

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего
образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

по организации и проведению самостоятельной работы

по дисциплине «Основы микропроцессорной техники»

для студентов направления подготовки

11.03.04 Электроника и нанoeлектроника

Направленность (профиль)

Интеллектуальные программируемые системы и устройства

Ставрополь 2026 г.

Печатается по решению
Учебно-методического совета
Северо-Кавказского федераль-
ного университета

УДК
ББК

Рецензент: Валюхов Д.П.

Лалин А.А. **Основы микропроцессорной техники:** методические указания для студентов по организации самостоятельной работы. – Ставрополь: Изд-во СКФУ, 20___. – 13 с.

Настоящие методические указания являются руководством по организации самостоятельной работы студентов по дисциплине «Основы микропроцессорной техники» для студентов очной формы обучения направления подготовки 11.03.04 Электроника и нанoeлектроника. В методических указаниях приведены: общая характеристика СРС, план-график выполнения, контрольные точки и виды отчетности, необходимые методические рекомендации и указания.

Методические указания составлены в соответствии с требованиями ФГОС ВО по направлению подготовки 11.03.04 Электроника и нанoeлектроника, рабочим учебным планом и программой дисциплины «Основы микропроцессорной техники».

УДК
ББК

© ФГАОУ ВО «Северо-Кавказский
федеральный университет», 20__

Введение

Целью дисциплины является формирование набора требуемых компетенций будущего бакалавра по направлению подготовки 11.03.04 Электроника и нанoeлектроника.

Задачи дисциплины: ознакомить студентов с основами микропроцессорной техники, принципами программирования микропроцессоров, возможностью использования микропроцессоров и устройств на их основе для решения различных прикладных задач, системой команд, организацией интерфейсов связи с внешним оборудованием.

Самостоятельная работа студентов является необходимым компонентом процесса обучения и может быть определена как творческая деятельность студентов, направленная на приобретение ими новых знаний и навыков. Настоящие методические указания помогут студентам выполнить указанный вид работ.

Содержание

1. Пояснительная записка	5
1.1. Цель и задачи СРС	5
1.2. Рекомендации и указания по выполнению СРС и форме контроля ...	5
2. План-график выполнения СРС	6
3. Методические рекомендации к СРС	6
4. Список рекомендуемой литературы	11

1 ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

1.1 Цель и задачи СРС

Цель самостоятельной работы студентов – систематическое изучение дисциплин в течение семестра, закрепление и углубление полученных знаний и навыков, подготовка к предстоящим занятиям, а также формирование культуры умственного труда и самостоятельности в поиске и приобретении новых знаний и умений, и, в том числе, формирование компетенций.

Основная тенденция инноваций в области образования определяется как переход от «научения к изучению»

Именно систематическое изучение учебной дисциплины позволяет студенту достигнуть уровня требований федерального государственного образовательного стандарта высшего образования (ФГОС ВО) к профессиональной подготовленности, которые сформулированы следующим образом: знать, уметь, владеть.

1.2 Тематика выполнения СРС, указания по форме контроля

Тематика СРС соответствует рабочей программе дисциплины «Основы микропроцессорной техники» по направлению подготовки 11.03.04 «Электроника и нанoeлектроника». Темы дисциплины приведены в таблице 1.

Таблица 1 – Темы дисциплины

№	Наименование разделов и тем дисциплины, их краткое содержание; вид самостоятельной работы	Форма контроля	Зачетные единицы (часы)
7 семестр			
1	Архитектура микропроцессора		0,(8)/ 32
1.1	Подготовка к практическим занятиям	Опрос	0,(2)/ 8
1.2	Подготовка к лабораторным занятиям	Опрос	0,1(6)/ 6
1.3	Углубленное изучение темы «Система команд»: микропроцессоров I80xx; микропроцессора Z80; современных 32- и 64-разрядных микропроцессоров. Обзор, сравнение, основные тенденции развития	Опрос	0,(2)/ 8
1.4	Конспектирование источников по теме «Система	Презентация	0,02(7)/ 1

	команд		
1.5	Углубленное изучение темы «Система команд»: современные языки программирования. Обзор, сравнение, основные тенденции развития	Опрос	0,(2)/ 8
1.6	Конспектирование источников по теме «Система команд»: современные языки программирования	Презентация	0,02(7)/ 1
2	Организация ввода/вывода в микропроцессорной системе		0,36/ 13
2.1	Подготовка к лабораторным занятиям	Опрос	0,(1)/ 4
2.2	Углубленное изучение тематики «Организация ввода-вывода в микропроцессорной системе»: современные интерфейсы связи с внешними устройствами. Обзор, сравнение, основные тенденции развития	Опрос	0,(2)/ 8
2.3	Конспектирование источников по тематике «Организация ввода-вывода в микропроцессорной системе»	Презентация	0,02(7)/ 1
3	Современные микропроцессоры и однокристальные микроЭВМ		0,25/ 9
3.1	Углубленное изучение темы «Типы, структуры и сравнительные характеристики современных микропроцессоров и однокристальных микроЭВМ»	Опрос	0,(2)/ 8
3.2	Конспектирование источников по теме «Типы, структуры и сравнительные характеристики современных микропроцессоров и однокристальных микроЭВМ»	Презентация	0,02(7)/ 1
4	Проектирование микропроцессорных систем		1,2(8)/ 46
4.1	Подготовка к практическим занятиям	Опрос	0,(1)/ 4
4.2	Углубленное изучение темы «Проектирование микропроцессорных систем»	Опрос	0,(2)/ 8
4.3	Конспектирование источников по теме «Проектирование микропроцессорных систем»	Презентация	0,02(7)/ 1
4.4	Углубленное изучение темы «Отладка программ»	Опрос	0,(2)/ 8
4.5	Конспектирование источников по теме «Отладка программ»	Презентация	0,02(7)/ 1
4.6	Углубленное изучение темы «Комплексная отладка микропроцессорных систем»	Опрос	0,(2)/ 8
4.7	Конспектирование источников по теме «Комплексная отладка микропроцессорных систем»	Презентация	0,02(7)/ 1
4.8	Контрольная работа	Защита, презентация результатов	0,41(6)/ 15
Итого за 7 семестр			2,78 (100)
ИТОГО			2,78 (100)

2 ПЛАН-ГРАФИК ВЫПОЛНЕНИЯ СРС

Подготовка к лекциям, лабораторным и практическим занятиям осуществляется непосредственно перед каждым занятием.

Контрольная/домашняя работа выполняется в 7 семестре (сроки выполне-

ния: 7 семестр; срок сдачи: сессия; формы контроля: защита, презентация результатов).

Темы контрольной/домашней работы выдаются в первые две недели учебного семестра. Защита работы проводится минимум за две недели до зачетно-экзаменационной сессии. Примерный календарный план ее выполнения приведен в таблице 2.

Таблица 2 – Примерный календарный план выполнения контрольной/домашней работы

Ход выполнения контрольной работы	Недели семестра
Получение задания	2 или 3
Ознакомление с основной и дополнительной литературой	3
Разработка алгоритмов работы и блок-схем	4-5
Написание программ на языке Ассемблера	6-10
Отладка программ	11-12
Оформление	13
Защита контрольной работы	14-15

3 МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ К СРС

Опрос по темам, изученным самостоятельно, может проходить в интерактивной форме в виде дискуссий или презентаций.

Выполнение заданий ведется в тетрадях для СРС или в электронном виде. Второй вид может быть использован для конспектирования.

Подготовка к лабораторным и практическим занятиям представляет собой внеаудиторную самостоятельную работу студентов и заключается в чтении:

- конспекта соответствующей лекции (если она читалась по данной теме);
- соответствующего раздела учебника и первоисточников (основная и дополнительная литература, интернет-ресурсы).

Контрольная/домашняя работа выполняется в 7 семестре (сроки выполнения: 7 семестр; срок сдачи: летняя сессия; формы контроля: защита, презентация результатов).

Примерная тема контрольной работы: «Изучение основ программирования на языке Ассемблер».

Для каждого студента предлагаются различные варианты задания, имеющие различные начальные условия:

а) обработка массивов данных:

- 1) способы сортировки элементов массива;
- 2) поиск элементов массивов по различным признакам;
- 3) вычисление элементов массива (массивов) по различным формулам и условиям;
- 4) выполнение различных операций над массивами по законам алгебры и алгебры логики;
- 5) размер массива (массивов) данных;
- 6) расположение массива (массивов) и результатов выполнения расчетов в памяти;

б) реализация типовых функций управления:

- 1) адрес размещения данных в области памяти;
- 2) номер ячейки дисплея порта вывода;
- 3) кнопки управления операциями и т. д.

Контрольная работа призвана путем решения практических задач обучить студентов:

- работать в среде эмулятора микропроцессора;
- разрабатывать программы на языке Ассемблер для выполнения арифметических операций над многоразрядными числами;
- разрабатывать программы на языке Ассемблер для выполнения логических операций над многоразрядными числами;
- разрабатывать программы на языке Ассемблер для обработки элементов массивов данных: сортировка и поиск элементов; вычисление элементов и т. д.;

- разрабатывать программы на языке Ассемблер для обработки массивов данных: выполнение операций над массивами данных; вычисление их размеров; перемещение массивов и т. д.;

- разрабатывать программы на языке Ассемблер, предназначенных для реализации типовых функций управления: сбор данных с порта ввода; размещение данных в области памяти, доступной пользователю; вывод данных в порт вывода; ожидание команд управления и т. д.

