МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

«ДАГЕСТАНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра радиотехники и телекоммуникаций

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению лабораторных работ по дисциплине «Микропроцессорная техника» для студентов направления подготовки магистров 11.04.01 «Радиотехника», программа «Системы и устройства передачи, приема и обработки сигналов»

Махачкала - 2020

УДК 621.395

Методические указания к выполнению лабораторных работ по дисциплине «Микропроцессорная техника» для студентов направления подготовки магистров 210400.68 «Радиотехника», программа «Системы и устройства передачи, приема и обработки сигналов». – Махачкала: ИПЦ ДГТУ, 2020. – 36 с.

Данные методические указания являются учебным руководством к выполнению лабораторных работ по дисциплине «Микропроцессорная техника». Лабораторная работа №1 посвящена ознакомлению с работой учебной микроЭВМ. Лабораторная работа №2 посвящена ознакомлению с записью и выполнением простых программ. Лабораторная работа №3 исследует особенности записи и обращения к подпрограммам, а также способы организации стека.

Составители: к.т.н., доцент Гаджиев Х.М.
к.ф.-м.н., доцент Гаджиева С.М.
к.т.н., ст. преп. Челушкина Т.А.

Рецензенты:
зав. кафедрой ИВТ филиала МГТУ МИРЭА
в г. Махачкала, д.т.н., профессор Гусейнов Р.В.
проф. кафедры БиМАС
ФГБОУ ВПО «ДГТУ», д.т.н., профессор Магомедов Д. А.

(Рег. №_____)

Печатается согласно постановлению Ученого совета Дагестанского государственного технического университета от «___» _____2020

Лабораторная работа № 1.
Ознакомление с работой учебной микроЭВМ

Цель работы: ознакомление со структурой учебной микроЭВМ, органами управления и режимами ее работы.

1.1. Краткие теоретические сведения

1.1.1. Архитектура микроЭВМ

В микроЭВМ, построенных на базе микропроцессоров (МП), все связи между отдельными функциональными блоками осуществляются шинами. Под шиной подразумевается физическая группа линий передачи сигналов, обладающих функциональной общностью (по каждой линии передается один двоичный разряд информации). Физически шины реализуются в виде параллельных проводящих полосок печатной платы или в виде связанных в жгут проводов. Так, например, в учебном микропроцессорном комплекте (прил. 1), организованном на основе микропроцессора серии 580, данные передаются к различным его функциональным узлам параллельно по восьми линиям. Эта группа из восьми линий передачи данных называется восьмиразрядной шиной данных. Кроме шины данных, в микроЭВМ выделяют шину адресов и шину управления, служащих соответственно для передачи адресов и управляющих сигналов. Микро-ЭВМ с такой организацией связей относят к системам, основанным на архитектуре с тремя шинами.

Линии шины адреса (ША) и шины управления (ШУ) являются однонаправленными, то есть сигналы по линиям этих шин передаются только в одном направлении. Передаваемые по ША сигналы формируются в МП. Они необходимы для определения пути передачи данных внутри микроЭВМ, например для выбора ячейки памяти, куда необходимо занести или откуда необходимо считать информацию. В определении тракта передачи данных могут принимать участие и управляющие сигналы, подсоединяющие или, напротив, блокирующие то или иное устройство микроЭВМ.

Разрядность шины адреса определяет количество ячеек памяти и внешних устройств, к которым может обратиться микропроцессор. В микроЭВМ организованной на основе микропроцессора серии 580, шина адреса является шестнадцатиразрядной, что позволяет обращаться к 65536 ячейкам памяти и внешним устройствам.

В отличие от ША и ШУ шина данных (ШД) является шиной двунаправленной. Данные по линиям шины могут передаваться от микропроцессора к какому-либо устройству микроЭВМ или пересылаться в МП от какого-либо устройства, доступ к которому обеспечивают сигналы адресной шины.

Естественно, что в каждый момент времени данные могут передаваться лишь в одном направлении, определяемом режимом работы микропроцессора. К основным режимам работы МП можно отнести:

1) запись данных в память машины;
2) чтение данных из памяти машины;
3) пересылку данных в устройство ввода/вывода;
4) чтение данных с устройства ввода/вывода;
5) выполнение операций с содержимым внутренних регистров

микропроцессора.

При реализации последнего режима внешние по отношению к МП шины микроЭВМ не используются, то есть все действия происходят внутри МП. Реализация первых четырех режимов оказывает определяющее влияние на работу шины данных.

Работа по реализации программы любой микроЭВМ, построенной по типу архитектуры с тремя шинами, состоит в выполнении следующих действий для каждой команды программы:

1) микропроцессор формирует адрес, по которому хранится команда, переводя в соответствующее состояние шину адреса;
2) команда считывается из памяти по сформированному адресу и пересылается в микропроцессор;
3) команда дешифрируется (идентифицируется) микропроцессором;
4) микропроцессор "настраивается" на выполнение одного из перечисленных выше пяти основных режимов в соответствии с результатами дешифрирования считанного из памяти кода команды.

Перечисленные выше пять режимов являются основными, но не единственно возможными.

1.1.2. Функциональная схема микропроцессора КР580ИК80А

При работе с микроЭВМ пользователю необходимо иметь информацию о структуре МП: числе и назначении регистров и специальных указателей; о системе команд. Число, назначение регистров, флагов и команд пользователь изменить не может. Он может изменить лишь содержимое регистров и использовать команды в любой нужной ему комбинации.

Под регистром подразумевается специальное запоминающее устройство, состоящее из элементов (триггеров), имеющих два устойчивых состояния. Число элементов регистра определяет разрядность. Большинство регистров микропроцессора восьмиразрядные и лишь некоторые шестнадцатиразрядные. Все регистры разбиты на группы и отличаются различным функциональным назначением. Некоторые регистры имеют специальное назначение, другие - многоцелевое. Количество и назначение регистров в микропроцессоре зависит от его архитектуры. Кроме того различают программно-недоступные и программно-доступные регистры.

Рассмотрим структуру восьмиразрядного микропроцессора (рис. 1.1). Основными блоками МП являются:

1) блок регистров общего назначения со схемой выборки регистров;
2) регистр команд с дешифратором команд и формирователем машинных циклов;
3) арифметическо-логическое устройство (АЛУ) с регистром А (аккумулятором), выполняющее арифметические и логические операции;
4) регистр временного хранения данных W и Z (РВХД);
5) флаговый регистр;
6) устройство управления и синхронизации;
7) буферы шины данных (БШД) и адреса (БША);
8) буфер аккумулятора (БА);

9) схема приращения и уменьшения (СПИУ).

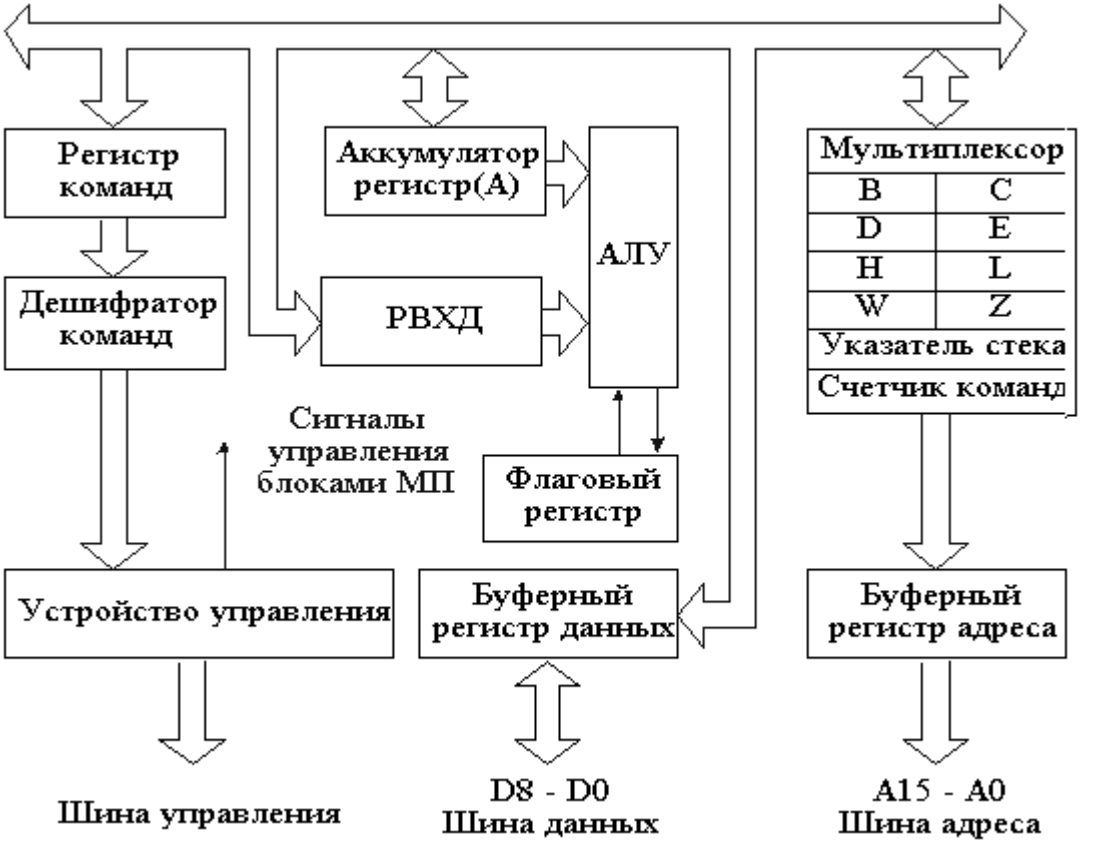

Рис. 1.1. Типовая структура микропроцессора

Программно-доступными являются следующие регистры:

1) восьмиразрядные регистры B, C, D, E, H, L, адресуемые по одному или парами;

2) восьмиразрядный регистр A (аккумулятор);

3) шестнадцатиразрядный указатель стека;

4) шестнадцатиразрядный счетчик команд (программный счетчик).

В некоторых специальных случаях могут быть доступными данные следующих двух регистров: восьмиразрядного регистра команд, пятиразрядного флагового регистра.

Программно недоступными пользователю являются регистры W и Z. Они используются для временного хранения данных при выполнении команд микропроцессором.

Рассмотрим назначение основных блоков и регистров.

Арифметическо-логическое устройство - выполняет арифметические и логические операции под воздействием устройства управления.

