Министерство науки и высшего образования Российской Федерации
ФГБОУ ВО «Удмуртский государственный университет»
Институт нефти и газа им. М.С. Гуцериева
Кафедра теплоэнергетики

С. А. Хорьков

Программирование микропроцессора

Лабораторный практикум

Ижевск
2021

УДК 004.431.4(075.8)
ББК 32.973.21я73-5
Х831

Рецензенты: старший преподаватель кафедры теплоэнергетики УдГУ **В.В. Зиновьев**,
к.т.н., доцент кафедры бурения нефтяных и газовых скважин УдГУ **А.Г. Миловзоров**

Хорьков, С.А.
Х831 Программирование микропроцессора. Лабораторный практикум / С. А. Хорьков. – Ижевск: Издательский центр «Удмуртский университет», 2021. – 60 с.

В учебно-методическом пособии приведено описание учебного микропроцессорного комплекта на базе восьмиразрядного микропроцессора, описаны основные правила составление программ на ассемблере, представлены пять лабораторных работ, выполнение которых позволяет изучить методы и получить навыки ручного ассемблирования – необходимого инструмента для начального обучения программированию микропроцессора. Пособие предназначено для студентов магистерской программы «Измерительные системы и информационные технологии в электроэнергетике и электротехнике», изучающих дисциплины «Микропроцессорная интегрированная противоаварийная автоматика» и «Теория автоматического управления».

УДК 004.431.4(075.8)
ББК 32.973.21я73-5

Введение

В настоящее время программирование микропроцессора (МП) ведется на языке высокого уровня или на языке Ассемблера, который относят к машинно-ориентированным языкам низкого уровня. Каждой команде языка Ассемблера соответствует, как правило, одна машинная команда. Каждый тип МП имеет свою систему команд и свой язык Ассемблера. Команды, адреса и данные записываются в виде буквенно-цифровых символов. Мнемонические обозначения команд в англоязычном написании легко ассоциируются с реально выполняемыми действиями. После составления нескольких простых программ мнемонические обозначения наиболее употребительных команд легко запоминаются.

Программирование на Ассемблере позволяет создавать максимально эффективные с точки зрения быстродействия и потребного объема памяти программы.

Ассемблирование может выполняться вручную или с привлечением специальной программы – транслятора.

Ручное ассемблирование, т.е. перевод команд (операторов) в машинные коды при помощи специальной таблицы, можно рекомендовать лишь на этапе начального обучения проектированию прикладных программ, так как процесс этот трудоемок. Однако его эффективность для первоначального знакомства с программированием МП чрезвычайно высока и его применение, в этом случае, не вызывает сомнений.

Написанная на Ассемблере программа может быть переведена на машинный язык с помощью транслирующей программы, называемой также Ассемблер. При этом автоматически проводится проверка синтаксиса языка с указанием ошибок, которые необходимо исправить.

Целью учебного пособия является методическое обеспечение ручного ассемблирования при выполнении лабораторных работ на учебном микропроцессорном комплекте на базе МП КР580ВМ80А.

Пособие предназначено для студентов магистерской программы «Измерительные системы и информационные технологии в электроэнергетике и электротехнике», изучающих дисциплины «Микропроцессорная интегрированная противоаварийная автоматика» и «Теория автоматического управления». Оно способствует формированию следующих компетенций: ПК-5. Способность выбирать серийные и проектировать новые объекты профессиональной деятельности; ПК-7. Способность выбирать структуру и параметры элементов межсистемных электрических связей и рассчитывать показатели надежности электроэнергетических систем.

1. Учебный микропроцессорный комплект (УМК)

В состав УМК входят: 1) микроЭВМ, 2) пульт оператора, 3) блок питания[1].

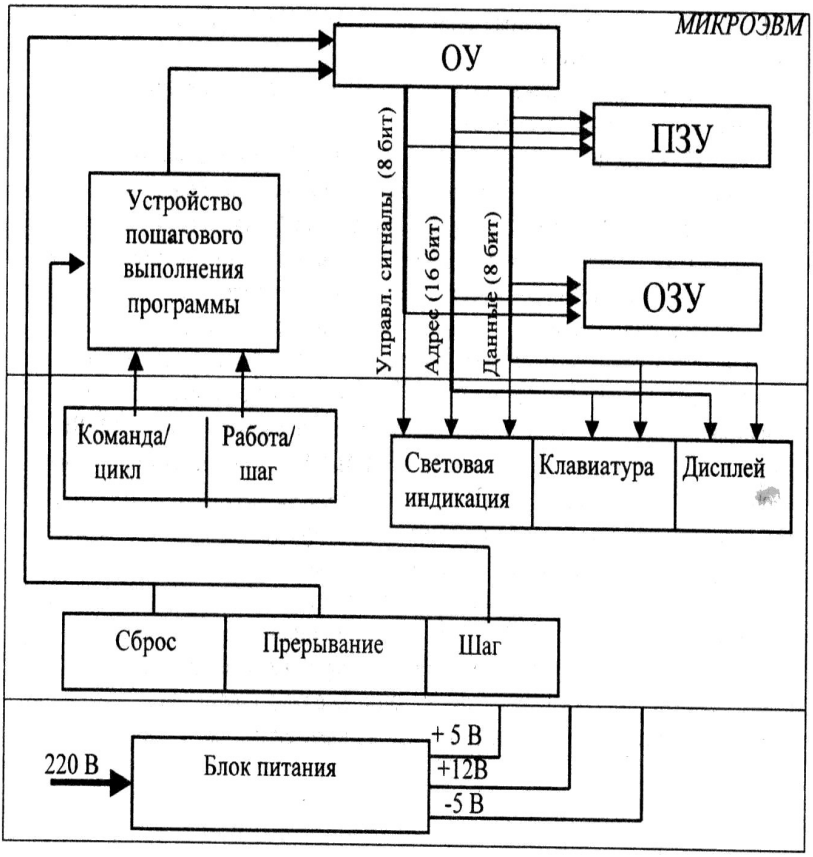


Рис. 1.1. Структурная схема УМК

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

2. Пульт оператора предназначен для взаимодействия оператора с микроЭВМ.

3. Блок питания обеспечивает постоянными стабилизированными напряжениями микроЭВМ и пульт оператора.

МикроЭВМ состоит из операционного устройства (ОУ) постоянного и оперативного запоминающих устройств и устройства пошагового выполнения программ.

Пульт оператора состоит из клавиатуры, шестиразрядного дисплея, световой индикации и управляющих кнопок

ОУ реализовано на базе МП КР580ВМ80А. Оно производит все операции по обработке информации. Информация о состоянии ОУ фиксируют в регистре состояния в начале каждого машинного цикла.

Для управления МПв ПЗУ хранится программа "Монитор", которая обеспечивает ввод информации с клавиатуры пульта оператора и вывод ее на дисплей. Эта программа позволяет выполнять чтение и изменение содержимого памяти, содержимого регистров, выполнение программ пользователя, копирование областей памяти, вычисление контрольных сумм; заполнение массива памяти константой, управление обменом с внешними устройствами; отладку программ пользователя.

Карта памяти УМК представлена в таблице 1.1.

Таблица 1.1.

Карта памяти УМК

Адрес памяти		Объем памяти	Назначение
Начало	Конец		
0000H	03FFH	1 кбайт	ПЗУ программы «Монитор»
0400H	07FFH	1 кбайт	ПЗУ пользователя
0800H	0BD9H	1 кбайт	ОЗУ записи программ
0BDAH	0DFFH	54 кбайта	Область стека программы «Монитор»

Программа "Монитор" занимает 1 кбайт постоянной памяти (ПЗУ), а также использует последние 54 ячейки ОЗУ для области стека, еще 1 кбайт ПЗУ зарезервирован за пользователем. ОЗУ предназначена для хранения программ пользователя и имеет емкость 1 кбайт.

Внешний вид лицевой панели УМК представлен на рис. 1.2.

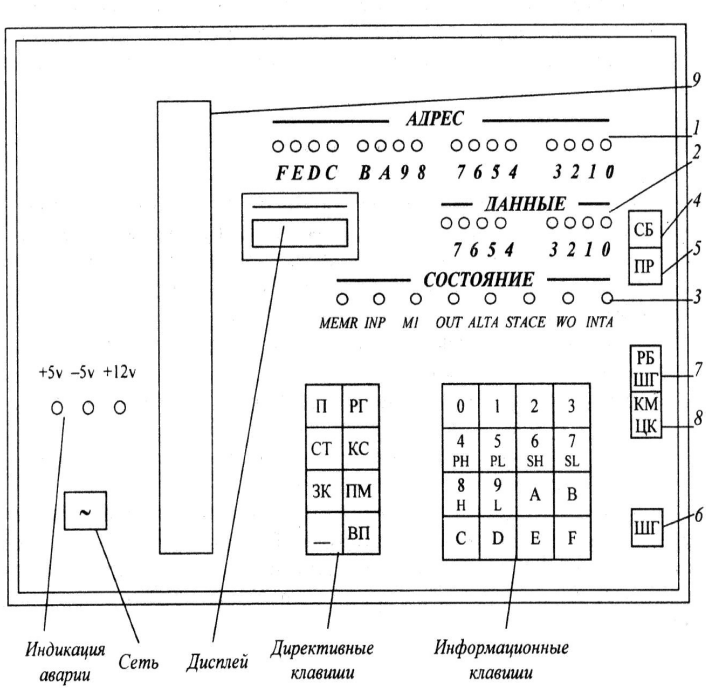


Рис. 1.2. Передняя панель УМК

1 – индикация шины адреса; 2 – индикация шины данных; 3 – индикация регистра состояний; 4 – СБ (сброс); 5 – ПР (прерывание); 6 – ШГ (шаг); 7 – РБ/ШГ (работа / шаг); 8 – КМ/ЦК (команда / цикл); 9 – разъем для подключения макетной платы

Опишем элементы и органы управления, расположенные на передней панели стенда УМК [1].

Клавиша «~» – включение питания три светодиода цепей питания, 16 светодиодов-индикаторов шины адреса (0. 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F), 8 светодиодов-индикаторов шины данных (0. 1, 2, 3, 4, 5, 6, 7), 8 светодиодных индикаторов состояния микропроцессора, описанных в таблице 1.2.

Индикаторы +5V, –5V, +12V– индикаторы аварии.

Для подключения макетной платы на панели УМК имеется разъем.

Шестиразрядный семисегментный индикатор (дисплей).

Клавиши управления: СБ – начальная установка (обнуление – сброс) программного счетчика, ПР – прерывание выполнения программы пользователя, РБ/ШГ – установка режима пошагового выполнения программы; КМ/ЦК – установка режима поциклового выполнения программы; ШГ – управление выполнения программы в пошаговом режиме.

Восемь директивных клавиш: П – чтение содержимого ОЗУ и ПЗУ и измерение содержимого ОЗУ; РГ – чтение и изменение содержимого регистров микропроцессора; СТ – передача управления программе пользователя; КС – определение контрольной суммы массива памяти; ЗК – заполнение массива памяти (ОЗУ) константой; ПМ – перемещение массива памяти ОЗУ; «пробел» – разделение переменных при вводе (клавиша пробела); ВП – выполнить, обозначение инструкции «Конец директивы»

Шестнадцать информационных клавиш имеют двойное значение. Они служат для ввода чисел в шестнадцатеричном коде (0. 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F) и для вызова идентификаторов регистров микропроцессора: PH, PL – старший и младший байты счетчика команд, SH, SL–старший и младший байты указателя стека, H–регистр H, L–регистр L, второе, кроме ввода чисел, назначение клавиш A – аккумулятор B, C, D, E – одноименные регистры.

