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).

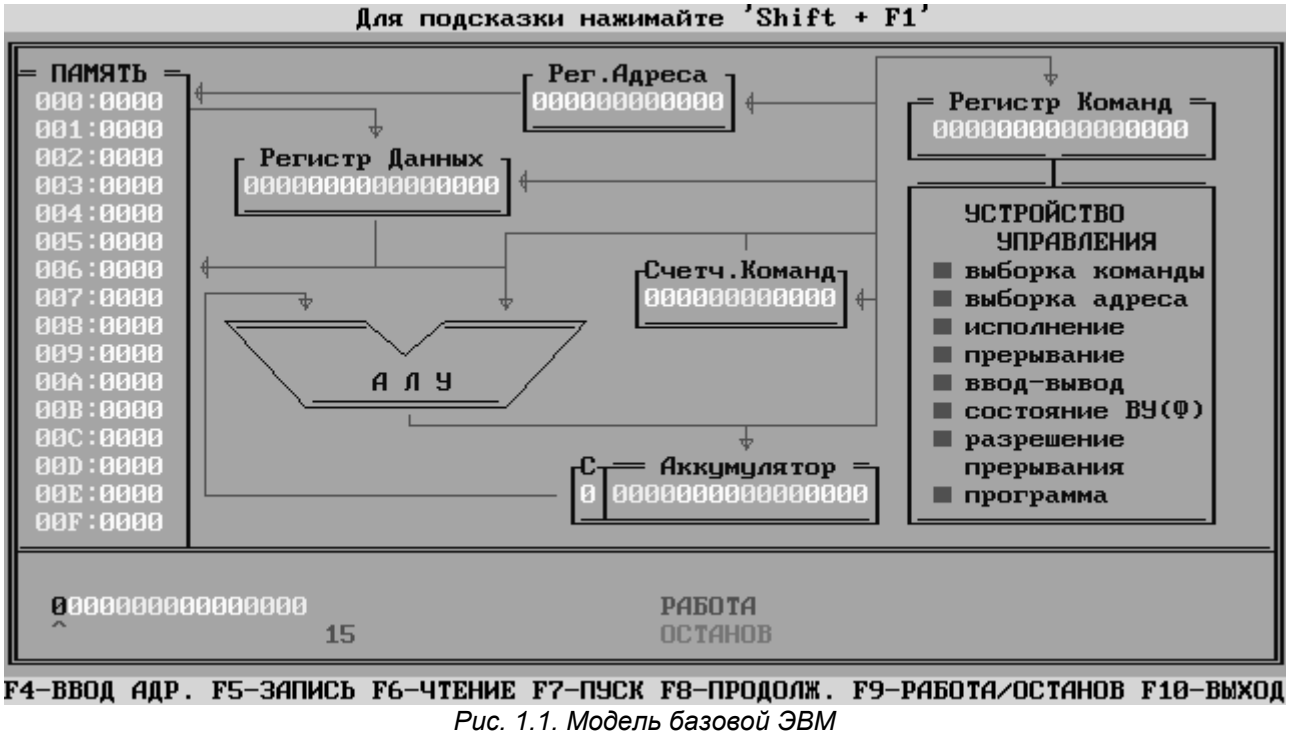


Рис. 1.1. Модель базовой ЭВМ

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 = 2^{11} 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).

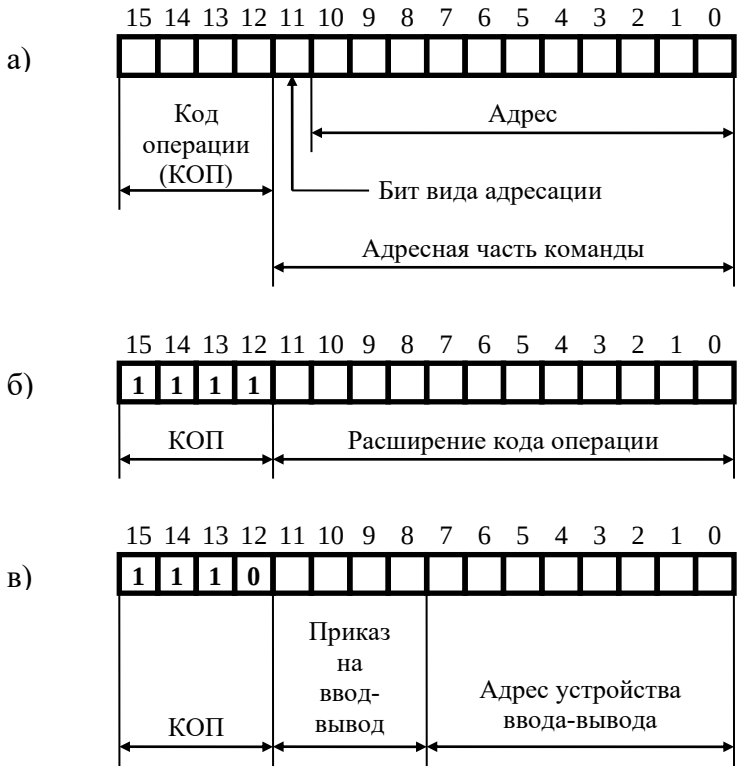


Рис 1.2. Форматы команд: а - адресных, б - безадресных, в - команд ввода-вывода

В командах обращения к памяти на адрес отведено 11 бит. Следовательно, можно прямо адресоваться к $2^{11} = 2048$ 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 $(1111\ 1111\ 1111\ 1111)_2 = (1FFF)_{16} = 2^{15} - 1 = 65535$.