Регистры общего назначения размерностью в один байт обозначаются B, C, D, E, H, L. Они используются для хранения данных и промежуточных результатов вычислений, выполняемых с помощью арифметическо-логического устройства. При обработке шестнадцатиразрядных слов возможно обращение к парам регистров (B, C) ; (D, E); (H, L).

Аккумулятор - специальный однобайтовый регистр, обозначаемый A. При

выполнении арифметических и логических операций служит источником одного из операндов и местом запоминания результата выполнения операции.

Регистр команд - однобайтовый регистр, в котором хранится код выполняемой команды. Этот регистр непосредственно пользователю недоступен. Это означает, что не существует команды, которая бы явным образом могла изменить его содержимое. После выполнения очередной команды в регистр команд автоматически записывается код следующей команды из ячейки оперативной памяти, адрес которой содержится в счетчике команд.

Счетчик команд - двухбайтовый (шестнадцатиразрядный) регистр, обозначаемый PC. Этот регистр хранит адрес следующей команды, которая должна быть выполнена вслед за предыдущей. Счетчик команд автоматически получает приращение хранимого в нем адреса в зависимости от того, какую по длительности команду (в один, два или три байта) микропроцессор считывает из памяти, указывая всегда на 1-й байт следующей команды. На содержимое этого регистра пользователь может повлиять только с помощью команд, изменяющих последовательное выполнение программы (например, команд безусловного перехода), а также с помощью некоторых специальных команд.

Указатель стека - двухбайтовый (шестнадцатиразрядный) регистр, обозначаемый SP. Этот регистр хранит адрес вершины стека. Стеком называется специально ориентированная область памяти, используемая для временного хранения данных или адресов. Особенностью работы стека является то, что число, записанное в стек последним, извлекается из него первым. Используя стек, легко в программах организовывать переходы, вызов подпрограмм и выполнение прерываний работы микропроцессора.

Флаговый регистр представляет собой набор флагов. Каждый флаг предназначен для хранения одного признака результата выполнения операции.

Микропроцессор КР580ИК80А содержит флаговый регистр, состоящий из следующих флагов: флага нуля Z; флага переноса C; флага знака S; флага четности P; флага дополнительного переноса AC.

Флаги всегда устанавливаются или сбрасываются автоматически после выполнения очередной команды, влияющей на флаги, в зависимости от результата операции. При этом флаг считается установленным, если флаговый разряд принимает значение 1, и сброшенным, если значение разряда 0. Состояния флагов используются в командах условного перехода.

Результаты выполнения арифметических и логических операций оказывают влияние на флаги следующим образом:

1) флаг нуля устанавливается в состояние 1, если после выполнения какой-либо команды получен нулевой результат и сбрасывается в 0 в случае не нулевого результата;

2) флаг переноса устанавливается в состояние 1, если в результате операций сложения и сдвига появляется единица переноса из старшего разряда байта данных, а также, если появляется заем из старшего разряда после выполнения операций вычитания или сравнения, в противном случае, флаг сбрасывается в 0;

3) флаг знака устанавливается в состояние 1, если в результате выполнения операций появляется логическая единица в старшем разряде байта данных (указание на отрицательный результат), и сбрасывается в 0 в случае нулевого значения старшего разряда (указание на положительный результат);

4) флаг четности устанавливается в состояние 1, если после выполнения операций сумма единиц в байте данных, подсчитываемых с помощью операции сложения по модулю 2, четна (значение суммы по модулю 2 равно 0), и сбрасывается в 0 в противном случае (число единиц - нечетное);

5) флаг дополнительного переноса устанавливается в состояние 1, если в результате выполнения команды появляется сигнал переноса из третьего разряда в четвертый в байте данных результата. Если такого переноса нет, флаг дополнительного переноса сбрасывается в 0. Сигнал этого флага используется во многих схемах вычислений, однако он особенно необходим при сложении чисел в двоично-десятичной форме.

1.2. Подготовка в выполнению лабораторной работы

Перед началом выполнения лабораторной работы № 1, используя указанную литературу, необходимо выполнить следующее:

1) ознакомиться с описанием учебного микропроцессорного комплекта и порядком работы с ним (прил. 1);

2) ознакомиться с типовой минимальной структурой микроЭВМ, методами организации магистралей, подключением памяти и внешних устройств к магистралям [1 - 3];

3) изучить состав и назначение внутренних регистров микропроцессора КР580ИК80 [1 - 3].

1.3. Порядок выполнения работы

Лабораторная работа выполняется в соответствии с вариантом задания (подраздел 1.5) и включает в себя три задания, выполняемые последовательно.

1.3.1. Задание 1**.** Изучение порядка включения микроЭВМ

Порядок выполнения:

1) подключить шнур питания к сети 220 В;

2) установить переключатель **"РБ/ШГ"** в состояние **"РБ"**;

3) включить УМК, нажав кнопку СЕТЬ (~);

4) нажать управляющую кнопку **"СБ".**

В результате в крайней левой позиции дисплея должен появиться знак **"-"**. Находясь в этом режиме, УМК реагирует на нажатие директивных клавиш. Вызов различных директив из этого состояния УМК определяется алгоритмом управляющей программы "Монитор". Из любого места управляющей программы можно вернуть УМК к начальному состоянию нажатием клавиши **"СБ"**.

1.3.2. Задание 2. Исследование содержимого памяти и запись данных в память УМК

Порядок выполнения:

1) произвести перевод УМК в начальное состояние кнопкой **"СБ"**;

2) произвести исследование области памяти согласно варианта задания. В результате исследования необходимо заполнить табл. 1.1;

3) произвести запись данных, считанных в предыдущем пункте задания, в оперативное запоминающее устройство (ОЗУ), начиная с ячейки памяти с адресом 0800.

Таблица 1.1

Результаты исследования области памяти УМК

Адрес ячейки памяти	Содержимое ячейки памяти
.	.

1.3.3. Задание 3. Исследование содержимого и запись данных в программно-доступные регистры микропроцессора

Порядок выполнения:

1) произвести перевод УМК в начальное состояние кнопкой **"СБ"**;

2) произвести исследование содержимого всех программно-доступных регистров. По результатам исследования необходимо заполнить табл. 1.2;

3) произвести запись в счетчик команд шестнадцатиразрядного адреса и в регистр А согласно варианту задания (подраздел 1.5).

Таблица 1.2

Результаты исследования программно-доступных регистров

Идентификатор регистра	Содержимое регистра
.	.

1.4. Содержание отчета

Отчет по лабораторной работе оформляется в соответствии с требованиями государственного стандарта и должен содержать:

1) титульный лист;
2) описание и цели работы;
3) структурную схему учебного микропроцессорного комплекта;
4) типовую структурную схему микропроцессора;
5) таблицу распределения памяти учебного микропроцессорного комплекта;
6) результаты исследования области памяти;
7) информацию о содержимом программно-доступных регистров микропроцессора;
8) выводы о проделанной работе.

1.5. Варианты заданий к лабораторной работе

Варианты заданий на выполнение лабораторной работы представлены в табл. 1.3. Все адреса и данные в таблице представлены в шестнадцатеричной системе счисления.

Таблица 1.3

Варианты задания на выполнение лабораторной работы № 1

Номер варианта	Область исследуемой памяти	Шестнадцатиразрядный адрес	Восьмиразрядное данное
1	005A - 0078	0101	11
2	0079 - 009C	0022	AF
3	009D - 00BE	0354	F2
4	00D9 - 00FE	05A5	5A
5	00FF - 0018	02FE	9B
6	0119 - 013B	06C1	C7
7	015C - 0178	09A0	69
8	0179 - 0192	0A0D	91
9	0193 - 01B0	067F	1F
10	01B1 - 01D1	0C2B	23
11	01DF - 0203	03A9	35
12	0204 - 021E	0555	DD
13	021F - 0238	04AA	00
14	0239 - 0258	07FF	F0
15	0259 - 0275	0B49	7B

Лабораторная работа № 2.
Запись и выполнение простых программ

Цель работы: изучение различных способов адресации микропроцессора, изучение основ программирования микропроцессора, исследование выполнения отдельных команд и простых программ, использование различных методов адресации в программах, запись программ в память микроЭВМ и их выполнение.

2.1. Краткие теоретические сведения

2.1.1. Программирование микроЭВМ

Любая микроЭВМ может делать только то, что ей предписывает человек. Программа для микроЭВМ - это последовательность команд, которые микроЭВМ распознает и в соответствии с этим выполняет определенные действия. Каждая команда инициирует выполнение определенного действия.

Каждая микроЭВМ способна воспринимать и выполнять точно определенный для нее набор команд. Количество и тип команд изменяются в зависимости от возможностей и назначения микроЭВМ.

Различают три основных уровня программирования:

1) программирование в машинных кодах;
2) программирование в кодах языка Ассемблера;
3) программирование на языках высокого уровня.

Программирование в машинных (двоичных) кодах - трудоемкая и сложная для человека задача. Будучи языком цифр, он неудобен для описания вычислительных процессов, требует от программиста больших усилий при написании и отладке программ.

К достоинствам данного вида программирования можно отнести следующее: для написания программ требуется лишь знания системы команд конкретной микроЭВМ, а для выполнения написанных таким образом программ микроЭВМ не нуждается ни в каком дополнительном программном обеспечении. Кроме того, при использовании машинного языка можно достигнуть максимальной гибкости в реализации технических возможностей микроЭВМ.

Для упрощения процесса написания, отладки и чтения программы используют мнемонический (символический) код. Каждую машинную команду представляют простым трех- или четырехбуквенным мнемоническим символом. Мнемонические символы значительно легче связать с машинными операциями, потому что их можно выбрать таким образом, чтобы они напоминали названия команд.

Программу, написанную в мнемонических кодах, необходимо транслировать в машинные коды. Сделать это можно двумя способами: вручную с помощью таблицы соответствия системы команд для конкретной микроЭВМ (прил. 3) или специальной программой, называемой Ассемблером. Ассемблер сравнивает каждую мнемоническую команду со списком команд и заменяет ее двоичным эквивалентом. Такой процесс получил название программной трансляции.

Язык Ассемблера, получивший свое название от имени программы, преобразующей программу на таком языке, в машинные коды, имеет ряд преимуществ. Он позволяет, наряду с программированием в машинных кодах, гибко и полно реализовать технические возможности микроЭВМ. В то же время, использование языка Ассемблера избавляет программиста от трудоемкой работы с машинными двоичными кодами.