Регистр состояний МП показывает, какие управляющие сигналы генерируются микропроцессором в каждом машинном цикле (таблицу 1.2.).

Таблица 1.2.

Разряды регистра состояний

Машинный цикл	INTA	WO	STACKE	ALTA	OUT	MI	INP	MEMR
1	2	3	4	5	6	7	8	9
Выборка кода операции	0	1	0	0	0	1	0	1
Считывание из памяти	0	1	0	0	0	0	0	1
Запись в память	0	0	0	0	0	0	0	0
Считывание из стека	0	1	1	0	0	0	0	1
Запись в стек	0	1	0	0	0	0	1	0
Ввод	0	1	0	0	0	0	1	0
Вывод	0	0	0	0	1		0	0
Прерывание	1	1	0	0	0	1	0	0
Останов	0	1	0	1	0	0	0	1
Прерывание при останове	1	1	0	1	0	1	0	0

Результат выполнения команды МП отображается на дисплее:

В 4-х левых разрядах отображается адрес, а в двух крайних правых разрядах – данные.

Если адрес был выведен с ошибкой, то «Монитор» выдает сообщение в виде знака «?». Сброс ошибочной команды осуществляется нажатием клавиши СБ.

Общий вид команды «Монитор» УМК имеет вид:

КОП[ПАР1 «пробел» ПАР2 «пробел» ПАР3] «ВП»,

где КОП – код операции, соответствующий нажатию одной из директивных клавиш (П, РГ, СТ, КС, ЗК, ПМ) [1];

[ПАР1, ПАР2, ПАР3] – вводимые параметры, которые являются адресами или данными; поскольку адреса и данные – это числа, то их удобно выделить квадратными скобками.

«пробел» – клавиша пробела;

ВП – клавиша выполнения.

Рассмотрим команды программы «Монитор»

1. Индикация содержимого ПЗУ и ОЗУ. Формат команды

П [адрес] «пробел».

По этой команде на дисплей выводится содержимое ячейки памяти по указанному адресу. Для просмотра ячеек начиная с указанного адреса, последовательно нажимают клавиши «пробел». Выход из этого режима осуществляют нажатием клавиши «ВП». Поскольку адрес это число, то его в формате команды удобно выделить квадратными скобками

2. Модификация содержимого ОЗУ.

Формат команды **П [адрес] «пробел» данные.**

Для модификации данных по тому же адресу, необходимо нажать клавишу СБ, а затем повторить вызов ячейки памяти с тем же адресом и записать в неё новые данные. Выход из режима осуществляется нажатием клавиши ВП, при этом в память загружается выведенное на дисплее число. Изменить содержание ПЗУ нельзя.

3. Индикация содержимого регистров общего назначения – РОН.

Формат команды **РГ [РОН]**. Например, РГА, при этом на дисплее отразится содержимое регистра, который называется аккумулятор, а его идентификатором является буква «А». Нажать можно любую информационную клавишу, вызывающую тот или иной регистр (PH,PL, SH, SL, H, L, D, E, B, C, A). Выход из режима достигается нажатием клавиши ВП.

4. Модификация содержимого РОН.

Формат команды – **РГ [РОН] «пробел» [данные] ВП**.

5. Заполнение ОЗУ константой.

Формат команды – **ЗК [начальный адрес «пробел» конечный адрес «пробел» данные] ВП**.

Предупреждение: категорически запрещается занимать область ОЗУ с адресами OBDA…OBFFH, так как это приведет к разрушению стека, используемого программой «Монитор».

6. Копирование областей памяти ОЗУ.

Формат команды – **ПМ [начальный адрес копируемого массива «пробел» его конечный адрес «пробел» начальный адрес начала массива ОЗУ для размещения копии] ВП**.

Область ОЗУ с адресами OBDA…OBFFH занимать запрещается, так как это вызывает разрушение стека Монитора.

7. Выполнение контрольной суммы массива памяти.

Формат команды **КС [начальный адрес «пробел» конечный адрес] ВП**. По этой команде подсчитывается контрольная сумма массива памяти в указанном массиве. Контрольную сумму содержимого всех ячеек подсчитывают по модулю 256. Результат выводят на дисплей. Эту команду используют для проверки правильности копирования и при отладке программ пользователя.

8. Выполнение программ пользователя.

Формат команды **СТ [начальный адрес] ВП.** После выполнение программы пользователя управление передается программе «Монитор». Формат команды может быть **СТ [начальный адрес «пробел» адрес 1 «пробел» адрес 2] ВП**. Такой формат используется при отладке программы, пользователь устанавливает точки останова работы программы (адрес 1 и адрес 2), для продолжения выполнения программы с адреса останова вводят команду **СТ ВП**.

9. Прерывание выполнения программы пользователя.

Формат команды **ПР**.

По этой команде происходит прерывание выполнения программы пользователя. Используется два выхода из выполняемой программы. После нажатия клавиши ПР содержимое всех регистров микропроцессора записывается в стек, а управление передается программе «Монитор». На дисплей выводится содержание счетчика команд, значение которого соответствует команде, следующей за той, на которой произошел останов команды.

После нажатия клавиши ПР можно вызвать выполнение любой команды программы Монитор. Продолжение выполнения прерванной команды возможно либо с адреса останова, либо с другого адреса, используя клавишу СТ.

Предупреждение: запрещается пользоваться клавишей ПР для прерывания команд, выполняемой программой Монитор.

10. Отладка программы пользователя.

Для отладки программы пользователя УМК снабжен средствами **пошагового выполнения программы**. Эти средства целесообразно использовать в двух режимах: **в поцикловом и командном.**

В поцикловом режиме микропроцессор переходит в состояние «ожидания» после выполнения каждого рабочего цикла программы. Режим используется для проверки и отладки программы содержащей циклы.

В командном режиме микропроцессор переходит в состояние «ожидания» после выполнения каждого байта команды. Режим используют для проверки и отладки всей программы пользователя.

При отладке программы в этих режимах к УМК подключаются светодиодные индикаторы, на которых после выполнения каждого шага отображаются в двоичной форме адрес ячейки памяти, ее содержимое и содержимое регистра состояний.

Для выхода **в пошаговый режим** необходимо выполнить следующее:

1. Установить клавишу РБ/ШГ и положение ШГ (нажать), при этом произойдет подключение к микропроцессору светодиодных индикаторов.

2. Установить необходимый режим работы. Для этого клавишей КМ/ЦК установить **поцикловой режим** (клавиша нажата) или **командный режим** (клавиша отжата).

3. Набрать команды «Монитора» – выполнение программы пользователя **СТ[адрес]ВП** и нажатием клавиши **ШГ** осуществлять выполнение программы в пошаговом режиме[1].

2. Основные правила составления программ на ассемблере

2.1. Коды команд

МП работает только с двоичными кодами (машинными кодами). Определенный набор этих кодов называют кодом команды. Команда «заставляет»МП выполнить определенное действие. Код любой команды представлен в запоминающем устройстве двоичным восьмиразрядным числом (байтом). В каждом разряде байта записан 1 бит информации.

Набор команд определяет систему команд микропроцессора. Всего с помощью одного байта можно сформировать $2^8 = 256$ различных кодовых комбинаций. Микропроцессор КР580ВМ80А имеет 244 команды. Набор команд микропроцессора представлен в приложении 1 и 2. Для запоминания команд каждому коду ставится в соответствие мнемоническое название (мнемоника) команды, которое является сокращением от английских слов, описывающих ее действие. Мнемонический код команд позволяет запомнить их функции и упрощает написание программ. Такой язык называют Ассемблером.

После того, как программа написана на Ассемблере, ее необходимо перевести машинные коды. Другими словами программу, записанную в мнемонике, необходимо перевести в последовательность двоичных восьмиразрядных чисел. Этот перевод может происходить автоматически с помощью специальных программ-трансляторов («кросс-ассемблер» или «ассемблер») или вручную.

Ручная трансляция необходимый инструмент для начального обучения программированию микропроцессора. Для ручной трансляции необходимо использовать таблицу «Коды команд микропроцессора КР580ВМ80А» приложения 2. С помощью этой таблицы можно сопоставить мнемонику команды с ее кодом. Код каждой команды приведен в шестнадцатеричной системе счисления. Например, команда ADDB имеет код 80H, команда XRAC – код A9H.

2.2. Программная модель микропроцессорной системы

Разработчику программы необходимо запомнить систему команд микропроцессора (МП) и представлять действия, которые будет выполнять МП при их обработке [2-4].

Система команд МП представляет собой полный перечень элементарных действий, которые может выполнить МП. Он выполняет очень простые действия, но с помощью этих действий можно запрограммировать любую сложную операцию.

Программная модель микропроцессорной системы (МПС) позволяет понять действия, которые выполняет микропроцессор при обработке команд.

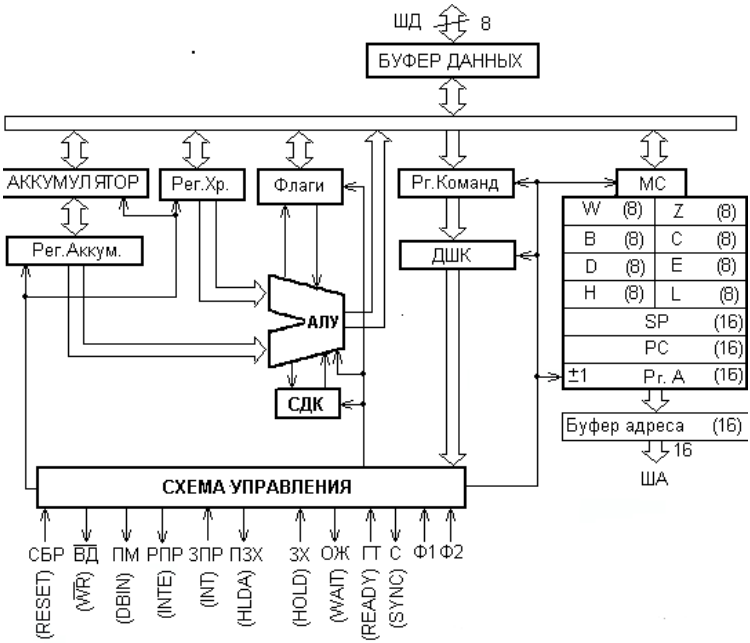


Рис. 2.1. Структурная схема микропроцессора КР580ВМ80А

SP – указатель стека; PC – программный счетчик; СДК – Схема десятичной коррекции; АЛУ – арифметико-логическое устройство: Флаги – регистр признаков; ШД – шина данных: ША – шина адреса; ШУ – шина управления.

Программная модель микропроцессорной системы, состоит из следующих блоков: 1) программно-доступных регистров; 2) программно-доступных восьмиразрядных ячеек памяти; 3) программно-доступных восьмиразрядных регистров.

1. Программно-доступные регистры микропроцессора – это регистры общего назначения, регистр указателя стека, регистр признаков (флагов) и регистр счетчика команд. **Эти регистры** показаны на структурной схеме микропроцессора КР580ВМ80Ана рис. 2.1.

Регистр А (аккумулятор) предназначен для обмена данными с другими регистрами МП и с внешними устройствами. **Регистры общего назначения** (РОН) –**это** шесть восьмиразрядных регистров микропроцессора, обозначенных буквами B, C, D, E, H, L, Регистры B и C, D и E, H и L в некоторых командах объединяют в шестнадцатиразрядные регистры, которые называют регистровыми парами. Регистры B, D и H образуют старшие восемь разрядов регистровых пар, а регистры C, E и L – восемь младших.