Контрольная работа оформляется в виде пояснительной записки и приложений общим объемом от 25 до 35 листов формата А4. Алгоритмы и их блок-схемы, способ оформления программы на языке Ассемблер, схемы электрические принципиальные и рисунки выполняются с соблюдением требований соответствующих государственных стандартов, ЕСПД, ЕСКД, ЕСТД.

К защите должны быть представлены разработанные программы и в электронном виде с целью демонстрации преподавателю их работоспособности.

Общие требования к оформлению текстовой части работы приведены в ГОСТ 2.105 – 95. ЕСКД. Общие требования к текстовым документам (с поправкой от 05.12.2011 г.).

Текст работы выполняют с использованием компьютера на одной стороне листа белой бумаги формата А4, шрифт – Times New Roman, размер – 14 пт, межстрочный интервал – 1,5. Номер страницы проставляют в центре нижней части листа. Страницы текстового материала следует нумеровать арабскими цифрами, соблюдая сквозную нумерацию по всему документу. Титульный лист текстового документа и листы задания включают в общую нумерацию страниц. Номер страницы на указанных листах не проставляют. Расстояние от края бумаги до границ текста следует оставлять: в начале строк – 30 мм; в конце строк – 10 мм; от верхней или нижней строки текста до верхнего или нижнего края бумаги – 20 мм. Размер абзацного отступа должен быть одинаковым по всему тексту работы и равным 12,5 мм.

Алгоритм и его блок-схема должны быть оформлены в соответствии с требованиями Единой системы программной документации – ЕСПД (ГОСТ 19.002-80. ЕСПД. Схемы алгоритмов и программ. Правила выполнения; ГОСТ 19.003-80.

ЕСПД. Схемы алгоритмов и программ. Обозначения условные графические; ГОСТ 19.701-90. ЕСПД. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения; ГОСТ 19.503-79. ЕСПД. Руководство системного программиста. Требования к содержанию и оформлению; ГОСТ 19.504-79. ЕСПД. Руководство программиста. Требования к содержанию и оформлению; ГОСТ 19.505-79. ЕСПД. Руководство оператора. Требования к содержанию и оформлению).

4 СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

4.1 Основная

1. Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд., испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

4.2. Дополнительная литература:

1. Нарышкин, А. К. Цифровые устройства и микропроцессоры : учеб. пособие / А. К. Нарышкин. - 2-е изд., стер. - М. : Издательский центр "Академия", 2008. - 320 с. : ил., рис., табл., сх. - (Высшее профессиональное образование). - Библиогр.: с. 312-314.

2. Калабеков, Б. А. Цифровые устройства и микропроцессорные системы : учебник для средн. спец. уч. зав. связи / Б. А. Калабеков. - 2-е изд., перераб. и доп. - М. : Горячая линия - Телеком, 2007. - 336 с. : ил. - Библиогр. с. 334.

3. Балашов, Е. П. Микро- и мини-ЭВМ : учеб. пособие для вузов / Е. П. Балашов, В. Л. Григорьев, Г. А. Петров. - Л. : Энергоатомиздат, 1984. - 376 с. : ил. - (для студентов вузов). - Библиогр.: с. 371-372. - Предм. указ.: с. 373-374.

4. Токхайм, Р. Л. Микропроцессоры : курс и упражнения / Р. Токхайм ; пер. с англ. В. Н. Грасевича, Л. А. Ильяшенко. - М. : Энергоатомиздат, 1988. - 336 с. - Библиогр. с. 334. - ISBN 5-283-02487-3 (русск.). - ISBN 0-07-064958-8 (англ.).
5. Майоров, С. А. Введение в микроЭВМ / С. А. Майоров, В. В. Кириллов, А. А. Приблуда. - Л. : Машиностроение, 1988. - 304 с. - Библиогр.: с. 298-299. - Предм. указ.: с. 300-302.
6. Лю, Ю-Ч. Микропроцессоры семейства 8086/8088 : архитектура, программирование и проектирование микрокомпьютерных систем / Ю-Чжен Лю, Г. Гибсон ; пер. с англ. В. Л. Григорьева. - М. : Радио и связь, 1987. - 512 с. - Прил.: с. 500-504. - Библиогр.: с. 505-.
7. Гибсон, Г. Аппаратные и программные средства микро-ЭВМ / Г. Гибсон, Ю-Ч. Лю ; пер. с англ. В. Л. Григорьева ; под ред. В. В. Сташина. - М. : Финансы и статистика, 1983. - 255 с. - Библиогр.: с. 249-252.
8. Микро-ЭВМ : учебное пособие / под ред. А. Дирксена ; пер. с англ. В. В. Сташина. - М. : Энергоиздат, 1982. - 328 с. : ил. - Библиогр.: с. 318. - Предм. указ.: с. 318-323.
9. Хоровиц. Искусство схемотехники. Т1/ У.Хилл. - 3-е изд., стереотип. - М.: Мир, 1986. - 598с.:ил.
10. Хоровиц, П. Искусство схемотехники. Т2/ У.Хилл. - 3-е , стереотип. - М.: Мир, 1986. - 590с.:ил. - с494.
11. Мячев, А. А. Персональные ЭВМ и микроЭВМ: основы организации : [справочник] / А. А. Мячев, В. Н. Степанов ; под ред. А. А. Мячева. - М. : Радио и связь, 1991. - 320 с. - Прил.: с. 275-299. - Слов. основных терм.: с. 300-308. - Слов. иностр. терм. и сокр.
12. Щелкунов, Н. Н. Микропроцессорные средства и системы / Н. Н. Щелкунов, А. П. Дианов. - М. : Радио и связь, 1989. - 288 с. - Библиогр.: с. 284-286. - ISBN 5-256-00256-2.
13. Микропроцессорный комплект К1810 : структура, программирование, применение : справочная книга / [Ю. М. Казаринов и др.] ; под ред. Ю. М. Казаринова. - М. : Высшая школа, 1990. - 269 с. - Прил.: с. 249-264. - Библиогр.: с. 265-

266. - ISBN 5-06-000821-5.

14. Казаринов, Ю. М. Применение микропроцессоров и микроЭВМ в радиотехнических системах : учеб. пособие для вузов / Ю. М. Казаринов, В. Н. Номоконов, Ф. В. Филиппов. - М. : Высшая школа, 1988. - 207 с. : ил. - Библиогр.: с. 206. - ISBN 5-06-001271-9.

ОСНОВЫ МИКРОПРОЦЕССОРНОЙ ТЕХНИКИ

**Методические указания для студентов
по организации самостоятельной работы
по направлению подготовки
11.03.04 Электроника и нанoeлектроника**

Квалификация выпускника

Бакалавр

Составитель: Лапин А.А.

Редактор:

Формат _____	Подписано в печать _____	Уч.-изд. л. – _____.
Бумага _____.	Усл. п. л. – _____	Заказ _____
Печать _____.	Тираж _____ экз.	

ФГАОУ ВО «Северо-Кавказский федеральный университет»
355029, г. Ставрополь, пр. Кулакова, 2

Издательство Северо-Кавказского федерального университета
Отпечатано в типографии СКФУ

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего
образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

по выполнению практических работ

по дисциплине «Основы микропроцессорной техники»

для студентов направления подготовки

11.03.04 Электроника и нанoeлектроника

Направленность (профиль)

Интеллектуальные программируемые системы и устройства

Ставрополь 2026 г.

Печатается по решению
Учебно-методического совета
Северо-Кавказского
федерального университета

УДК
ББК

Рецензент: Валюхов Д.П.

Лапин А.А. **Основы микропроцессорной техники:** методические указания по выполнению практических работ (часть 1). – Ставрополь: Изд-во СКФУ, 20___. – 50 с.

Настоящие методические указания являются руководством по выполнению практических работ по дисциплине «Основы микропроцессорной техники» для студентов очной формы обучения направления подготовки 11.03.04 Электроника и нанoeлектроника. В методических указаниях приведены необходимые теоретические обоснования, перечень оборудования, варианты заданий, примеры их решения, требования к содержанию отчета, его форме, литература.

Методические указания составлены в соответствии с требованиями ФГОС ВО по направлению подготовки 11.03.04 Электроника и нанoeлектроника, рабочим учебным планом и программой дисциплины «Основы микропроцессорной техники».

УДК
ББК

© ФГАОУ ВО «Северо-Кавказский
федеральный университет», 20__

Введение

Целью дисциплины является формирование набора требуемых компетенций будущего бакалавра по направлению подготовки 11.03.04 Электроника и наноэлектроника.

Задачи дисциплины: ознакомить студентов с основами микропроцессорной техники, принципами программирования микропроцессоров, возможностью использования микропроцессоров и устройств на их основе для решения различных прикладных задач, системой команд, организацией интерфейсов связи с внешним оборудованием.

Выполнение практических работ способствует получению знаний и умений, согласно заявленным для дисциплины компетенциям, на базе теоретического материала, изучаемого на лекционных занятиях.

Содержание

Практическое занятие 1, 2. Изучение микропроцессора и основ программирования на языке Ассемблер	5
Практическое занятие 3. Изучение методов обработки массивов данных	16
Практическое занятие 4, 5, 6. Изучение арифметических операций ...	24
Практическое занятие 7, 8. Изучение методов работы с разрядами и операциями сдвига	35

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ 1, 2

Тема: Изучение микропроцессора и основ программирования на языке Ассемблер.

Цель работы: закрепление знаний, полученных на лекциях по разделу: «Архитектура микропроцессора».

Знания, приобретаемые студентом в результате освоения темы:

- аппаратная и логическая структуры типового микропроцессора;
- режимы адресации и система команд.

