020  021  022  023 | 0625  0FA7  + F200  4016  4017  9020  F200  3022  F100  F000  3022  C01F  1111  0000 | + C01A  АСАВ  001F  0000  F200  4017  4018  A020  F200  F100  3022  F000  CCCC  0000 | СF0B  F0F5  F000  + F200  4016  4017  B020  F200  3018  F000  4016  3018  C01F  0000 | 0000  + C01B  0018  0019  1000  F200  4019  401A  8022  F200  3018  301A  F000  0000 | 0000  5417  + F200  4022  4023  9020  F200  3017  F100  F000  3017  C01F  FF0F  0031 | 0000  + C01B  001B  FF20  00DF  F200  4019  401A  A021  F200  F100  3023  F000  C008 |

### **Лабораторная работа № 3**

***Исследование работы ЭВМ при выполнении циклических программ.***

Цель работы - изучение способов организации циклических программ и исследование порядка функционирования ЭВМ при выполнении циклических программ.

Подготовка к выполнению работы.

1. Восстановить текст заданного варианта программы.
2. Составить описание программы.

Порядок выполнения работы. Занести в память базовой ЭВМ заданный вариант программы и заполнить таблицу трассировки, выполняя эту программу по командам.

Содержание отчета по работе. Текст программы с комментариями, таблица трассировки; описание программы.

Варианты программ (первая команда программы помечена знаком "+").

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Адрес | Варианты программ | | | | | |
|  | 1 | 2 | 3 | 4 | 5 | 6 |
| 00A  00B  00C  00D  00E  00F  010  011  012  013  014  015  016  017  018  019  01A  01B  01C  01D  01E  01F | 0000  0000  0000  0000  001C  0000  0000  0000  FFFC  + F200  480E  B018  4011  3011  0012  C013  F000  0378  0000  F0EB  0377  0000 | 0000  0000  0000  0000  0000  001C  0000  0000  FFFC  + F200  480F  A018  4011  3011  0012  C013  F000  7F02  DECA  30AE  7F01  0000 | 0000  0000  001B  0000  0000  0000  0000  + F200  480C  9016  401D  301D  0019  C011  F000  FFFC  8778  1777  8788  1111  FFA1  0000 | 0011  0000  0000  0000  0000  0000  3355  71BC  ABBA  63CD  FFFC  0000  + F200  480A  A01D  F200  F800  4015  3015  0014  C016  F000 | 0000  001C  0000  0000  0000  0000  0000  FFFC  + F200  480B  9019  F200  F800  401C  301C  0011  C012  F000  0000  B0B0  5B0B  CF11 | 0000  0000  0000  0010  0000  0000  0000  0010  0000  0707  0000  FFFC  + F200  480D  B01A  C01D  F800  4011  3011  0015  C016  F000 |

### **Лабораторная работа № 4**

***Исследование работы ЭВМ при выполнении комплекса программ.***

Цель работы - изучение способов связи между программными модулями, команды обращения к подпрограмме и исследование порядка функционирования ЭВМ при выполнении комплекса взаимосвязанных программ.

Подготовка к выполнению работ.

1. Восстановить текст заданного варианта программы и подпрограммы (программного комплекса).
2. Составить описание программного комплекса.

Порядок выполнения работы. Занести в память базовой ЭВМ заданный вариант программы и заполнить таблицу трассировки, выполняя эту программу по командам.

Содержание отчета по работе. Текст программы с комментариями, таблица трассировки; описание программы.

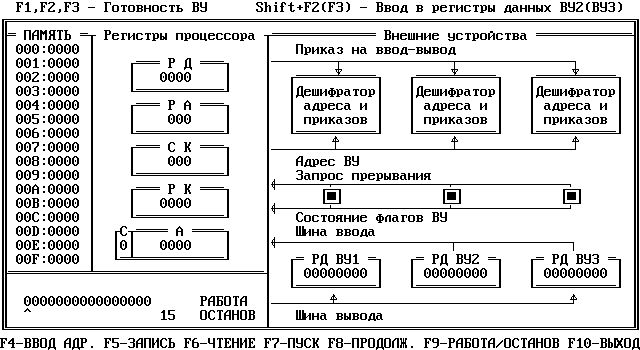
Варианты программ (первая команда программы помечена знаком "+").

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Адрес | Варианты программ | | | | | |
|  | 1 | 2 | 3 | 4 | 5 | 6 |
| 00A  00B  00C  00D  00E  00F  010  011  012  013  014  015  016  017  018  019  01A  01B  01C  01D  . . .  045  046  047  048  049  04A | 0010  0000  0000  0000  0000  0000  8080  ABDA  630D  71B0  FFFC  0000  + F200  480A  A01A  2045  0014  C016  F000  0000  . . .  0000  F200  F800  4015  3015  C845 | 0000  001A  0000  0000  0000  0000  0000  FFFE  + F200  480B  9016  2045  0011  C012  F000  0000  CF01  B0BA  5B1B  0000  . . .  0000  F200  F800  4019  3019  C845 | 0000  0000  0012  0000  0000  0000  0000  FFFD  0000  0707  0000  0000  + F200  480C  B01A  C01B  2045  C011  C016  F000  . . .  0000  F200  F800  4015  3015  C845 | 0000  0000  0000  0019  0000  0000  + F200  480D  B014  2045  0018  C010  F000  0000  FFFD  8018  0000  81FF  0000  0000  . . .  0000  F200  F800  4017  3017  C845 | 0000  0000  0000  0000  0010  0000  0000  6789  CACA  8A7C  FFFC  + F200  480E  A019  2045  0014  C015  F000  0000  0000  . . .  0000  F200  F800  4010  3010  C845 | 0000  0000  0000  0000  0000  0011  F200  4816  F800  + F200  480F  9017  2045  001A  C013  F000  FFFE  0000  0000  0000  . . .  0000  F200  F800  401B  301B  C845 |

# Раздел 2. Организация ввода-вывода в базовой ЭВМ

## 2.1 Устройства ввода-вывода базовой ЭВМ

Модель базовой ЭВМ с устройствами ввода-вывода представлена на рис 2.1. В базовой ЭВМ используются простейшие внешние устройства (ВУ): одно устройство вывода (ВУ-1) и два устройства ввода (ВУ-2 и ВУ-3). В модели устройства ввода-вывода представлены 8-разрядными регистрами данных (РД ВУ). Через регистры данных ВУ-2 и ВУ-3 информация может быть введена в базовую ЭВМ, а в регистр данных ВУ-1 принята из базовой ЭВМ.



*Рис. 2.1. Модель базовой ЭВМ с устройствами ввода-вывода*

Между ВУ и процессором включены простейшие контроллеры, каждый из которых содержит: *дешифратор адреса*, позволяющий выделить обращение к данному ВУ среди всех обращений к устройствам ввода-вывода, подключенных к процессору; *дешифратор приказов*, декодирующий приказы от процессора на выполнение тех или иных операций; *регистр состояния*, в котором хранится информация о готовности ВУ к обмену данными с процессором. В контроллерах простейших ВУ обычно используются однобитовые регистры готовности, которые часто называют *флагом* или *флажком*. Это название используется и в контроллерах базовой ЭВМ. Контроллеры ВУ связаны с процессором шинами ввода и вывода информации, шиной адреса и шиной управления, по которым передаются приказы от процессора и сведения о состояния ВУ.

## 2.2 Программно-управляемая передача данных.

При использовании программно-управляемого обмена должна быть составлена программа, обеспечивающая пересылку данных из памяти ЭВМ в аккумулятор и далее в регистр памяти контроллера ВУ (вывод данных) или из регистра данных контроллера ВУ в аккумулятор и затем в память ЭВМ (ввод данных). В такое программе можно реализовать один из трех типов обмена: синхронный, асинхронный и по прерыванию. Синхронный обмен очень редко используется в ЭВМ и не будет рассматриваться в данном пособии, остальные виды обмена рассматриваются в п.п. 2.4 и 2.5.

Формат команд ввода-вывода приведен на рис. 1.2.в. Код операции (1110)2 служит для отличия этих команд от других команд ЭВМ. Между собой они отличаются кодом приказа: пересылка данных (IN В - ввод и OUT В - вывод), *проверка готовности ВУ* (TSF B) и *сброс состояния готовности* (CLF B), где В - адрес ВУ. Адрес позволяет связать процессор с одним из подключенных к нему ВУ (их может быть до 28=256).

*Флажок* - однобитовый регистр готовности ВУ, устанавливаемый в единичное состояние, когда ВУ готово к обмену информацией. Если флажок сброшен (установлен в ноль), ВУ занято: устройство вывода еще обрабатывает предыдущую команду, а устройство ввода готовит данные для передачи в процессор.

Команда CLF B (E0xx, где хх - две последние 16-ричные цифры адреса ВУ) служит для установки в нуль флажка ВУ с адресом В.

Команда TSF B (E1xx) служит для проверки готовности к обмену ВУ с адресом В. Если флажок этого ВУ сброшен (ВУ не готово к обмену), то выполняется команда, расположенная вслед за TSF В. В противном случае эта команда *пропускается* и выполняется команда, расположенная через одну за TSF В.

Команда IN B (E2хх) служит для пересылки содержимого регистра данных контроллера ВУ с адресом B в восемь младших разрядов аккумулятора.

Команда OUT B (E3хх) служит для пересылки содержимого восьми младших разрядов аккумулятора в регистр данных контроллера ВУ с адресов В.