Регистр указатель стека (SP) представляет собой шестнадцатиразрядный регистр, который содержит адрес вершины стека. **Стек** – это динамическая последовательная структура данных в ОЗУ. Она организована таким образом, что очередная запись данных всегда осуществляется в вершину (начало) стека. В вершину стека записывают только шестнадцатибитные данные. При записи данных в стек содержимое указателя стека уменьшается на 2, а при считывании – увеличивается на 2.

Регистр признаков (флагов)(F) – это восьмиразрядный регистр, который содержит признаки результата выполнения команды (рис. 2.2).

Распределение признаков по разрядам

7	6	5	4	3	2	1	0
S	Z	**0**	**V**	**0**	**P**	1	**C**

S–признак знака;
Z–признак нуля;
V–признак дополнительного переноса;
P–признак четности;
C–признак переноса.

Рис. 2.2. Регистр признаков

Признаки устанавливаются следующим образом:

- **признак S**–единица, если седьмой разряд результата равен единице, в противном случае – ноль;

- **признак нуля Z**–единица, если во всех разрядах результата ноли, в противном случае – ноль;

- **признак дополнительного переноса V**–единица при переносе из третьего разряда или при заёме в третий разряд результата, в противном случае – ноль;

- **признак четности** P–единица, если результат в двоичном коде содержит четное количество единиц, в противном случае – ноль;

- **признак переноса C**–единица при переносе из седьмого разряда или при заёме в седьмой разряд результата, в противном случае – ноль.

Аккумулятор и регистр признаков образуют **слово состояния процессора,** обозначенное буквами **PSW**. Аккумулятору соответствуют восемь старших разрядов, а регистру признаков – восемь младших.

Регистр счетчика команд (PC) – это шестнадцатиразрядный регистр, он указывает адрес следующей команды, которая должна быть выполнена МП.

2. Программно-доступные восьмиразрядные ячейки памяти

используют в качестве памяти микропроцессорной системы (МПС). Разряды ячейки памяти нумеруют справа налево целыми числами, начиная с ноля. Максимальная емкость памяти, реализуемой запоминающим устройством, равна $2^{16} = 65536$ байт.

3. Программно-доступные восьмиразрядные регистры используют для ввода и вывода. Максимальное число регистров для ввода данных составляет 256, для вывода данных – столько же.

Команды микропроцессора используют только те **элементы данных**, обработка которых осуществляется непосредственно МП, т. е. восьми – и шестнадцатибитовые.

Восьмибитовые данные – это восемь бит (один байт), которые хранятся в РОН или ячейке памяти и обрабатываются МП как одно целое.

Шестнадцатибитовые данные – это 16 бит (два байта) данных, которые могут храниться в шестнадцатиразрядном регистре или в двух последовательных ячейках памяти и обрабатываются МП как одно целое, причем в памяти старшие восемь бит записывают по адресу на единицу большему, чем адрес младших. При адресации шестнадцатибитовых данных указывают адрес младшей ячейки памяти, содержащей эти данные.

Операнды команд могут храниться в программно-доступных регистрах микропроцессора или в памяти МПС. **Операнд** – это символические имена регистров, адресов, меток или константы. Для **указания операнда в регистре** используют: регистровая адресация и регистровая неявная адресация.

При **регистровой адресации** в коде команды существует поле для указания регистра, содержащего операнд.

При **регистровой неявной адресации** специального поля в коде команды нет.

Для указания операнда в памяти используют **непосредственную, прямую, косвенную и стековую адресации.**

При **непосредственной адресации** операнд задается в команде.

При **прямой адресации** в команде указывается адрес операнда.

При **косвенной регистровой адресации** адрес операнда указывается в регистровой паре или для команды XTHL– в регистре указателя стека.

2. 3. Команды микропроцессора КР580ВМ80А

Команды МП делят на группы:1)команды пересылки и загрузки данных; 2)команды арифметических операций; 3)команды логических операций; 4)команды передачи управления;5)команды стека, ввода – вывода и управления [2-4].

Команды имеют длину один, два или три байта.

Общее число кодов операций МП серии КР580 – 244 (Приложение 2). Поскольку коды некоторых операций можно объединить в одну команду, то общее число команд этого МП – 78.

Время выполнения одной команды МП составляет 4-18 тактов.

2.4. Формат ассемблерной строки

Запись программы на Ассемблере требует соблюдения определенных правил. Программа пишется в виде последовательности команд. Каждая команда записывается в одной строке и имеет четыре поля:

МЕТКА ОПЕРАЦИЯ ОПЕРАНД КОММЕНТАРИЙ

Некоторые поля не заполняют, но они должны быть представлены в указанном порядке [2-4].

МЕТКА – это символическое имя адреса команды, находящейся в данной строке. Метки придаются командам, к которым предусматривается обращение в процессе выполнения программы. Метка начинается с буквы и она не должна превышать 8 символов. В качестве меток нельзя записывать имена регистров, коды команд и другие зарезервированные сочетания. Разделителем между меткой и операцией является **двоеточие.**

ОПЕРАЦИЯ – в этом поле содержатся мнемокоды команд или директивы ассемблера. Разделителем между кодом команды и операндами является пробел.

ОПЕРАНД – это символические имена регистров, адресов, меток или константы. Целочисленные константы могут быть определены в двоичной, шестнадцатеричной или десятеричной системах счисления. При наличии двух операндов разделителем служит **запятая.** У некоторых команд операнды отсутствуют и поле операндов остается пустым.

КОММЕНТАРИИ начинается с **точки с запятой.** Если строка начинается с **точки с запятой,** то вся строка считается комментарием. Комментарий является вспомогательным средством, используемым для пояснения и документирования программ.

3. Лабораторные работы

3.1. Лабораторная работа № 1.
Изучение УМК и приобретение навыков работы с его клавиатурой и дисплеем

Задание на работу

1. Изучить структуру и возможности УМК.

2. Приобрести практические навыки работы с клавиатурой и дисплеем УМК.

3. Исследовать содержимое ячеек памяти ПЗУ и ОЗУ, записать константы в ОЗУ.

4. Выполнить операции перемещения константы ОЗУ, вычисления контрольной суммы и записи чисел в РОН.

Подготовка к лабораторной работе

1. Изучить УМК по разделу 1 и 2 настоящего пособия (структура УМК, расположение и назначение директивных и управляющих клавиш УМК; описание режимов работы микропроцессора, которые отображаются регистром состояний УМК; карта памяти УМК.)

2. Предоставить результаты изучения преподавателю на проверку.

Порядок выполнения работы

1. Получить разрешение на включение УМК. После получения разрешения включить УМК (перед включением все клавиши отжаты, УМК подключен к сети 220В).

1.1. Нажать клавишу со знаком «~».

1.2. Кратковременно нажать клавишу СБ. появление в левом крайнем индикаторе знака «-» свидетельствует о готовности УМК к работе.

2. Проверить работу дисплея. Для этого нажать последовательно клавиши П.0.2.3.6.«пробел». При правильной работе на дисплее должно появиться 02367А. Это означает, что в ячейке памяти 0236 хранятся данные 7А. Из карты памяти УМК (стр. 5, таблица 1.1)видно, что в памяти с адресом 0236 хранятся данные программы «Монитор» и эти данные находятся в ПЗУ. Код операции П перед цифрами адреса означает, что оператор обращается к памяти УМК. Программа «Монитор» записанная в ПЗУ позволяет оператору пользоваться панельными клавишами УМК.

3. Научиться работать с памятью УМК. Нажать клавиши П.0.0.0.0. «пробел». На дисплее в 2-х крайних правых разрядах высветится содержимое ячейки ПЗУ с адресом 0000. Провести аналогичное исследование ОЗУ для ячейки памяти с адресом 0800.

3.1.Используя клавишу «пробел», записать содержимое ОЗУ и ПЗУ в таблицу 3.1.1. (по три значения для ОЗУ и три значения для ПЗУ (смотри карту памяти). Если оператор обратился к некоторой ячейки памяти УМК, то при нажатии на клавишу «пробел» программа «Монитор» позволяет перейти к следующей по порядку ячейке памяти.

Таблица 3.1.1.

Содержимое ячеек памяти

Адрес памяти в кодах			Содержимое ячейки памятивкодах		
шестнадцатерич-ный	деся-тич-ный	двоич-ный	шестнадцатерич-ный	деся-тич-ный	двоич-ный
0000Н					
0001Н					
0002Н					
0800Н					
0801Н					
0802Н					

3.2. Ввести новые данные в ячейку по адресу 0800. Для этого нужно ввести адрес 0800 и новые данные, например 55. Для загрузки данных в память нажать «пробел», затем ВП (выполнить). Убедиться, что данные записаны в ячейку памяти. Для этого вновь набрать П0800«пробел». Если все сделано правильно, то на дисплее появятся цифры 0800 55. Затем в ячейку с адресом 0800 можно будет записать другие данные. При этом ранее записанные данные исчезнут, т.е. новые данные будут записаны «поверх» старых. Данные в ячейке будут храниться до новой перезаписи. При выключении УМК данные записанные в ПЗУ остаются сохраненными, данные записанные в ОЗУ перезаписываются случайным образом.

4. Записать константу 33 в массив памяти 0820…082F. Для этого нужно набрать команду ЗК 0820«пробел»082F«пробел»33 ВП. Убедиться, что в массиве записана константа. Для этого необходимо проверить содержимое ячеек памяти 0820…082F. Если все сделано правильно, то во всех ячейках памяти 0820…082F будет записано число 33.

5. Переместить константу из массива памяти 0820…082F в область памяти ОЗУ, начиная с адреса 0900. Для этого набрать ПМ 0820«пробел»082F«пробел»0900ВП. Убедиться, что константа перемещена. Для этого необходимо проверить содержимое ячеек памяти, начиная с адреса 0900.

6. Вычислить контрольную сумму массива памяти 0900…090F, набрав команду КС0900…090F ВП. Вычислить контрольную сумму для заданных областей памяти. Результаты записать.

7. Проверить последовательность вывода содержимого РОН на дисплей командой РГ РОН, записывая вместо РОН последовательно идентификаторы РОН (H, L, A, B, C, D, E.). Записать выведенные на дисплей значения всех РОН в таблицу 3.1.1.

7.1. Изменить содержимое РОН, используя пример РГ А77 ВП. По этой команде в регистр А (аккумулятор) будет записано число 77.

8. Получить разрешение на выключение УМК. После получения разрешения выключить УМК

Отчет по работе должен содержать: название работы; задание на выполнение работы; описание назначения клавиш УМК; описание режимов работы микропроцессора, которые отображаются регистром состояний УМК; заполненную таблицу 3.1.1, карту памяти УМК; выводы.

Контрольные вопросы

1. Из каких функциональных частей состоит УМК?
2. Какие виды клавиш имеет УМК?
3. Какие виды индикаторов имеет УМК?
4. Какие виды команд имеет программа «Монитор»?
5. Каков формат команды для записи операндов в память ОЗУ?
6. Каков формат команды для выполнения программы пользователя?
7. Каковы виды представления информации в УМК?
8. В каких режимах может работать УМК? Как устанавливают эти режимы?

3.2. Лабораторная работа № 2.
Запись и выполнение простых команд

Задание на работу

1. Изучить программу, размещенную в таблице 3.2.1., на языке Ассемблера и в машинных кодах, Ввести её в УМК и выполнить.