Умения, приобретаемые студентом в результате освоения темы:

- разрабатывать алгоритмы работы и программное обеспечение для аппаратной части микропроцессорных систем.

Формируемые компетенции: ОК-6, ОК-7, ОПК-6, ОПК-7, ОПК-8, ОПК-9, ПК-1, ПК-3.

Актуальность темы: тема является актуальной, так как многие современные электронные устройства построены на основе микропроцессоров разных поколений и семейств. Поэтому студент должен знать основные элементы структурных схем МП, уметь составлять блок-схемы алгоритмов программ, а так же сами программы функционирования.

Теоретическая часть

Изучаемый микропроцессор (МП) K580BM80 (далее BM80) является аналогом микропроцессора 8080, выпущенного фирмой *Intel* в 1974 г. МП представляет собой функционально законченный однокристалльный универсальный восьмиразрядный микропроцессор с фиксированной системой команд. Структурная схема МП приведена на рисунке 1.1.

В структурной схеме можно выделить следующие ее части: арифметико-логическое устройство (АЛУ), блок регистров, буферные схемы, управляющее устройство. О назначении каждой части можно прочитать в

соответствующей литературе: основной [1] и дополнительной [1 - 6].

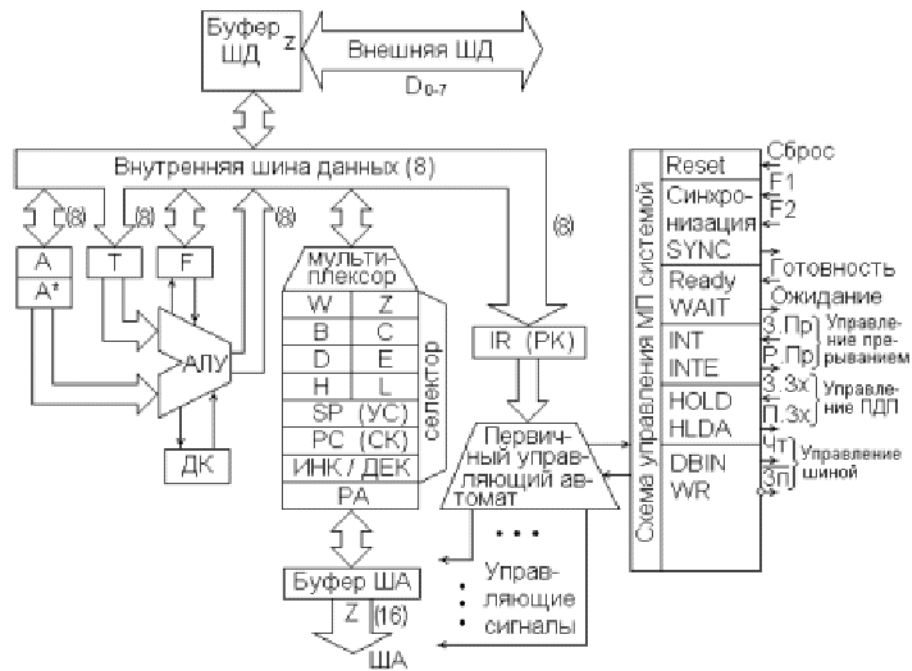


Рисунок 1.1 – Структурная схема восьмиразрядного МП

Блок регистров VM80 содержит 8-разрядные программно-доступные регистры: *A* (регистр-аккумулятор); *B*, *C*, *D*, *E*, *H*, *L*. (регистры общего назначения); *F* (регистр признаков-флагов); 16-разрядные специализированные регистры; программный счетчик СК (*PC*); регистр-указатель стека УС (*SP*); сдвоенный регистр косвенного адреса *HL* (*H* – регистр старшего полуадреса, *L* – регистр младшего полуадреса). Также имеются непосредственно недоступные программе регистры: 8-разрядный регистр команд (ПК); 8-разрядные регистры временного хранения *T*, *W*, *Z*; 16-разрядный регистр адреса *РА*. Содержимое пар регистров *B* и *C*, *D* и *E*, *H* и *L* можно использовать в качестве слов двойной длины, т.е. 16-разрядных слов. Более подробное описание – в соответствующей (указана выше).

Модель микропроцессора содержит только узлы, наиболее важные для понимания процесса его работы. Модель может содержать программно-доступные и программно-недоступные узлы. На рисунке 1.2 представлены программно-доступные (адресуемые в командах в явной или неявной форме)

узлы МП, памяти и устройств ввода-вывода. На рисунке 1.3 указаны программно-недоступные узлы МП, наиболее существенные для процесса выполнения команд.

В программной модели выделены регистры *B, C, D, E, H, L, A*. Эти регистры предназначены для хранения данных. Они находятся в полном распоряжении программиста и единообразно участвуют в арифметических и логических операциях. К этой же группе отнесена так называемая ячейка памяти *M*, адрес которой помещен в регистровую пару *HL*. Ячейка памяти *M* также предназначена для хранения данных и находится в полном распоряжении программиста после того, как ее адрес загружен в регистровую пару *HL*.

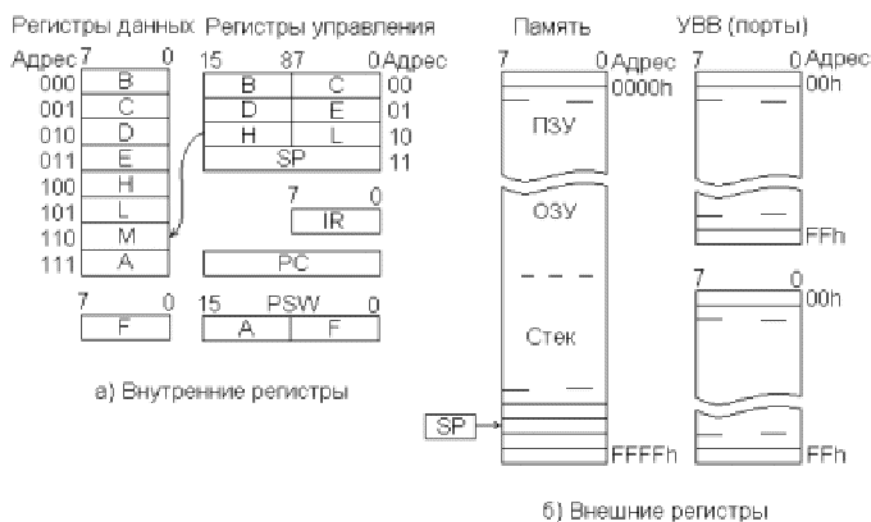


Рисунок 1.2 – Программно-доступные узлы МП (а), памяти и устройств ввода-вывода (б)



Рисунок 1.3 – Программно-недоступные узлы МП

Регистр признаков F , показанный на рисунке 1.4, содержит признаки выполнения операций. В регистре фиксируются признаки:

- CY – признак переноса заема. Если при выполнении команды возник перенос из старшего разряда или заем в старший разряд, то $CY = 1$, иначе $CY = 0$;
- P – признак паритета. Если число единиц в байте результата четно, то $P = 1$, иначе $P = 0$;
- AC – признак вспомогательного переноса. Если есть перенос между тетрадами байта, то $AC = 1$, иначе $AC = 0$;
- Z – признак нуля. Если результат равен нулю, то $Z = 1$, иначе $Z = 0$;
- S – признак «знака» принимает значение старшего разряда результата.

D7	D6	D5	D4	D3	D2	D1	D0
S	Z	0	AC	0	P	1	CY

Рисунок 1.4 – Регистр признаков F

Команды микропроцессора VM80 имеют однобайтный, двухбайтный и трехбайтный форматы. Поле команды содержит информацию (в кодах) о типе операции (код операции), об адресе операнда, типе обращения к памяти, о количестве байт в команде. Примеры команд приведены на рисунке 1.5.

Способы определения источников и приемников операндов называют режимами адресации данных. Для МП разработано около двух десятков режимов адресации. В МП VM80 используются только четыре (рисунок 1.6):

- прямая (абсолютная) адресация. Это наиболее простая, но наименее экономичная адресация. В поле самой команды содержится полный 16-битный адрес операнда в памяти (рисунок 1.6, а). С помощью прямой адресации можно обращаться к любой ячейке в адресном пространстве;
- непосредственная адресация. Данные (байт или два байта) находятся непосредственно в команде во втором или во втором и третьем ее байтах

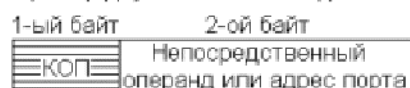
(рисунок 1.6, б);

Примеры однобайтных команд:



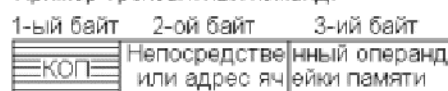
MOV r_d, r_s , где r_d и r_s - имя регистра-приемника и имя регистра-источника операнда соответственно.
Пример конкретной однобайтной команды (переслать содержимое регистра A в регистр C):
MOV C, A ; (A)→C
sss=111 - код регистра A,
ddd=001 - код регистра C,
01=КОП (код операции).
Машинный код команды
MOV C,A: 01 001 111.