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	Мнемо-ника	Код	Описание
Адресные команды			
Логическое умножение	AND M	1XXX	(M) & (A) ➔ A
Пересылка	MOV M	3XXX	(A) ➔ M
Сложение	ADD M	4XXX	(M) + (A) ➔ A
Сложение с переносом	ADC M	5XXX	(M) + (A) + (C) ➔ A
Вычитание	SUB M	6XXX	(A) – (M) ➔ A
Переход, если перенос	BCS M	8XXX	Если (C) = 1, то M ➔ CK
Переход, если плюс	BPL M	9XXX	Если (A) >= 0, то M ➔ CK
Переход, если минус	BMI M	AXXX	Если (A) < 0, то M ➔ CK
Переход, если ноль	BEQ M	BXXX	Если (A)и(C) = 0, то M ➔ CK
Безусловный переход	BR M	CXXX	M ➔ CK
Приращение и пропуск	ISZ M	0XXX	(M) + 1 ➔ M, если (M) >= 0, то (CK) + 1 ➔ CK
Обращение к подпрограмме	JSR M	2XXX	(CK) ➔ M, M + 1 ➔ CK
Безадресные команды			
Очистка аккумулятора	CLA	F200	0 ➔ A
Очистка рег. переноса	CLC	F300	0 ➔ C
Инверсия аккумулятора	CMA	F400	(!A) ➔ A
Инверсия рег. переноса	CMC	F500	(!C) ➔ C
Циклический сдвиг влево на 1 разряд	ROL	F600	Содержимое A и C сдвигается влево, A(15) ➔ C, C ➔ A(0)
Циклический сдвиг вправо на 1 разряд	ROR	F700	Содержимое A и C сдвигается вправо, A(0) ➔ C, C ➔ A(15)
Инкремент аккумулятора	INC	F800	(A) + 1 ➔ A
Декремент аккумулятора	DEC	F900	(A) – 1 ➔ A
Останов	HLT	F000	
Нет операции	NOP	F100	
Разрешение прерывания	EI	FA00	
Запрещение прерывания	DI	FB00	
Команды ввода-вывода			
Очистка флага	CLF B	E0XX	0 ➔ флаг устр. B
Опрос флага	TSF B	E1XX	Если (флаг устр. B) = 1, то (CK) + 1 ➔ CK
Ввод	IN	E2XX	(B) ➔ A
Вывод	OUT	E3XX	(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	—	—
0 000 0000 0000 0001	+1	1 111 1111 1111 1111	-1
0 000 0000 0000 0010	+2	1 111 1111 1111 1110	-2
0 000 0000 0000 0011	+3	1 111 1111 1111 1101	-3
...	...	...	...
0 111 1111 1111 1110	+32766	1 000 0000 0000 0010	-32766
0 111 1111 1111 1111	+32767	1 000 0000 0000 0001	-32767
—	—	1 000 0000 0000 0000	-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 A, 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 про-

граммы 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 A и C 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 C 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 A, к содержимому ячейки с адресом M добавляется 1 и если результат меньше нуля, то выполняется команда, следующая за ISZ M (команда, расположенная по адресу A+1), в противном случае эта команда *попускается, т.е. выполняется команда, расположенная по адресу A+2. Программа с использованием команды ISZ приведена в табл. 1.4.

Таблица 1.3

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

Адрес	Содержимое		Комментарии
	Код	Мнемоника	
5	0078	Y	Множимое (здесь – десятичное значение 120)
6	0000	Z	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
7	0032	M	Множитель 50 = $(32)_{16}$
8	0000	C	Ячейка, используемая для накопления числа выполненных
. . .	. . .	. . .	циклов, - счетчик циклов
10	F200	CLA	
11	4006	ADD 6	
12	4005	ADD 5	К промежуточному результату, находящемуся в ячейке 6,
13	3006	MOV 6	добавляется еще одно значение множимого Y
14	F200	CLA	
15	4008	ADD 8	Содержимое счетчика циклов увеличивается на 1,
16	F800	INC	а его копия пока сохраняется в аккумуляторе
17	3008	MOV 8	
18	6007	SUB 7	Если содержимое счетчика циклов меньше значения
19	A010	BMI 10	множителя, то выполняется переход к командам, осуществляющим новое суммирование Y с промежуточным значением Z
1A	F000	HLT	Останов. В ячейке 6 хранится искомый результат

Таблица 1.4

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

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

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

Адрес	Содержимое		Комментарии
	Код	Мнемоника	
5	0000		Ячейка, отведенная для накопления результата
6	FFE0		Отрицательное число элементов массива
7	0010		Текущий адрес элемента массива
. . .			
10			
.			
.			Численные значения элементов массива
.			

2F			
30	F200	CLA	Очистка аккумулятора
31	4807	ADD (7)	Суммирование очередного элемента массива
32	0007	ISZ 7	Текущий адрес элемента массива наращивается на 1
33	F100	NOP	Команда “Нет операции”
34	0006	ISZ 6	Наращивание на 1 содержимого счетчика элементов
35	C031	BR 31	массива и переход к команде 31, пока оно < 0
36	3005	MOV 5	Запись результата в ячейку 5
37	F000	HLT	Останов ЭВМ

Таблица 1.7

Третий вариант программы суммирования элементов массива

Ад-рес	Содержимое		Комментарии
	Код	Мнемо-ника	
5	0000		Ячейка, отведенная для накопления результата
6	FFE0		Отрицательное число элементов массива
. . .			
F	0010		Текущий адрес элемента массива
10			
.			
.			Численные значения элементов массива
.			
2F			
30	F200	CLA	Очистка аккумулятора
31	480F	ADD (F)	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.
32	0006	ISZ 6	Наращивание на 1 содержимого счетчика элементов
33	C031	BR 31	массива и переход к команде 31, пока оно < 0
34	3005	MOV 5	Запись результата в ячейку 5
35	F000	HLT	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 M 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

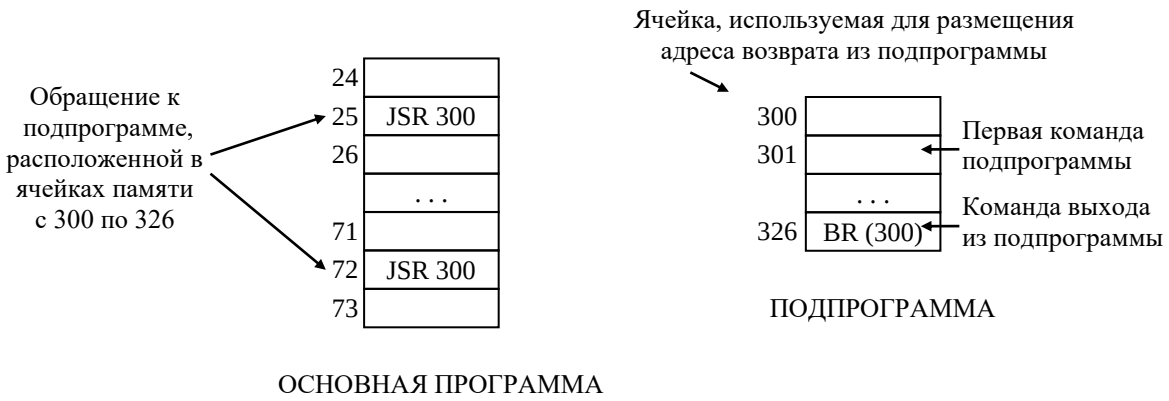
Рис. 1.3. Циклические сдвиги: а - влево, б - вправо

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 M в команде JSR M). Последней командой подпрограммы должна быть команда выхода (команда BR (M) для подпрограммы, размещенной начиная с ячейки M). По ней осуществляется переход к команде, адрес которой сохраняется в первой ячейке подпрограммы.