Язык Ассемблера является машинно-зависимым (машинно-ориентированным) и, следовательно, отражает аппаратурные особенности той микроЭВМ, для которой создан. Поэтому программы, составленные с использованием языка Ассемблера (или машинного языка), обладают наименьшей мобильностью по отношению к различным микроЭВМ.

Большей мобильностью обладают программы, составленные на алгоритмических языках высокого уровня (например Фортран, Бейсик, ПЛ/М, Паскаль, Ада, СИ и др.), занимающих верхнее положение в иерархии языков программирования. Будучи приближенными к привычной математической нотации и, в ряде случаев обеспечивая естественную форму описания вычислительных процессов, они достаточно просты и удобны в программировании, но не всегда позволяют в полной мере реализовать технические возможности ЭВМ. Кроме того, результирующие машинные программы, получаемые после трансляции программ с алгоритмических языков, обычно менее эффективны с точки зрения объема и быстродействия, по сравнению с программами, написанными в кодах языка Ассемблера и в машинных кодах. Применение языков высокого уровня предполагает обязательное наличие транслятора, представляющего собой сложный программный комплекс.

2.1.2. Выполнение команд микропроцессором

Микропроцессор КР580ИК80А имеет фиксированный набор команд. Работа микропроцессора по реализации каждой команды программы основана на принципе микропрограммного управления. То есть, каждая команда реализуется как некоторая последовательность микрокоманд (или микроопераций), приводящая к требуемому результату.

Считываемая из памяти микропроцессором команда, точнее ее восьмиразрядный двоичный код (код операции), поступает в регистр команд, где и хранится в течение времени ее выполнения. По результату дешифрирования кода операции происходит формирование последовательности микроопераций, процесс выполнения которой и определяет все последующие операции по выполнению считанной команды. При этом выполнение отдельных микроопераций синхронизируется сигналами тактового генератора.

В микропроцессоре выполнение каждой команды можно разбить на ряд основных операций. Время, отведенное на выполнение операции обращения к памяти или устройству ввода-вывода, составляет машинный цикл. Следовательно, процесс выполнения команды состоит из стольких машинных циклов, сколько обращений к памяти или к устройствам ввода-вывода требуется для ее выполнения. Каждая команда в зависимости от ее вида может

занимать от одного до пяти машинных циклов.

Микропроцессор КР580ИК80А имеет 10 типов машинных циклов:

1) выборка команды из памяти;
2) чтение из памяти;
3) запись в память;
4) чтение из стека;
5) запись в стек;
6) ввод с внешнего устройства;
7) вывод на внешнее устройство;
8) прерывание;
9) останов;
10) прерывание во время останова.

В свою очередь, каждый машинный цикл может состоять из 3 - 5 машинных тактов. Под машинным тактом подразумевается интервал времени, соответствующий одному периоду тактовых импульсов.

На рис. 2.1 приведена временная диаграмма команды LDA (Загрузить данные в аккумулятор из яяейки памяти, адрес которой находится во втором и третьем байте команды). Данная команда состоит из четырех машинных циклов: С1 - выборка команды (код команды помещается в регистр команд); С2 - обращение к памяти (восемь младших разрядов адреса выбираются из памяти и помещаются в регистр Z); С3 - обращение к памяти (восемь старших разрядов адреса выбираются из памяти и помещаются в регистр W); С4 - аккумулятор загружается данными из памяти по адресу, содержащемуся в регистровой паре WZ. В свою очередь первый машинный цикл С1 команды LDA состоит из четырех машинных тактов Т1 ? Т4, а машинные циклы С2 ? С4 - из трех.

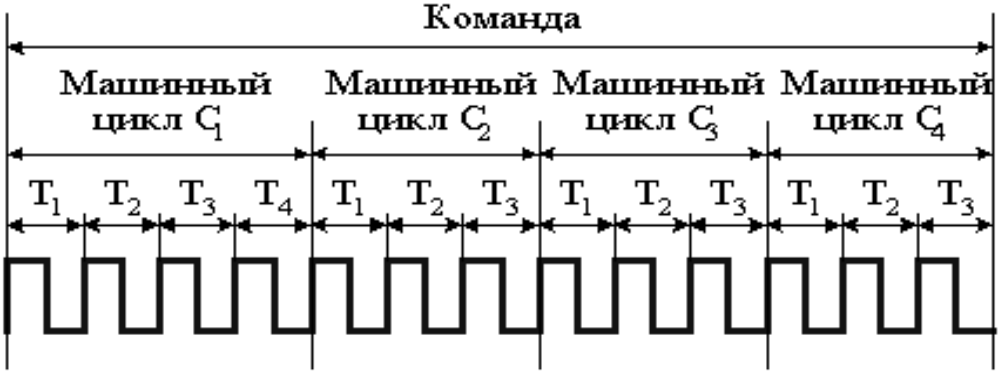

Рис. 2.1. Временная диаграмма команды LDA

Система команд микропроцессора КР580ИК80А представлена 244 кодами операций. Полный перечень команд микропроцессора приведен в прил. 2.

Все команды можно классифицировать по трем основным признакам: по длине команды; по виду адресации к данным; по функциональному признаку.

По длине команды делятся на однобайтовые, двухбайтовые и трехбайтовые. Двухбайтовые и трехбайтовые команды в памяти занимают соответственно две или три последовательно расположенные ячейки. При этом первый байт всегда отводится под код операции, а второй или второй и третий байты содержат либо данные, либо адрес, по которому они находятся в памяти.

2.1.3. Система команд и способы адресации микропроцессора КР580ИК80А

В зависимости от способа адресации различают команды с непосредственной, прямой, регистровой и косвенной адресацией.

При непосредственной адресации необходимые данные содержит сама команда. Эти данные содержатся во втором и третьем байтах трехбайтовой команды или во втором - двухбайтовой. В случае трехбайтовой команды младшие разряды шестнадцатибитового числа содержатся во втором байте команды, а старшие - в третьем (рис.2.2, а).

В случае прямой адресации во втором и третьем байтах команд содержится полный шестнадцатиразрядный адрес памяти. Младшим байтом является второй байт команды В2, а старшим - третий (рис. 2.2, б). Таким способом можно обратиться к любой ячейке адресного пространства памяти.

При регистровой адресации код команды содержит указания на регистр или пару регистров, в которых содержатся данные. Используемые в регистровой адресации команды являются однобайтовыми (рис. 2.2, в).

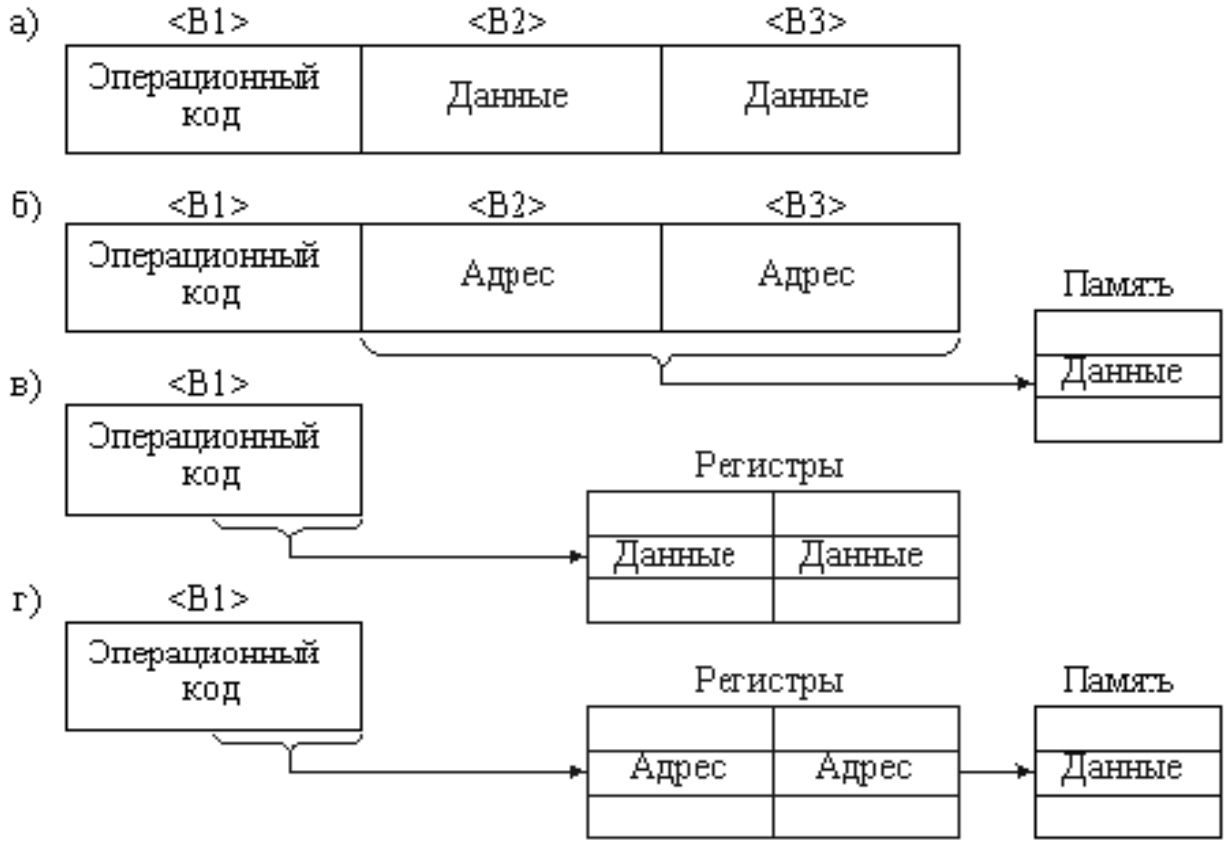

Рис. 2.2. Способы адресации: а) непосредственная; б) прямая; в) регистровая; г) косвенная

Последним способом является косвенная адресация, при которой в регистровой паре микропроцессора, определяемой кодом команды, содержится полный шестнадцатиразрядный адрес ячейки памяти, содержащей данные (рис. 2.2, г). Старший байт адреса записывается в первом регистре пары, а младший байт - во втором.