Пример двухбайтных команд:



MVI B, 75h ; 75→B
OUT, 87h ; (A)→87h

Пример трехбайтных команд:



LXI B, 7C, 24h ; 7Ch→B; 24h→C
JNZ, b_3b_2 ; если результат не нулевой, т.е. Z=0,
то b_3b_2 →PC, т.е. переход

Рисунок 1.5 – Форматы команд

– регистровая адресация. Операндом является содержимое адресуемого в команде регистра (рисунок 1.6, в). Команды с регистровой адресацией имеют однобайтный формат, выполняются достаточно быстро. В однобайтных командах с неявной адресацией подразумевается, что операнд находится в определенном внутреннем регистре МП и его специально адресовать не надо;

– косвенная регистровая адресация (рисунок 1.6, г) позволяет компактно адресовать все пространство памяти. В этом режиме в поле команды содержится указание на регистровую пару, содержащую адрес операнда (данных). Косвенная адресация широко применяется при обращении к структурам данных типа массивов.

Систему команд МП можно представить в виде 76 мнемочкодов (ассемблерных кодов). Здесь команды ВМ80 рассматриваются с позиции описания действий микропроцессора. Всю систему команд можно по

функциональным признакам разделить на группы:

- пересылки/загрузки (рисунки 1.7, ..., 1.12);
- положительных/отрицательных приращений;
- арифметические (рисунок 1.13);
- логические (рисунок 1.14);
- сдвига в аккумуляторе;
- передачи управления (таблица 1.1, рисунки 1.15 и 1.16);
- ввода-вывода;
- специальные.

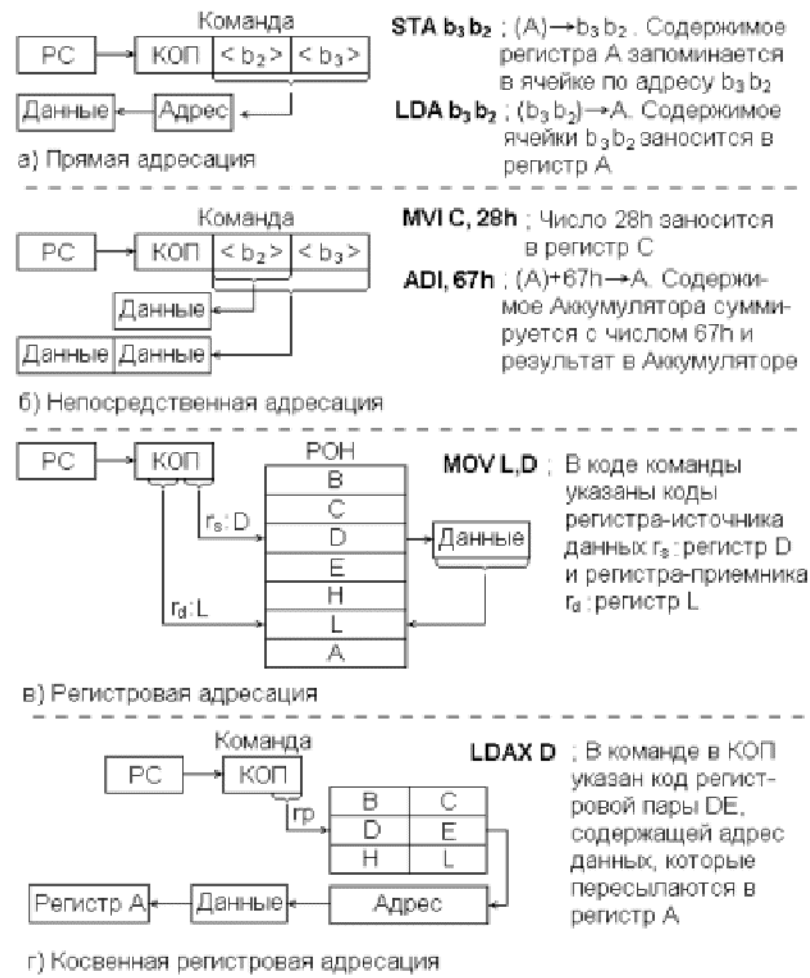


Рисунок 1.6 – Режимы адресации данных

Приведенные далее рисунки показывают действия лишь некоторых команд языка Ассемблер и предназначены для понимания действий МП при

их выполнении.

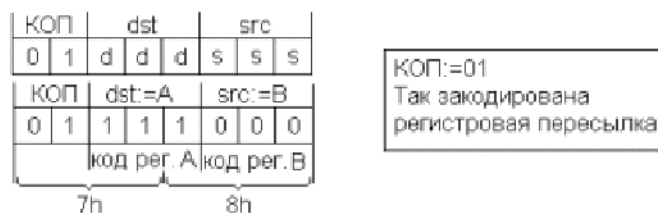


Рисунок 1.7 – Машинный код команды *MOV A,B*

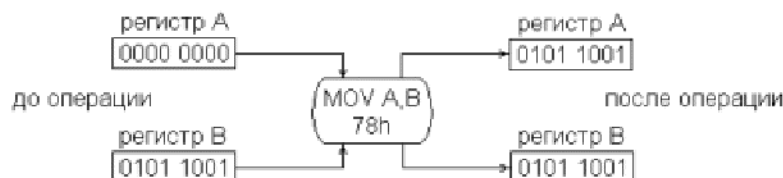


Рисунок 1.8 – Действие команды *MOV A,B*

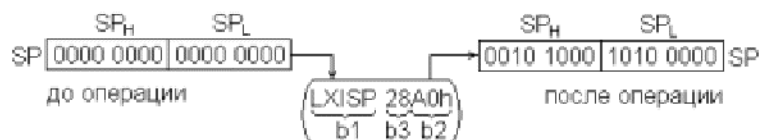


Рисунок 1.9 – Загрузка указателя стека *SP*

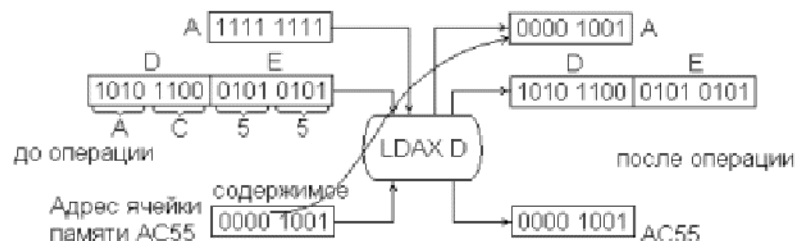


Рисунок 1.10 – Выполнение команды *LDAX D*

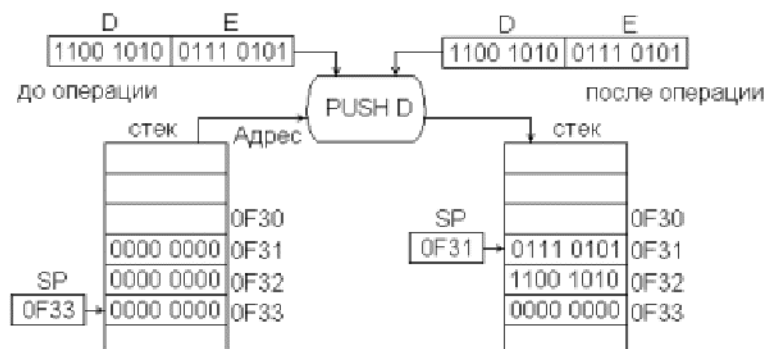


Рисунок 1.11 – Выполнение команды *PUSH D*

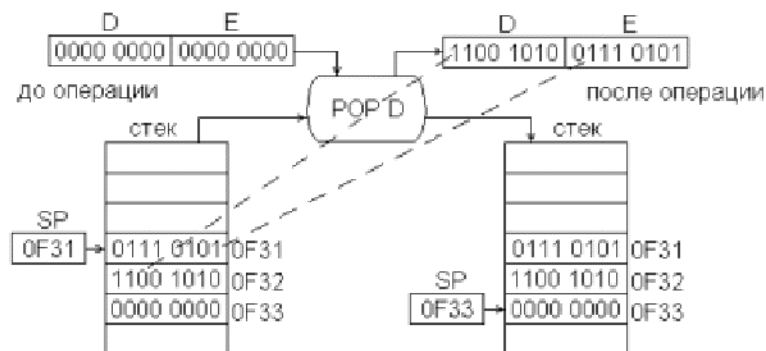
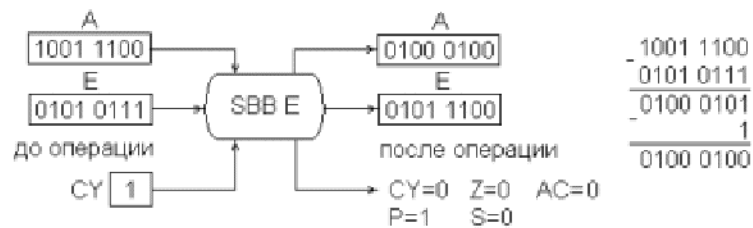
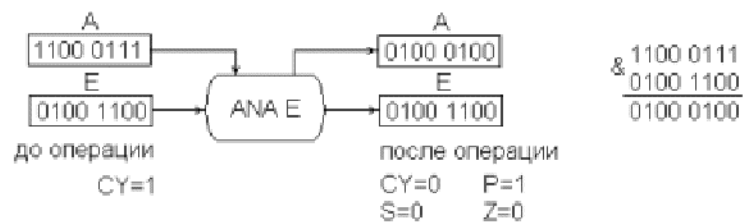
Рисунок 1.12 – Выполнение команды *POP D*Рисунок 1.13 – Выполнение команды *SBB E*Рисунок 1.14 – Выполнение команды *ANA E*