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.

4. 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 чте-

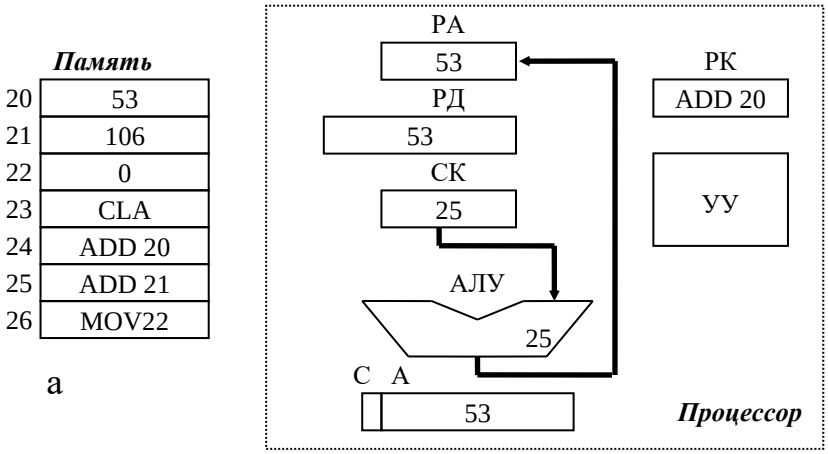
ния из памяти адреса операнда, результата или перехода и состоит из следующих шагов.

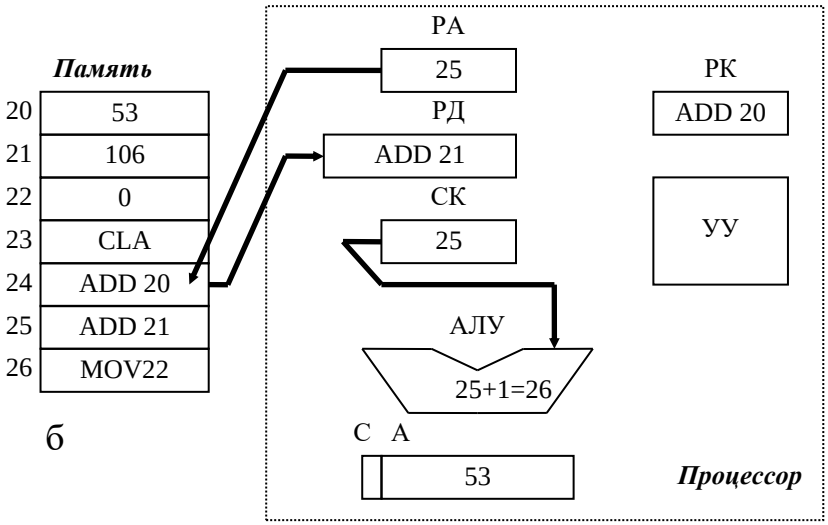
1) Адресная часть команды пересылается из регистра данных, где пока еще сохраняется копия команды, в регистр адреса.
2) Содержимое ячейки памяти, указываемой регистром адреса, читается в регистр данных. Теперь в этом регистре находится либо адрес операнда, либо адрес результата, либо адрес перехода, который будет использоваться в цикле исполнения команды. Если косвенно адресуется одна из индексных ячеек (адреса 8...F), то цикл выборки адреса операнда (результата) продолжается.
3) Содержимое регистра данных увеличивается на единицу.
4) Измененное содержимое регистра данных пересылается в ячейку памяти по адресу, указанному регистром адреса.
5) Содержимое регистра данных уменьшается на единицу.

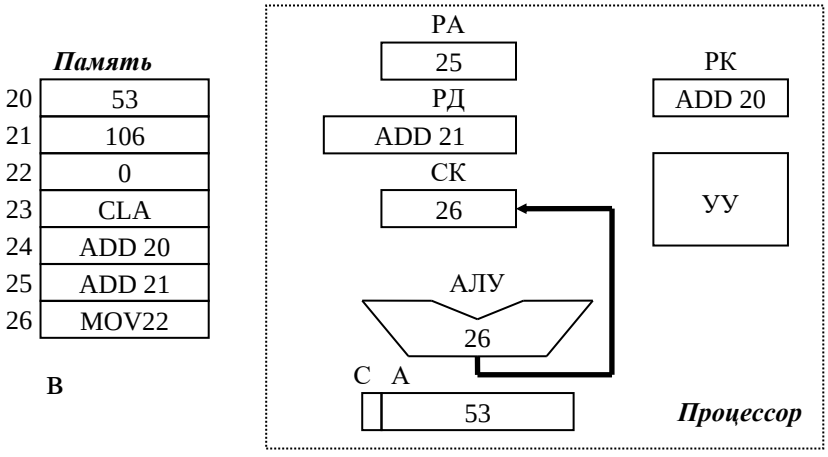
После этой операции в регистре данных восстанавливается значение адреса, находившегося в индексной ячейке до выполнения шага 3. Содержимое же индексной ячейки увеличилось на 1 и при следующем обращении к ней будет выбран новый адрес операнда (результата).

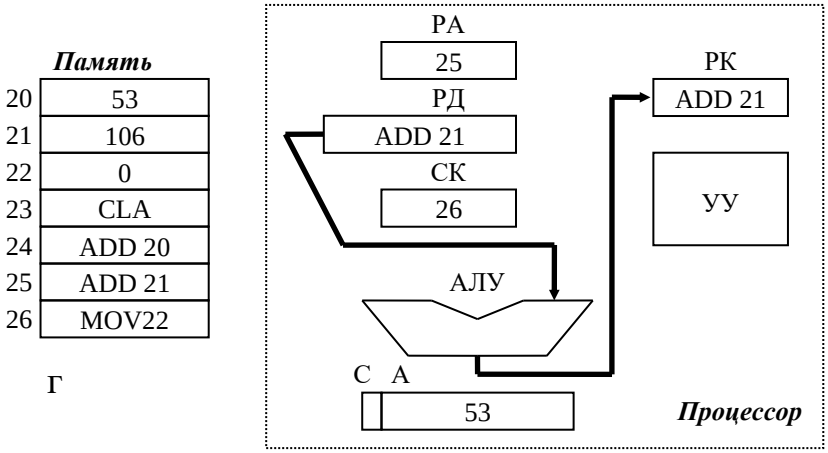
Исполнение. Последовательность действий, выполняемых в этом цикле, определяется типом выполняемой адресной команды.