Для организации обмена с ВУ в состав устройства управления базовой ЭВМ включены два устройства: регистр состояний внешних устройств (Ф) и контроллер прерываний. Связь контроллеров ВУ с этими устройствами осуществляется по линиям "Состояние флага" и "Запрос прерывания". Данные передаются по шинам ввода и вывода.

## 2.3 Асинхронный обмен.

Программа такого обмена строится так: сначала проверяется готовность ВУ к обмену и если оно готово, то дается команда на обмен. ВУ сообщает о готовности установкой флага.

Пример 2.1 С помощью ВУ-2 записать в ячейку 006 коды символов слова "ДА".

Программа для выполнения этого задания имеет вид:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Содержимое | |  |
| Адрес | Код | Мнемоника | Комментарии |
| 05 | FFF8 |  | Константа -8, используемая для сдвига |
| 06 | 0000 |  | Ячейка для записи слова "ДА" |
| ........ | ........ |  |  |
| 20 | E102 | TSF 2 | Опрос флага контроллера ВУ-2 и повторение этой операции: если ВУ-1 не готовок обмену (флаг=0) |
| 21 | C020 | BR 20 |  |
| 22 | E202 | IN 2 | Это действие выполняется лишь после готовности ВУ-2, т.е. когда при выполнении TSF 2 выяснятся, что флаг=1 и пропускается BR 20, По команде IN 2 содержимое регистра данных контроллера ВУ-2 пересылается в восемь младших разрядов аккумуляторов. |
| 23 | E002 | CLF 2 | Сброс готовности ВУ-2 (очистка флага ВУ-2) |
| 24 | F600 | ROL | Код первого введенного символа сдвигается на восемь разрядов влево и освобождается место для ввода следующего символа. |
| 25 | 0005 | ISZ 5 |  |
| 26 | C024 | BR 24 |  |
| 27 | E102 | TSF 2 | Опрос флага контроллера ВУ-2 ..... (см. Комментарии к командам 20 и 21) |
| 28 | C027 | BR 27 |  |
| 29 | E202 | IN 2 | Ввод кода символа (если подан сигнал готовности ВУ-2) |
| 2А | E002 | CLF 2 | Сброс готовности ВУ-2 |
| 2В | 3006 | MOV 6 | Пересылка кода слова "ДА" в ячейку 006 |
| 2С | F000 | HLT | Останов ЭВМ |

Две первые команды этой программы "заставляют" ЭВМ ожидать его готовности ВУ-2 к выдаче данных. Поэтому необходимо ввести код символа "Д" в регистр данных ВУ-2. После сброса готовности ВУ-2 (команда CLF 2), которая подтверждает, что данные из регистра данных контроллера ВУ-2 переписаны в аккумулятор можно приступить к набору кода символа "А". В процессе набора этого кода ЭВМ занята сдвигом кода символа "Д" в старшие разряды аккумулятора, чтобы подготовиться к приему символа "А", и ожиданием поступления нового сигнала готовности ВУ-2 к выдаче информации. Так как ЭВМ выполняет эти операции значительно быстрее, чем человек, набирающий код нового символа. Теперь в аккумулятор перепишется все слово "ДА", затем оно перепишется в ячейку 006 и выполнение программы прекратиться.

Легко заметить, что при асинхронном обмене ЭВМ должна тратить время на ожидание момента готовности, а так как готовность проверяется командным путем (команда TSF), то в это время ЭВМ не может выполнять никакой другой работы по преобразованию данных.

## 2.4 Обмен по прерыванию программы.

Этот вид обмена отличается от асинхронного тем, что сигнал готовности ВУ к обмену анализируется не программным, а аппаратным путем. ЭВМ может выполнять любую не связанную с обменом программу (будем называть ее основной), а когда из ВУ по линии "Запрос прерывания" (рис. 1.1) поступит сигнал готовности ВУ к приему или выдаче информации, прервать (приостановить) выполнение этой программы на время выполнения программы обмена данными. Все эти действия осуществляются с помощью контроллера прерываний, входящего в состав устройства управления базовой ЭВМ.

Команды EI (Разрешение прерывания) и DI (Запрещение прерывания) переводят контроллер прерываний в одно из двух состояний, в которых он соответственно реагирует или не реагирует на сигналы готовности ВУ, передаваемые по линии "Запрос прерывания". Если контроллер прерываний установлен в состояние разрешения прерывания, то выполняются следующие действия.

*Шаг 1*. По завершению выполнения текущей команды основной программы управление передается контроллеру прерываний. Если в этот момент на линии "Запрос прерывания" нет сигнала о готовности какого-либо ВУ, то начинается выборка и исполнение следующей команды основной программы и данный шаг повторяется. При наличии запроса прерывания выполняется второй шаг.

*Шаг 2.* Контроллер прерываний переходит в состояние запрещения прерывания, в ячейку памяти с адресом 000 заносится содержимое СК (адрес следующей команды основной программы, которая выполнялась бы при отсутствии запроса прерывания), и управление передается команде расположенной в ячейке 001. Так происходит переход к подпрограмме обработки прерывания (с первой командой в ячейке 001), функции которой определяются содержанием следующих шагов.

*Шаг 3.* Производится запоминание в памяти содержимого аккумулятора и регистра переноса. Для этого требуется минимум три команды: пересылка содержимого аккумулятора в специально отведенную буферную ячейку (например, В1), циклический сдвиг содержимого аккумулятора влево (для того, чтобы содержимое регистра переноса попало в аккумулятор) и запись этого содержимого в другую буферную ячейку (например, В2). Таким образом, необходимый минимум информации о прерванной программе сохраняется - в ячейке 000 хранится адрес продолжения прерванной программы, а в ячейках В1 и В2 хранится содержимое двух других основных регистров А и С.

*Шаг 4.* Производится поиск источника прерывания. Для этого в любой, наиболее целесообразной, последовательности опрашиваются флаги всех ВУ (команда TSF). При обнаружении ВУ с установленным флагом (флаг=1) выполняется переход к тому участку подпрограммы, в котором описаны действия по обмену данными с этим ВУ.

*Шаг 5.* Выполняется передача данных и их предварительная обработка, если это необходимо.

*Шаг 6.* Восстанавливается содержимое регистра переноса и аккумулятора. Для этого требуется минимум пять команд: очистка аккумулятора, вызов содержимого ячейки В2 в очищенный аккумулятор, циклический сдвиг вправо для восстановления содержимого регистра переноса , очистка аккумулятора и вызов содержимого буферной ячейки В1 в очищенный аккумулятор.

*Шаг 7.* Контроллер прерываний вновь переводится в состояние разрешение прерывания (команда ЕI) и осуществляется возврат к выполнению прерванной программы, т.е. к команде, адрес которой хранится в ячейке 000 (команда BR (0)). Здесь следует отметить, что команда BR () должна располагаться непосредственно за командой ЕI. Иначе при появлении во время обработки прерывания будет стерт (заменен на новый) адрес возврата и путь возврата к прерванной программе будет разрушен.

Пример 2.2 Составить программу, которая периодически (с периодом в три цикла команды) наращивает на 1 содержимое аккумулятора. Восемь младших разрядов аккумулятора должны выводиться на ВУ-1 по его запросу, а по запросу ВУ-3 код, записанный в регистр данных контроллера ВУ-3, должен помещаться в ячейку 25.

Основная программа решения задачи примера 2.2

|  |  |  |  |
| --- | --- | --- | --- |
|  | Содержимое | |  |
| Адрес | Код | Мнемоника | Комментарии |
| 20 | FA00 | EI | Установка состояния разрешения прерывания |
| 21 | F200 | CLA | Очистка аккумулятора |
| 22 | F800 | INC | Цикл для наращивания содержимого аккумулятора |
| 23 | F100 | NOP |  |
| 24 | C022 | BR 22 |  |
| 25 | 0000 |  | Ячейка для хранения кодов, поступающих с ВУ-1 |