Таблица 1.1 – Условия передачи управления

cop	CCC	Условия передачи управления
NZ	000	Если результат не нулевой, т. е. Z=0
Z	001	Если результат нулевой, т.е. Z=1
NC	010	Если не было переноса/заема, т. е. CY=0
C	011	Если был перенос/заем, т. е. CY=1
PO	100	Если результат нечетный, т. е. P=0
PE	101	Если результат четный, т. е. P=1
P	110	Если результат положительный, т. е. S=0
M	111	Если результат отрицательный, т. е. S=1

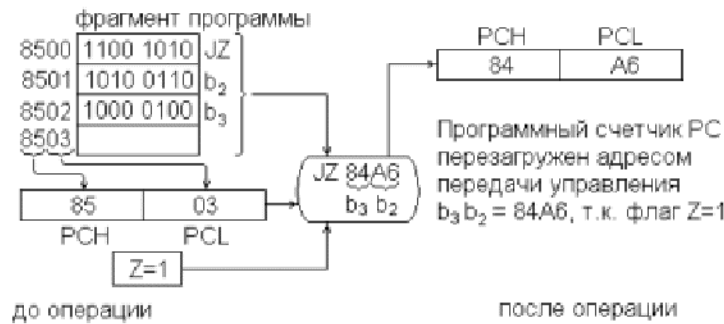


Рисунок 1.15 – Выполнение команды передачи управления по условию $Z=1$

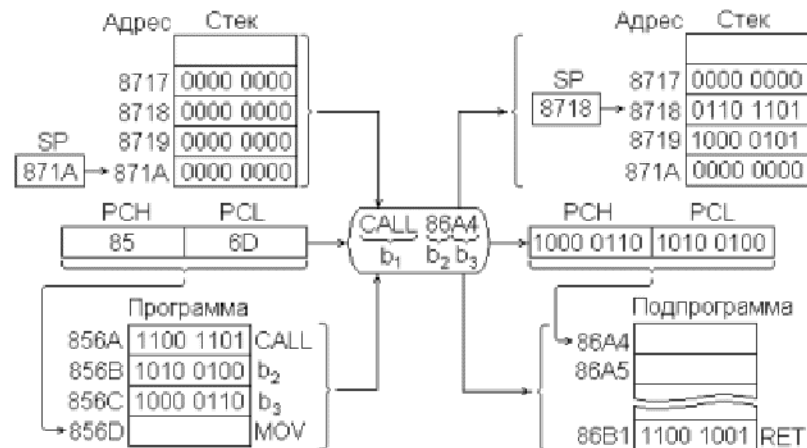


Рисунок 1.16 – Выполнение команды *CALL* 86A4

Запись программ на языке Ассемблер принято вести определенным образом. Это позволяет понимать программы, написанные разными программистами. Пример оформления приведен в таблице 1.2.

Таблица для записи программы имеет следующие поля: адреса, кода команды, ассемблерного кода (мнемокода) и комментариев. Поле метки используется при наличии команд передачи управления и в данной программе остается пустым.

Поле мнемокода содержит символьное обозначение кода команды на языке ассемблера.

Поле операнда содержит информацию о регистрах, данных и адресах, объединенных соответствующей операцией. Программа ассемблер может выдать соответствующий машинный код, используя только поля мнемокода и операнда. Списку команд желательно назначить ячейки памяти.

Таблица 1.2 – Пример оформления программы на языке Ассемблер

Адрес (h-код)	Машин- ный код (h-код)	Мет- ка	Мнемо- код	Операнд	Комментарий
8500 8501 8502	21 00 84		LXI	H, 8400h	Загрузка указателя адреса, т.е. 8400h → HL
8503 8504	DB 01		IN	01	Ввод числа из порта 01 в аккумулятор
8505	86		ADD	M	Сложение содержимого ячейки M и регистра A, т.е. (A)+(M) → A
8506	23		INX	H	Увеличить указатель адре- са, т.е. (HL)+1 → HL
8507			MOV	M, A	Заполнить сумму в ячейке 8401
8508 8509	D3 02		OUT	02	Вывод суммы в порт 02, т.е. (A) → 02
850A	76		HLT		Останов ЭВМ

Поле комментариев не учитывается ассемблером и ограничивается его перепечаткой. Это поле очень важно для понимания процессов и событий в программе.

После составления программы на ассемблере мнемокоды и операнды можно ручным способом перевести на машинный язык, используя систему команд выбранного микропроцессора. Каждому байту кода команды и операнду назначить последовательно ячейки памяти. Эту операцию определения машинных кодов и назначения ячеек памяти может выполнить ассемблер. Но для достижения вышеуказанных целей на начальных этапах изучения микропроцессоров и их программирования рекомендуется эти преобразования выполнять вручную.

Полный список команд микропроцессора K580BM80 приведен в

приложении А.

Для написания сложных программ рекомендуется составлять алгоритм в виде блок-схемы, примерный вид которой показан на рисунке 1.17. Алгоритм значительно упрощает написание программы, точно указывая, какие операции и в какой последовательности надо использовать.

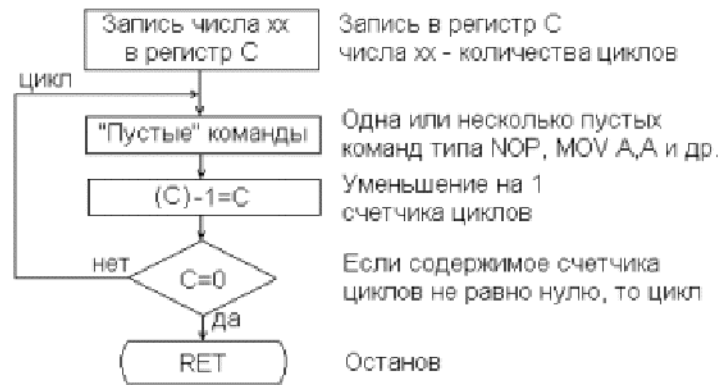


Рисунок 1.7 – Пример блок-схемы алгоритма программы

Оборудование и материалы

Микро-ЭВМ представляет собой виртуальное устройство, выполненное в виде программного обеспечения для персонального компьютера. Назначение устройства – эмуляция существующей учебной микро-ЭВМ на микропроцессорном комплекте серии K580.

В ходе выполнения практических работ по дисциплине «Основы микропроцессорной техники» может быть использован любой из двух эмуляторов на выбор студента: «Эмулятор работы микропроцессора KP580BM80» (далее ПМК – по заголовку главного окна эмулятора), либо «Программная модель стенда УМПК-80 на базе микропроцессора K580BM80» (далее УМПК – по заголовку главного окна эмулятора). Первый эмулятор находится в папке *CPU580*, второй – в папке *UMPK80*.

При входе в первый эмулятор открывается окно ПМК KP580BM80, в котором изображена структура «стенда» (рисунок 1.8).

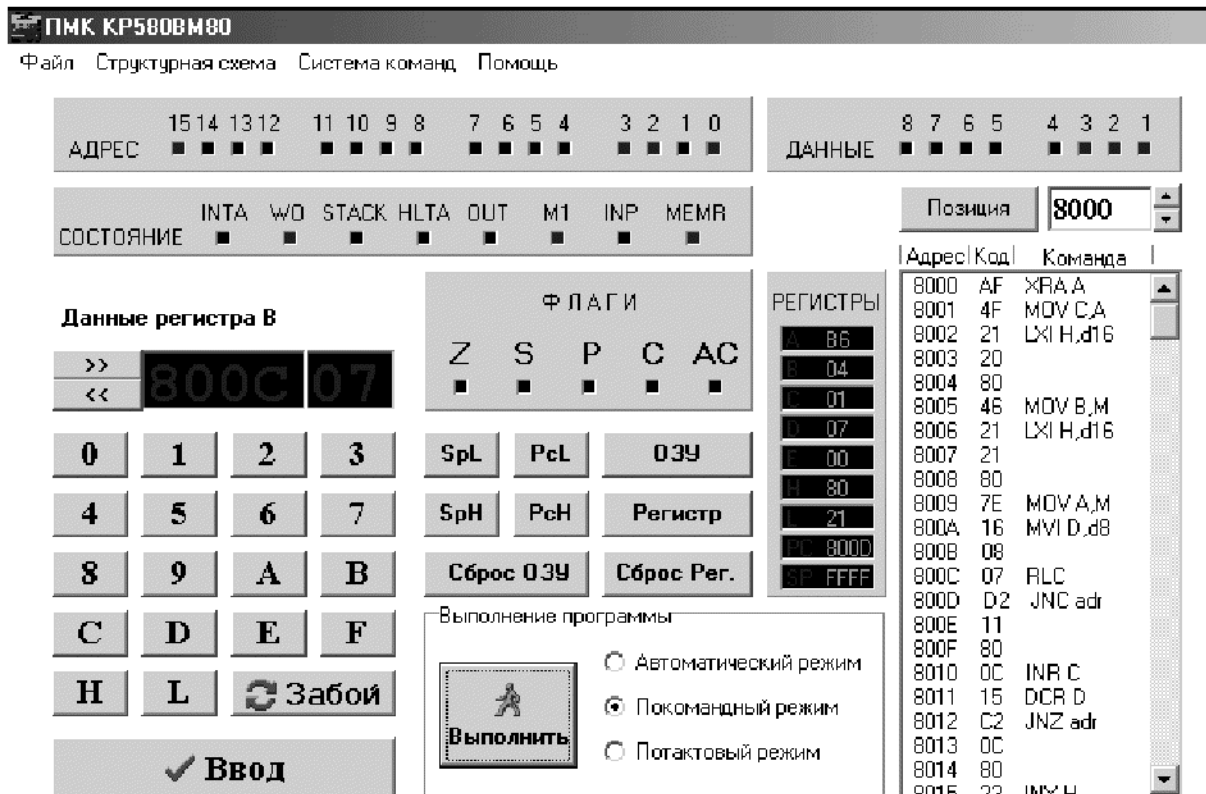


Рисунок 1.8 – Главное окно эмулятора ПМК KP580BM80

В верхней части окна указываются режимы работы: «Файл», «Структурная схема», «Система команд», «Помощь».