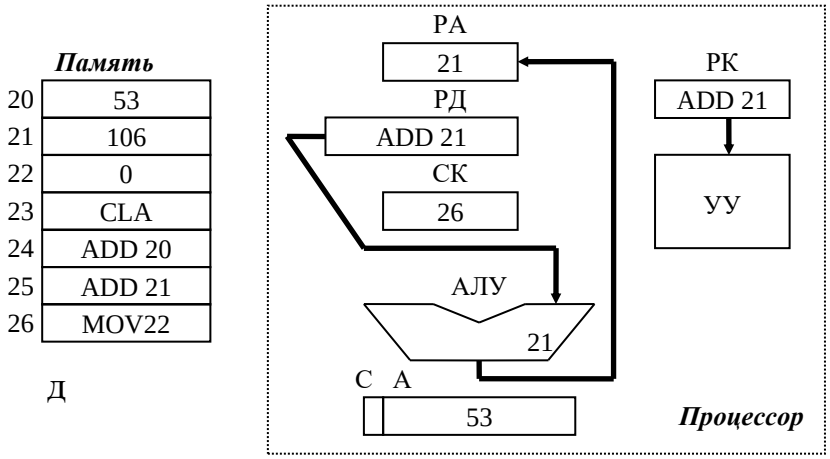
1. Для команд, при выполнении которых требуется выборка операнда из памяти ЭВМ (AND, ADD, ADC, SUB, ISZ), цикл исполнения используется для чтения операнда в регистр данных и выполнения операции, указываемой кодом операции команды. Пример цикла исполнения команды ADD 21 приведен на рис. 1.4, д, е, ж, 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.

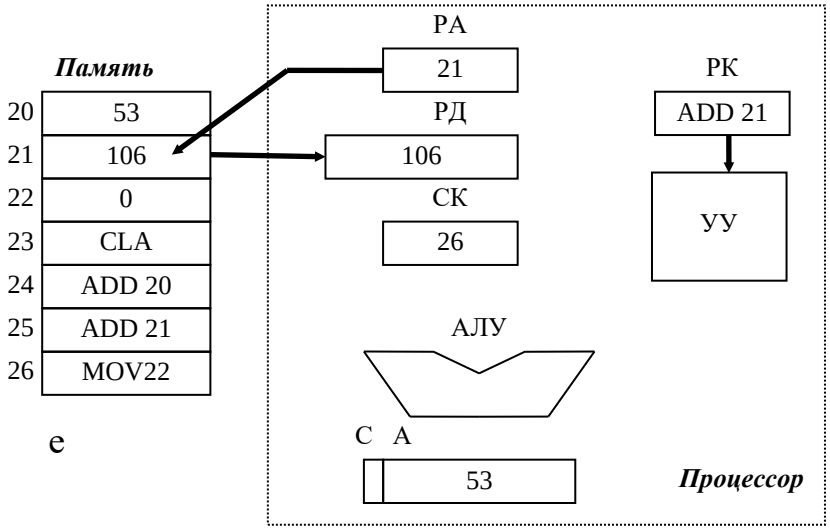

Память
20 53
21 106
22 0
23 CLA
24 ADD 20
25 ADD 21
26 MOV22
а
РА
53
РД
53
СК
25
АЛУ
25
С А
53
РК
ADD 20
УУ
Процессор

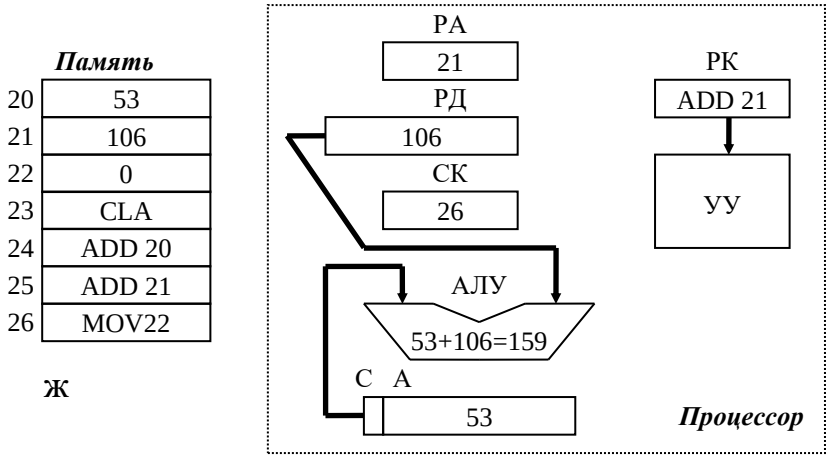

Память
20 53
21 106
22 0
23 CLA
24 ADD 20
25 ADD 21
26 MOV22
б
РА
25
РД
ADD 21
СК
25
АЛУ
25+1=26
С А
53
РК
ADD 20
УУ
Процессор

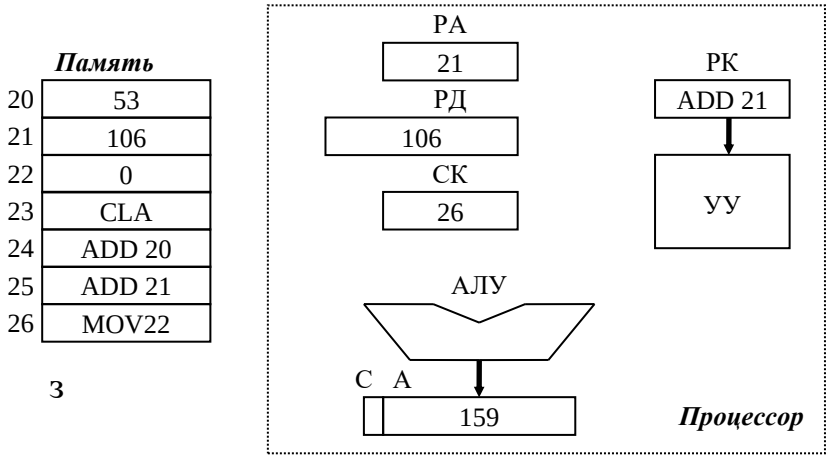

Память
20 53
21 106
22 0
23 CLA
24 ADD 20
25 ADD 21
26 MOV22
в
РА
25
РД
ADD 21
СК
26
АЛУ
26
С А
53
РК
ADD 20
УУ
Процессор


Память
20 53
21 106
22 0
23 CLA
24 ADD 20
25 ADD 21
26 MOV22
г
РА
25
РД
ADD 21
СК
26
АЛУ
С А
53
РК
ADD 21
УУ
Процессор