Подпрограмма обработки прерываний для примера 2.2

|  |  |  |  |
| --- | --- | --- | --- |
|  | Содержимое | |  |
| Адрес | Код | Мнемоника | Комментарии |
| 00 |  |  | Ячейка для хранения адреса возврата (этот адрес будет занесен сюда на 2-м шаге) |
| 01 | C030 | BR 30 | Первая команда подпрограммы - переход к основному ее тексту, размещенному в ячейках 30-4С |
| .......... | ......... |  |  |
| 30 | 304B | MOV 4B | Сохранение в буферных ячейках 4В и 4С содержимого аккумулятора и регистра переноса **(ШАГ 3)** |
| 31 | F600 | ROL |  |
| 32 | 304C | MOV 4C |  |
| 33 | E103 | TSF 3 | Опрос флага ВУ-3. Если он сброшен, то переход к опросу флага ВУ-1. В противном случае переход на ввод данных из ВУ-3 |
| 34 | C036 | BR 36 |  |
| 35 | C039 | BR 39 |  |
| 36 | E101 | TSF 1 | Опрос флага ВУ-1. Если он сброшен, то переход к сбросу флага ВУ-2. В противном случае переход на вывод данных в ВУ-1. |
| 37 | C043 | BR 43 |  |
| 38 | C03E | BR 3E |  |
| 39 | F200 | CLA | Ввод данных из ВУ-3, пересылка их в ячейку 25, сброс флага ВУ-3, переход к восстановлению содержимого основных регистров и выходу из подпрограммы |
| 3A | E203 | IN 3 |  |
| 3B | E003 | CLF 3 |  |
| 3C | 3025 | MOV 25 |  |
| 3D | C044 | BR 44 |  |
| 3E | F200 | CLA | Пересылка в аккумулятор содержимого буферной ячейки 4В, вывод на ВУ-1 восьми младших разрядов аккумулятора, сброс флага ВУ-1, переход восстановлению А и С и выходу из подпрограммы. |
| 3F | 404B | ADD 4B |  |
| 40 | E301 | OUT 1 |  |
| 41 | E001 | CLF 1 |  |
| 42 | C044 | BR 44 |  |
| 43 | E002 | CLF 2 | Очистка флага ВУ-2 **(ШАГ 5)** |
| 44 | F200 | CLA | Восстановление содержимого регистра переноса и аккумулятора **(ШАГ 6)** |
| 45 | 404C | ADD 4C |  |
| 46 | F700 | ROR |  |
| 47 | F200 | CLA |  |
| 48 | 404B | ADD 4B |  |
| 49 | FA00 | EI | Возобновление состояния разрешения прерывания и выход из подпрограммы **(ШАГ 7)** |
| 4A | C800 | BR (0) |  |
| 4B | 0000 |  | Ячейки для сохранения содержимого аккумулятора и регистра переноса |
| 4C | 0000 |  |  |

Если команды этой программы занести в память базовой ЭВМ, установить в СК пусковой адрес 20 и нажать кнопку ПУСК, то начнет выполняться бесконечный цикл наращивания содержимого аккумулятора. Когда же на пульте управления (рис 1.1) будет нажата любая из трех кнопок ("Готов" ВУ1, ВУ2 или ВУ3), то будет выполнен переход к подпрограмме обработки прерываний. Она может быть построена по стандартной схеме (как в таблице) или в другой форме, учитывающий конкретные особенности реализуемой задачи.

### **Домашнее задание № 3**

***Программирование обмена данными с внешними устройствами***

Написать комплекс программ, обеспечивающий обмен данными с ВУ в режиме прерывания программы. Основная программа должна наращивать на 1 (начиная с 0) содержимое (обозначим его буквой Х) какой-либо ячейки памяти. Цикл для наращивания Х не должен содержать более трех команд. Вывод всегда осуществляется на ВУ-3 в асинхронном режиме. Выводится только восемь младших разрядов результата.

Варианты задания:

1. По запросу ВУ-1 вывести -2Х+5, а по запросу ВУ-2 вывести 3Х/4.
2. По запросу ВУ-3 вывести (3Х-2)/2, а по запросу ВУ-2 вывести Х/2+10.
3. По запросу ВУ-2 вывести (Х/2)+5, а по запросу ВУ-1 вывести –(5Х/2)+1.
4. По запросу ВУ-3 вывести –(Х+1)/4, а по запросу ВУ-1 вывести (2Х+3)/2.
5. По запросу ВУ-2 вывести (3Х+3)/8, а по запросу ВУ-1 вывести –(5Х+7)/2.
6. По запросу ВУ-1 вывести (5Х+1)/2, а по запросу ВУ-3 вывести (Х/2)-6.

Составить методику проверки правильности выполнения разработанного комплекса на базовой ЭВМ, т.е. написать последовательность действий оператора (пользователя) базовой ЭВМ, которые необходимо выполнить, чтобы проверить все возможные режимы работы комплекса программ (при появлении запроса прерывания от любого ВУ) и получить заданное количество результатов.

Пример. Начальный фрагмент методики проверки

1. Загрузить комплекс программ в память базовой ЭВМ.
2. Запустить основную программу в автоматическом режиме с адреса XXX.
3. Установить "Готовность ВУ-3".
4. После сброса "Готовность ВУ-3", что означает ... (указать конкретно что именно), сделать следующее (указать что именно) и т.д. .

### **Лабораторная работа № 5**

***Исследование работы ЭВМ при асинхронном обмене данными с ВУ***

Цель работы - изучение организации системы ввода-вывода базовой ЭВМ, команд ввода-вывода и исследование процесса функционирования ЭВМ при обмене данными по сигналам готовности внешних устройств.

Подготовка к выполнению работы.

Закодировать заданную программу и составить ее описание. Команды программы надо разместить, начиная с ячейки 10, а коды символов - начиная с ячейки 20.

Порядок выполнения работы

1. Занести программу в память базовой ЭВМ.
2. Перевести ЭВМ в режим автоматического выполнения программы и ввести в память четыре первых символов заданного слова.
3. Перевести ЭВМ в режим покомандного выполнения программы и ввести в ее память еще два символа заданного слова, заполняя таблицу трассировки.

Содержание отчета по работе. Текст программы, заданное слово и коды его символов, таблица с результатами трассировки и описание программы.

Исходные данные к лабораторной работе

1. Программа асинхронного обмена данными

|  |  |  |
| --- | --- | --- |
| Адрес | Мнемоника | Комментарии |
| A: | TSF 1 | Опрос флага ВУ-1 и повторение этой операции, если ВУ-1 не готово к обмену (флаг=0) |
|  | BR A |  |
|  | IN 1 | Ввод данных в аккумулятора, если флаг=1 |
|  | CLF 1 | Сброс флага ВУ-1 |
|  | MOV (B) | Пересылка содержимого аккумулятора в память и увеличение на 1 адреса элемента массива (В=В+1) |
|  | ISZ C | Наращивание на 1 содержимого счетчика элементов массива и переход по адресу А, пока оно < 0. |
|  | BR A |  |
|  | HLT | Останов ЭВМ |

*Примечание*. Здесь А, В, С - адреса начала программы, ячейки с начальным адресом массива (любая индексная ячейка) и ячейки содержащей счетчик количества еще не введенных символов.

2. Варианты вводимых слов:

1) КРЕМЕНЬ; 2) КАМЕНЬ; 3) МАРШРУТ; 4) ПРОПАН; 5) ПРОРУБЬ; 6) ТРЕСК.

1. Коды используемых символов

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Символ** | **А** | **Б** | **Д** | **Е** | **И** | **Й** | **К** | **М** | **Н** | **О** | **П** | **Р** | **У** | **Т** | **Ч** | **Ш** | **Ь** | **С** | **Я** |
| **Код** | E1 | E2 | E4 | E5 | E9 | EA | EC | ED | EE | EF | F0 | F2 | F3 | F4 | FE | FB | F8 | F3 | F1 |

### **Лабораторная работа № 6**

***Исследование работы ЭВМ при обмене данными с ВУ***

***в режиме прерывания программы.***

Цель работы - изучение организации процесса прерывания программы и исследования порядка функционирования ЭВМ при обмене данными в режиме прерывания программы. Работа является практический проверкой домашнего задания №3.

Подготовка к выполнению работы. Выполнить домашнее задание №3.

Порядок выполнения работы. Используя методику проверки разработанной программы, получит три пары результатов, указывая для каждого выведенного значения величину Х. Результаты работы программного комплекса представить в виде таблицы.

Содержание отчета по работе. Домашнее задание №3, таблицу с результатами работы комплекса программ.

# РАЗДЕЛ 3. МИКРОПРОГРАММНОЕ УСТРОЙСТВО УПРАВЛЕНИЯ

## 3.1. Микропрограммное управление вентильными схемами.

Процесс выборки, дешифрации и исполнения команд ЭВМ состоит из последовательности элементарных операций (например, пересылка содержимого одного регистра в другой регистр или проверка определенного бита в каком-либо регистре). Для выполнения таких микроопераций, как правило, достаточно подать открывающий сигнал на одну или несколько вентильных схем, связывающих между собой два регистра, регистр и АЛУ и (или) перестраивающих АЛУ на выполнение заданной операции (сложения, логического умножения и т.п.). Требуемая последовательность сигналов на вентильные схемы ЭВМ вырабатывается ее устройством управления, связанным с тактовым генератором.

Микропрограммное устройство управления (МПУ) базовой ЭВМ - это, в свою очередь, очень простая ЭВМ, для которой регистры и вентильные схемы процессора являются как бы устройствами ввода-вывода (рис. 3.1).

Операционная микрокоманда 0 (ОМК0)

15

14

13

12

11

10

9

8

7

6

5

4

3

2

1

0

0/1

0/1

0/1

0/1

xxx

xxx

0/1

0/1

0

0

0/1

0/1

0/1

0/1

0/1

0/1

Код

операц.

Левый вход

Правый

вход

Обрат-ный код

Опера-ция

Сдвиги

Память

0-0, 1-А,

2-РС, 3-КР

0-0, 1-РД,

2-РК, 3-СК

0-лев.вх+пр.вх., 1-лев.вх+пр.вх.+1, 2- лев.вх&пр.вх.

0-нет обмена, 1-чтение, 2-запись

0-не сдвигать, 1-сдвиг вправо,2-сдвиг влево

0-не вычислять, 1-левый вход, 2-правый вход

Операционная микрокоманда 1 (ОМК1)

15

14

13

12

11

10

9

8

7

6

5

4

3

2

1

0

0/1

0/1

0/1

0/1

0/1

0/1

xxx

xxx

1

0

0/1

0/1

0/1

0/1

0/1

0/1

Код

операц.