При всевозможных пересылках данных: из регистров в регистры или из памяти в регистры и обратно, различают регистры-источники и регистры-приемники данных. При использовании регистровых пар B,C; D,E и H,L старшими являются первые регистры пар, то есть B,D и H соответственно. Коды регистров общего назначения, пар регистров и флагов строго фиксированы (табл. 2.1).

Таблица 2.1

Коды регистров и обозначение условий в мнемокодах

Регистр	Код	Пара регистров	Код	Обозначение условия в мнемокоде	Код
A	111	B (B, C)	00	NZ (Z = 0)	000
B	000	D (D, E)	01	Z (Z = 1)	001
C	001	H (H, L)	10	NC (CY = 0)	010
D	010	SP	11	C (CY = 1)	011
E	011			PO (P = 0)	100
H	100			PE (P = 1)	101
L	101			P (S = 0)	110
M (память)	110			M (S = 1)	111

По функциональному признаку все команды могут быть разделены на пять групп:

1) команды пересылки данных;
2) арифметические команды;
3) логические команды;
4) команды переходов;
5) команды управления и работы со стеком.

К группе команд пересылки данных относятся команды, пересылающие содержащиеся в них данные в регистры или в память, команды пересылки данных между регистрами микропроцессора и между регистрами и памятью. В мнемонике команд регистр-приемник указывается вслед за символическим кодом команды, а через запятую - регистр-источник (прил. 2). Например: команда MOV M, A - пересылает данные из регистра A (аккумулятора) в ячейку памяти по адресу указанному в регистровой паре H,L.

Арифметические команды предназначены для выполнения арифметических операций над данными, хранящимися в регистрах и ячейках памяти. Все команды этой группы оказывают влияние на значения разрядов флагового регистра (прил. 4), так как при их выполнении меняются знаки используемых чисел, возникают сигналы переноса, появляются нулевые результаты и т.п.

Логические команды служат для выполнения логических (или булевых) операций над данными, содержащимися в регистрах, ячейках памяти, а так же над флагами регистров. К ним относятся следующие операции: инверсия (НЕ),

логическое суммирование (ИЛИ), логическое умножение (И), суммирование по модулю 2, сравнение, сдвиг, дополнение до 1 и до 2. Как и команды предыдущей группы, все логические команды оказывают влияние на флаги состояния.

Команды переходов предназначены для организации правильной последовательности выполнения программы. В эту группу входят команды безусловного и условного переходов, команды вызова подпрограммы и возвращения к главной программе. Все команды этой группы влияния на флаги состояния не оказывают. Команды безусловного перехода выполняют специальные операции над содержимым счетчика команд. Команды условного перехода обеспечивают необходимое ветвление программы путем анализа состояния одного из четырех флагов: нуля, знака, четности и переноса.

Команды управления и работы со стеком служат для управления работой микропроцессора, устройствами ввода-вывода и стеком. Команды этой группы не оказывают влияния на флаги состояния.

2.1.4. Пример программирования

Рассмотрим простейшую программу производящую извлечение числа из ячейки памяти с адресом 0B0016, операцию инвертирования данного числа и запись полученного результата в ячейку памяти с адресом 0B0A16. При составлении программы используем прямой вид адресации. Пользуясь набором команд микропроцессора КР580ИК80А, составим программу (табл. 2.2).

Таблица 2.2

Программа записанная в мнемокодах языка Ассемблера

Мнемокод	Комментарий
LDA 0B00	Записать в аккумулятор содержимое ячейки памяти 0B00
CMA	Инвертировать содержимое аккумулятора (логическая операция НЕ)
STA 0B0A	Записать содержимое аккумулятора в ячейку памяти 0B0A
HLT	Остановить программу

При разработке программ все данные и адреса в командах записываются в шестнадцатеричной системе счисления.

Для записи программы в память микроЭВМ необходимо перевести мнемокоды команд в машинные коды. Так как команды в программе могут быть одно-, двух- или трехбайтовые, следовательно они должны занимать соответственно одну, две или три ячейки памяти. Пользуясь таблицей соответствия мнемокодов и машинных кодов (прил. 3), запишем программу в машинных кодах с указанием адресов ячеек памяти каждого байта программы (табл. 2.3).

Таблица 2.3

Программа записанная в машинных кодах с размещением по адресам памяти

Адрес	Число	Комментарий
0800	3A	Код команды LDA
0801	00	Младший байт адреса
0802	0B	Старший байт адреса
0803	2F	Код команды CMA
0804	32	Код команды STA
0805	0A	Младший байт адреса
0806	0B	Старший байт адреса
0807	76	Код команды HLT

Предварительную запись программ удобно проводить в более компактной форме. В программе указывается начальный адрес каждой команды, при этом в зависимости от длины, команды в памяти будут занимать от одной до трех последовательных ячеек. При такой записи в левом столбце указываются адреса команд в программе (табл. 2.4).

Таблица 2.4

Программа записанная в общем виде

Адрес	Машинный код	Метка	Мнемокод	Комментарий
0800	3A 000B		LDA, 0B00	Записать в аккумулятор содержимое яяейки памяти 0B00
0803	2F		CMA	Инвертировать содержимое аккумулятора
0804	32 0A0B		STA, 0B0A	Записать содержимое аккумуля-тора в ячейку памяти 0B0A
0807	76		HLT	Остановить программу

2.2. Подготовка к выполнению лабораторной работы

Перед началом выполнения лабораторной работы № 2, используя указанную литературу, необходимо выполнить следующие задания:

1) изучить форму записи чисел в различных системах счисления: двоичной, восьмеричной, шестнадцатеричной [1, 3, 4].

2) изучить структуру команд и способы адресации микропроцессора КР580ИК80А (прил. 2), [4 , 5];

3) ознакомиться со способами записи программ (табл. 2.2 - 2.4).

2.3. Порядок выполнения

Лабораторная работа выполняется в соответствии с вариантом задания (подраздел 2.5) и состоит из трех заданий, выполняемых последовательно.

2.3.1. Задание 1.Составление программы с использованием прямой адресации

Порядок выполнения:

1) составить и записать программу решения примера варианта задания в мнемокодах языка Ассемблера, максимально используя команды с прямой адресацией;

2) пользуясь таблицей соответствия мнемокодов языка Ассемблера и машинных кодов (прил. 3), записать составленную программу в машинных кодах с указанием адресов ячеек памяти;

3) записать программу в общем виде.

2.3.2. Задание 2. Исследование программы

Порядок выполнения:

1) подготовить УМК к работе;

2) ввести в память УМК составленную программу;

3) записать по указанным в варианте задания адресам исходные данные;

4) осуществить пуск программы в пошаговом поцикловом режиме с начального адреса программы;

5) исследовать процесс выполнения программы. После выполнения каждого шага программы производить считывание со световой индикации УМК адреса ячейки памяти, к которой обращается микропроцессор, данных и состояния операционного устройства УМК. На основе кода состояния определить тип машинного цикла микропроцессора (прил. 1, табл. 2). Результаты записать в табл. 2.5;

6) после завершения работы программы проверить результат считыванием числа, записанного по адресу, указанному в варианте задания. Результат записать в табл. 2.6.

Таблица 2.5

Результаты исследования процесса выполнения программы

Шаг	Адрес	Данные	Тип машинного цикла
1 2 n	. . .	. . .	. . .

Таблица 2.6

Результаты выполнения программ

Используемая адресация	Результат
1. Прямая	...
2. Косвенная	...

2.3.3. Задание 3. Исследование различных способов адресации

Порядок выполнения:

1) составить и записать в общем виде программу решения примера варианта задания, максимально используя команды с косвенной адресацией;

2) загрузить программу в память УМК;

3) осуществить пуск программы в рабочем цикловом режиме;

4) после останова программы произвести считывание результата выполнения программы, полученный результат записать в табл. 2.6.

2.4. Содержание отчета

Отчет по лабораторной работе оформляется в соответствии с требованиями государственного стандарта и должен содержать:

1) титульный лист;

2) описание и цели работы;

3) задание на лабораторную работу;

4) программу, использующую прямую адресацию, записанную в мнемокодах, в машинных кодах и в общем виде;

5) результаты исследования программы, использующей прямую адресацию;

6) общий вид программы, написанной с использованием косвенной адресации;

7) результаты выполнения программ с различной адресацией к данным.

2.5. Варианты заданий к лабораторной работе

В вариантах заданий на лабораторную работу все адреса и данные представлены в шестнадцатеричной системе счисления.

Вариант 1. Произвести сложение чисел 10 и A6, хранящихся по адресам 0C23 и 0C24. Полученный результат инвертировать и конечный результат записать по адресу 0C30. Остановить программу.

Вариант 2. Инвертировать число EA, хранящееся по адресу 0E93. Полученный результат вычесть из числа B6, хранящегося по адресу 0E90. Конечный результат записать по адресу 0E96. Остановить программу.

Вариант 3. Инвертировать число 12, хранящееся по адресу 0D23, полученный результат сложить с числом 21, хранящимся по адресу 0D32, конечный результат записать по адресу 0D22. Остановить программу.

Вариант 4. Выполнить логическое сложение чисел 45 и B4, хранящихся по адресам 0E11 и 0C29. Полученный результат инвертировать и конечный результат записать по адресу 0C30. Остановить программу.

Вариант 5. Выполнить логическое умножение чисел 8A и A3, хранящихся по адресам 0C09 и 0C11. Полученный результат инвертировать и конечный результат записать по адресу 0C13. Остановить программу.

Вариант 6. Выполнить операцию инвертирования чисел 92 и 48, хранящихся по адресам 0CC0 и 0CC2. Полученные результаты сложить и конечный результат записать по адресу 0CC5. Остановить программу.

Вариант 7. Выполнить операцию инвертирования чисел FA и 21, хранящихся по адресам 0F11 и 0C22. Над полученными результатами выполнить операцию логического сложения и конечный результат записать по адресу 0F22. Остановить программу.

Вариант 8. Выполнить операцию инвертирования числа 02, хранящегося по адресу 0E18. Из полученного результата вычесть число CC, хранящееся по адресу 0D19. Результат операции вычитания инвертировать и конечный результат записать по адресу 0C20. Остановить программу.

Вариант 9. Выполнить логическое сложение чисел E6 и 67, хранящихся по адресам 0E99 и 0D76. Полученный результат инвертировать. К результату инвертирования прибавить число A2. Конечный результат записать по адресу 0D78. Остановить программу.