Память
20 53
21 106
22 0
23 CLA
24 ADD 20
25 ADD 21
26 MOV22
Д
РА
21
РД
ADD 21
СК
26
АЛУ
21
С А
53
РК
ADD 21
УУ
Процессор


Память
20 53
21 106
22 0
23 CLA
24 ADD 20
25 ADD 21
26 MOV22
е
РА
21
РД
106
СК
26
АЛУ
С А
53
РК
ADD 21
УУ
Процессор


Память
20 53
21 106
22 0
23 CLA
24 ADD 20
25 ADD 21
26 MOV22
ж
РА
21
РД
106
СК
26
АЛУ
53+106=159
С А
53
РК
ADD 21
УУ
Процессор


Память
20 53
21 106
22 0
23 CLA
24 ADD 20
25 ADD 21
26 MOV22
з
РА
21
РД
106
СК
26
АЛУ
С А
159
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: X1=A, X2=C, X3=A+C, X4=A+C+C, X5=C-A, X6=65536-X4, X7= -X1, X8= -X2, X9= -X3, X10= -X4, X11= - X5, X12= -X6. Выполнить перевод десятичных чисел X1,...,X12 в двоичную систему счисления, получив их двоичные эквиваленты B1,...,B12 соответственно. Для представления двоичных чисел B1,...,B12 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 X1, B1, X7 и B7.
2. Выполнить следующие сложения двоичных чисел: B1+B2, B2+B3, B7+B8, B8+B9, B2+B7, B1+B8. 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
A	2006	6390	4186	1818	5238	2262	6582
C	15452	14940	15772	16924	15900	16028	17436
Операнд	Номер варианта						
	8	9	10	11	12	13	14
A	4154	2902	1722	2774	5302	2294	1978
C	16162	18006	16988	15388	14972	16064	15516
Операнд	Номер варианта						
	15	16	17	18	19	20	21
A	2998	6518	2678	5238	4314	2422	1754
C	16288	15260	16160	14932	15420	17500	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
xxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	x	xxx	xxxx

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	0000	0000	+ F200	0000	0000	0000
018	F1AA	+ F200	4022	4017	4015	0018
019	7C89	4021	4021	2009	4019	+ F200
01A	2A5A	6022	3020	00F4	+ F200	4023
01B	0000	3024	F200	+ F200	4018	6024
01C	+ F200	F200	4023	4024	6024	3018
01D	4018	4023	1020	6018	3017	F200
01E	501A	1024	3020	301A	F200	4022
01F	301B	3024	F000	F200	4019	1018
020	F200	F000	0000	401A	1023	3018
021	4019	1377	7C89	1019	3017	F000
022	101B	2295	01AA	301A	F000	21AA
023	301B	7C90	A299	F000	0001	0255
024	F000	301A	0000	C000	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	0625	+ C01A	CF0B	0000	0000	0000
017	0FA7	ACAB	F0F5	+ C01B	5417	+ C01B
018	+ F200	001F	F000	0018	+ F200	001B
019	4016	0000	+ F200	0019	4022	FF20
01A	4017	F200	4016	1000	4023	00DF
01B	9020	4017	4017	F200	9020	F200
01C	F200	4018	B020	4019	F200	4019
01D	3022	A020	F200	401A	3017	401A
01E	F100	F200	3018	8022	F100	A021
01F	F000	F100	F000	F200	F000	F200
020	3022	3022	4016	3018	3017	F100
021	C01F	F000	3018	301A	C01F	3023
022	1111	CCCC	C01F	F000	FF0F	F000
023	0000	0000	0000	0000	0031	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	0000	0000	0000	0011	0000	0000
00B	0000	0000	0000	0000	001C	0000
00C	0000	0000	001B	0000	0000	0000
00D	0000	0000	0000	0000	0000	0010
00E	001C	0000	0000	0000	0000	0000
00F	0000	001C	0000	0000	0000	0000
010	0000	0000	0000	3355	0000	0000
011	0000	0000	+ F200	71BC	FFFC	0010
012	FFFC	FFFC	480C	ABBA	+ F200	0000
013	+ F200	+ F200	9016	63CD	480B	0707
014	480E	480F	401D	FFFC	9019	0000
015	B018	A018	301D	0000	F200	FFFC
016	4011	4011	0019	+ F200	F800	+ F200
017	3011	3011	C011	480A	401C	480D
018	0012	0012	F000	A01D	301C	B01A
019	C013	C013	FFFC	F200	0011	C01D
01A	F000	F000	8778	F800	C012	F800
01B	0378	7F02	1777	4015	F000	4011
01C	0000	DECA	8788	3015	0000	3011
01D	F0EB	30AE	1111	0014	B0B0	0015
01E	0377	7F01	FFA1	C016	5B0B	C016
01F	0000	0000	0000	F000	CF11	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	0010	0000	0000	0000	0000	0000
00B	0000	001A	0000	0000	0000	0000
00C	0000	0000	0012	0000	0000	0000
00D	0000	0000	0000	0019	0000	0000
00E	0000	0000	0000	0000	0010	0000
00F	0000	0000	0000	0000	0000	0011
010	8080	0000	0000	+ F200	0000	F200
011	ABDA	FFFE	FFFD	480D	6789	4816
012	630D	+ F200	0000	B014	CACA	F800
013	71B0	480B	0707	2045	8A7C	+ F200
014	FFFC	9016	0000	0018	FFFC	480F
015	0000	2045	0000	C010	+ F200	9017
016	+ F200	0011	+ F200	F000	480E	2045

017	480A	C012	480C	0000	A019	001A
018	A01A	F000	B01A	FFFD	2045	C013
019	2045	0000	C01B	8018	0014	F000
01A	0014	CF01	2045	0000	C015	FFFE
01B	C016	B0BA	C011	81FF	F000	0000
01C	F000	5B1B	C016	0000	0000	0000
01D	0000	0000	F000	0000	0000	0000
. . .	. . .	. . .	. . .	. . .	. . .	. . .
045	0000	0000	0000	0000	0000	0000
046	F200	F200	F200	F200	F200	F200
047	F800	F800	F800	F800	F800	F800
048	4015	4019	4015	4017	4010	401B
049	3015	3019	3015	3017	3010	301B
04A	C845	C845	C845	C845	C845	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 B - ввод и OUT B - вывод), *проверка готовности ВУ* (TSF B) и *сброс состояния готовности* (CLF B), где B - адрес ВУ. Адрес позволяет связать процессор с одним из подключенных к нему ВУ (их может быть до 2^8=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 xx - две последние 16-ричные цифры адреса ВУ) служит для установки в нуль флажка ВУ с адресом B.