Управление

обменом с ВУ

Регистр

С

N

Выход АЛУ

(содерж. БР)

0-не измен., 1-перенос,2-сброс, 3-установ.

0-не изменять, 1-записать результат

0-не изменять состояние, 1-остановить ЭВМ

Z

0-не пересылать, 1-в РА, 2-в РД, 3-в РК, 4-в СК,

5-в АКК, 7-переслать в РА,РД,РК и АКК.

Управляющая микрокоманда (УМК)

15

14

13

12

11

10

9

8

7

6

5

4

3

2

1

0

0/1

0/1

0/1

0/1

0/1

0/1

0/1

0/1

0/1

1

0/1

0/1

0/1

0/1

0/1

0/1

Код

оп.

Поле выбора

проверяемого бита

Адрес перехода

однобитовое поле сравнения

Поле выбора проверяемого регистра: 0-РС, 1-РД, 2-РК, 3-АК

Операционная микрокоманда 0 (ОМК0)

15

14

13

12

11

10

9

8

7

6

5

4

3

2

1

0

0/1

0/1

0/1

0/1

xxx

xxx

0/1

0/1

0

0

0/1

0/1

0/1

0/1

0/1

0/1

Код

операц.

Левый вход

Правый

вход

Обрат-ный код

Опера-ция

Сдвиги

Память

0-0, 1-А,

2-РС, 3-КР

0-0, 1-РД,

2-РК, 3-СК

0-лев.вх+пр.вх., 1-лев.вх+пр.вх.+1, 2- лев.вх&пр.вх.

0-нет обмена, 1-чтение, 2-запись

0-не сдвигать, 1-сдвиг вправо,2-сдвиг влево

0-не вычислять, 1-левый вход, 2-правый вход

Операционная микрокоманда 1 (ОМК1)

15

14

13

12

11

10

9

8

7

6

5

4

3

2

1

0

0/1

0/1

0/1

0/1

0/1

0/1

xxx

xxx

1

0

0/1

0/1

0/1

0/1

0/1

0/1

Код

операц.

Управление

обменом с ВУ

Регистр

С

N

Выход АЛУ

(содерж. БР)

0-не измен., 1-перенос,2-сброс, 3-установ.

0-не изменять, 1-записать результат

0-не изменять состояние, 1-остановить ЭВМ

Z

0-не пересылать, 1-в РА, 2-в РД, 3-в РК, 4-в СК,

5-в АКК, 7-переслать в РА,РД,РК и АКК.

Управляющая микрокоманда (УМК)

15

14

13

12

11

10

9

8

7

6

5

4

3

2

1

0

0/1

0/1

0/1

0/1

0/1

0/1

0/1

0/1

0/1

1

0/1

0/1

0/1

0/1

0/1

0/1

Код

оп.

Поле выбора

проверяемого бита

Адрес перехода

однобитовое поле сравнения

Поле выбора проверяемого регистра: 0-РС, 1-РД, 2-РК, 3-АК

Программа работы такой ЭВМ называется микропрограммой, а ее команды, содержащие информацию об элементарных действиях, выполняемых в течение одного рабочего такта ЭВМ, - микрокомандами.

В одном из вариантов реализации МПУ используется всего два типа микрокоманд - операционная и управляющая:

0

Код

операции

31

28

24

20

16

12

8

4

0

Биты управления отдельными вентильными схемами

Операционная микрокоманда (ОМК)

РК

РС

РД

1

А

Код

операции

31

28

24

25

23

16

15

0

Управляющая микрокоманда (УМК)

Поле выбора проверяемого регистра (А,РК,РД,РС)

Однобитное поле сравнения

Адрес перехода

Поле выбора проверяемого бита

0

Код

операции

31

28

24

20

16

12

8

4

0

Биты управления отдельными вентильными схемами

Операционная микрокоманда (ОМК)

РК

РС

РД

1

А

Код

операции

31

28

24

25

23

16

15

0

Управляющая микрокоманда (УМК)

Поле выбора проверяемого регистра (А,РК,РД,РС)

Однобитное поле сравнения

Адрес перехода

Поле выбора проверяемого бита

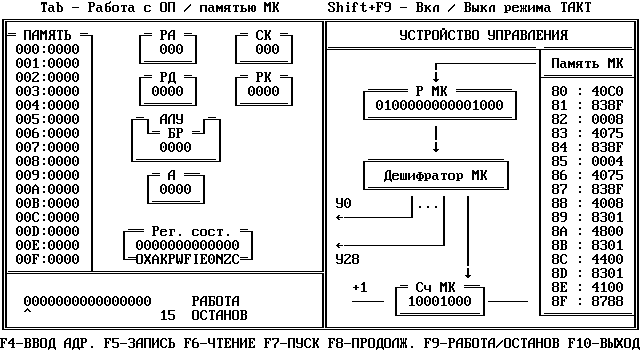
Микропрограмма хранится в постоянном запоминающем устройстве - памяти микрокоманд. В каждом такте работы ЭВМ из этой памяти в регистр микрокоманд (РМК) пересылается очередная микрокоманда, т.е. микрокоманда, на которую указывает счетчик микрокоманд (СчМК), одновременно выполняющий функции регистра адреса микрокоманд. Затем содержимое СчМК наращивается на единицу.

Если из памяти микрокоманд выбрана операционная микрокоманда, то в 31-ый бит РМК записывается 0 (код операции ОМК). Этот сигнал через инвертор НЕ открывает вентильную схему ВРО и обеспечивает передачу на В0-В28 состояний соответствующих битов РМК (управляющих сигналов У0-У28).

Разряды РМК, содержащие 1, создают открывающий управляющий сигнал, а содержащие 0 - закрывающий. Подобная структура микрокоманды, где каждый бит используется для создания отдельного управляющего сигнала, называется горизонтальной.

Вентильные схемы В1, В2, В3 предназначены, соответственно, для передачи содержимого РД, РК, СК на правый вход АЛУ. Если все эти схемы закрыты (У1=У2=У3=0), то сигнал на правом входе АЛУ соответствует коду числа 0. Аналогично используются вентильные схемы В4, В5, В6, позволяющие передать на левый вход АЛУ содержимое А, РС, КР или кода числа 0.

Управляющие сигналы У7-У10 перестраивают АЛУ на выполнение различных микроопераций. При У7=...У10=0 в 17-разрядный буферный регистр АЛУ (БР) записывается сумма входных сигналов АЛУ: при У7=У8=У9=0 и У10=1 к такой сумме добавляется 1; при У7=У8=У10=0 и У9=1 в БР записывается результат логического умножения входных сигналов АЛУ; при У7=1 и (или) У8=1 можно получить аналогичные результаты, но для инверсных значений одного или двух входных сигналов.



*Рис. 3.1. Модель базовой ЭВМ с микропрограммным устройством управления*

Рассмотрим несколько примеров операционных микрокоманд.

1. Для вычитания содержимого РД из содержимого А и записи результата в буферный регистр (А-РД=>БР) следует выполнить микрокоманду (0000 0000 0000 0000 0000 0101 0001 0010)2 = (0000 0512)16, т.е. одновременно подать единичные управляющие сигналы на В1, В4, В8 и В10. Тогда к уменьшаемому прибавится обратный код вычитаемого и к этой добавится единица, что эквивалентно суммированию уменьшаемого с дополнительным кодом вычитаемого.
2. Для вычитания 1 из содержимого аккумулятора (А-1=>БР) надо выполнить микрокоманду (0000 0110)16, т.е. подать единичные управляющие сигналы на В4 и В8 и сложить содержимое А с обратным кодом числа 0 или (что то же самое) с дополнительным кодом числа -1.
3. Для увеличения на 1 содержимого СК (СК+1=>БР) надо выполнить микрокоманду (0000 0408)16, т.е. открыть В3 и В10.

Вентильные схемы В11 и В12 позволяют записать в БР сдвинутое на один разряд вправо или влево содержимое аккумулятора. При этом "лишний" разряд БР заполняется содержимым регистра переноса С.

Вентильные схемы В13-В15 используются для передачи в однобитовые регистры С, N, Z признаков результата операции, выполненной в АЛУ: двух старших разрядов 17-битного БР (перенос и знак), а также выходного сигнала специальной схемы, который равен 1 лишь в том случае, когда содержимое БР равно 0. Управляющие сигналы У16 и У17 позволяют установить регистр С в 0 или 1 независимо от результата выполнения операции, сохраняемого в БР.

Вентильные схемы В18-В22 позволяют переписать содержимое 16 или 11 младших разрядов в РА, РД, РК, СК и А соответственно.

Вентильные схемы В23-В28 используются для организации обмена информацией между регистрами процессора и другими подсистемами ЭВМ (памятью и устройствами ввода-вывода). И, наконец, вентильная схема В0 используется для передачи сигнала прекращения выполнения программы (команда HLT).

Если из памяти микрокоманд выбрана управляющая микрокоманда, то в 31-ый бит РМК записывается 1 (код операции УМК). Этот сигнал открывает вентильную схему БР1 и тем самым создает условия для выполнения УМК. Теперь по сигналу, создаваемому каким-либо битом поля выбора проверяемого регистра (У1, У2, У4 или У5) открывается из вентильных схем В1, В2, В4 или В5 и на вентили ВВ0-ВВ15 поступает через АЛУ содержимое соответствующего регистра (РД, РК, А или РС). Одновременно на эти же вентили поступает с РМК содержимое поля выбора проверяемого бита. Так как в этом поле записана только одна 1 (на месте, соответствующем проверяемому биту), то открывается лишь один из вентилей ВВ0-ВВ15, через который на схему сравнения поступает содержимое проверяемого бита из проверяемого регистра. На другой вход этой схемы поступает содержимое однобитового поля сравнения (24-ый бит УМК), в которое при кодировании УМК записали 0 или 1.