Строкой ниже указывается адрес (состояние шины адреса в шестнадцатеричной системе счисления) и данные (состояние шины данных в шестнадцатеричной системе счисления).

Далее показано окно, в котором указывается адрес ОЗУ и его содержимое, или номер регистра и его содержимое.

В окне флагов записывается состояние флагов *Z*, *S*, *P*, *C*, *AC* на данный момент выполнения команды процессором.

Для ввода информации используется клавиатура, которая подразделяется на цифровую: от 0 до *F* и режимную: *SpL*, *SpH*, *PcL*, *PcH*, ОЗУ, Сброс ОЗУ, Регистр, Сброс Рег.

Для ввода информации ОЗУ необходимо нажать кнопку «ОЗУ», затем кнопку «Ввод». После этого ввести в данную ячейку код команды или

данные.

Для ввода информации в регистры требуется нажать клавишу «Регистр», клавишу «Ввод», а затем на цифровой клавиатуре произвести ввод данных.

Клавиши *SpL*, *SpH* и *PcL*, *PcH* позволяют записывать адреса в указатель стека и счетчик команд.

При вводе любой информации необходимо выполнить нажатия левой кнопкой мыши на клавишу «Ввод».

В правой части окна эмулятора расположены регистры процессора:

- *A* – аккумулятор;
- *B, C, D, E, H, L* – регистры общего назначения (РОН);
- *PC* – программный счетчик;
- *SP* – указатель стека.

Указатель стека указывает вершину стека *FFFF*.

В области окна «Выполнение программы» задается режим:

- автоматический режим. При нажатии этой кнопки программа выполняется до команды *HLT*;
- покомандный режим. При нажатии этой кнопки реализуется весь цикл исполнения текущей команды;
- потактовый режим. При каждом нажатии кнопки «Выполнить» осуществляется выполнение одного такта работы процессора.

В правой части окна эмулятора расположено окно «Позиция» адресного пространства ОЗУ, в котором производится набор программы: адрес команды, код, ассемблерный код команды. «Позиция» указывает область адресного пространства ОЗУ, в котором прокруткой «» или «» можно указывать начало требуемого адресного пространства.

При указании курсором мыши позиции «Структурная схема» раскрывается следующее окно, в котором показана структурная схема микропроцессора (рисунок 1.9).

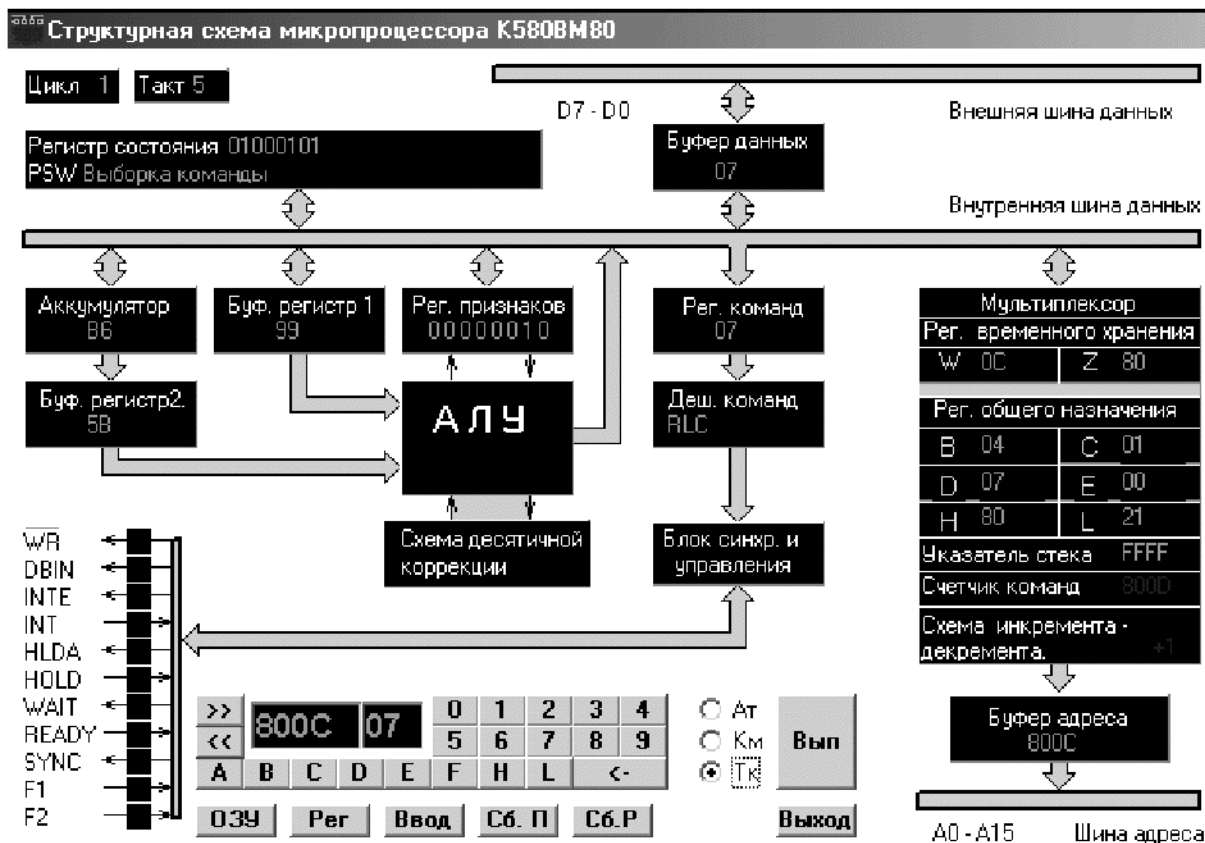


Рисунок 1.9 – Структурная схема микропроцессора K580BM80

Подробное описание структуры приведено в файле справки эмулятора.

Для организации режимов работы, как и в предыдущем окне, в данном расположена клавиатура ввода информации, аналогичная клавиатуре предыдущего окна, и соответствующие режимные клавиши, а также клавиши режима работы.

В потактовом режиме можно исследовать выполнение любой команды. При каждом нажатии кнопки «Вып.» реализуется очередной такт работы процессора. При этом фиксируется на экране все состояния его регистров, номер такта, номер цикла, состояния управляющих сигналов, регистр состояния и *PSW*.

При нажатии левой кнопки мыши в окне «ПМК KP580BM80» на позицию «Система команд», раскрывается окно «Команды» в котором представлена таблица системы команд. Код команды определяется по

матрице в шестнадцатеричной системе счисления по столбцу и строке таблицы.

В окне команд можно получить «помощь» по назначению команды и правилам ее выполнения, указав эту команду курсором мыши и щелкнув ее два раза левой кнопкой.

Полное описание правил работы с эмулятором ПМК приведено в файле справки, вызываемом в главном окне при нажатии кнопки «Помощь», а также в прилагаемой к эмулятору документации «Описание эмулятора». Файл находится в той же папке, что и файл запуска эмулятора. Важным является приложение Б файла документации, так как в нем указаны особенности поведения эмулятора, которые обязательно необходимо учитывать, во избежание некорректного выполнения кода составленной программы.

При входе во второй эмулятор открывается окно эмулятора «Модель лабораторного стенда УМПК-80», в котором изображена структура «стенда» (рисунок 1.10). Основные элементы главного окна показаны на рисунке 1.11.

В состав окна входит:

- 1 – Меню модели;
- 2 – Панель инструментов;
- 3 – Секция машинных кодов;
- 4 – Секция мнемокодов;
- 5 – Секция сообщений;
- 6 – Строка состояния.

Поскольку эмулятор имеет достаточно подробный файл справки по функциям, управлению, назначению кнопок управления и т. д., то в данном методическом указании эта информация не приводится. Файл справки можно вызвать нажатием правой кнопки мыши на кнопке «Помощь» Меню модели. Так же файл находится в той же папке, что и файл запуска эмулятора.