Вариант 10. Выполнить сложение чисел 53 и 99, хранящихся по адресам 0F80 и 0F88. Из результата сложения вычесть число 3F. Полученный результат инвертировать и конечный результат записать по адресу 0FFF. Остановить программу.

Вариант 11. Выполнить логическое сложение числа 9F, хранящегося по адресу 0C00, и числа 0A. Из полученного результата вычесть число 7D, хранящееся по адресу 0C61. Конечный результат записать по адресу 0E01. Остановить программу.

Вариант 12. Инвертировать число 30. К результату прибавить число 19, хранящееся по адресу 0D31. Полученный результат инвертировать. Из результата инвертирования вычесть число 6B, хранящееся по адресу 0D42. Конечный результат записать по адресу 0C00. Остановить программу.

Вариант 13. Выполнить логическое умножение числа 69, хранящегося по адресу 0CCC, и инвертированного значения числа A6. К результату прибавить число B9, хранящееся по адресу 0D00. Результат сложения записать по адресу 0C40. Остановить программу.

Вариант 14. Выполнить инвертирование чисел 04 и 3E, хранящихся по адресам 0D0F и 0D1C. Произвести логическое сложение инвертированных значений чисел 04 и 3E. К результату логического сложения прибавить число 03. Конечный результат записать по адресу 0D22. Остановить программу.

Вариант 15. Инвертировать числа AC и B4, хранящиеся по адресам 0C2F и 0C48. Полученные результаты сложить. Над результатом сложения и числом DA выполнить операцию логического умножения. Конечный результат записать по адресу 0C20. Остановить программу.

Лабораторная работа № 3.
Подпрограмма и стек

Цель работы: исследование особенностей записи и обращения к подпрограммам; изучение способов организации стека и методов использования стека при создании программ.

3.1. Краткие теоретические сведения

3.1.1. Подпрограммы

Команды микропроцессора с функциональной точки зрения довольно просты. Поэтому при написании программ для микроЭВМ даже несложные процедуры приходится реализовывать сравнительно длинными последовательностями команд. Очевидно, что многие из этих процедур встречаются в алгоритмах решения задачи неоднократно. Поэтому, целесообразно оформлять эти процедуры в виде независимых программных модулей - подпрограмм, которые выполняются при обращении к ним по специальным командам перехода к выполнению подпрограмм.

Главным достоинством использования подпрограмм является:

- возможность оформления каждой подпрограммы в виде относительно короткой конструкции, удобной для проверки и отладки;
- простота организации связи основной программы и подпрограммы;
- сокращение объема памяти, занимаемой программой.

Процесс передачи управления подпрограмме называется вызовом. Данные и адреса, требуемые для работы подпрограммы, называются входными параметрами, а результаты работы подпрограммы, передаваемые по окончании ее работы в основную программу, называются выходными параметрами.

Использование подпрограмм настолько эффективно, что в ряде случаев целесообразно организовать основную программу в виде последовательности только переходов к подпрограммам, каждая из которых может вызывать новые подпрограммы и т.д.

В качестве команд перехода к подпрограммам используются команды:

- ***безусловного перехода*** к подпрограммам с возвратом (БПВ);
- ***условных переходов*** к подпрограммам с возвратом (УПВ).

Команда БПВ содержит адрес, загружаемый в счетчик команд. Это вызывает передачу управления соответствующей ячейке памяти, с которой начинается последовательность команд подпрограммы. После окончания выполнения подпрограммы команда БПВ предполагает возврат к продолжению прерванной основной программы, поэтому должна заканчиваться командой ВОЗВРАТ.

Выполнение команды УПВ отличается от выполнения команды БПВ предварительным анализом указанного в команде признака (анализируется один из разрядов флагового регистра) и переходом к подпрограмме при его определенном значении. Наличие команды УПВ нескольких модификаций позволяет упростить переход к подпрограммам в случаях, когда такой переход должен происходить лишь при выполнении некоторых условий. Аналогично для упрощения возврата из подпрограммы, заканчивающейся циклами, в системе команд микропроцессора имеется группа команд УСЛОВНЫЙ ВОЗВРАТ. По этим командам возврат в основную программу происходит

только при выполнении соответствующих условий.

Кроме команд перехода к подпрограммам в набор команд микропроцессора входят команды безусловных и условных переходов в заданную область памяти. По команде безусловного перехода в регистр счетчика команд записывается указанный в команде адрес. Следовательно следующей командой, которую будет выполнять микропроцессор, будет команда, находящаяся по адресу, занесенному в регистр счетчика команд. При появлении в программе команды условного перехода передача управления по адресу, указанному в команде, происходит только в результате выполнения определенного условия (аналогично выполнению команд условных переходов к подпрограммам с возвратом).

3.1.2. Назначение и устройство стека

Мощным средством повышения эффективности использования подпрограмм является стек.

Стек - это набор регистров или ячеек оперативной памяти, служащий для временного хранения адресов возврата из подпрограмм или состояния внутренних регистров при обработке прерываний. Работа стека организована таким образом, что данное или адрес, записанное в стек последним, будет считано первым.

В микропроцессорах используются два вида организации стека: встроенный и автономный.

Встроенный стек размещается полностью на кристалле микропроцессора и представляет собой набор внутренних регистров. Работа встроенного стека показана на рис. 3.1. Если в стек загружается новый элемент (адрес или данное) **D5**, то он записывается в верхнем регистре, а каждый из элементов **D1...D4**, уже находившихся в стеке, перемещается в соседние нижние регистры. Если же **D5** извлекается из стека, то каждый из элементов **D1...D4** перемещается в соседние верхние регистры. Таким образом, нельзя извлечь из стека элемент **D4** раньше **D5**. Емкость (или глубина) стека при такой организации не может быть большой и, как правило, не превышает 16 - 32 элемента.

Отличие автономного стека от встроенного заключается в том, что в качестве самого стека используется область ОЗУ, являющаяся внешней по отношению к микропроцессору. При такой организации стека необходим специальный регистр микропроцессора - указатель стека SP, для хранения адреса последнего по времени поступления элемента в стек (вершины стека). Разрядность указателя стека равна разрядности шины адреса.

Рассмотрим работу стека при автономной организации. На рис. 3.2 указатель стека представляет собой шестнадцатиразрядный регистр. Первоначально указатель стека содержит адрес $0FEC_{16}$. Это означает, что последний элемент находится в ячейке памяти с адресом $0FEC_{16}$. Для записи в стек нового элемента **D5** используется ячейка памяти, адрес которой на единицу меньше адреса $0FEC_{16}$, то есть адрес $0FEB_{16}$. При записи **D5** в стек, содержимое указателя стека уменьшается на единицу. В результате указатель стека содержит адрес ячейки памяти, в которую был записан последний элемент **D5**. При извлечении элемента из стека производятся обратные действия. Из ячейки

памяти, адрес которой указан в указателе стека, считывается последний по времени записи элемент, а содержимое указателя стека увеличивается на единицу.

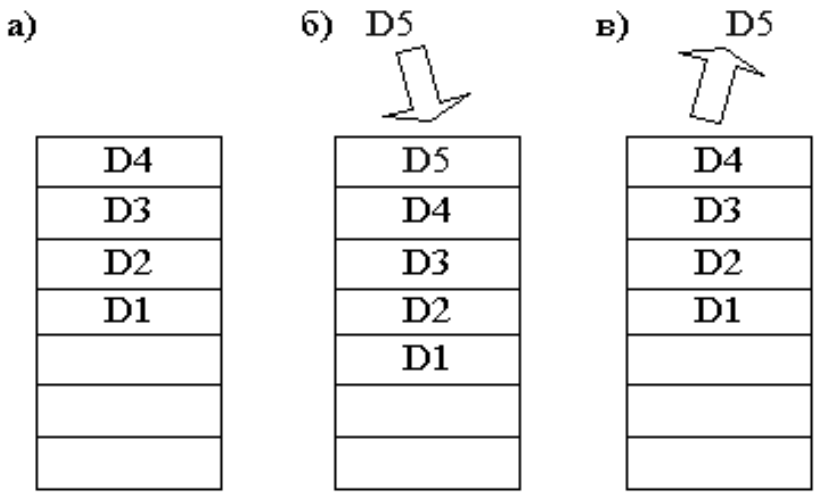

Рис.3.1. Принцип работы встроенного стека: а)исходное состояние; б) загрузка данного D5 в стеке; в) извлечение данного D5 из стека

Для вызова подпрограмм используются команды управления **Call** (например, CALL <B2> <B3>, CNZ <B2> <B3> и др.), а для возврата - **Return** (например, RET, RNZ и др.).

Каждый раз, когда процессор получает для исполнения команду **Call**, в содержимом счетчика команд происходит приращение и его информация записывается в ячейку памяти, адрес которой сохраняется в указателе стека. Затем в счетчик команд загружается содержимое второго и третьего байтов команды (<B2> и <B3>), причем содержимое второго байта <B2> записывается в младший байт счетчика команд, а третьего <B3> - в старший. Последняя команда подпрограммы **Return** обеспечивает считывание адреса команды возврата в основную программу из памяти по адресу, хранящемуся в указателе стека. Так как разрядность ячейки памяти равна восьми и для сохранения адреса возврата требуется две ячейки памяти, при каждой записи в стек содержимое указателя стека уменьшается на два, а при каждом считывании - увеличивается на два.

Автоматическое сохранение и восстановление адреса основной программы при обращении к подпрограммам позволяет сделать подпрограммы вложенными, то есть осуществлять вызов одной подпрограммы из другой. Уровень вложенности определяется лишь размером стека.

Помимо команд вызова подпрограмм и возврата из них, со стеком можно обмениваться информацией с помощью команд PUSH r (запись в стек содержимого регистра r) и POP r (запись данных из стека в регистр r). Эти команды являются однобайтовыми, и в них содержится указание пары регистров микропроцессора.

При записи с стек содержимого пары регистров содержимое старшего регистра регистровой пары записывается по адресу (SP-1), а содержимое

младшего регистра - по адресу (SP-2).