Если проверяемый бит и бит из поля сравнения идентичны, то схема сравнения формирует единичный сигнал, который открывает вентильную схему ВА и на СчМК пересылается адрес перехода (16-24-ый биты УМК). В противном случае на СчМК сохраняется адрес микрокоманды, расположенной вслед за исполняемой, так как после выборки текущей микрокоманды содержимое СчМК увеличивается на единицу.

При организации разветвлений в микропрограмме используется содержимое регистра состояний, являющегося объединением однобитовых регистров признаков и состояний ЭВМ (табл. 3.1). Такое объединение сделано с целью формального уменьшения числа регистров, с которыми работает МПУ, что позволяет сократить разрядность УМК. На структурной схеме (рис. 3.1) РС изображен (для удобства описания) в виде самостоятельного регистра, хотя его разряды лишь дублируют состояние регистров С, N, Z и т.д.

Таблица 3.1

|  |  |
| --- | --- |
| **Разряд** | **Содержимое** |
| **0** | Перенос |
| **1** | Нуль |
| **2** | Знак |
| **3** | 0 - используется для организации безусловных переходов в МПУ |
| **4** | Разрешение прерывания |
| **5** | Прерывание |
| **6** | Состояние ВУ (Ф) |
| **7** | Состояние тумблеров РАБОТА/ОСТАНОВ (1 - РАБОТА) |
| **8** | Программа |
| **9** | Выборка команды |
| **10** | Выборка адреса |
| **11** | Исполнение |
| **12** | Ввод-вывод |

Рассмотрим две управляющих микрокоманды:

1. После увеличения на 1 содержимого РД в команде ISZ надо проверить знаковый разряд РД (разряд с номером 15). Если этот разряд равен 1 (содержимое РД меньше нуля), то выполнение команды ISZ завершается. В противном случае необходимо прибавить 1 к содержимому СК, т.е. организовать пропуск команды, следующей за ISZ. Это разветвление (переход по адресу 8F) осуществляется с помощью микрокоманды (858F 8000)16.

2. Для организации безусловного перехода (например, по тому же адресу 8F) используется 3-й бит регистра состояний, содержащий константу 0. Сравнение этого разряда с нулем, записанным в 24-ый разряд УМК, всегда дает положительный результат и позволяет переслать в СчМК нужный адрес перехода. Микрокоманда, реализующая эту операцию, имеет вид: (828F 0008)16.

## 3.2 Интерпретатор базовой ЭВМ.

Полный текст микропрограммы (интерпретатора команд) приведен в табл. 3.2. В этой таблице есть один "лишний" столбец (ВЕРТ.), содержание которого будет описано ниже.

Первые микрокоманды интерпретатора служат для выборки команды из основной памяти (ОП) базовой ЭВМ и определения ее типа: адресная, безадресная или ввода-вывода. Для этого содержимое СК (в котором хранится адрес исполняемой команды) пересылается через БР в РА (СК=>БР и БР=>РА). Затем из ячейки ОП, на которую указывает РА, пересылается в РД команда, а содержимое СК увеличивается на единицу и пересылается в БР: ОП(РА)=>РД, СК+1=>БР.