2. Составить и изучить новые программы аналогичные программе 3.2.1, используя вместо команды CMA команды INRA, DCRA. ANAA, ORAA, XRAA.

Пример простейшей линейной программы

Для изучения простейших линейных программ, составленных на языке ассемблера, рассмотрим пример из таблицы 3.2.1. Требуется извлечь число из ячейки памяти с адресом 0B00, инвертировать его и записать результат в ячейку памяти с адресом 0B01. Программу составить в мнемокодах и машинных кодах.

Таблица 3.2.1.

Программа в мнемокодах

Метка	Мнемокод	Операнд	Комментарий
	LDA	0B00H	;Загрузить число из ячейки с адресом 0B00 в аккумулятор
	CMA		;Инвертировать содержимое аккумулятора
	STA	0B01H	;Переслать содержимое аккумулятора в ячейку по адресу 0B01
	RST7		;Прервать выполнение команды

Для записи программы в память УМК и последующего ее выполнения необходимо перевести мнемокоды программы в машинные коды и выбрать участок памяти УМК для внесения программы. Для этого необходимо воспользоваться приложениями 1 и 2. В таблице 3.2.2. приведен перевод программы таблицы 3.2.1. из мнемокодов и в машинные коды.

При записи программ все числа представляют в шестнадцатеричной системе исчисления, а мнемокоды команд по приложению 2 переводят в шестнадцатеричные коды операций (КОП). Адреса ячеек памяти распределяют в соответствии с форматом каждой конкретной команды. В память УМК вводят только данные второго столбца таблицы по адресам ее первого столбца в режиме «модификация содержимого ОЗУ». В линейной команде адреса из первого столбца переключаются счетчиком команд автоматически, после того как записан начальный адрес программы.

Таблица 3.2.2.

Программа примера 3.2.1 в машинных кодах и в мнемокодах

Адрес	Код	Метка	Мнемокод	Операнд	Комментарий
0800	3A		LDA	0B00H	;Код команды LDA
0801	00				;Младший байт адреса 0B00H
0802	0B				;Старший байт адреса 0B00H
0803	2F		CMA		;Код команды CMA
0804	32		STA	0B01H	;Код команды STA
0805	01				;Младший байт адреса0B01H
0806	0B				;Старший байт адреса0B01H
0807	FF		RST	7	;Код команды RST7

Подготовка к лабораторной работе

1. Изучить способы адресации команд микропроцессора
2. Изучить программу по таблицам 3.2.1 и 3.2.2.

3. Составить пять новых программ аналогичных программе 3.2.1, используя вместо команды CMA команду INRA, DCRA. ANAA, ORAA, XRAA.

4. Предоставить результаты записи программ преподавателю на проверку.

Порядок выполнения работы

1. Получить разрешение на включение УМК. После получения разрешения включить УМК (перед включением все клавиши отжаты, УМК подключен к сети 220В).

2. Ввести программу 3.2.1. в ОЗУ с адреса 0800.

3. Записать по адресу 0B00 заданное число, например 07.

4. Запустить программу на выполнение с адреса 0800.

5. Проверить результат выполнения программы 3.2.1. Для этого извлечь число из ячейки памяти с адресом 0B00, затем – 0B01. Записать результаты в тетрадь. Сравнить результаты.

4. Исследовать по аналогии с программой 3.2.1. пять новых программ, в которых вместо команды CMA, как в программе 3.2.1., применены команды INRA, DCRA. ANAA, ORAA, XRAA, соответственно.

5. После выполнения работы получить разрешение на выключение УМК. После получения разрешения выключить УМК.

Указание: новые программы можно рассматривать как модификации программы 3.2.1.

Отчет по работе должен содержать: название работы и задание на выполнение работы; тексты всех программ в мнемокодах и машинных кодах; выводы.

Контрольные вопросы

1. Какие форматы команд языка ассемблера применяют в микропроцессоре серии КР 580?

2. Как располагают в ячейках памяти 2-х и 3-х байтные команды?

3. На какие группы делят команды микропроцессоров КР 580?

4. Приведите примеры команд пересылки.

5. Приведите примеры логических команд.
6. Приведите примеры арифметических команд.
7. Поясните функции, выполняемые командами CMA, INRA, DCRA.
8. Поясните функции, выполняемые командами ANAA, ORAA, XRAA

3.3. Лабораторная работа № 3.
Способы адресации микропроцессора

Задание на работу

1. Изучить способы адресации команд МП.

2. Исследовать работу программ, размещенных в таблицах 3.3.1 и 3.3.2, с непосредственной, неявной, регистровой и косвенной адресациями.

Описание прямой, непосредственной, неявной, регистровой и косвенной адресациями микропроцессора

Метод задания операнда в команде называют прямой, непосредственной, неявной, регистровой и косвенной регистровой адресацией [2-4].

Команды **прямой адресации** имеют трехбайтный формат. Адрес указывает на местоположение данных в памяти МП. Например, команда LДА0В00 пересылает данные из ячейки памяти с адресом 0В00 в аккумулятор. В памяти программ МП эта команда занимает три последовательных ячейки: первая – код операции 3А, вторая – младший байт адреса 00, третья – старший байт адреса 0В, т.е. запись команды в машинных кодах имеет вид: 3А 000В. Другими словами, после кода операции вводится сначала младший байт адреса, затем старший байт адреса.

Команды **непосредственной адресации** имеют двухбайтный формат. Первый байт содержит код операции, а второй – данные. Например, команда MVIA,15 – записывает в аккумулятор шестнадцатеричное число 15Н. В памяти программ МП эта команда занимает две последовательные ячейки: первая – код операции 3Е, вторая – шестнадцатеричное число 15.

Команды **неявной адресации** имеют однобайтный формат. В этом байте содержится только код операции, указывающий на то, какие данные должны участвовать в обработке. По этой адресации в команде отсутствуют как адрес

обрабатываемых данных, так и непосредственные данные, находящиеся в одном из регистров общего назначения (РОН) или в регистре признаков (флагов). Например, команда STC – устанавливает признак переноса С в регистре признаков (флагов) в положение 1. В памяти программ микропроцессора эта команда занимает одну ячейку, в которой записан код операции 37Н.

Команды **регистровой адресации** имеют однобайтный формат. В этом байте содержатся код операции и адрес регистра-источника, в котором находятся данные. Регистром источником данных является один из РОН. Эти команды используются при выполнении арифметических или логических операций. МП выполняет действия над двумя операндами. Один из этих операндов находится в РОН, а второй – обязательно помещен в аккумулятор. Результат выполнения операции помещают в аккумулятор. Например, команда ANA C – выполняет поразрядную логическую операцию И над содержимым аккумулятора и данными, находящимися в регистре С. В памяти программ микропроцессора эта команда занимает одну ячейку, в которой записан ее код – А1. При выполнении команд пересылок при регистровой адресации в команде помимо кода операции указывается регистр-приемник и регистр-источник. По команде пересылки данные из регистра-источника пересылаются регистр-приемник. Например, команда MOVB,D – пересылает данные из регистра D в регистр В, код этой команды – 42.

Команды **косвенной регистровой адресации** имеют однобайтный формат. В этом байте содержится только код операции, указывающий, какие действия выполняют над данными, адрес расположения которых содержится в одной из регистровых пар РОН. По этой команде МП извлекает из заданной регистровой пары РОН шестнадцатеричный адрес, обращается по нему к ячейке памяти, из которой считывает данные и выполняет над ними действия, указанные в коде операции. Например, команда ADDM – складывает содержимое ячейки памяти, адрес которой указан в регистровой паре HLc содержимым аккумулятора, результат вычислений помещается в аккумулятор.

В памяти программ микропроцессора эта команда занимает одну ячейку, в которой записан ее код – 86Н.

Команды с прямой адресацией занимают в памяти 3 байта, и на их выполнение микропроцессором затрагивается много времени. Поэтому стремятся использовать команды с другими способами адресации. Однако следует учитывать, что в командах регистровой и косвенной адресации требуется предварительная загрузка в регистры РОН данных или адреса, где они хранятся.

В программе, размещенной в таблице 3.2.1,лабораторной работы 2 использована команда с прямой адресацией LDA 0B00. По этой команде некоторое число (операнд), предварительно помещенное в ячейку памяти с адресом 0B00 загружается в аккумулятор.

Изучение непосредственной, неявной, регистровой и косвенной адресации проводят на основе программ, размещенных в таблицах 3.3.1 и 3.3.2, записанных в мнемокодах.

Таблица 3.3.1

Метка	Мнемокод	Операнд	Комментарий
	MVI	A, 05	
	CMA		
	STA	0B01H	
	RST7		

Таблица 3.3.2

Метка	Мнемокод	Операнд	Комментарий
	MVI	A, 04	
	MVI	D, 05	
	ADD	D	
	LXIH	0A00H	
	MOV	M, A	
	RST7		

Подготовка к лабораторной работе

1. Изучить способы адресации команд МП

2. Исследовать работу программы, размещенной в таблице 3.3.1; заполнить графу «Комментарий» таблицы 3.3.1; записать программу, размещенную в таблице 3.3.1, в машинных кодах аналогично программе, размещенной в таблице 3.2.2, из лабораторной работы № 2.

3. Исследовать работу программы, размещенной в таблице 3.3.2.; заполнить графу «Комментарий» таблицы 3.3.2; записать программу, размещенную в таблице 3.3.1, в машинных кодах аналогично программе из таблицы 3.2.2. из лабораторной работы № 2.

4. Предоставить результаты записи программ преподавателю на проверку.

Порядок выполнения работы

1. Получить разрешение на включение УМК. После получения разрешения включить УМК (перед включением все клавиши отжаты, УМК подключен к сети 220В).

2. Ввести программу из таблицы 3.3.1 в ОЗУ с адреса 0800.

3. Выполнить программу из таблицы 3.3.1 в режиме пользователя.

4. Проверить результат выполнения программы из таблицы 3.3.1. Результат, находящийся в ячейке памяти 0B01, записать в отчет.

5. Ввести программу из таблицы 3.3.2.

6. Выполнить программу из таблицы 3.3.2 в режиме пользователя.

7. Проверить результат выполнения программы из таблицы 3.3.2.

8. После выполнения работы получить разрешение на выключение УМК. После получения разрешения выключить УМК.

Отчет по работе должен содержать: название работы и задание на выполнение работы; тексты всех программ в мнемокодах и машинных кодах; выводы.

Контрольные вопросы

1. В каком способе адресации операнд следует за кодом операции?
2. В каком способе адресации указывают адрес операнда после кода операции?
3. Какой способ адресации в командах межрегистровых пересылок?
4. В каком способе адресации адрес операнда указывают в регистровой паре?
5. Укажите все возможные способы адресации, которые используют в программах из таблиц 3.3.1 и 3.3.2.
6. Укажите различия в способах адресации, используемых в МП КР580 ВМ 80А.
7. Какие недостатки характерны для прямой адресации?
8. Каковы особенности применения косвенной адресации?
9. Как определяется операнд, задаваемых символическим адресом «М»?

3.4. Лабораторная работа № 4.
Команды переходов микропроцессора

Задание на работу

1. Изучить команды переходов МП.

2. Исследовать работу программу, заданную блок- схемой рис. 3.4.1.и таблицей 3.4.3.

Описание команд переходов микропроцессора

МП выполняет команды последовательно. Он считывает их из памяти одну за другой. В счетчике команд (PC) при этом хранится адрес следующей извлекаемой из памяти команды. К окончанию текущей команды в счетчике команд сформирован адрес следующей по порядку команды. Это справедливо для всех линейных программ.