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

Команда IN B (E2xx) служит для пересылки содержимого регистра данных контроллера ВУ с адресом B в восемь младших разрядов аккумулятора.

Команда OUT B (E3xx) служит для пересылки содержимого восьми младших разрядов аккумулятора в регистр данных контроллера ВУ с адресов B.

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:

Адрес	Содержимое		Комментарии
	Код	Мнемоника	
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)
2A	E002	CLF 2	Сброс готовности ВУ-2
2B	3006	MOV 6	Пересылка кода слова "ДА" в ячейку 006
2C	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, B1), циклический сдвиг содержимого аккумулятора влево (для того, чтобы содержимое регистра переноса попало в аккумулятор) и запись этого содержимого в другую буферную ячейку (например, B2). Таким образом, необходимый минимум информации о прерванной программе сохраняется - в ячейке 000 хранится адрес продолжения прерванной программы, а в ячейках B1 и B2 хранится содержимое двух других основных регистров A и C.

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 B2 в очищенный аккумулятор, циклический сдвиг вправо для восстановления содержимого регистра переноса , очистка аккумулятора и вызов содержимого буферной ячейки B1 в очищенный аккумулятор.

Шаг 7. Контроллер прерываний вновь переводится в состояние разрешение прерывания (команда EI) 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 EI. 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

Адрес	Содержимое		Комментарии
	Код	Мнемоника	
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

Адрес	Содержимое		Комментарии
	Код	Мнемоника	
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 30-4C
.....	.....		

30	304B	MOV 4B	Сохранение в буферных ячейках 4B и 4C 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 4B, вывод на ВУ-1 восьми младших разрядов аккумулятора, сброс флага ВУ-1, переход восстановлению A и C 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 X) какой-либо ячейки памяти. Цикл для наращивания X не должен содержать более трех команд. Вывод всегда осуществля-

ется на ВУ-3 в асинхронном режиме. Выводится только восемь младших разрядов результата.

Варианты задания:

1. По запросу ВУ-1 вывести -2X+5, а по запросу ВУ-2 вывести 3X/4.
2. По запросу ВУ-3 вывести (3X-2)/2, а по запросу ВУ-2 вывести X/2+10.
3. По запросу ВУ-2 вывести (X/2)+5, а по запросу ВУ-1 вывести –(5X/2)+1.
4. По запросу ВУ-3 вывести –(X+1)/4, а по запросу ВУ-1 вывести (2X+3)/2.
5. По запросу ВУ-2 вывести (3X+3)/8, а по запросу ВУ-1 вывести –(5X+7)/2.
6. По запросу ВУ-1 вывести (5X+1)/2, а по запросу ВУ-3 вывести (X/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 (B=B+1)
	ISZ C	Наращивание на 1 содержимого счетчика элементов массива и переход по адресу A, пока оно < 0.

```
BR A
HLT          Останов ЭВМ
```

Примечание. Здесь A, B, C - 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.

3. 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 X. Результаты работы программного комплекса представить в виде таблицы.

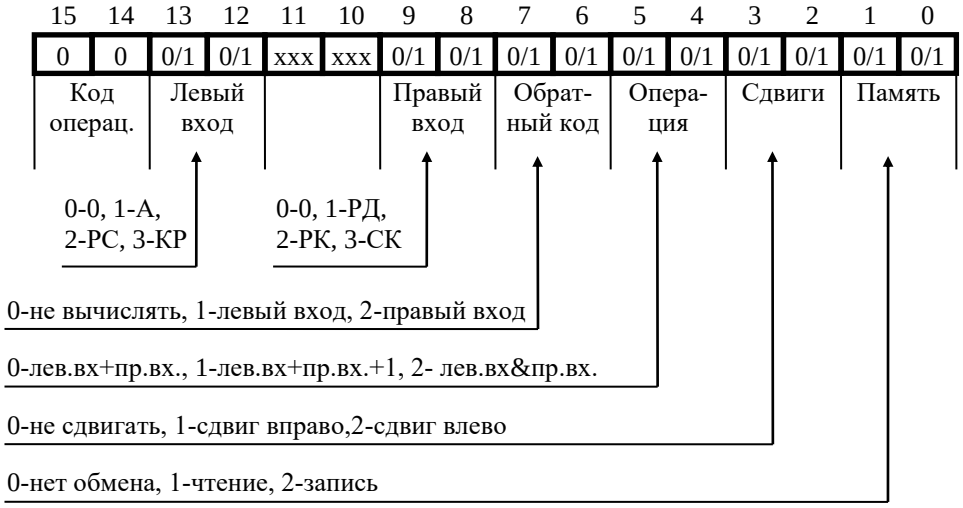
Содержание отчета по работе. Домашнее задание №3, таблицу с результатами работы комплекса программ.

РАЗДЕЛ 3. МИКРОПРОГРАММНОЕ УСТРОЙСТВО УПРАВЛЕНИЯ

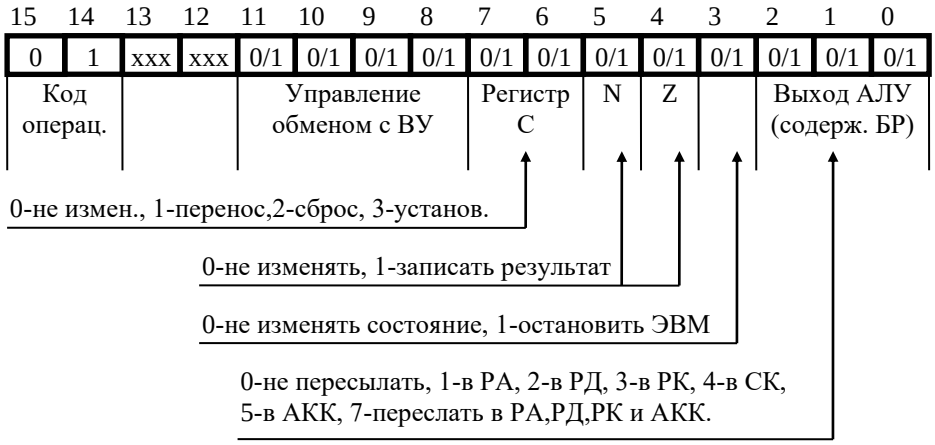
3.1. Микропрограммное управление вентильными схемами.

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.