*Таблица 3.2*

*Интерпретатор базовой ЭВМ (микропрограмма)*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Адрес** | **Микрокоманды** | | **Комментарии** | |
|  | **Горизонт.** | **Верт.** | **Метка** | **Действие** |
| **1** | **2** | **3** | **4** | **5** |
| **Цикл выборки команды** | | | | |
| 01 | 0000 0008 | 0300 | нач | СК ==> БР |
| 02 | 0004 0000 | 4001 |  | БР ==> РА |
| 03 | 0080 0408 | 0311 |  | ОП(РА) ==> РД, СК + 1 ==> БР |
| 04 | 0020 0000 | 4004 |  | БР ==> СК |
| 05 | 0000 0002 | 0100 |  | РД ==> БР |
| 06 | 0010 0000 | 4003 |  | БР ==> РК |
|  |  |  |  | **Определение типа команды** |
| 07 | 880C 8000 | AF0C |  | IF BIT(15,PK) = 0 THEN АДЦ(0C) |
| 08 | 880C 4000 | AE0C |  | IF BIT(14,PK) = 0 THEN АДЦ(0C) |
| 09 | 880C 2000 | AD0C |  | IF BIT(13,PK) = 0 THEN АДЦ(0C) |
| 0A | 895E 1000 | EC5E |  | IF BIT(12,PK) = 1 THEN БАД(5Е) |
| 0B | 828E 0008 | 83BE |  | GOTO B/B(8E) |
|  |  |  |  | **Определение вида адресации** |
| 0С | 881D 0800 | AB1D | АДЦ | IF BIT(11,PK) = 0 THEN АДР(1D) |
| **Цикл выборки адреса операнда** | | | | |
| 0D | 0000 0002 | 0100 |  | РД ==> БР |
| 0E | 0004 0000 | 4001 |  | БР ==> РА |
| 0F | 0080 0000 | 0001 |  | ОП(РА) ==> РД |
| 10 | 881D 0008 | A31D |  | IF BIT(3,PK) = 0 THEN АДР(1D) |
| 11 | 891D 0010 | E41D |  | IF BIT(4,PK) = 1 THEN АДР(1D) |
| 12 | 891D 0020 | E51D |  | IF BIT(5,PK) = 1 THEN АДР(1D) |
| 13 | 891D 0040 | E61D |  | IF BIT(6,PK) = 1 THEN АДР(1D) |
| 14 | 891D 0080 | E71D |  | IF BIT(7,PK) = 1 THEN АДР(1D) |
| 15 | 891D 0100 | E81D |  | IF BIT(8,PK) = 1 THEN АДР(1D) |
| 16 | 891D 0200 | E91D |  | IF BIT(9,PK) = 1 THEN АДР(1D) |
| 17 | 891D 0400 | EA1D |  | IF BIT(10,PK) = 1 THEN АДР(1D) |
| 18 | 0000 0402 | 0110 |  | РД ==> БР |
| 19 | 0008 0000 | 4002 |  | БР ==> РД |
| 1A | 0100 0000 | 0002 |  | РД ==> ОП(РА) |
| 1B | 0000 0082 | 0140 |  | РД + СОМ(0) = РД - 1 ==> БР |
| 1C | 0008 0000 | 4002 |  | БР ==> РД |
| **Цикл исполнения адресных команд** | | | | |
|  |  |  |  | **Декодирование адресных команд** |
| 1D | 892D 8000 | EF2D | АДР | IF BIT(15,PK) = 1 THEN ПРХ(2D) |
| 1E | 0000 0002 | 0100 |  | РД ==> БР |
| 1F | 0004 0000 | 4001 |  | БР ==> РА |
| 20 | 8927 4000 | EE27 |  | IF BIT(14,PK) = 1 THEN АРФ(27) |
| 21 | 8824 2000 | AD24 |  | IF BIT(13,PK) = 0 THEN A1(24) |
| 22 | 8857 1000 | AC57 |  | IF BIT(12,PK) = 0 THEN JSR(57) |
| 23 | 8238 0008 | 8338 |  | GOTO MOV(38) |
| 24 | 0080 0000 | 0001 | A1 | ОП(РА) ==> РД |
| 25 | 8850 1000 | AC50 |  | IF BIT(12,PK) = 0 THEN ISZ(50) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** |
| 26 | 8235 0008 | 8335 |  | GOTO AND(35) |
| 27 | 0080 0000 | 0001 | АРФ | ОП(РА) ==> РД |
| 28 | 882B 2000 | AD2B |  | IF BIT(13,PK) = 0 THEN СУМ(2B) |
| 29 | 8843 1000 | AC43 |  | IF BIT(12,PK) = 0 THEN SUB(43) |
| 2A | 82B0 0008 | 83B0 |  | GOTO P - A(B0) |
| 2B | 883C 1000 | AC3C | СУМ | IF BIT(12,PK) = 0 THEN ADD(3C) |
| 2C | 823F 0000 | 833F |  | GOTO ADC(3F) |
| 2D | 8830 4000 | AE30 | ПРХ | IF BIT(14,PK) = 0 THEN УПХ(30) |
| 2E | 8847 1000 | AC47 |  | IF BIT(12,PK) = 0 THEN BR(47) |
| 2F | 82D0 0008 | 83D0 |  | GOTO Р - П(D0) |
| 30 | 8833 2000 | AD33 | УПХ | IF BIT(13,PK) = 0 THEN П1(33) |
| 31 | 884С 1000 | AC4C |  | IF BIT(12,PK) = 0 THEN BMI(4C) |
| 32 | 824E 0008 | 834E |  | GOTO BEQ(4E) |
| 33 | 8846 1000 | AC46 | П1 | IF BIT(12,PK) = 0 THEN BCS(46) |
| 34 | 824A 0008 | 834A |  | GOTO BPL(4A) |
|  |  |  |  | **Исполнение адресных команд** |
| 35 | 0000 0212 | 1120 | AND | A & РД ==> БР |
| 36 | 0040 C000 | 4035 |  | БР ==> A, N, Z |
| 37 | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
| 38 | 0000 0010 | 1000 | MOV | A ==> БР |
| 39 | 0008 0000 | 4002 |  | БР ==> РД |
| 3A | 0100 0000 | 0002 |  | РД ==> ОП(РА) |
| 3B | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
| 3C | 0000 0012 | 1100 | ADD | A + РД ==> БР |
| 3D | 0040 E000 | 4075 |  | БР ==> A, C, N, Z |
| 3E | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
| 3F | 823C 0001 | 803C | ADC | IF BIT(0,PC) = 0 THEN ADD(3C) |
| 40 | 0000 0412 | 1110 |  | A + РД + 1 ==> БР |
| 41 | 0040 E000 | 4075 |  | БР ==> A, C, N, Z |
| 42 | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
| 43 | 0000 0512 | 1190 | SUB | A + COM(РД) + 1 = A - РД ==> БР |
| 44 | 0040 E000 | 4075 |  | БР ==> A, C, N, Z |
| 45 | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
| 46 | 828F 0001 | 808F | BCS | IF BIT(0,PC) = 0 THEN ПРЕ(8D) |
| 47 | 0000 0002 | 0100 | BR | РД ==> БР |
| 48 | 0020 0000 | 4004 |  | БР ==> СК |
| 49 | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
| 4A | 838F 0004 | C28F | BPL | IF BIT(2,PC) = 1 THEN ПРЕ(8F) |
| 4B | 8247 0008 | 8347 |  | GOTO BR(47) |
| 4C | 828F 0004 | 828F | BMI | IF BIT(2,PC) = 0 THEN ПРЕ(8F) |
| 4D | 8247 0008 | 8347 |  | GOTO BR(47) |
| 4E | 828F 0002 | 818F | BEQ | IF BIT(1,PC) = 0 THEN ПРЕ(8F) |
| 4F | 8247 0008 | 8347 |  | GOTO BR(47) |
| 50 | 0000 0402 | 0110 | ISZ | РД + 1 ==> БР |
| 51 | 0008 0000 | 4002 |  | БР ==> РД |
| 52 | 0100 0000 | 0002 |  | РД ==> ОП(РА) |
| 53 | 858А 8000 | DF8F |  | IF BIT(15,РД) = 1 THEN ПРЕ(8F) |
| 54 | 0000 0408 | 0310 |  | СК + 1 ==> БР |
| 55 | 0020 0000 | 4004 |  | БР ==> СК |
| 56 | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
| 57 | 0000 0402 | 0110 | JSR | РД + 1 ==> БР |
| 58 | 0010 0000 | 4003 |  | БР ==> РК |
| 59 | 0000 0008 | 0300 |  | СК ==> БР |
| 5A | 0008 0000 | 4002 |  | БР ==> РД |
| 5B | 0100 0004 | 0202 |  | РД ==> ОП(РА), РК ==> БР |
| 5C | 0020 0000 | 4004 |  | БР ==> СК |
| 5В | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
| **Продолжение цикла выборки команды**  **декодирование и исполнение безадресных команд** | | | | |
| 5E | 8861 0800 | AB61 | БАД | IF BIT(11,PK) = 0 THEN Б0(61) |
| 5F | 886C 0400 | AA6C |  | IF BIT(10,PK) = 0 THEN Б1(6C) |
| 60 | 82E0 0008 | 83E0 |  | GOTO Р - Б(E0) |
| 61 | 8867 0400 | AA67 | Б0 | IF BIT(10,PK) = 0 THEN Б2(67) |
| **1** | **2** | **3** | **4** | **5** |
| 62 | 8865 0200 | A965 |  | IF BIT(9,PK) = 0 THEN Б3(65) |
| 63 | 8882 0100 | A882 |  | IF BIT(8,PK) = 0 THEN ROL(82) |
| 64 | 8285 0008 | 8385 |  | GOTO ROR(85) |
| 65 | 887B 0100 | A87B | Б3 | IF BIT(8,PK) = 0 THEN CMA(7B) |
| 66 | 827E 0008 | 837E |  | GOTO CMC(7E) |
| 67 | 886A 0200 | A96A | Б2 | IF BIT(9,PK) = 0 THEN Б4(6A) |
| 68 | 8876 0100 | A876 |  | IF BIT(8,PK) = 0 THEN CLA(76) |
| 69 | 8279 0008 | 8379 |  | GOTO CLC(79) |
| 6A | 8888 0100 | A888 | Б4 | IF BIT(8,PK) = 0 THEN HLT(88) |
| 6B | 8287 0008 | 8387 |  | GOTO NOP(87) |
| 6C | 886F 0200 | A96F | Б1 | IF BIT(9,PK) = 0 THEN Б5(6F) |
| 6D | 888A 0100 | A88A |  | IF BIT(8,PK) = 0 THEN EI(8A) |
| 6E | 828C 0008 | 838C |  | GOTO DI(8C) |
| 6F | 8873 0100 | A873 | Б5 | IF BIT(8,PK) = 0 THEN INC(73) |
| 70 | 0000 0110 | 1080 | DEC | A + COM(0) = A - 1 ==> БР |
| 71 | 0040 E000 | 4075 |  | БР ==> A, C, N, Z |
| 72 | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
| 73 | 0000 0410 | 1010 | INC | A + 1 ==> БР |
| 74 | 0040 E000 | 4075 |  | БР ==> A, C, N, Z |
| 75 | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
| 76 | 0000 0200 | 0020 | CLA | 0 ==> БР |
| 77 | 0040 C000 | 4035 |  | БР ==> A, N, Z |
| 78 | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
| 79 | 0001 0000 | 4080 | CLC | 0 ==> C |
| 7A | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
| 7B | 0000 0090 | 1040 | CMA | COM(A) ==> БР, инверсия A |
| 7C | 0040 C000 | 4035 |  | БР ==> A, N, Z |
| 7D | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
| 7E | 8280 0001 | 8080 | CMC | IF BIT(0,PC) = 0 THEN Б6(80) |
| 7F | 8279 0008 | 8379 |  | GOTO CLC(79) |
| 80 | 0002 0000 | 40C0 | Б6 | 1 ==> C |
| 81 | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
| 82 | 0000 1000 | 0008 | ROL | RAL(A) ==> БР, сдвиг влево |
| 83 | 0040 E000 | 4075 |  | БР ==> A, C, N, Z |
| 84 | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
| 85 | 0000 0800 | 0004 | ROR | RAR(A) ==> БР, сдвиг вправо |
| 86 | 0040 E000 | 4075 |  | БР ==> A, C, N, Z |
| 87 | 828F 0008 | 838F | NOP | GOTO ПРЕ(8F) |
| 88 | 0000 0001 | 4008 | HLT | Останов машины |
| 89 | 8201 0008 | 8301 |  | GOTO НАЧ(01) |
| 8A | 1000 0000 | 4800 | EI | Разрешение прерывания |
| 8B | 8201 0008 | 8301 |  | GOTO НАЧ(01) |
| 8C | 0800 0000 | 4400 | DI | Запрещение прерывания |
| 8D | 8201 0008 | 8301 |  | GOTO НАЧ(01) |
| **Продолжение цикла выборки команды**  **декодирование и исполнение команд ввода-вывода** | | | | |
| 8E | 0200 0000 | 4100 | B/B | Организация связей с ВУ |
| **Цикл прерывания** | | | | |
| 8F | 8288 0080 | 8788 | ПРЕ | IF BIT(7,PC) = 0 THEN HTL(88) |
| 90 | 8201 0020 | 8501 |  | IF BIT(5,PC) = 0 THEN НАЧ(01) |
| 91 | 0000 0200 | 0020 |  | 0 ==> БР |
| 92 | 0004 0000 | 4001 |  | БР ==> РА |
| 93 | 0000 0008 | 0300 |  | СК ==> БР |
| 94 | 0008 0000 | 4002 |  | БР ==> РД |
| 95 | 0100 0400 | 0012 |  | РД ==> ОП(РА), 1 ==> БР |
| 96 | 0020 0000 | 4004 |  | БР ==> СК |
| 97 | 0800 0000 | 4400 |  | Запрещение прерывания |
| 98 | 8201 0008 | 8301 |  | GOTO НАЧ(01) |
| **Пультовые операции** | | | | |
|  |  |  |  | **Ввод адреса** |
| 99 | 0000 0040 | 3000 | В/А | КР ==> БР |
| 9A | 0020 0000 | 4004 |  | БР ==> СК |
| 9B | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
| **1** | **2** | **3** | **4** | **5** |
|  |  |  |  | **Чтение** |
| 9C | 0000 0008 | 0300 | ЧТ | СК ==> БР |
| 9D | 0004 0000 | 4001 |  | БР ==> РА |
| 9E | 0080 0408 | 0311 |  | ОП(РА) ==> РД, СК + 1 ==> БР |
| 9F | 0020 0000 | 4004 |  | БР ==> СК |
| A0 | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
|  |  |  |  | **Запись** |
| A1 | 0000 0008 | 0300 | ЗАП | СК ==> БР |
| A2 | 0004 0000 | 4001 |  | БР ==> РА |
| A3 | 0000 0040 | 3000 |  | КР ==> БР |
| A4 | 0008 0000 | 4002 |  | БР ==> РД |
| A5 | 0100 0408 | 0312 |  | РД ==> ОП(РА), СК + 1 ==> БР |
| A6 | 0020 0000 | 4004 |  | БР ==> СК |
| A7 | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
|  |  |  |  | **Пуск** |
| A8 | 0000 0200 | 0020 | ПУС | 0 ==> БР |
| A9 | 005C E000 | 4077 |  | БР ==> A, C, N, Z, РА, РД, РК |
| AA | 0400 0000 | 4200 |  | Сброс флагов ВУ |
| AB | 0800 0000 | 4400 |  | Запрещение прерывания |
| AC | 828F 0008 | 838F |  | GOTO ПРЕ(8F) |
| . . . |  |  |  |  |
| B0 |  |  | Р - А | Арифметическая команда 7### |
| . . . |  |  |  |  |
| D0 |  |  | Р - П | Команда перехода D### |
| . . . |  |  |  |  |
| E0 |  |  | Р - Б | Безадресная команда FC## |
| . . . |  |  |  |  |
| FF |  |  |  |  |

Далее содержимое БР, т.е. адрес следующей команды, пересылается в СК, а команда пересылается из РД в РК, после чего начинается ее дешифрация.