В реальных программах почти всегда нарушается линейный порядок работы. Это нарушение создают команды условных и безусловных переходов, а также команды вызова подпрограмм. При их помощи реализуют циклические и разветвляющиеся алгоритмы. Адрес ячейки, на которой необходимо перейти МП загружают в счетчик команд PC. Такую процедуру называют передачей управления [2-4].

К командам переходов относят: 1) команды вызова подпрограмм CALL addr(B3B2) и команда возвращения из подпрограмм RET; 2) команды безусловных переходов JMP addr (B3B2); 3) команды условных переходов J (условие) addr (B3B2) или команды вызова подпрограмм по условию – C (условие) addr (B3B2) и возврата из подпрограмм по условию – R (условие) addr (B3B2) [2-4].

1. Команда вызова подпрограммы CALL для передачи подпрограмме процедуры вычислений требует три байта памяти. Первый байт содержит код

программы CD, во втором байте содержится младший байт адреса подпрограммы (B2) и в третьем – старший байт адреса подпрограммы(B3). При считывании этой команды МП автоматически запоминает содержимое счетчика команд (PC), записывая его в стек по принципу «автоматного магазина» - первым вошел, последним вышел. Английская аббревиатура этого принципа LIFO. Стек содержит свой счетчик SP (указатель стек), в котором записан адрес вершины стека. При записи в стек указатель стека автоматически уменьшается, а при чтении увеличивается.

По команде CALL МП загружает в счетчик команд PC число, записанное как адрес перехода. Одновременно в стек записывается содержимое младшего PCL и старшего PCH байтов счетчика команд. Это необходимо для того, чтобы после возврата из подпрограммы продолжить работу основной программы с прерванной команды.

Для возврата из подпрограммы в основную программу служит команда RET, являющаяся последней командой в любой программе. Команда RET однобайтовая, её код C9. По этой команде микропроцессор автоматически извлекает из стека два последних записанных туда числа и записывает последнее (перед записью в стек) содержимое счетчика команд PC.

2. Команда безусловного перехода имеет формат JMP addr (B2 B3)и располагается в трех байтах памяти: первый байт содержит код программы C3; второй - младший байт адреса перехода addrB2; третий байт – старший байт адреса перехода addrB3. При считывании этой команды микропроцессор записывает в счетчик команд PC адрес перехода.

Дальнейшее выполнение команд программы микропроцессор продолжит с указанного адреса.

3. Команды условных переходов осуществляют передачу управления только при выполнении некоторых условий. Если условие не выполняется, то передача управления не происходит. Организация установки условий осуществляется с помощью регистра признаков (флагов) (F). Регистр имеет 8 разрядов, в три из которых записаны константы (таблица 3.4.1). В каждый из пя-

ти оставшихся разрядов устанавливается в 1 или 0 по результатам выполнения текущей команды.

Таблица 3.4.1

Таблица регистра признаков

Д7	Д6	Д5	Д4	Д3	Д2	Д1	Д0
S	Z	0	AC	0	P	1	CY

S – разряд знака («сигнум»). Устанавливается в 1, если при выполнении арифметической или логической команды результат получился отрицательным, т.е. если в старшем разряде аккумулятора записана 1.

Z – разряд знака («зеро»). Устанавливается в 1 при нулевом результате выполнения команды, т.е. если во всех разрядах аккумулятора установлен 0.

AC – разряд десятичной коррекции устанавливается в 1, если при переработке двоично-десятичных чисел в аккумуляторе возникает перенос из разряда Д3 в разряд Д4 или заем в разряде Д4.

P – разряд знака («паритет»), разряд контроля по четности. Устанавливается в 1, если в результате выполнения команды количество единиц, установленных во всех разрядах аккумулятора, будет четным.

CY – разряд признака переноса. Устанавливается в 1, если при выполнении арифметических операций или команд сдвига возникает перенос из старшего разряда, а при вычитании происходит заем.

Проверяемым условием в командах условных переходов является значение одного из разрядов регистра состояния (флага). При этом могут проверяться следующие результаты выполнения команд: нулевой и ненулевой, положительный, отрицательный, четный, нечетный, наличие или отсутствие переноса. Переход может осуществляться как при наличии 1 в определённом разряде признаков, так и по нулю.

Все команды условных переходов являются трехбайтными. При считывании этих команд МП проверяет выполнение требуемого кодом команды

условия по состоянию нужного флага регистра. Если условие не выполняется, то выполняется следующая команда, адрес которой задан в PC, т.е. переход не происходит. Если условие выполнено, то в счетчик команд PC записывается адрес, указанный в команде, и дальнейшее выполнение команд программы, микропроцессора осуществляет с указанного адреса.

Мнемокод команд условного перехода состоит из первой буквы сдвига перехода J(JMP) и мнемокода условия перехода по признакам регистра F. После мнемокода команд условного перехода записывают адрес перехода – addr (B2 B3). Команда условного перехода, как и команда безусловного перехода, располагается в трех байтах памяти: первый байт содержит код программы; второй - младший байт адреса перехода addr B2; третий байт – старший байт адреса перехода addr B3.

ВМПКР 580 ВМ 80А помимо команд условного перехода, есть команды вызова подпрограмм по условию – C (условие) addr(B3B2) и возврата из подпрограмм по условию R (условие) addr(B3B2). Мнемокоды этих команд приведены в таблице 3.4.2.

Таблица 3.4.2

Мнемокоды команд переходов

Условие	Мнемокод		
Результат не равен нулю	JNZ	CNZ	RNZ
Результат равен нулю	JN	CZ	RZ
Нет переноса	JNC	CNC	RNC
Есть перенос	JC	CC	RC
Нечетный результат	JPO	CPO	RPO
Четный результат	JPE	CPE	RPE
Положительный результат	JP	CP	RP
Отрицательный результат	JM	CM	RM

Особенность использования команд условных переходов заключается в обязательной установке флагов регистра признаков. Эта установка должна быть выполнена до применения команд проверки условия. Поэтому при написании программы, содержащей команды условных переходов, проверяют выполнения установки флагов регистра признаков. Эта установка происхо-

дит только при выполнении арифметических и логических операций. Поэтому после команд пересылок или ввода данных в аккумулятор сразу использовать команды условных переходов нельзя, т.к. ни один из флагов регистра признаков F не изменяется. В этом случае перед командой условного перехода обязательно записывается команда, производящая установку признаков флага регистра F. Рекомендуется применять для этого логические команды типа ANAA и ORAA.

Отладка программы пользователя на УМК, содержащих команды переходов, осуществляется в поцикловом режиме работы МП.

Подготовка к лабораторной работе

1. Изучить команды переходов МП

2. Проанализировать работу программы, заданной рис.3.4.1 и таблицей 3.4.3.

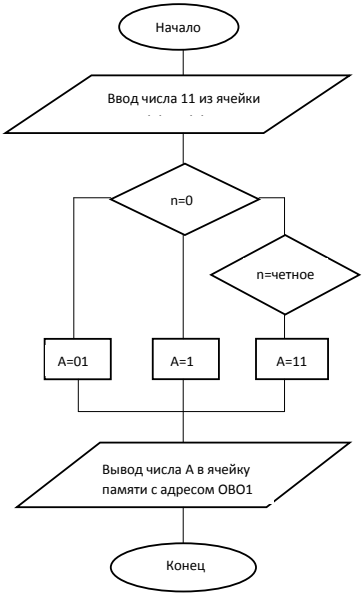


Рисунок 3.4.1 – Блок схема программы для изучения команд переходов

Таблица 3.4.3

Программа в машинных кодах и в мнемокодах

Адрес	Код	Метка	Мнемокод	Операнд	Комментарий
0800	21010B	XIH	LXIH	0B01	
0803	3A 01 0B		LDA	0B00	
0806	B7		ORAA		
0807	CA 10 08		JZ	ДAL1:”	
080A	EA 13 08		JPE	ДAL2:”	
080D	36 11		MVIM	11	
080F	76		RST7		
0810	36 01	ДAL1:	MVIM	01	
0812	76		RST7		
0813	36 10	ДAL2:	MVIM	10	
0815	76		RST7		

3. Заполнить столбец комментария таблицы 3.4.3.

4. Предоставить результаты подготовки преподавателю на проверку.

Порядок выполнения работы

1.1 Получить разрешение на включение УМК. После получения разрешения включить УМК (перед включением все клавиши отжаты, УМК подключен к сети 220В).

1.2 Ввести программу из таблицы 3.4.3.

1.3 Записать число в ячейку памяти 0B00, например, 05.

1.4 Выполнить программу в режиме пользователя.

1.5 Проверить последовательно ячейки памяти 0B00, затем 0B01. Результат записать в отчет.

1.6 Выполнить программу пользователя не менее трех раз, вводя различные числа в ячейку 0B00 так, чтобы в ячейке 0B01 получались числа 01, 10, 11. Результаты записать в отчет.

1.7 После выполнения работы получить разрешение на выключение УМК. После получения разрешения выключить УМК.

Отчет по работе должен содержать: название работы и задание на выполнение работы; тексты программ в мнемокодах и машинных кодах; выводы.

Контрольные вопросы

1. Какие команды переходов имеются в МП КР580ВМ80А?
2. Как изменяется содержимое счетчика команд РС при выполнении команд условных и безусловных переходов?
3. Как изменяется содержимое счетчика команд РС при переходе на подпрограмму и возврате в основную программу?
4. Какие признаки фиксируются в регистре признаков?
5. Поясните выполнение команд, приведенных в таблице 3.4.3.

3.5. Лабораторная работа № 5.
Изучение принципа функционирования дисплея УМК

Задание работу

1. Изучить программу из таблицы 3.5.6 на языке Ассемблера и в машинных кодах, ввести её в УМК и выполнить.

2. Изучить программу из таблицы 3.5.8 на языке Ассемблера, составить аналог программы в машинных кодах, ввести её в УМК и выполнить.

3. Изучить программу из таблицы 3.5.9 на языке Ассемблера и в машинных кодах, ввести ее в УМК и выполнить при времени задержки FFFF, а затем при времени задержки 0101.

Описание дисплея и управления дисплеем УМК

Дисплей УМК состоит из 6 семисегментных индикаторов АЛС 3245 расположенных в ряд, причем 4 первых индикатора слева предназначены для индикатора адреса, два последующих – для индикации байта данных. Индикаторы управляются через параллельный программируемый интерфейс (ППИ) – микросхема КР580ВВ55[4]. Порт А ППИ управляет соответствующими индикаторами. Порт В ППИ управляет свечением соответствующего сегмента в индикаторе. В дисплее УМК реализован режим динамической индикации, поэтому все одноименные сегменты всех шести индикаторов соединены параллельно.

В общем случае на дисплей УМК можно выводить любые знаки (буквы и цифры), которые позволяет реализовать структура семисегментного индикатора.

Для того чтобы некоторый знак высветился в каком-то индикаторе дисплея необходимо в порт А ППИ записать число, соответствующее номеру индикатора дисплея и настроить порт А на вывод информации, а в порт В

ППИ записать число, соответствующее выводимому знаку, т.е. число определяющие какие сегменты индикатора должны быть включены и настроить порт В на вывод информации.

ППИ, управляющий индикацией, имеет адреса:1) регистра управляющего слова (РУС); 2) порта А; 3) порта В.

1. Регистр управляющего слова (РУС) ППИ имеет порт вывода с адресом FB. Управляющее слово 80, выведенное по указанному адресу настраивает порты А и В ППИ на вывод информации.