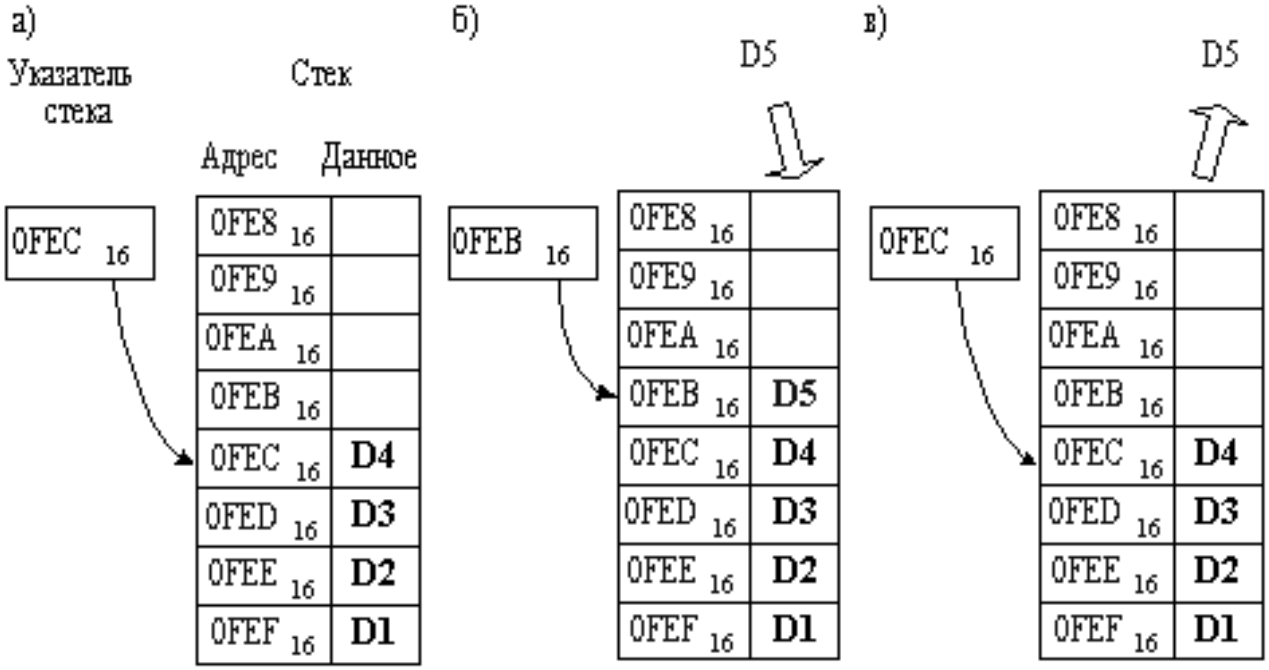

Рис. 3.2. Принцип работы автономного стека:
а) исходное состояние;
б) операция загрузки данного в стек;
в) операция извлечения данного из стека

При записи из стека данных в пару регистров в младший регистр пары записывается число из ячейки памяти по адресу (SP), а в старший регистр пары - число, записанное по адресу (SP+1). В результате выполнения команды **POP** данные в памяти не изменяются, а лишь происходит их чтение и увеличение содержимого указателя стека.

Все операции со стеком должны быть сбалансированы, то есть каждая подпрограмма должна содержать равное количество команд PUSH и POP и оканчиваться командой выхода из подпрограммы. В противном случае, выполнение команды возврата в конце подпрограммы приведет к записи в счетчик команд случайного числа из стека, что приведет к нарушению последовательности выполнения программы.

3.1.3. Блок-схема программы

Блок-схемы используются для наглядного представления структуры программы и являются графической формой представления алгоритма программы. Алгоритм изображается в виде последовательно связанных между собой функциональных блоков, каждый из которых соответствует выполнению одного или нескольких действий (команд).

Для изображения блок-схем используются специальные графические символы, служащие для обозначения функциональных блоков алгоритма программы (прил. 5). Последовательность выполнения пунктов алгоритма, описываемого схемой, устанавливается путем упорядочения размещения

блоков на схеме и объединения их линиями, которые называются линиями потока информации.

В зависимости от поставленной задачи блок-схема программы может изображаться с различной степенью детализации. При необходимости в ней могут быть помещены различные словесные комментарии и примечания.

Разработка алгоритма в виде блок-схемы является начальным этапом составления программы. На основе разработанной блок-схемы составляется текст программы, поэтому при разработке блок-схемы важно не допустить неточностей в алгоритме программы, так как они повлияют на правильность работы программы.

3.2. Подготовка к выполнению лабораторной работы

Перед началом выполнения лабораторной работы необходимо, используя предложенную литературу, выполнить следующие задания:

1) изучить назначение и способы использования подпрограмм [1, 2];

2) изучить команды вызова подпрограмм, возврата из подпрограмм и команды переходов языка Ассемблера (прил. 2), [3, 4];

3) изучить способы организации стека в микропроцессорной системе и методы его использования при программировании [1, 4];

4) изучить назначение и порядок построения блок-схем программы. Ознакомиться с примерами построения блок-схем [4, 6].

3.3. Порядок выполнения работы

Лабораторная работа выполняется в соответствии с вариантом задания (подраздел 3.5) и включает выполнение следующих этапов:

1) разработать алгоритм выполнения указанных в задании действий. При разработке алгоритма необходимо наиболее полно использовать условные переходы и подпрограммы. Используя условные графические обозначения (прил. 5), оформить разработанный алгоритм в виде блок-схемы;

2) составить программу в командах языка Ассемблера на основе разработанного на первом этапе алгоритма программы. Записать программу в общем виде;

3) загрузить программу в память машины. В пошаговом режиме работы, на уровне машинных циклов, осуществить проверку правильности работы программы. Если в процессе проверки обнаружатся ошибки в алгоритме или в тексте программы, необходимо их устранить (уточнить алгоритм или текст программы). После устранения ошибок повторить выполнение этапа 3;

4) осуществить пуск программы в рабочем режиме работы.

3.4. Содержание отчета

Отчет по лабораторной работе оформляется в соответствии с требованиями государственного стандарта и должен содержать:

1) титульный лист;

2) описание и цели работы;

3) задание на лабораторную работу;

4) алгоритм выполнения программы, оформленный в виде блок-схемы;

5) программу на языке Ассемблера, записанную в общем виде;

6) список использованных в программе команд переходов, вызова

подпрограмм и команд возврата из подпрограмм;

7) схему организации стека в УМК.

3.5. Варианты заданий к лабораторной работе

В заданиях на лабораторную работу все адреса и данные представлены в шестнадцатеричной системе счисления.

Вариант 1. Выполнить операцию сложения константы 7A с содержимым ячейки памяти 043B . При наличии переноса из старшего разряда выполнить логическую операцию инвертирования. Полученный результат записать по адресу 0C09. Остановить программу.

Вариант 2. Переписать массив данных длиной 16 байт, начиная с ячейки памяти 02F0, в область памяти начиная с адреса 0B28. При переписывании осуществлять контроль 4-го разряда каждого данного, при равенстве его нулю, над данным выполнить логическую операцию инвертирования. Остановить программу.

Вариант 3. Организовать стек в области памяти, начиная с ячейки 0CFF. Загрузить в стек содержимое всех программно-доступных регистров микропроцессора. Значение вершины стека сохранить по адресам 0B0B (старший байт адреса) и 0B0C (младший байт адреса). Остановить программу.

Вариант 4. Переписать содержимое ячейки памяти 02D6 в аккумулятор. Наращивать содержимое аккумулятора на 2A до появления признака переноса. В регистре B сохранять количество приращений. При появлении признака переноса содержимое регистра B переписать в яяейку памяти 0F00. Остановить программу.

Вариант 5. Из числа A6 вычесть содержимое яяейки памяти 02D6. При получении положительного результата, сложить его с содержимым ячейки памяти 02D3. Полученный результат записать по адресу 0EAB. Остановить программу.

Вариант 6. Организовать стек в области памяти, начиная с адреса 0CFF. Переписать в стек содержимое 10 яяеек памяти, начиная с адреса 0314. Остановить программу.

Вариант 7. Изменить содержимое 3-го разряда данного, хранящегося в ячейке памяти 01A6. При получении результата с четным количеством единиц, записать его в ячейку памяти с адресом 0CC0, в противном случае результат записать в ячейку памяти 0CD0. Остановить программу.

Вариант 8. Над данными, хранящимися в пяти ячейках памяти с адресами от 030A до 030E, выполнить следующие действия: переслать данное в микропроцессор, осуществить циклический сдвиг числа на два разряда вправо; полученный результат сохранить в памяти (в ячейках памяти с адресами от 0C0A до 0C0E). Остановить программу.

Вариант 9. В регистр B записать число C1. Над каждым данным, хранящимся в ячейках памяти, начиная с адреса 031A и заканчивая адресом 0322, выполнить следующие операции: считать данное; увеличить значение данного на 01; сравнить полученный результат с содержимым регистра B; при равенстве - содержимое ячейки памяти оставить неизменным, в противном случае - записать в нее полученный результат. Остановить программу.

Вариант 10. Сложить содержимое ячейки памяти 01EF и число 2A. При четном количестве единиц и отсутствии переноса из старшего разряда полученный результат записать по адресу 0F00, иначе результат записать в регистр микропроцессора E. Остановить программу.

Вариант 11. Массив памяти длиной 16 байт, начиная с адреса 01CC, разделить на два массива по признаку: если данное больше числа 0800, записать в первый массив, начинающийся с адреса 0C10, иначе - во второй, начинающийся с адреса 0C30. Организовать подсчет элементов каждого массива: в регистре D - количество элементов в первом массиве, а в регистре E - во втором. Остановить программу.

Вариант 12. Изменить значение 4-го разряда данного, хранящегося в ячейке памяти 0234, на противоположное. При получении результата с нечетным количеством единиц, записать его в ячейку памяти 0C0C, в противном случае результат записать в ячейку памяти с адресом 0C10. Остановить программу.

Вариант 13. Переписать содержимое ячейки памяти 01CA в регистр микропроцессора B. Уменьшать содержимое регистра B на 2F до появления отрицательного результата. В регистре E сохранять количество шагов уменьшения. При появлении признака отрицательного результата содержимое регистра E переписать в ячейку памяти 0A0F. Остановить программу.

Вариант 14. Из данного, хранящегося в ячейке памяти 035F, вычесть число 30. При получении отрицательного результата обнулить ячейку памяти 035F, иначе записать в нее полученный результат. Остановить программу.