Так как адресные команды (команды с кодами операции от 0 до D) обязательно содержат ноль в 15, 14 или 13 бите, то проверкой этих битов РК можно выделить адресную команду и перейти к проверке ее 2-го бита (бита вида адресации). Для разделения команд ввода-вывода (код операции Е) и безадресных команд (код операции F) достаточно проанализировать 12-ый бит РК: если этот бит равен 1, то надо переходить к микрокомандам продолжения дешифрации безадресных команд, расположенных, начиная с адреса 5E (метка БАД). В комментариях микрокоманда анализа 12-го бита РК записана в виде:

IF BIT(12,РК)=1 THEN БАД(5E).

В памяти микрокоманд нет полных микропрограмм для адресных команд с кодами операций 7 и D, а также для безадресных команд FC00, FD00, FE00 и FF00. Когда при декодировании команды выясняется, что выбрана команда 7xxx, управление передается ячейке с адресом В0. Начиная с этой ячейки, могут располагаться микрокоманды какой-либо новой арифметической команды (например, умножения). Для микропрограмм реализации команды перехода и безадресных команд выделены участки памяти микрокоманд с начальными адресами D0 и E0.

В базовой ЭВМ реализован и другой вариант интерпретатора, использующий более короткие - вертикальные микрокоманды (столбец "ВЕРТ." табл. 3.2). Эти микрокоманды состоят из полей, в которых закодированы требуемые наборы управляющих сигналов (рис. 3.2). Для декодирования используются дополнительные устройства - дешифраторы.

### **Домашнее задание №4**

***Расширение системы команд ЭВМ.***

Цель задания - изучение микрокоманд базовой ЭВМ, микропрограмм выполнения отдельных команд, а так же овладение навыками составления микропрограмм для новых команд.

Часть I. Написать последовательность адресов микрокоманд, которые должны быть выполнены при реализации заданного фрагмента программы, начинающегося с команды, расположенной по адресу 002 (перед выполнением программы исполняется команда "Пуск", очищающая аккумулятор и регистр переноса).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Адрес | Номер варианта | | | | | |
|  | 1 | 2 | 3 | 4 | 5 | 6 |
| 1  2  3  4  5 | 0  CMA  BMI 05  NOP  + MOV 01 | 1  INC  BLP 05  NOP  + ADD 01 | 1  DEC  BMI 05  NOP  + ADD 01 | 1  ADD 01  + BPL 05  NOP  DEC | 1  + BEQ 05  NOP  ADD 01  INC | 1  CMC  BCS 05  NOP  + ADC 01 |

Результаты сводятся в таблицу вида:

|  |  |  |
| --- | --- | --- |
| Команда | Машинный цикл | Последовательность адресов микрокоманд |
| AND 01  (1001) | —  Выборка команды  Исполнение  — | 89  01, 02, 03, 04, 05, 06, 07, 0C  1D, 1E, 1F, 20, 21, 24, 25, 26, 35, 36, 37, 8F  88 |
| CLC  (F300) | —  Выборка  команды  — | 89  01, 02, 03, 04, 05, 06, 07, 08, 09, 0A, 5E,  61, 67, 68, 69, 79, 7A, 8F  88 |
| . . . | . . . | . . . |

В этой таблице символом "-" отмечены микрокоманды остановки и перехода к циклу "ВЫБОРКА КОМАНДЫ", используемые при пошаговом выполнении программы.

Кроме того необходимо описать поля шести последних микрокоманд цикла "ИСПОЛНЕНИЕ" команды, отмеченной знаком "+". Описания каждой микрокоманды выполнить в виде рисунков:

Часть II.

Микрокоманда: СК+1=>БР

Горизонтальная: 0000 0408

0

0

0

0

0

0

0

0

0

0

0

0

0

0

0

0

0

0

0

0

0

1

0

0

0

0

0

0

1

0

0

0

Код операции

Открыть В10 (+1)

Открыть В3 (правый вход АЛУ соединен с СК)

Вертикальная: 0310

00

00

00

11

00

01

00

00

Код операции

На левый вход АЛУ подан 0

Правый вход АЛУ соединен с СК

Обратный код не вычислять

Обмен информацией с памятью не осуществлять

Не сдвигать

Лев.вх.+Пр.вх.+1. те 0+СК+1

Микрокоманда: СК+1=>БР

Горизонтальная: 0000 0408

0

0

0

0

0

0

0

0

0

0

0

0

0

0

0

0

0

0

0

0

0

1

0

0

0

0

0

0

1

0

0

0

Код операции

Открыть В10 (+1)

Открыть В3 (правый вход АЛУ соединен с СК)

Вертикальная: 0310

00

00

00

11

00

01

00

00

Код операции

На левый вход АЛУ подан 0

Правый вход АЛУ соединен с СК

Обратный код не вычислять

Обмен информацией с памятью не осуществлять

Не сдвигать

Лев.вх.+Пр.вх.+1. те 0+СК+1

А. Написать завершающие вертикальные микрокоманды цикла "ИСПОЛНЕНИЕ" следующих команд:

Команда 7ххх

1 вариант - ЗАГРУЗКА(записать в аккумулятор содержимое ячейки памяти, на которую указывает адресная часть команды);

2 вариант - ПЕРЕСЫЛКА СО СБРОСОМ(записать содержимое аккумулятора в ячейку памяти, на которую указывает адресная часть команды, а затем очистить аккумулятор);

3 вариант - СРАВНЕНИЕ(вычесть содержимое аккумулятора из содержимого ячейки памяти, на которую указывает адресная часть команды, и, не изменяя содержимое аккумулятора, установить признаки результата вычитания: C, N, Z);

4 вариант - ЗАГРУЗКА ДОПОЛНИТЕЛЬНАЯ(записать в аккумулятор дополнительный код содержимого ячейки, на которую указывает адресная часть команды);

5 вариант - ПЕРЕСЫЛКА ДОПОЛНИТЕЛЬНАЯ(записать дополнительный код содержимого аккумулятора в ячейку памяти, на которую указывает адресная часть команды);

6 вариант - ПЕРЕСЫЛКА УДВОЕННАЯ (записать в ячейку памяти, на которую указывает адресная часть команды, удвоенное содержимое аккумулятора).

Команда Dxxx

Организовать переход к команде, расположенной по адресу, на которую указывает адресная часть команды, если:

1 вариант - аккумулятор содержит четное число;

2 вариант - аккумулятор содержит нечетное число;

3 вариант - аккумулятор содержит число, большее чем 16383;

4 вариант - аккумулятор содержит число, меньшее чем -16384;

5 вариант - 7-й бит аккумулятора(старший бит младшего байта) равен нулю;

6 вариант - 7-й бит аккумулятора равен единице;

Безадресные команды

1 вариант - циклический сдвиг влево на 2 разряда (FC00);

2 вариант - циклический сдвиг вправо на 2 разряда (FD00);

3 вариант - получение дополнительного кода аккумулятора(FE00);

4 вариант - запись единицы в аккумулятор(FC00);

5 вариант - циклический сдвиг влево с очисткой регистра С(FD00);

6 вариант - циклический сдвиг вправо с очисткой регистра С(FЕ00);

Б. Написать тестовые программы для проверки правильности исполнения всех трех синтезированных команд базовой ЭВМ и подготовиться к выполнению лабораторной работы №8. Тестовые программы должны отвечать следующим требованиям:

1. Для синтезированных арифметических и без адресных команд результат их выполнения должен быть зафиксирован в памяти базовой ЭВМ, а не только в регистрах,
2. Если проверяемая арифметическая или безадресная команда устанавливает признаки результата (C,Z,N), необходимо проверить правильную установку одного из них, используя соответствующую команду перехода. Результат проверки признака зафиксировать в памяти базовой ЭВМ,
3. Для синтезированных команд переходов необходимо проверить команду как при выполнении условия перехода, так и при его невыполнении. Результат проверки в обоих случаях зафиксировать в памяти базовой ЭВМ.

Таким образом, после выполнения правильно разработанной тестовой программы в автоматическом режиме в памяти базовой ЭВМ будет размещена информация, позволяющая однозначно подтвердить правильность выполнения синтезированной команды.