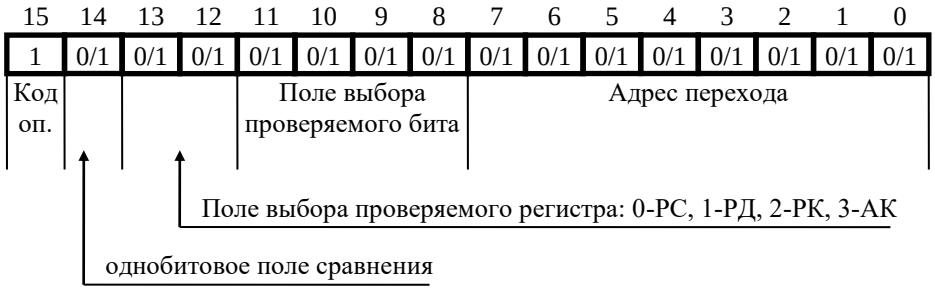
Операционная микрокоманда 0 (ОМК0)



Операционная микрокоманда 1 (ОМК1)



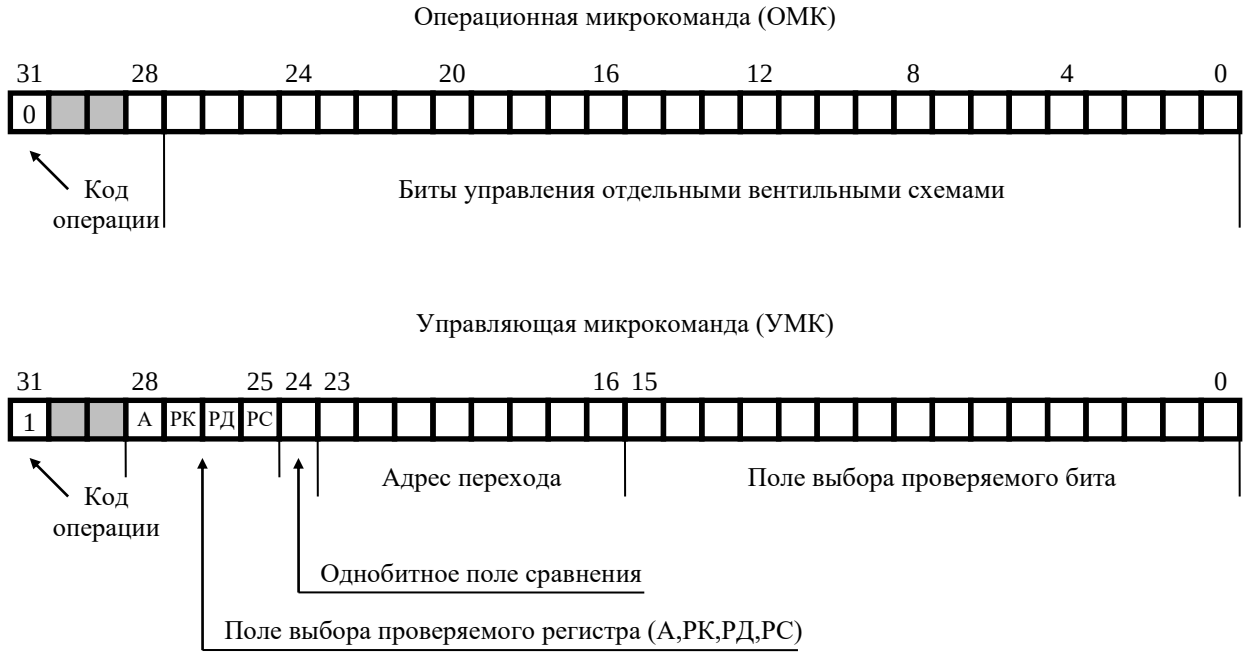
Управляющая микрокоманда (УМК)



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).

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:



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.

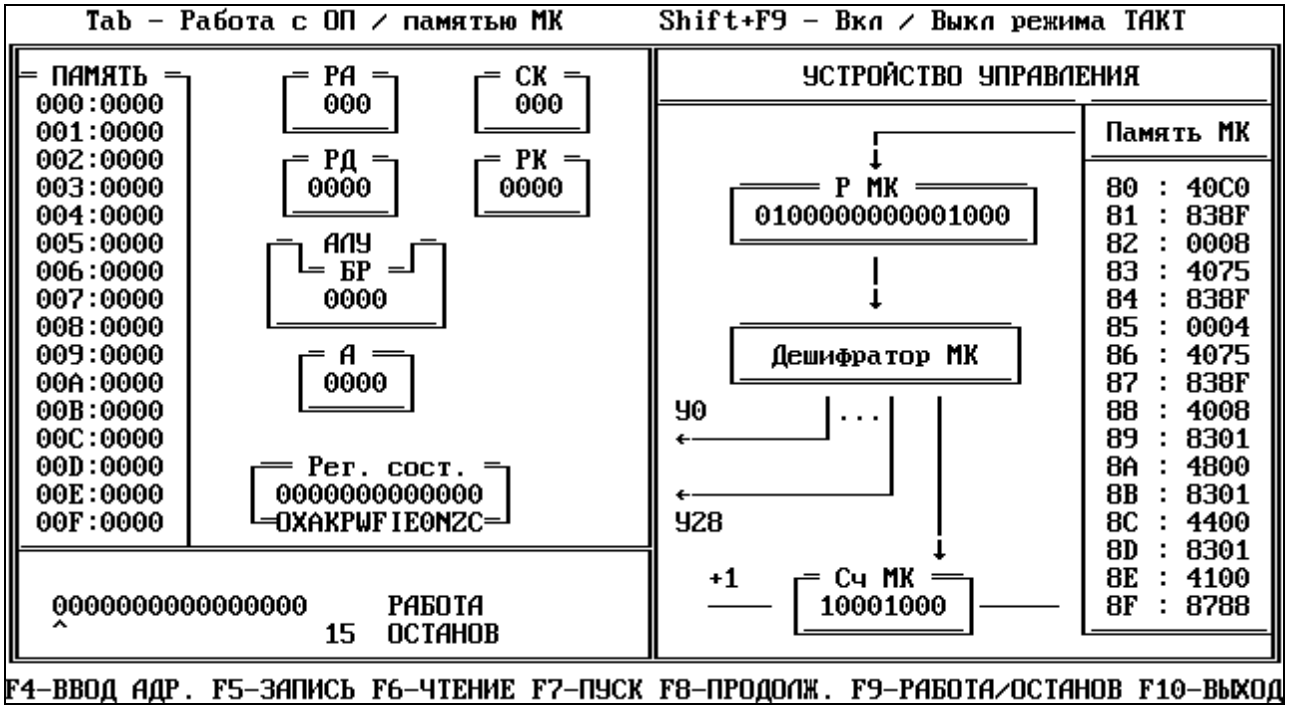


Рис. 3.1. Модель базовой ЭВМ с микропрограммным устройством управления

Рассмотрим несколько примеров операционных микрокоманд.