2. Порт АППИ имеет адрес порта вывода F8. Соответствие индикаторов (счет идет слева направо) разрядам порта А представлено в таблице 3.5.1.

Таблица 3.5.1

Соответствие семисегментных индикаторов
шестиразрядного дисплей УМК разрядам порта АППИ –
микросхема КР580ВВ55

Индикаторы	HG1	HG2	HG3	HG4	HG5	HG6
Разряды порта А	Д0	Д1	Д2	Д3	Д4	Д5
Двоичный номер индикатора	$1*2^0$=00 000001	$1*2^1$=00 000010	$1*2^2$=00 000100	$1*2^3$=00 001000	$1*2^4$=00 010000	$1*2^5$=00 100000
Шестнадцатеричный номер индикатора	01	02	04	08	10	20
Десятичный номер индикатора	1	2	4	8	16	32

Из таблицы 3.5.1. следует, что если в порту А ППИ будет 01, то управляется индикатор HG1, если 02, то управляется индикатор HG2, 04 – HG3, 08 – HG4, 10 – HG5, 20 – HG6. Следует заметить, что номера здесь шестнадцатеричные, а также то, что нахождение в порту А шестнадцатеричного числа 03 вызовет свечение HG1 иHG2 одновременно.

Поясним сказанное примером. Для того чтобы было обеспечено управление 4-ым индикатором дисплея необходимо занести в аккумулятор код для включения 4-го индикатора (MV1 A, 08), а затем вывести этот код по адресу порта A (OUTF8). Поместим фрагмент программы примера в таблицу 3.5.2.

Таблица 3.5.2

Результат вывода в порт A ППИ кода,
управляющего свечением 4-ого индикатора

Метка	Мнемокод	Операнд	Комментарий
	MVI	A, 08	;В аккумулятор выводится число 08
	OUT	F8	;Содержимое аккумулятора выводится по адресу F9

3. Порт B ППИ имеет адрес F9. Этот порт управляет сегментами индикаторов. Соответствие разрядов порта B номеру сегмента индикатора представлено в таблице 3.5.3.

Таблица 3.5.3

Соответствие разрядов порта B сегментам индикатора

Д0	Д1	Д2	Д3	Д4	Д5	Д6	Д7
a	b	c	d	e	f	g	h

Пример. Чтобы вывести число «4» надо зажечь сегменты b, c, g, f какого либо индикатора. Результат представим в таблице 3.5.4.

Таблица 3.5.4

Активизация сегментов b, c, g, f индикатора дисплея из примера

Д7	Д6	Д5	Д4	Д3	Д2	Д1	Д0
a	b	c	d	e	f	g	h
0	1	1	0	0	1	1	0

Записи нулей и единиц из таблицы 3.5.4 соответствует шестнадцатеричное число 66H.

Поместим фрагмент программы примера в таблицу 3.5.5.

Таблица 3.5.5

Фрагмент программы, управляющей свечением числа «4»

Метка	Мнемокод	Операнд	Комментарий
	MVI	A, 05	;В аккумулятор выводится число 66
	OUT	F9	;Содержимое аккумулятора выводится по адресу F9

На основе фрагментов программ из таблиц 3.5.2 и 3.5.5. приведем новый обобщающий пример.

Требуется засветить число «5» в третьем разряде дисплея. Программа на языке ассемблера и в машинных кодах приведена в таблице 3.5.6. Программа в машинных кодах записана, начиная с адреса 0A00.

Таблица 3.5.6.

Программа на языке ассемблера и в машинных кодах для свечения числа «5» в третьем разряде дисплея

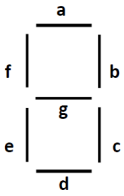
Адрес	Код	Метка	Мнемокод	Операнд	Комментарий
0A00	3E 80		MVI	A, 80	;В аккумулятор выводится число 80H
0A02	D 3 FB		OUT	FB	;В РУС через аккумулятор выводят 80H, что означает: порт А ППИ настраивается на вывод; порт В ППИ настраивается на ввод.
0A04	3E 6D		MVI	A, 6D	;В аккумулятор выводится число 6 D H
0A06	D 3 F9		OUT	F9	;В порт В через аккумулятор выводится 6 D H -код числа «5».
0A08	3E 04		MVI	A, 04	;В аккумулятор выводится число 04 H
0A0A	D3 F8		OUT	F8	;В порт А через аккумулятор выводится число 04H-код третьего индикатора.
0A0C	76		HLT		;Останов

На основе предыдущих примеров рассмотрим этапы создания программы для того, чтобы засветить на дисплее УМК слово «Пень», используя 4-е первых его индикатора.

Таблица 3.5.7

Подготовка задачи к программированию

	h	g	f	c	e	d	b	a	
	D7	D6	D5	D4	D3	D2	D1	D0	
П	0	0	1	1	0	1	1	1	= 37 H
Е	0	1	1	1	1	0	0	1	= 79 H
Н	0	1	1	1	0	1	1	0	= 76 H
Ь	0	1	1	1	1	1	0	0	= 7CH



Из таблицы 3.5.7. видно, что букве «П» соответствует шестнадцатеричное число 37 H; «Е» - 79 H; «Н» - 76 H; «Ь» - 7CH. Порт вывода РУС FB. Для настройки интерфейса (порты A и B на вывод) в него выводится число 80.

Порт A имеет адрес F8, в него должны быть выведены одно за другим шестнадцатеричные числа 01H, 02H,04H,08H, для управления свечением соответственно HG1, HG2, HG3, HG4. Порт B имеет адрес F9, в него должны быть выведены одно за другим шестнадцатеричные числа 37H, 79H, 76H, 7CH.

На основе произведенной подготовки запишем программу в мнемокодах для слова «ПЕНЬ» и поместим её в таблицу 3.5.8.

Таблица 3.5.8

Программа для свечения слова «ПЕНЬ» в четырех разрядах дисплея

Метка	Мнемокод	Операнд	Комментарий
«начало»:	MVI	A, 80	
	OUT	FB	
	MVI	A, 01	
	OUT	F8	
	MVI	A, 37	
	OUT	F9	
	MVI	A, 80	
	OUT	FB	
	MVI	A, 02	
	OUT	F8	
	MVI	A, 79	

Окончание таблицы 3.5.8

	OUT	F9	
	MVI	A, 80	
	OUT	FB	
	MVI	A, 04	
	OUT	F8	
	MVI	A, 76	
	OUT	F9	
	MVI	A, 80	
	OUT	FB	
	MVI	A, 08	
	OUT	F8	
	MVI	A, 7C	
	OUT	F9	
	JMP	«начало»addr	

Описание программных средств организации временных задержек

В практике микропроцессорных систем автоматического управления очень часто возникает необходимость в организации временных задержек. Наряду с аппаратными, существуют и программные средства их организации. В этом случае в основной программе создается подпрограмма временной задержки переход, к которой и возврат из нее осуществляется при помощи команд условных переходов. Для вызова программы используется команда CALL (адрес). Сама программа должна заканчиваться командой возврата из программы RET. При обращении к программе микропроцессор использует для запоминания содержимого счетчика команд и регистров область ОЗУ, называемую стеком.

Рассмотрим блок-схему алгоритма временной задержки.

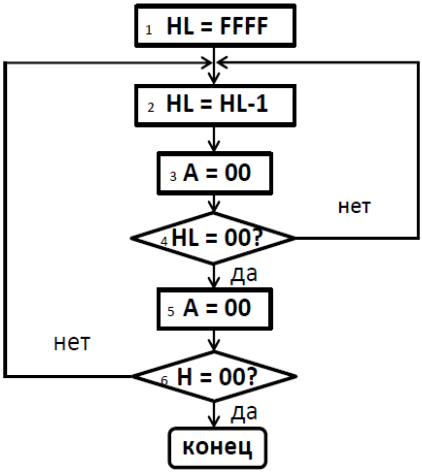


Рис. 3.5.1. Блок-схема временной задержки

Блок-схему можно пояснить следующим образом. Первый блок означает, что в регистровую пару HL загружено число, обеспечивающее требуемую задержку, в общем случае число может быть любым в диапазоне от 0001H до FFFFH; максимальную задержку дает наибольшее число. Блок 2 означает действие по уменьшению (декрементированию) содержимого регистра L регистровой пары HL на 1, блок 3 представляет загрузку в аккумуляторе числа 00, блок 4 обеспечивает сравнение содержимого регистра регистровой пары HL с содержимым аккумулятора, если равенства нет, то идет возврат к блоку 2 с выполнением в нем операции последующего декрементирования. После завершения указанного цикла (а каждое обращение к блоку 2 требует машинного времени) осуществляется переход к блоку 5, где вновь аккумулятор загружается числом 00. Блок 6 обеспечивает сравнение содержимого аккумулятора с регистром H регистровой пары HL, при отсутствии равенства будет осуществляться переход к блоку 2. Таким образом, пока содержимое регистровой пары HL не обнулится будет реализовываться вычисление в соответствии с алгоритмом временной задержки.

Рассмотрим пример, в котором вывод слова «ПЕНЬ» осуществляется при помощи программы, включающей подпрограмму временной задержки. Это позволяет получать либо «бегущие» буквы, при обеспечении большой временной задержки, либо «стоящие» буквы, как в программе примера 3.5.8, при создании короткого времени задержки.

Требуется высветить слово «ПЕНЬ», используя 4-е первых индикатора дисплея, причем слово должно быть представлено «бегущими» буквами.

Подготовка задачи к программированию осуществляется аналогично задаче из таблицы 3.5.8.

Приведем программу на языке ассемблера и мнемокодах в таблице 3.5.9. Основная программа в машинных кодах записана, начиная с адреса 0800, подпрограмма временной задержки TIME- с адреса 0900.

Таблица 3.5.9

Программа, высвечивающая «бегущими» буквами слово «ПЕНЬ»,
в мнемокодах и в машинных кодах

Адрес	Код	Метка	Мнемокод	Операнд	Комментарий
0800	31 000B	BEGIN:	LX1 S P	0B00	
0803	3E 80		MVI	A, 80	
0805	D3 FB		OUT	FB	
0807	3E 01		MVI	A, 01	
0809	D3 F8		OUT	F8	
080B	3E 37		MVI	A, 37	
0800	D3 F9		OUT	F9	
080F	CD 0009	TIME:	CALL	«TIME» addr0009	
0812	3E 80		MVI	A, 80	
0814	D3 FB		OUT	FB	
0816	3E 02		MVI	A, 02	
0818	D3 F8		OUT	F8	
081A	3E 79		MVI	A, 79	
081C	D3F9		OUT	F9	
081E	CD 0009	TIME:	CALL	«TIME» addr0009	
0821	3E 80		MVI	A, 80	
0823	D3 FB		OUT	FB	

Окончание таблицы 3.5.9

0825	3E 04		MVI	A, 04	
0827	D3 F8		OUT	F8	
0829	3E 76		MVI	A, 76	
082B	D3 F9		OUT	F9	
082D	CD 0009	TIME:	CALL	«TIME» addr 0009	
0830	3E80		MVI	A, 80	
0832	D3 FB		OUT	FB	
0834	3E 08		MVI	A, 08	
0836	D3 F8		OUT	F8	
0838	3E 7C		MVI	A, 7C	
083A	D3 F9		OUT	F9	
083C	CD 0009	TIME:	CALL	«TIME» addr 0009	
083F	C3 0008	BEGIN:	JMP	«BEGIN»addr 0008	
0900	21 FFFF	TIME:	LX1H	FFFF H	
0903	2B	M1:	DCXH		
0904	3E 00		MVI	A, 00	
0906	BD		CMP L		
0907	CZ 0309		JNZ	«M1»addr 0309	
090A	3E 00		MVI	A, 00	
090C	BC		CMPH		
090D	CZ 0309		JNZ	«M1»addr	
0910	C9		RET		

Подготовка к лабораторной работе

1. Изучить программу из таблицы 3.5.6 на языке ассемблера и в машинных кодах,

2. Изучить программу из таблицы 3.5.8 на языке ассемблера, записать аналог программы в машинных кодах

3. Изучить программу из таблицы 3.5.9 на языке ассемблера и в машинных кодах,

4. Предоставить результаты программ преподавателю на проверку.