Вариант 15. Организовать стек в области памяти, начиная с адреса 0C88. Сохранить в стеке содержимое регистровых пар BC, DE, HL. Вызвать подпрограмму, производящую сложение чисел 2F и 6A и сохранение результата в ячейке памяти с адресом 0A00. После выполнения подпрограммы осуществить восстановление значений регистровых пар. Остановить программу.

СПИСОК ЛИТЕРАТУРЫ

1. Напрасник М. В. Микропроцессоры и микроЭВМ: Учебное пособие для техн.- М.: Высшая школа, 2006.- 192 с.
2. Микропроцессоры. В 3-х кн. Кн. 1: Архитектура и проектирование микро-ЭВМ. Организация вычислительных процессов / П. В. Нестеров, В. Ф. Шаньгин, В. А. Горбунов и др. - М.: Высшая школа, 2008.- 495 с.
3. Горбунов В.Л., Панфилов Д.И., Преснухин Д.Л. Справочное пособие по микропроцессорам и микроЭВМ / Под ред. Л.Н.Преснухина. - М.: Высшая школа, 2009.- 272 с.
4. Андреев В.В., Куликов П.Б., Марквардт Г.Г. и др. Вычислительная и микропроцессорная техника в устройствах электрических железных дорог: Учебник для вузов ж.-д. трансп./ Под ред. Г.Г.Марквардта. - М.: Транспорт, 2008.- 287 с.
5. Микропроцессоры. В 3-х кн. Кн. 3: Средства отладки, лабораторный практикум и задачник / Н. В. Воробьев, В. Л. Горбунов, А. В. Горячев и др. - М.: Высшая школа, 2007.- 351 с.

ПРИЛОЖЕНИЕ 1

Техническая характеристика и принцип работы учебного микропроцессорного комплекта

1. Техническая характеристика

Учебный микропроцессорный комплект (УМК) представляет собой законченную микроЭВМ, реализованную на микропроцессорном комплекте серии КР580.

УМК предназначен для обучения основам программирования микропроцессора серии КР580, изучения основ проектирования и обслуживания микро-ЭВМ.

Техническая характеристика УМК представлена в табл. 1.

Таблица 1

Технические характеристики УМК

Параметр	Характеристика
Тип применяемого микропроцессора	КР580ИК80А
Объем оперативного запоминающего устройства	2 Кбайта
Объем постоянного запоминающего устройства	2 Кбайта
Возможность прерывания	1 вектор
Программное обеспечение	Системная программа "Монитор"
Напряжение питания, В	220 ± 22
Потребляемая мощность, ВА (не более)	55

Блок питания УМК имеет встроенную защиту от перегрузок по току и от увеличения напряжения на выходах.

УМК предназначен для эксплуатации в следующих климатических условиях:

- температура окружающего воздуха - от +10 до +35 ° С;
- относительная влажность - не более 80 % при +25 ° С.

2. Функции выполняемые учебным микропроцессорным комплектом

Ввод информации в микроЭВМ и вызов директив осуществляется с клавиатуры, расположенной на лицевой панели УМК.

Отображение вводимой и выводимой информации производится на дисплее в шестнадцатеричном коде.

С клавиатуры пульта осуществляется вызов следующих директив:

1) чтение и модификация содержимого ячеек памяти;
2) чтение и модификация содержимого регистров микропроцессора;
3) вычисление контрольной суммы массива памяти;
4) заполнение массива памяти константой;
5) перемещение заданного массива памяти в адресном пространстве;
6) выполнение программ пользователя с возможностью установки до двух точек останова.

В УМК предусмотрено пошаговое выполнение программ, при этом для отображения состояния шины адреса, данных и регистра состояний в двоичном коде используется световая индикация на светодиодах.

3. Устройство и принцип работы составных частей учебного микропроцессорного комплекта

УМК состоит из микроЭВМ, пульта оператора и блока питания. Структурная схема представлена на рис. 1.

МикроЭВМ является основной составной частью и управляет работой всего УМК. Все обращения к памяти , операции ввода-вывода, вычисления выполняются микроЭВМ или ею же инициируются.

МикроЭВМ состоит из операционного устройства (ОУ), постоянного запоминающего устройства (ПЗУ), оперативного запоминающего устройства (ОЗУ) и устройства пошагового выполнения программ.

Основой микроЭВМ является ОУ, которое производит все операции по обработке информации. Исходным состоянием ОУ является чтение информации по нулевому адресу ПЗУ. ОУ принимает это состояние после нажатия управляющей клавиши "СБ" на пульте оператора.

Информация о состоянии ОУ фиксируется в регистре состояния в начале каждого машинного цикла. В табл. 2 приведены возможные состояния ОУ. В зависимости от состояния этого регистра формируются сигналы, управляющие работой всей микроЭВМ. В табл. 3 дано определение каждого бита регистра состояний.

В ПЗУ записана программа "Монитор", обеспечивающая ввод информации с клавиатуры пульта оператора и вывод ее на дисплей. Программа "Монитор" занимает 1 Кбайт ПЗУ и использует последние 54 ячейки ОЗУ.

ОЗУ используется для хранения программ пользователя и имеет объем 2 Кбайта.

Устройство пошагового выполнения программ переводит ОУ в состояние "Ожидания" после выполнения очередного шага.

Возможны два пошаговых режима работы: покомандный шаг и поцикловый шаг.

Вызов пошагового режима работы осуществляется переключателем "РБ/ШГ", выбор величины шага - переключателем "КМ/ЦК". Для последующего шага необходимо нажать кнопку "ШГ", при этом после выполнения очередного шага на световой индикации отображается состояние
адресной шины, шины данных и регистра состояния ОУ в двоичном коде.

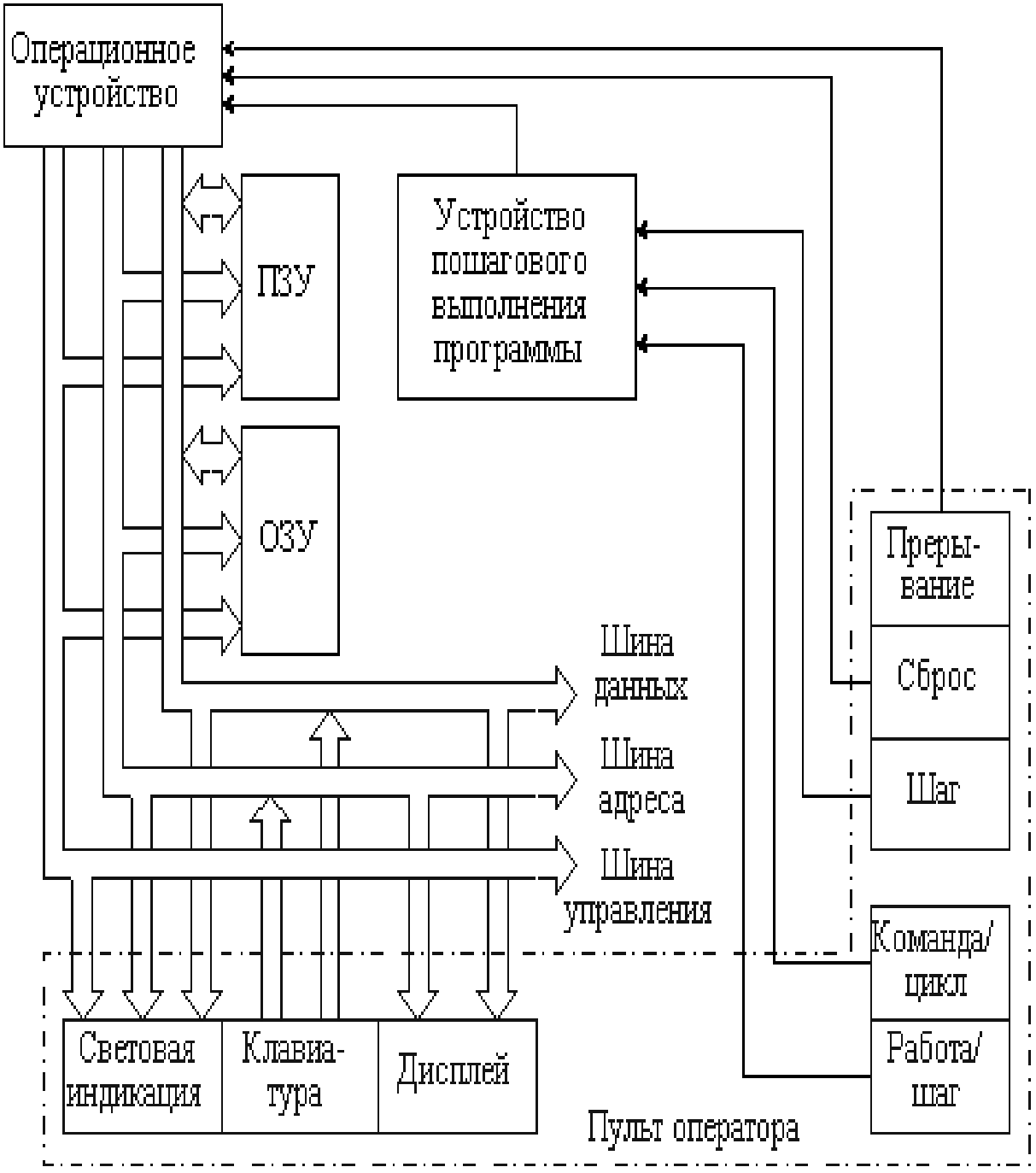

Рис. 1. Структурная схема УМК

Таблица 2

Возможные состояния ОУ

Состояние ОУ	Разряды регистра состояния ОУ							
	D0 INTA	D1 WO/	D2 STACK	D3 HLTA	D4 OUT	D5 MI	D6 INP	D7 MEMR
Выбор команды	0	1	0	0	0	1	0	1
Чтение памяти	0	1	0	0	0	0	0	1
Запись в память	0	0	0	0	0	0	0	0
Чтение стека	0	1	1	0	0	0	0	1
Запись в стек	0	0	1	0	0	0	0	0
Ввод	0	1	0	0	0	0	1	0
Вывод	0	0	0	0	1	0	0	0
Прерывание	1	1	0	0	0	1	0	0
Останов	0	1	0	1	0	0	0	1
Прерывание в останове	1	1	0	1	0	1	0	0

Таблица 3

Значение разрядов регистра состояния

Наименование сигнала	Разряд регистра состояния ОУ	Пояснения
INTA	D0	Сигнал подтверждения запроса прерывания. Используется для ввода на шину данных команды RST
W0	D1	Указывает, что в текущем машинном цикле выполняется запись в память или операция ввода
STACK	D2	Означает наличие на шине адреса содержимого указателя стека
HLTA	D3	Сигнал подтверждения команды HLT
OUT	D4	Указывает, что в текущем машинном цикле выполняется операция вывода
MI	D5	Указывает, что текущий машинный цикл служит для выборки первого байта команды
INP	D6	Указывает, что в текущем машинном цикле выполняется операция ввода
MEMR	D7	Указывает, что в текущем машинном цикле будет производиться чтение памяти

Примечание. Активным состоянием сигнала WO является ноль, а остальных сигналов - единица.