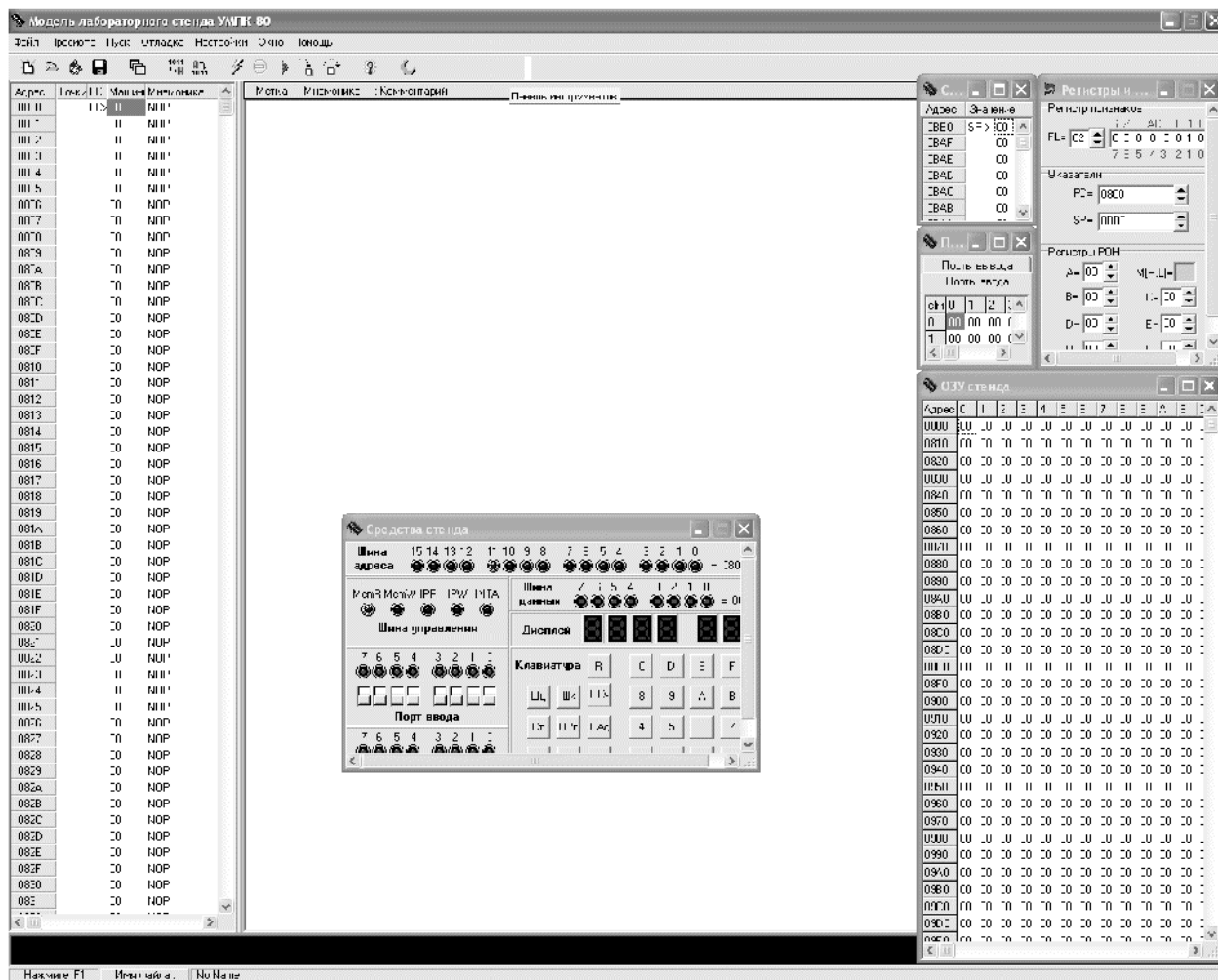


Рисунок 1.10 – Главное окно эмулятора «Модель лабораторного стенда УМПК-80»

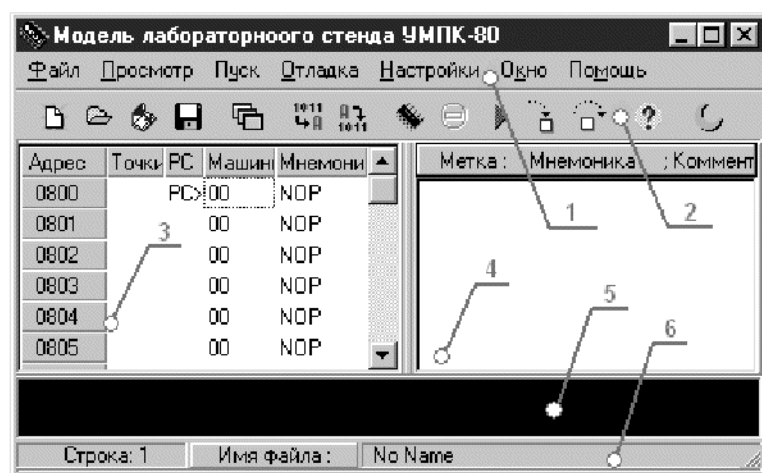


Рисунок 1.11 – Основные элементы главного окна эмулятора УМПК

В обоих эмуляторах микро-ЭВМ реализована адресация с помощью карты памяти: каждому физическому устройству макета ставится в соответствие адрес, и обращение к нему при таком типе адресации аналогично обращению к ячейке памяти с использованием всего набора команд МП БИС. На рисунке 1.12 приведена карта памяти микро-ЭВМ.

Биты в старшем байте адреса			Адреса		
15 14	13 12 11	10 9 8			
0 0	0 0 0	0 0 0	0000	Управляющая программа	2КБ
0 0	0 0 0	1 1 1	07FF	Демонстрационные программы ПЗУ	
0 0	0 0 1	0 0 0	0800	Область адресов для записи исследуемых программ	1КБ
			0AFF	Область адресов для записи данных исследуемых программ и стек	
			0BB0	Область адресов для записи данных управляющей программы ОЗУ	
			0BFF		
0 0	0 0 1	1 1 1	0FFF		
0 0	0 1 0	0 0 0	1000	Управление защитой первых 314 адресов ОЗУ от записи УПР	
0 0	0 1 0	1 1 1	17FF		
0 0	0 1 1	0 0 0	1800	Входной регистр чтения клавиатуры РгЧК	
0 0	0 1 1	1 1 1	1FFF		
0 0	1 0 0	0 0 0	2000	Входной регистр РгВВ	
0 0	1 0 0	1 1 1	27FF		
0 0	1 0 1	0 0 0	2800	Регистр сканирования дисплея и клавиатуры	
0 0	1 0 1	1 1 1	2FFF	РгСК	
0 0	1 1 0	0 0 0	3000	Выходной регистр РгВыв	
0 0	1 1 0	1 1 1	37FF		
0 0	1 1 1	0 0 0	3800	Регистр сегментов дисплея РгСг	
0 0	1 1 1	1 1 1	3FFF		
1 0	0 0 0	0 0 0	8000	Регистр звукового выхода СхВЗС	
1 0	0 0 0	0 0 1	8001		
1 1	1 1 1	1 1 1	FFFF	Неиспользуемые адреса	

Рисунок 1.12 – Карта памяти эмулируемой микро-ЭВМ

На рисунке 1.13 приведена схема возможных переходов от одного режима работы микро-ЭВМ к другому.

В режиме ожидания ввода команд управления микро-ЭВМ может реагировать на нажатие кнопок: отыскание адреса, отыскание регистра и программный счетчик. Все последующие возможные переходы от режима к режиму можно проследить по рисунку, где сплошными линиями показаны возможные пути изменения режимов работы по командам пользователя, вводимым с клавиатуры управления, а штриховыми – изменения в состояниях микро-ЭВМ, вызванные выполнением соответствующих подпрограмм команд управления.

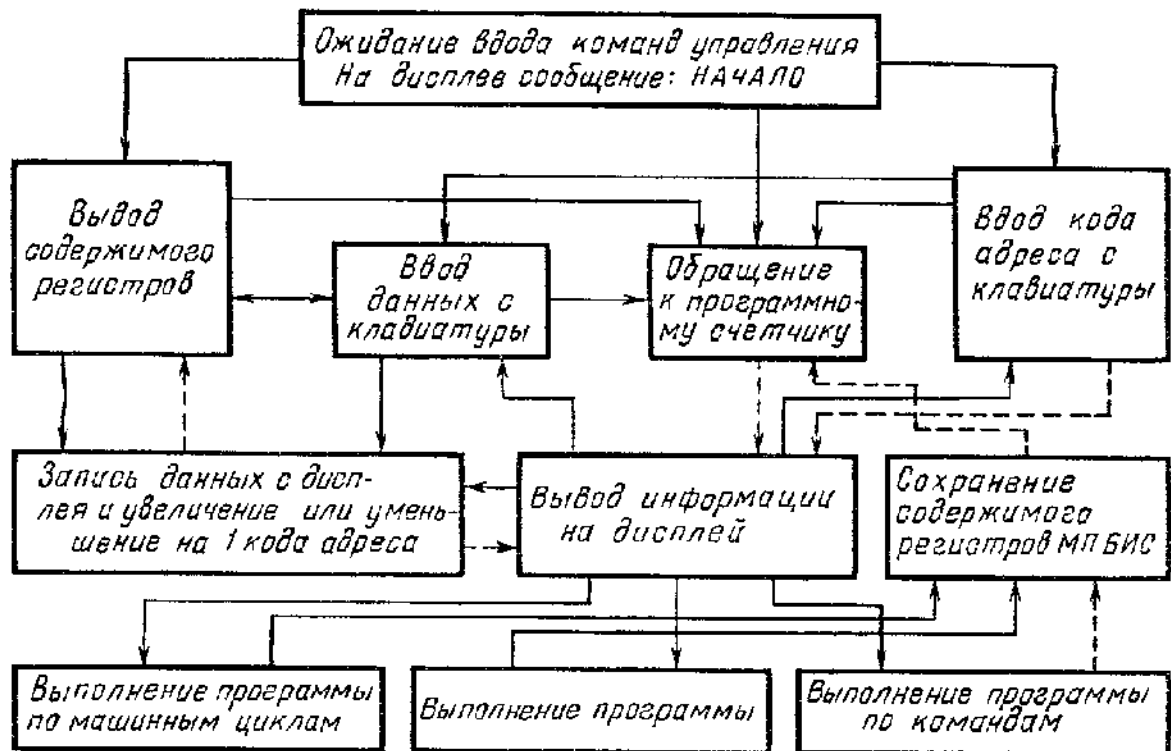


Рисунок 1.13 – Схема перехода к различным режимам работы в микро-ЭВМ

С помощью любого эмулятора (по выбору студента) необходимо выполнить задания данной и следующих практических работ.