Порядок выполнения работы

1. Получить разрешение на включение УМК. После получения разрешения включить УМК (перед включением все клавиши отжаты, УМК подключен к сети 220В).

2. Ввести в УМК и выполнить программу из таблицы 3.5.6.

3. Ввести в УМК и выполнить программу 3.5.8

4. Ввести в УМК и выполнить программу 3.5.9 при времени задержки FFFF, затем при времени задержки 0101.

5. После выполнения работы получить разрешение на выключение УМК. После получения разрешения выключить УМК

Отчет по работе должен содержать: название работы и задание на выполнение работы; тексты программ из таблиц 3.5.6, 3.5.8, 3.5.9 в мнемокодах и машинных кодах, блок-схему временной задержки; выводы.

Контрольные вопросы и задания

1. Каковы адреса РУС, портов А и В?

2. Каков порядок вывода определенного знака (буквы или цифры) на дисплее УМК?

3. Пояснить блок-схему алгоритма временной задержки.

4. Пояснить программу из таблицы 3.5.6.

5. Пояснить программу из таблицы 3.5.8.

6. Пояснить программу из таблицы 3.5.9.

Литература

1. Эксплуатационная документация УМК. Альбом 1. 1986 . – 90с.

2. Нарышкин, А.К. Цифровые устройства и микропроцессоры: учебное пособие для студентов высших учебных заведений радиотехнических специальностей / А.К.Нарышкин. – М.: Академия. – 2008. – 320с.

3. Мышляева, И.М. Информатика и вычислительная техника. Цифровая схемотехника / И.М. Мышляева. – М.: Академия. – 2005. – 400с.

4. Фурунжиев, Р.И. Микропроцессорная техника в автоматике / Р.И Фурунжиев., Н.И. Бохан.– Минск, Ураджай, 1991. – 280 с.

Приложение 1

Набор команд микропроцессора КР580ВМ80А

Число байт	Мнемокод	Такты	Символическое обозначение операций	Выполняемая функция
1	2	3	4	5
Команды пересылки				
1	MOVR1, R2	5	(R1)<– (R2)	Пересылка данных из регистра R2 в регистр R1
1	MOVR, M	7	(R)<–((H) (L))	Содержимое ячейки памяти, адрес которой хранится в регистрах H и L, пересылается в регистр R
1	MOVM, R	7	((H) (L)) <– (R)	Содержимое регистра R пересылают в ячейку памяти, адрес которой в регистрах H и L
2	MVI R, data 8	2	(R)<– (байт2)	Содержимое второго байта (B2) команды пересылают в регистр R
2	MVI M, data 8	2	((H) (L))<– (байт2)	Содержимое второго байта (B2) команды пересылают в ячейку памяти, адрес которой указан в регистрах H и L
3	LX1гр, data 16	10	(гH) <– (байт3) (гL) <– (байт2)	Третий байт команды пересылают в старший регистр H пары регистров гр, второй–в младший регистр L пары регистров гр
3	LDA addr(B2 B3)	13	(A) <– (байт3) (байт2)	Содержимое ячейки памяти, адрес которой указан во втором B2 и третьем B3 байтах команды, загружают в аккумулятор
3	STA addr(B2 B3)	13	(байт3) (байт2)<–(A)	Пересылка содержимого аккумулятора в ячейку памяти, адрес которой указан во втором B2 и третьем B3 байтах команды
3	SHLD addr(B2 B3)	16	((байт3) (байт2))<–(L) ((байт3) (байт2)+1)<–H)	Содержимое регистра L пересылают в ячейку памяти, адрес которой определен во втором B2и третьем B3байтах команды. Содержимое регистра H пересылается в следующую ячейку
1	LDAX гр	7	(A) <– ((гр))	Загрузка в аккумулятор содержимого ячейки памяти, адрес которой, записан в регистровой паре гр

1	STAX гр	7	((гр))<–(A)	Пересылка содержимого аккумулятора в ячейку, адрес которой задан в регистровой паре гр
Команды арифметических операций				
1	ADDR	4	(A) <–(A) + (R)	Сложение содержимого регистра R и аккумулятора. Результат помещают в аккумулятор в этой команде и во всех последующих аналогичных арифметических командах.
1	ADDM	7	(A)<– (A)+ ((H) (L))	Сложение содержимого ячейки памяти, адрес которой содержится в регистрах H и L, и аккумулятора
2	ADIdata 8	7	(A)<– (A)+ (байт2)	Содержимое второго байта команды складывают с содержимым аккумулятора
1	ADC R	4	(A)<– (A)+ (R)+(CY)	Содержимое регистра R и бит флага переноса складывают с содержимым аккумулятора
1	ADC M	7	(A)<– (A)+ ((H) (L))+(CY))	Содержимое ячейки памяти, адрес которой содержится в регистрах H и L и содержимое флага переноса складывают с содержимым аккумулятора
2	ASIdata 8 (B2)	7	(A)<– (A)+ (байт 2)(CY)	Содержимое второго байта команды и бита флага переноса складывают с содержимым аккумулятора
1	SUB R	4	(A) <–(A) - (R)	Содержимое регистра R вычитают из содержимого аккумулятора
1	SUB M	7	(A)<– (A)-((H) (L))	Содержимое ячейки памяти, адрес которой содержится в регистрах H и L вычитают из содержимого аккумулятора
2	SUIdata 8 (B2)	7	(A)<– (A)- (байт 2)	Содержимое второго байта команды вычитают из содержимого аккумулятора
1	SBB R	4	(A)<– (A)-(R)- (CY)	Содержимое регистра R и бит флага переноса CY вычитают из содержимого аккумулятора
1	SBB M	7	(A)<– (A)- ((H) (L))-(CY))	Содержимое ячейки памяти, адрес которой содержится в регистрах H и L и содержимое флага переноса CY вычитают из содержимого аккумулятора

2	SBI data 8 (B2)	7	(A)<– (A)-(байт 2)-(CY)	Содержимое второго байта команды и бит флага переноса CY вычитают из содержимого аккумулятора
1	INRR	5	(R) <–(R) +1	Инкрементирование содержимого регистраR, т.е.содержимое регистра Rувеличивают на единицу
1	INRM	10	((H) (L)) <– ((H) (L)) +1	Содержимое ячейки памяти, адрес которой содержится в регистрах H и L увеличивают на единицу
1	INX гр	5	((RH) (RL)) <– ((RH) (RL)) +1	Инкрементирование содержания регистровой пары гр, т.е. содержимое пары регистров гр увеличивают на единицу
1	DCR R	5	(R) <–(R) - 1	Декрементирование содержимого регистра R, т.е. содержимое регистра R увеличивают на единицу
1	DCR M	10	((H) (L)) <– ((H) (L)) +1	Содержимое ячейки памяти, адрес которой содержится в регистрах H и L уменьшают на единицу
1	DCX гр	5	((RH) (RL)) <– ((RH) (RL)) -1	Инкрементирование содержания регистровой пары гр, т.е. содержимое пары регистров гр уменьшают на единицу
1	DAD гр	10	((H) (L)) <– ((H) (L)) +гр	Сложение содержимого регистровой пары гр и содержимого регистров H и L. Результат помещают в пару регистров H, L.
1	DAA	4		Восьми битное число в аккумуляторе дополняют до представления в виде 4-битных чисел в двоично-десятичном коде.
Команды логических операций				
1	ANA R	4	(A)<– (A) ^ (R)	Поразрядное И над содержимым регистра Rи аккумулятора. Результат этой команды и других аналогичных команд помещают в аккумулятор.
1	ANA M	7	(A)<– (A) ^ ((H) (L))	Над содержимым ячейки памяти, адрес которой содержится в регистрах H и L и содержимым аккумулятора выполняют операцию логического И.

2	ANIdata 8 (B2)	7	(A)<– (A) ^ (байт 2)	Над содержимым второго байта команды и аккумулятора выполняют операция логического И.
1	XRA R	4	(A)<– (A)⊕(R)	Поразрядное ИСКЛЮЧАЮЩЕЕ ИЛИ над содержимым регистра Ri аккумулятора.
1	XRA M	7	(A)<– (A)⊕ ((H) (L))	Над содержимым ячейки памяти, адрес которой содержится в регистрах H и L и содержимым аккумулятора выполняют операцию ИСКЛЮЧАЮЩЕЕ ИЛИ.
2	XRIdata 8 (B2)	7	(A)<– (A)⊕ (байт 2)	Над содержимым аккумулятора и второго байта команды выполняют операция ИСКЛЮЧАЮЩЕЕ ИЛИ.
1	ORA R	4	(A)<– (A)∨(R)	Поразрядное ИЛИ над содержимым регистра R и аккумулятора
1	ORA M	7	(A)<– (A)∨ ((H) (L))	Над содержимым ячейки памяти, адрес которой содержится в регистрах H и L и содержимым аккумулятора выполняют операцию ИЛИ.
2	ORI data 8 (B2)	7	(A)<– (A)∨ (байт 2)	Над содержимым аккумулятора и второго байта команды выполняют операцию ИЛИ.
1	CMP R	4	(A)–(R)	Содержимое регистра R вычитают из содержимого аккумулятора. Содержимое аккумулятора не изменяется. Если (A)= (R), то флаг нуля Z устанавливается в 1.
1	CMP M	7	(A)–((H) (L))	Содержимое ячейки памяти, адрес которой содержится в регистрах H и L, вычитают из содержимого аккумулятора. Содержимое аккумулятора не изменяется. Если (A)= ((H)(L)), то флаг нуля Z устанавливается в 1.
2	CPI data 8 (B2)	7	(A)–(байт 2)	Содержимое второго байта команды вычитают из содержимого аккумулятора. Если (A) – (байт2) то флаг нуля Z устанавливается в 1.
1	RLC	4	(A_{n+1})<– (A_n) (A_0)<– (A_7) (CY)<– (A_7)	Содержимое аккумулятора сдвигают влево на одну позицию. Содержимое старшего бита заносят в младший бит.