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.

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.

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(5E)
0B	828E 0008	83BE		GOTO В/В(8E)
				Определение вида адресации
0C	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 + COM(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,РК) = 0 THEN СУМ(2B)
29	8843 1000	AC43		IF BIT(12,РК) = 0 THEN SUB(43)
2A	82B0 0008	83B0		GOTO Р - А(B0)
2B	883C 1000	AC3C	СУМ	IF BIT(12,РК) = 0 THEN ADD(3C)
2C	823F 0000	833F		GOTO ADC(3F)
2D	8830 4000	AE30	ПРХ	IF BIT(14,РК) = 0 THEN УПХ(30)
2E	8847 1000	AC47		IF BIT(12,РК) = 0 THEN BR(47)
2F	82D0 0008	83D0		GOTO Р - П(D0)
30	8833 2000	AD33	УПХ	IF BIT(13,РК) = 0 THEN П1(33)
31	884C 1000	AC4C		IF BIT(12,РК) = 0 THEN BMI(4C)
32	824E 0008	834E		GOTO BEQ(4E)
33	8846 1000	AC46	П1	IF BIT(12,РК) = 0 THEN BCS(46)
34	824A 0008	834A		GOTO BPL(4A)
				Исполнение адресных команд
35	0000 0212	1120	AND	А & РД ==> БР
36	0040 C000	4035		БР ==> A, N, Z
37	828F 0008	838F		GOTO ПРЕ(8F)
38	0000 0010	1000	MOV	А ==> 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	А + 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,РС) = 0 THEN ADD(3C)
40	0000 0412	1110		А + РД + 1 ==> БР
41	0040 E000	4075		БР ==> A, C, N, Z
42	828F 0008	838F		GOTO ПРЕ(8F)
43	0000 0512	1190	SUB	А + СОМ(РД) + 1 = А - 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,РС) = 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,РС) = 1 THEN ПРЕ(8F)
4B	8247 0008	8347		GOTO BR(47)
4C	828F 0004	828F	BMI	IF BIT(2,РС) = 0 THEN ПРЕ(8F)
4D	8247 0008	8347		GOTO BR(47)
4E	828F 0002	818F	BEQ	IF BIT(1,РС) = 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	858A 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
5B	828F 0008	838F		GOTO ПРЕ(8F)
Продолжение цикла выборки команды декодирование и исполнение безадресных команд				
5E	8861 0800	AB61	БАД	IF BIT(11,РК) = 0 THEN Б0(61)
5F	886C 0400	AA6C		IF BIT(10,РК) = 0 THEN Б1(6C)
60	82E0 0008	83E0		GOTO Р - Б(E0)
61	8867 0400	AA67	Б0	IF BIT(10,РК) = 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	В/В	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 E) 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 B0. 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).

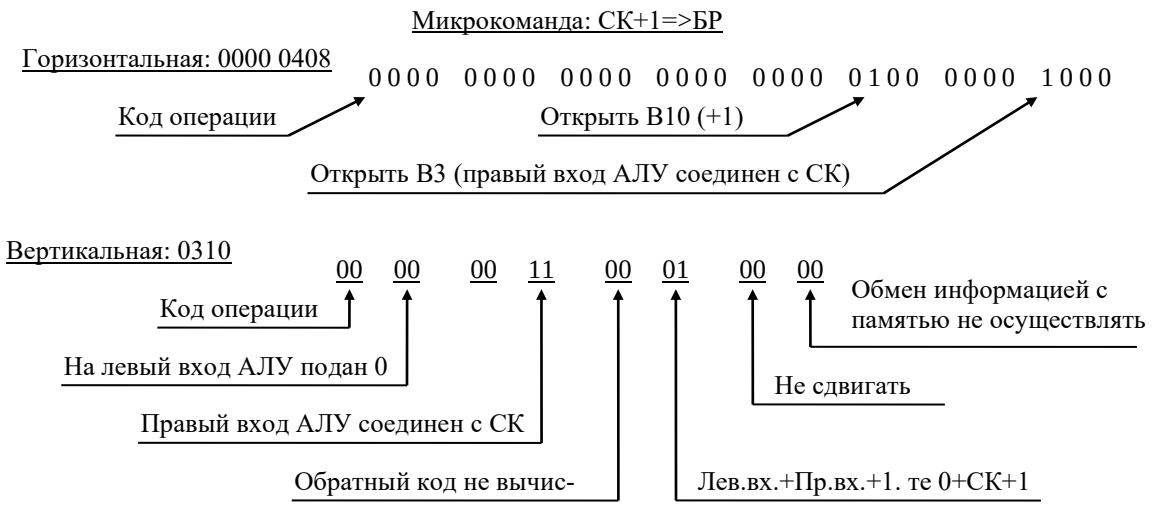
Адрес	Номер варианта					
	1	2	3	4	5	6
1	0	1	1	1	1	1
2	CMA	INC	DEC	ADD 01	+ BEQ 05	CMC
3	BMI 05	BLP 05	BMI 05	+ BPL 05	NOP	BCS 05
4	NOP	NOP	NOP	NOP	ADD 01	NOP
5	+ MOV 01	+ ADD 01	+ ADD 01	DEC	INC	+ ADC 01

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.

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

Команда 7xxx

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 C(FD00);

6 вариант - циклический сдвиг вправо с очисткой регистра C(FE00);

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 7xxx 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 7xxx.

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

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 E0. 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 Ex 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	A98F	IF BIT(9,РК)=0 THEN ПРЕ(8F): К окончанию цикла
E1	A88F	IF BIT(8,РК)=0 THEN ПРУ(8F): исполнения, если
		: дешифрируемая ко-
		: манда не FF00
E2	1040	СОМ(А)=>БР : Инверсия А
E1	4035	БР=>А : Пересылка резуль-
		: тата в А и регистр
		: признаков
E4	4080	0=>C : Очистка С
E5	838F	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 МК	Содержимое регистров после выборки и исполнения МК										
	ВМК	СК	РА	РК	РД	А	С	БР	N	Z	СчМК
хх	хххх	хххх	хххх	хххх	хххх	хххх	х	хххх	х	х	хххх

Порядок выполнения работы

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.

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

Занести в память ЭВМ текст тестовой программы.

Занести в память микрокоманд (ПМ) микрокоманды новых команд.

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