Выполнение программы может быть остановлено нажатием управляющей кнопки "ПР". При этом состояние всех регистров ОУ сохраняется в ОЗУ, откуда они опять могут быть загружены в ОУ и выполнение программы продолжится, начиная с точки останова.

Пульт оператора предназначен для взаимодействия оператора с микроЭВМ.

Пульт оператора состоит из клавиатуры, шестиразрядного дисплея, световой индикации и управляющих кнопок (сброс "СБ", прерывание "ПР", шаг "ШГ") и переключателей работы (работа/шаг "РБ/ШГ" и команда/цикл "КМ/ЦК"). Клавиатура состоит из 24 клавиш, из них 8 клавиш директивные, а 16 - информационные.

Директивные клавиши служат для вызова директив. Обозначение и назначение директивных клавиш приведены в табл. 4.

Таблица 4

Обозначение директивных клавиш

Обозначение	Назначение
П	Чтение и изменение содержимого памяти
РГ	Чтение и изменение содержимого регистров микропроцессора
СТ	Передача управления программе пользователя
КС	Определение контрольной суммы массива памяти
ЗК	Заполнение массива памяти константой
ПМ	Перемещение массива памяти в адресном пространстве
_	Клавиша пробела служит для разделения нескольких переменных при вводе
ВП	Выполнить, означает конец директивы

Информационные клавиши служат для ввода чисел в шестнадцатеричном коде. При неправильной работе с клавиатурой в крайней правой позиции дисплея индицируется знак "?".

Блок питания обеспечивает постоянным стабилизированным напряжением микроЭВМ и пульт.

4. Порядок работы с учебным микропроцессорным устройством

4.1. Идентификация и изменение содержимого памяти

Последовательно нажмите следующие клавиши:

"П" X1 X2 X3 X4 "_" D1 "_" D2 "_" DN "ВП ",

где X1, X2, X3 - адрес ячейки памяти, задается с помощью информационных клавиш. В качестве адреса фиксируются последние четыре введенные цифры; D1 ... DN - данные, подлежащие записи в память, задаются с помощью информационных клавиш. В качестве байта данных фиксируются последние две введенные цифры.

Для изменения содержимого индицируемой ячейки памяти наберите новое содержимое и нажмите клавишу "_". При этом индицируется содержимое следующей ячейки памяти.

Для перехода к следующей ячейке памяти без изменения содержимого индицируемой, не набирая новых данных, нажмите клавишу "_".

4.2. Индикация и изменение содержимого регистров

Нажмите клавишу "РГ", а затем идентификатор регистра. Идентификатором регистра являются символы, определяющие регистры микропроцессора (табл. 5).

Таблица 5

Идентификаторы регистра микропроцессора

Идентификатор регистра	Регистр	Емкость регистра, бит
A	Регистр A	8
B	Регистр B	8
C	Регистр C	8
D	Регистр D	8
E	Регистр E	8
H	Регистр H	8
F	Регистр условий	8
L	Регистр L	8
SL	Младший байт указателя стека	8
SH	Старший байт указателя стека	8
PL	Младший байт счетчика команд	8
PH	Старший байт счетчика команд	8

Ответом на ввод идентификатора является индикация содержимого указанного регистра в виде совокупности шестнадцатеричных цифр. Для изменения содержимого наберите новое значение с помощью информационных клавиш. В качестве нового содержимого регистров фиксируются последние две введенные цифры. После этого нажмите клавишу "_" и введите идентификатор следующего регистра. При необходимости перехода к следующему регистру без изменения содержимого индицируемого регистра, не набирая новых данных, нажмите клавишу "_" . Для завершения директивы нажмите клавишу "ВП".

4.3. Передача управления программе пользователя

Нажмите последовательно следующие клавиши:

"СТ" АДРЕС-1 "_" АДРЕС-2 "_" АДРЕС-3 "ВП",

где АДРЕС-1 - начальный адрес программы; АДРЕС-2 и АДРЕС-3 - адреса выполнения прерывания программы.

Производится передача управления программе по АДРЕСУ-1. АДРЕС-2 и АДРЕС-3 воспринимаются как адреса, до которых должна выполняться программа, и должны находиться в пределах адресов оперативного запоминающего устройства. Если АДРЕС-2 и АДРЕС-3 отсутствуют, то выполнение программы не прерывается. АДРЕС-1, АДРЕС-2 и АДРЕС-3 должны указывать первый байт команды. Состояние регистров микропроцессора при достижении АДРЕСА-2 или АДРЕСА-3 сохраняется в оперативном запоминающем устройстве и управление передается программе монитор. За пользователем остается выполнение любой директивы.

При передаче управления по АДРЕСУ-1 происходит восстановление состояния регистров микропроцессора, определенное в момент последнего прерывания. В случае отсутствия параметра АДРЕС-1 управление передается по адресу, находящемуся в счетчике команд.

4.4. Определение контрольной суммы массива памяти

Нажмите последовательно следующие клавиши:

"КС" АДРЕС-1 "_" АДРЕС-2 "ВП" ,

где АДРЕС-1 и АДРЕС-2 - соответственно начальный и конечный адреса массива памяти.

Контрольная сумма массива представляет собой сумму содержимого всех ячеек по модулю без учета переполнения. После выполнения директивы на экране дисплея индицируется контрольная сумма массива памяти.

4.5. Заполнение массива памяти константой

Нажмите последовательно следующие клавиши:

"ЗК" АЛРЕС1 "_" АДРЕС2 "_" D "ВП" ,

где АДРЕС-1 и АДРЕС-2 - соответственно начальный и конечный адреса массива памяти; D - байт данных, подлежащий занесению в память.

Подпрограмма директивы заполняет массив памяти данными от АДРЕСА-1 до АДРЕСА-2 включительно.

4.6. Перемещение массива памяти в адресном пространстве

Нажмите последовательно клавиши:

"ПМ" АДРЕС-1 "_" АДРЕС-2 "_" АДРЕС-3 "ВП" ,

где АДРЕС-1 и АДРЕС-2 - начальный и конечный адреса перемещаемого массива; АДРЕС-3 - начальный адрес массива размещения.

Массив памяти, ограниченный АДРЕСОМ-1 и АДРЕСОМ-2 включительно переписывается в область памяти, начиная с АДРЕСА-3. Массивы перемещения и назначения не должны перекрываться, в противном случае происходит утеря информации.

4.7. Прерывание выполнения программы пользователя

Для прерывания программы пользователя нажмите управляющую кнопку "ПР". При этом управление передается программе обработки прерываний командой RST7. Программа сохраняет состояние всех регистров процессора и производит передачу управления монитору.

Регистры сохраняются в стеке пользователя, в случае отсутствия такового в стеке монитора. На дисплее индицируется содержимое счетчика команд, которое на единицу больше адреса последнего байта последней выполненной

команды.

После этого пользователь может вызывать выполнение любой из существующих директив. Выполнение прерванной программы возможно, начиная с адреса останова или любого другого адреса.

4.8. Пошаговое выполнение программ

Имеются два режима пошагового выполнения программы: ***поцикловый*** и ***покомандный***. В поцикловом режиме оперативный блок переводится в состояние "ожидание" при выполнении каждого рабочего цикла, а в покомандном режиме лишь при чтении первого байта команды.

Для вызова пошагового режима необходимо выполнить следующие действия:

- установить переключатель “РБ/ШГ” в состояние “ШГ”, при этом происходит подключение световой индикации;
- выбрать переключателем “КМ/ЦК” один из режимов работы;
- передать управление программе пользователя.

После выполнения указанных действий на световой индикации отобразится начальный адрес программы, данные по этому адресу и состояние ОУ.

Для выхода из пошагового режима нужно переключатель “РБ/ШГ” установить в состояние “РБ”.

5. Адресация в Учебном Микропроцессорном Комплекте

В УМК первые 2 Кбайта адресов (0000_{16} - $07FF_{16}$) составляет ПЗУ, в котором записана управляющая программа “Монитор”. Следующие 2 Кбайта адресов (0800_{16} - $0FFF_{16}$) отведены ОЗУ.

Карта памяти УМК приведена в табл. 6. Таблица 6

Карта памяти УМК

Устройство	Адреса	Назначение области памяти
ПЗУ	0000 ... 07FF	Управляющая программа "Монитор"
ОЗУ	0800 ... 0BFF	Для записи исследуемых программ
ОЗУ	0C00 ... 0FFF	Для записи данных исследуемых программ и стека
	0FFF...FFFF	Не используемые адреса

6. Меры безопасности при работе с Учебным Микропрооцессорным Комплектом

При работе с УМК запрещается:

- эксплуатировать УМК при незакрепленной лицевой панели;
- соединять и разъединять разъемы УМК при включенном питании;
- оставлять УМК во включенном состоянии без наблюдения;
- устанавливать в вилку разъема для подключения ТЭЗ, расположенной на лицевой панели УМК, посторонние предметы;
- закрывать вентиляционные щели;
- самостоятельно вскрывать лицевую панель и ремонтировать УМК;
- применять самодельные предохранители или предохранители, рассчитанные на большие значения токов.

СОДЕРЖАНИЕ

Гаджиев Хаджимурат Магомедович
Гаджиева Солтанат Магомедовна
Челушкина Татьяна Алексеевна

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
к выполнению лабораторных работ
по дисциплине «Микропроцессорная техника» для студентов направления подготовки магистров 11.04.01 «Радиотехника», программа «Системы и устройства передачи, приема и обработки сигналов»

Формат 60х84 1/16. Бумага офсетная.
Печать ризограф. Усл. п. л. 3,0.
Тираж 50 экз. Заказ №

Отпечатано в ИПЦ ДГТУ.
367015, г. Махачкала, пр. Имама Шамиля, 70