1	RRC	4	$(A_n) \leftarrow (A_{n+1})$ $(A_7) \leftarrow (A_0)$ $(CY) \leftarrow (A_0)$	Циклический сдвиг содержимого аккумулятора
1	CMA	4	$(A) \leftarrow (\bar{A})$	Содержимое аккумулятора инвертируют (бит, равный 1, становится равным 0, бит равный 0, становится равным 1)
Команды передачи управления				
3	JMP addr (B2 B3)	10	(PC)<– (байт3) (байт2)	Управление передает команде адрес, который указан во втором и третьем байтах команды перехода
3	Jcondaddr (B2 B3)	10	(PC)<– (байт3) (байт2) Если условие истинно	Если указано условие (cond) истинно, то управление передают команде, адрес которой указан во втором и третьем байтах команды перехода. Если условие ложное, то последовательный код команды не изменяется.
3	JNZ addr (B2 B3)	10	(PC)<– (байт3) (байт2) Если условие истинно	Переход при условии NZ- не ноль (Z=0)
3	JZ addr (B2 B3)	10	(PC)<– (байт3) (байт2) Если условие истинно	Переход при условии Z- ноль (Z=1)
3	JNC addr (B2 B3)	10	(PC)<– (байт3) (байт2) Если условие истинно	Переход при условии отсутствия переноса(CY=0)
3	JC addr (B2 B3)	10	(PC)<– (байт3) (байт2) Если условие истинно	Переход при условии наличия переноса(CY=1)
3	JMaddr (B2 B3)	10	(PC)<– (байт3) (байт2) Если условие истинно	Переход при условии положительности(S=1)
3	JPE addr (B2 B3)	10	(PC)<– (байт3) (байт2) Если условие истинно	Переход, если четно
3	JP0 addr (B2 B3)	10	(PC)<– (байт3) (байт2) Если условие истинно	Переход, если нечетно

3	CALL addr (B2 B3)	10	((SP)-1)<– PCh) ((SP)-2)<– (PCl) (SP)<– (SP)-2 (PC)<– (байт3) (байт2)	Управление передают команде, адрес которой указан во втором и третьем байтах команды вызова. Старшие 8 бит адреса(PCh) пересылают в ячейку памяти, адрес которой на единицу меньше указателя стека(SP). Младшие 8 бит адреса (PC)пересылаются в ячейку памяти, адрес которой на 2 меньше указателя стека(SP). Содержимое указателя стека уменьшается на 2
3	Ccondaddr (B2 B3)	10	(PC)<– (байт3) (байт2) Если условие истинно	Если указанное условие (cond) аналогично условиям в командах J (условие), addr (B2 B3) истинно, то выполняют действия, описанные в команде CALL, в противном случае последовательность исполнения команды не изменяется
1	RET	10	(PCl) <– (SP) (PCh) <– (SP)+1) (SP)<– (SP)+2	Возврат. Содержимое ячейки памяти, адрес которой указан в указателе стека(SP), пересылают в 8 младших бит программного счетчика (PC). Содержимое ячейки памяти, адрес которой на единицу больше содержимого указателя стека, пересылают в 8 старших бит программного счетчика. Содержимое указателя стека увеличивают на 2.
1	R cond	5/11	(PCl) <– (SP) (PCh) <– (SP)+1) (SP)<– (SP)+2 Если условие истинно	Если указанное условие (cond) истинно, то выполняются действия, описанные в команде RET, в противном случае последовательность исполнения команды не изменяется
1	RSTn	11	(PCh) <– (L)	Старшие 8 бит адреса следующей команды пересылают в ячейку памяти, адрес которой на 1 меньше содержимого указателя стека(SP). Младшие 8 бит следующей команды пересылают в ячейку памяти, адрес которой на 2 меньше содержимого указателя стека. Содержимое указателя стека уменьшают на 2. Управление

				передают команде, адрес которой равен коду (NNN) умноженному на 8.
1	PCHL	5	PCl) <– (H) (PCh) <– (L)	Содержимое регистра (H) пересылают в 8 старших бит программного счетчика PC. Содержимое регистра (L) пересылают в 8 младших бит программного счетчика PC.
Команды стека, ввода – вывода и управления.				
2	IN port (B2)	10	(A) <– data8	Ввод данных. Данные, имеющиеся на восьми байтах двунаправленной шины данных указанного порта (port), пересылают в аккумулятор/
2	OUT port (B2)	10	data8 <– (A)	Вывод данных. Содержимое аккумулятора помещают на двунаправленную шину данных для передачи в указанный порт.
1	PUSH rp	11	((SP)-1)<– (rpH) ((SP)-2)<– (rpL) (SP)<– (SP)-2	Содержимое старшего байта регистровой пары rp пересылают в ячейку памяти, адрес которой на 1 меньше содержимого указателя стека SP. Содержимое младшего байта – в ячейку памяти, адрес которой на 2 меньше содержимого указателя стека. Содержимое указателя стека уменьшают на 2.
1	PUSH PSW	11	((SP)-1)<– (A) ((SP)-2)<– (CY, P, AC, Z) (SP)<– (SP)-2	Запись в стек слова состояния процессора. Содержимое аккумулятора пересылают в ячейку памяти, адрес которой на 1 меньше содержимого указателя стекаSP. Содержимое флагов объединяют в слово состояния (PSW) и пересылают в ячейку памяти, адрес которой на 2 меньше содержимого указателя стека. Содержимое указателя стека уменьшают на 2.
1	POP PSW	11	(A) <– ((SP)+1) ((SP)+2)<– (CY, P, AC, Z) (SP)<– (SP)+2	Чтение из стека слова состояния процессора. Содержимое ячейки памяти адрес которой определяется содержимым указателя стекаSP, используют для восстановления флагов.

				Содержимое ячейки памяти, адрес которой на 1 больше содержимого указателя стека, пересылают в аккумулятор. Содержимое указателя стека увеличивают на 2.
1	E1	4		Разрешение прерываний. Систему прерываний разрешают при исполнении следующей команды.
1	D1	4		Запрет прерываний. Систему прерываний запрещают непосредственно при выполнении следующей за D1 команды
1	HLT	7		Останов. В программный счетчик PC заносят адрес следующей команды. Процессор затем бездействует до прихода прерываний.
1	NOP	4		Пустая операция. В программный счетчик PC заносят адрес следующей команды и микропроцессор переходит к её обработке

Условные обозначения в наборе команд

R – один из регистров A, H, L, B, C, D, E;

M – ячейка памяти, адресуемая содержимым H-пары регистров;

rp – одна из регистровых пар B – (B,C), D – (D,E), H – (H,L) или указатель стека SP;

rpH, rpL – старший и младший регистры регистровой пары rp;

PCH, PCL – старший и младший регистры программного счетчика PC;

Приложение 2

КОДЫ КОМАНД МИКРОПРОЦЕССОРА КР580ВМ80А

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*A*	*B*	*C*	*D*	*E*	*F*	
0	NOP	LXIB (B2,B3)	STAXB	INXB	INRB	DCRB	MVIB,(B2)	RLC	-	DADB	LDAXB	DCXB	INRC	DCRC	MVI C,(B2)	RRC	*0*
1	-	LXI D (B2,B3)	STAX D	INX D	INR D	DCR D	MVI D,(B2)	RAL	-	DAD D	LDAX D	DCX D	INR E	DCR E	MVI E,(B2)	RAR	*1*
2	-	LXI H (B2,B3)	SHLD	INX H	INR H	DCR H	MVI H,(B2)	DAA	-	DAD H	LHLD	DCX H	INR L	DCR L	MVI L,(B2)	CMA	*2*
3	-	LXI SP (B2,B3)	STA (B2,B3)	INX SP	INR M	DCR M	MVI M,(B2)	STC	-	DAD SP	LDA (B2,B3)	DCX SP	INR A	DCR A	MVI A,(B2)	CMC	*3*
4	MOV B,B	MOV B,C	MOV B,D	MOV B,E	MOV B,H	MOV B,L	MOV B,M	MOV B,A	MOV C,B	MOV C,C	MOV C,D	MOV C,E	MOV C,H	MOV C,L	MOV C,M	MOV C,A	*4*
5	MOV D,B	MOV D,C	MOV D,D	MOV D,E	MOV D,H	MOV D,L	MOV D,M	MOV D,A	MOV E,B	MOV E,C	MOV E,D	MOV E,E	MOV E,H	MOV E,L	MOV E,M	MOV E,A	*5*
6	MOV H,B	MOV H,C	MOV H,D	MOV H,E	MOV H,H	MOV H,L	MOV H,M	MOV H,A	MOV L,B	MOV L,C	MOV L,D	MOV L,E	MOV L,H	MOV L,L	MOV L,M	MOV L,A	*6*
7	MOV M,B	MOV M,C	MOV M,D	MOV M,E	MOV M,H	MOV M,L	HLT	MOV M,A	MOV A,B	MOV A,C	MOV A,D	MOV A,E	MOV A,H	MOV A,L	MOV A,M	MOV A,A	*7*
8	ADD B	ADD C	ADD D	ADD E	ADD H	ADD L	ADD M	ADD A	ADC B	ADC C	ADC D	ADC E	ADC H	ADC L	ADC M	ADC A	*8*
9	SUB B	SUB C	SUB D	SUB E	SUB H	SUB L	SUB M	SUB A	SBB B	SBB C	SBB D	SBB E	SBB H	SBB L	SBB M	SBB A	*9*
A	ANA B	ANA C	ANA D	ANA E	ANA H	ANA L	ANA M	ANA A	XRA B	XRA C	XRA D	XRA E	XRA H	XRA L	XRA M	XRA A	*A*
B	ORA B	ORA C	ORA D	ORA E	ORA H	ORA L	ORA M	ORA A	CMP B	CMP C	CMP D	CMP E	CMP H	CMP L	CMP M	CMP A	*B*
C	RNZ	POP B	JNZ (B2,B3)	JMP (B2,B3)	CNZ (B2,B3)	PUSH B	ADI (B2)	PST 0	RZ	RET	JZ (B2,B3)	-	CZ (B2,B3)	CALL (B2,B3)	ACI (B2)	RST 1	*C*
D	RNC	POP D	JNC (B2,B3)	OUT (B2)	CNZ (B2,B3)	PUSH D	SUI (B2)	PST 2	RC	-	JC (B2,B3)	IN (B2)	CC (B2,B3)	-	SBI (B2)	RST 3	*D*
E	RPO	POP H	JPO (B2,B3)	XTHL	CPO (B2,B3)	PUSH H	ANI (B2)	PST 4	RPE	PCHL	JPE (B2,B3)	XCHG	CPE (B2,B3)	-	XRI (B2)	RST 5	*E*
F	RP	POP PSW	JP (B2,B3)	DI	CP (B2,B3)	PUSH PSW	ORI (B2)	PST 6	RM	SPHL	JM (B2,B3)	EI	CM (B2,B3)	-	CPI (B2)	RST 7	*F*
	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*A*	*B*	*C*	*D*	*E*	*F*	

Оглавление

Введение…….. 3
1. Учебный микропроцессорный комплект (УМК)……..….. 5
2. Основные правила составления программ на ассемблере……... 13
2.1. Коды команд…….…… 13
2.2. Программная модель микропроцессорной системы…….…… 14
2.3. Команды микропроцессора КР580ВМ80А…….…… 17
2.4. Формат ассемблерной строки…….…… 18
3. Лабораторные работы…….. 19
3.1. Лабораторная работа №1. Изучение УМК и приобретение навыков работы с его клавиатурой и дисплеем…….…… 19
3.2. Лабораторная работа №2. Запись и выполнение простых команд... 23
3.3. Лабораторная работа №3. Способы адресации микропроцессора... 27
3.4. Лабораторная работа №4. Команды переходов микропроцессора.. 32
3.5. Лабораторная работа №5. Изучение принципа функционирования дисплея УМК…… 39
Литература……..… 49
Приложение 1…… 50
Приложение 2…….…… 58

Учебное издание

Хорьков Сергей Алексеевич

Программирование микропроцессора

Подписано в печать 16.04.2021. Формат 60x84$^{1}/_{16}$
Усл. печ. л. 3,5. Уч.-изд. л. 1,8.
Тираж 50 экз. Заказ № 780.

Типография
Издательского центра «Удмуртский университет»
426034, Ижевск, ул. Университетская, 1, корп.2
Тел.68-57-18