В. При разработке микропрограмм заданных команд следует иметь в виду:

1. В процессе дешифрации команды 7ххх в РА записывается адрес операнда (может использоваться для команд пересылки), а в РД - сам операнд ( может использоваться для команд загрузки и сравнения). Затем осуществляется переход к ячейке памяти микрокоманд ВО, где надо разместить первую синтезируемую микрокоманду команды 7ххх.

2. После выборки команды перехода ххх в РД сохраняется адрес перехода (адресная часть команды), который может быть переписан в СК при выполнении условия перехода. Последняя микрокоманда дешифрации команды Dххх передает управление в ячейку с адресом D0, где надо разместить первую синтезируемую микрокоманду команды Dххх.

3. Когда в процессе дешифрации безадресных команд выясняется, что в 10-м и 11-м разрядах РК содержатся единицы(т.е. выбрана одна из команд:FC00, FD00, FE00 или FF00), управление передается в ячейку с адресом Е0. Здесь должны начинаться микрокоманды дополнительной дешифрации, выделяющие заданную команду путем анализа 9-го и 8-го разрядов РК и передающие управление в свободную область памяти микрокоманд(от Ех до FF), где следует разместить микрокоманды реализации безадресной команды.

4. Все микропрограммы реализуемых команд должны заканчиваться микрокомандой 838F (GOTO ПРЕ(8F)), осуществляющей переход к микрокомандам, завершающим исполнение любой команды базовой микро ЭВМ.

Пример. Для создания команды FF00, которая осуществляет инвертирование содержимого аккумулятора и очистку регистра переноса, можно написать следующую последовательность микрокоманд:

|  |  |  |
| --- | --- | --- |
| Адрес  МП | Микро-  команды | Комментарии |
| E0  E1  E2  E1  E4  E5 | A98F  A88F  1040  4035  4080  838F | IF BIT(9,PK)=0 THEN ПРЕ(8F): К окончанию цикла  IF BIT(8,PK)=0 THEN ПРУ(8F): исполнения, если  : дешифрируемая ко-  : манда не FF00  COM(A)=>БР : Инверсия А  БР=>А : Пересылка резуль-  : тата в А и регистр  : признаков  0=>C : Очистка С  GOTO ПРЕ(8F) : Выход |

### **Лабораторная работа № 7**

***Исследование микропрограммного устройства управления.***

Цель работы - исследование микропрограмм выполнения нескольких команд базовой ЭВМ, способов программирования отдельных машинных циклов и дешифрирования команд, а также принципа кодирования отдельных микрокоманд. Работа является завершением первой части домашнего задания №4. В ней производится проверка правильности анализа порядка выполнения микрокоманд заданной программы.

Подготовка к выполнению работы - завершить первую часть домашнего задания №4 и подготовить следующие таблицы:

а)для записи последовательности микрокоманд, которые будут выполняться базовой ЭВМ при реализации фрагмента программы первой части домашнего задания №4 (форма таблицы аналогична таблице этого задания);

б) для записи результатов выполнения шести последних микрокоманд цикла "ИСПОЛНЕНИЕ" команды, которая отмечена символом "+" в заданном фрагменте программы:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| СчМК до  выборки | Содержимое регистров после выборки и исполнения МК | | | | | | | | | | |
| МК | ВМК | СК | РА | РК | РД | А | С | БР | N | Z | СчМК |
| xx | xxxx | xxxx | xxxx | xxxx | xxxx | xxxx | x | xxxx | x | x | xxxx |

Порядок выполнения работы

1. Занести в память машины заданный фрагмент программы, ввести ее пусковой адрес, нажать "ПУСК" и после завершения начальной установки устройств ЭВМ перевести ее в режим потактового выполнения программы.
2. Последовательно выполнить все микрокоманды, записывая в подготовленные таблицы адреса выполняемых микрокоманд и для шести из них - содержимое регистров.

Содержание отчета по работе. В отчет надо поместить домашнее задание №4 (часть 1), указанные выше таблицы экспериментальных данных и схему алгоритма дешифрации команды, отмеченной символом "+".

### **Лабораторная работа № 8**

***Синтез команд базовой ЭВМ.***

Цель работы - практическое завершение второй части домашнего задания №4. В ней производится загрузка в память микропрограмм микрокоманд новых команд базовой ЭВМ, загрузка в память ЭВМ программы для проверки правильности выполнения синтезированных команд, а также проверка и отладка этих микропрограмм.

Подготовка к выполнению работы. Завершить домашнее задание №4 и подготовить две таблицы по форме, приведенной в лаб. работе №7. Строки первой из этих таблиц (теоретически) должны быть заполнены содержимым регистров базовой ЭВМ при пошаговом выполнении за нее тестовой программы (синтезированные команды должны выполняться по тактам, остальные - по командам). Строку с содержимым регистров ЭВМ после исполнения (или первой микрокоманды новой команды) следует предворять заголовком:

КОМАНДА хххх, РАСПОЛОЖЕННАЯ ПО АДРЕСУ ххх

Вторая таблица (экспериментальная) заполняется в лаборатории.

Порядок выполнения работы

1. Занести в память ЭВМ текст тестовой программы.
2. Занести в память микрокоманд (ПМ) микрокоманды новых команд.
3. Выполнить в пошаговом режиме тестовую программу, занося в таблицу содержимое регистров процессора после выполнения каждой команды (для синтезированных команд) или каждой команды (для остальных команд).

Содержание отчета по работе. Домашнее задание №4 (часть 2), таблицы с результатами выполнения тестовой программы(теоретическая и экспериментальная). Анализ расхождений между этими таблицами и описание процесса отладки программы и микропрограммы.

# Приложение 1

|  |  |
| --- | --- |
| ***Для перемещения в клавишном регистре***  ***используются следующие клавиши:*** | |
|  |  |
| RIGHT | Перемещение указателя на одну позицию вправо. |
| LEFT | Перемещение указателя на одну позицию влево. |
| UP | Инверсия бита (изменение значения на противоположное) по текущему положению указателя |
| 1 | Занесение 1 по текущему положению указателя и перемещение его на на следующую позицию |
| 0 | Занесение 0 по текущему положению указателя и перемещение его на на следующую позицию |
|  |  |
| ***В процессе работы также используются клавиши:*** | |
|  |  |
| F4 | Ввод адреса. По этой клавише содержимое клавишного регистра заносится в счетчик команд. |
| F5 | Запись. Информация из клавишного регистра заносится в память по текущему содержимому счетчика команд. |
| F6 | Чтение. Из ячейки памяти (по адресу расположенному в счетчике команд) информация читается в регистр данных. |
| F7 | Пуск. Действие этой клавиши различно в режимах "РАБОТА" и "ОСТАНОВ". В режиме "РАБОТА" по ней происходит обнуление всех регистров, кроме счетчика команд, и происходит запуск программы на выполнение. В режиме "ОСТАНОВ" происходит очистка регистров, кроме счетчика команд, а запуск не производится |
| F8 | Продолжение. В режиме "ОСТАНОВ" происходит исполнение одной инструкции, а в режиме "ОСТАНОВ" продолжение выполнения программы с адреса в регистре команд |
| F9 | Клавиша, управляющая переключением режима работы базовой ЭВМ. Производит переключение режимов "РАБОТА" и "ОСТАНОВ". |
| F10 | Выход из базовой ЭВМ. |
| Shift+F4 | Смена маски. |
|  |  |
| ***Работа с внешними устройствами обеспечивается клавишами:*** | |
|  |  |
| F1,F2,F3 | Готовность внешнего устройства 1,2,3 соответственно. |
| Tab | Переход в режим ввода в регистры данных ВУ2 и ВУ3. |
|  |  |
| ***Для работы с микрокомандами используйте клавиши:*** | |
|  |  |
| Tab | Переключение ввода в обычную память и память микрокоманд. При вводе в память микрокоманд слева от клавишного регистра загорается индикатор МК. |
| Shift+F9 | Включение/Отключение режима ТАКТ. В этом режиме при нажатии клавиши F8 (Продолжение) происходит выполнение одной микрокоманды. |

**Содержание**

РАЗДЕЛ 1. БАЗОВАЯ ЭВМ

1.1 Назначение базовой ЭВМ

1.2 Структура базовой ЭВМ

1.3. Система команд базовой ЭВМ

1.4 Арифметические операции

1.5 Управление вычислительным процессом, сдвиги и логические операции

1.6 Подпрограммы

1.7 Выполнение машинных команд

**Домашнее задание № 1**

**Домашнее задание № 2**

**Лабораторная работа № 1**

**Лабораторная работа № 2**

**Лабораторная работа № 3**

**Лабораторная работа № 4**

РАЗДЕЛ 2. ОРГАНИЗАЦИЯ ВВОДА-ВЫВОДА В БАЗОВОЙ ЭВМ

2.1 Устройства ввода-вывода базовой ЭВМ

2.2 Программно-управляемая передача данных.

2.3 Асинхронный обмен.

2.4 Обмен по прерыванию программы.

**Домашнее задание № 3**

**Лабораторная работа № 5**

**Лабораторная работа № 6**

РАЗДЕЛ 3. МИКРОПРОГРАММНОЕ УСТРОЙСТВО УПРАВЛЕНИЯ

3.1. Микропрограммное управление вентильными схемами.

3.2 Интерпретатор базовой ЭВМ.

**Домашнее задание №4**

**Лабораторная работа № 7**

**Лабораторная работа № 8**

ПРИЛОЖЕНИЕ 1