Указания по технике безопасности

Перед проведением практического занятия следует изучить инструкцию по технике безопасности при работе на компьютере. Также следует соблюдать правила и требования, установленные для данного компьютерного класса.

Задания (задачи)

- 1 Определите способы адресации: *STA 80C2h*; *IN 58h*; *ADI 4Ah*; *OUT*

58h; *MOV C, M; LHLD 94C2h; MOV E, D; PUSH PSW.*

2 Выберите любую команду и поясните, какие действия выполняются в каждом машинном цикле при ее выполнении.

3 Определите содержимое аккумулятора:

<i>LXI H, 35E7h;</i>	<i>LXI H, 35E7h;</i>
<i>INX H;</i>	<i>SHLD 854Ch;</i>
<i>SHLD 854Ch;</i>	<i>INX H;</i>
<i>LDA 854Ch;</i>	<i>LDA 854Ch.</i>

4 Определите содержимое ячейки памяти 1000h:

MVI A, 00h;
MOV B, A;
STA 1000h.

5 Определите содержимое указателя стека *SP* и ячейки *B06Ah*:

LXI SP B069h;
MVI A, 45h;
PUSH PSW.

6 Поясните на примерах выполнение команд *ADD* и *ADC*, *SUB* и *SBB* и какие признаки формируются в регистре *F*?

7 Поясните на примере команду *DAD*. Обратите внимание на признак переноса *CY*.

8 Перечислите команды, позволяющие сравнивать двоичные коды. Приведите примеры выполнения.

9 Поясните отличие команд: *INR H*; *DCR D*; *INX H*; *DCX D* на примерах.

10 Занесите в аккумулятор десятичное число, заданное преподавателем, в двоично-десятичном коде и определите содержимое аккумулятора и признаки результата после выполнения каждой из команд: *ORI 0Fh*; *ANI F0h*; *XRA A*.

11 Содержимое какого устройства изменяют команды *RLC*, *RRC*, *RAL* и

RAR? Поясните выполняемые ими операции на примерах.

12 Поясните, что общего и различного при выполнении команд: *JMP 000h*; *CALL 000h*; *RST 0*. Приведите примеры применения.

Выбираемые для заданий примеры и числа должны различаться у каждого студента.

Содержание отчета

Отчет по практической работе выполняется на листах формата А4 согласно требованиям ГОСТ и ЕСКД и должен содержать следующие пункты:

- а) титульный лист с названием работы, дисциплины, кода группы, ФИО исполнителя и преподавателя;
- б) цель и содержание работы;
- в) краткое теоретическое введение;
- г) условия, ход выполнения и решения задач;
- д) машинные коды (команды на языке Ассемблер) программ;
- е) выводы по результатам выполнения практического задания.

Список литературы, рекомендованный к использованию по данной теме

Основная учебная литература

1. Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд., испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

Дополнительная учебная литература

- 1 Шевкопляс Б.С. Микропроцессорные структуры. Инженерные

решения: Справочник. - М.: Радио и связь, 1990 – 512 с.

2 Щелкунов, Н. Н. Микропроцессорные средства и системы / Н. Н. Щелкунов, А. П. Дианов. – М. : Радио и связь, 1989. – 288 с.

3 Коффрон, Дж. Технические средства микропроцессорных систем: практический курс / Дж. Коффрон. Пер. с англ. – М. : Мир, 1983. – 344 с.

4 Шахнов, В. А. Микропроцессоры и микропроцессорные комплекты интегральных микросхем : справочник / В.-Б. Б. Абрайтис, Н. Н. Аверьянов, А. И. Белоус [и др.]: под ред. В. А. Шахнова. – Том 1. – М. : Радио и связь, 1988. – 368 с.

5 Соппа, И. В. Микропроцессоры и микроконтроллеры / И. В. Соппа, А. С. Самардак, В. А. Иванов. – Владивосток : Издательство Дальневосточного университета, 2009. – 211 с.

6 Алексенко, А. Г. Основы микросхемотехники / А. Г. Алексенко. 3-е изд. перераб. и доп. – М. : БИНОМ. Лаборатория знаний, 2009. – 448 с.

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ 3

Тема: Изучение методов обработки массивов данных.

Цель работы: закрепление знаний, полученных на лекциях по разделу: «Архитектура микропроцессора».

Знания, приобретаемые студентом в результате освоения темы:

- аппаратная и логическая структуры типового микропроцессора;
- система команд.

Умения, приобретаемые студентом в результате освоения темы:

- разрабатывать алгоритмы работы и программное обеспечение для аппаратной части микропроцессорных систем.

Формируемые компетенции: ОК-6, ОК-7, ОПК-6, ОПК-7, ОПК-8, ОПК-9, ПК-1, ПК-3.

Актуальность темы: тема является актуальной, так как многие современные электронные устройства выполняют сбор и обработку данных с различных объектов регистрации данных и управления. Поэтому студент должен знать основные методы обработки массивов данных, уметь составлять блок-схемы алгоритмов программ, а так же сами программы обработки на языке Ассемблер.

Теоретическая часть

Достаточно часто в современной электронной аппаратуре используется сбор данных с внешних, реже внутренних устройств. Примером может служить система контроля состояния датчиков.

В простейшем случае необходима только регистрация состояний, а значит, просто сохранение полученных данных. Данные представляются в виде, удобном для обработки, например, в виде нескольких бит, одного байта, слова и т. д. Проще всего выполнять хранение в виде упорядоченных определенным образом элементов информации. Упорядочивание часто

производится по времени поступления данных.

В более сложных системах производится обработка принятых данных и с целью принятия каких-либо решений. При этом над элементами данных могут производиться различные операции:

- выявление состояния отдельных бит;
- изменение элементов с использованием арифметических или логических операций;
- перемещение (упорядочивание) элементов;
- поиска минимальных и/или максимальных значений и т. д.

Поэтому важным является понимание способов упорядочивания элементов информации и методов их обработки. Элементы данных имеют определенное название – массив.

Массив – упорядоченный набор данных, для хранения данных одного типа, идентифицируемых с помощью одного или нескольких индексов (адресов, например, при хранении данных в оперативных запоминающих устройствах). В простейшем случае массив имеет постоянную длину и хранит элементы данных одного и того же типа.

Количество используемых индексов массива может быть различным. Массивы с одним индексом называют одномерными, с двумя — двумерными и т. д. Одномерный массив нестрого соответствует вектору в математике, двумерный – матрице. Чаще всего применяются массивы с одним или двумя индексами, реже – с тремя, ещё большее количество индексов встречается крайне редко.

В языках программирования, допускающих объявления программистом собственных типов, как правило, существует возможность создания типа «массив». В определении такого типа может указываться размер, тип элемента, диапазон значений и типы индексов (диапазоны адресов для хранения). В дальнейшем возможно определение переменных созданного типа. Все такие переменные-массивы имеют одну структуру.

Выделяют типы массивов:

- статические;
- динамические;
- гетерогенные;
- массивы массивов.

Динамическим называется массив, размер которого может меняться во время исполнения программы. Для изменения размера динамического массива язык программирования, поддерживающий такие массивы, должен предоставлять встроенную функцию или оператор. Динамические массивы дают возможность более гибкой работы с данными, так как позволяют не прогнозировать хранимые объёмы данных, а регулировать размер массива в соответствии с реально необходимыми объёмами.

Обычные, не динамические массивы называют статическими.

Гетерогенным называется массив, в разные элементы которого могут быть непосредственно записаны значения, относящиеся к различным типам данных. Массив, хранящий указатели на значения различных типов, не является гетерогенным, так как собственно хранящиеся в массиве данные относятся к единственному типу – типу «указатель». Гетерогенные массивы удобны как универсальная структура для хранения наборов данных произвольных типов.

Массивы массивов – многомерные массивы, как правило, реализованные как одномерные массивы, каждый элемент которых является ссылкой на другой одномерный массив.

Таким образом, умение организовывать, обрабатывать, управлять, преобразовывать массивы данных является достаточно важным. Поэтому в данном практическом занятии будут изучены различные методы обработки массивов данных, а также их программная реализация на языке Ассемблер.

Оборудование и материалы

В ходе выполнения практической работы может быть использован любой из двух эмуляторов на выбор студента: «Эмулятор работы микропроцессора КР580ВМ80» (далее ПМК – по заголовку главного окна эмулятора), либо «Программная модель стенда УМПК-80 на базе микропроцессора К580ВМ80» (далее УМПК – по заголовку главного окна эмулятора). Первый эмулятор находится в папке *CPU580*, второй – в папке *UMPK80*.

Указания по технике безопасности

Перед проведением практического занятия следует изучить инструкцию по технике безопасности при работе на компьютере. Также следует соблюдать правила и требования, установленные для данного компьютерного класса.

Задания (задачи)

1 Выполнить перенос элементов массива, расположенного в памяти с некоторого адреса по другому адресу. Исходные данные для выполнения задания приведены в таблице 2.1.

Таблица 2.1 – Исходные данные для задачи 1

Номер варианта	Начальный адрес массива	Смещение	Элементы массива
1	0BB1	5	BB, 87, 03, 67, 71, 6A, 27, 33
2	0BB5	6	09, ED, 37, FD, 87, 01, 35

3	0BBA	7	ED, 03, 45, DE, BE, 89, 10, 90
4	0BC1	8	67, 01, DE, 34, 91, AD, ED, 76
5	0BC5	9	11, 34, 2D, AE, 85, DE, 29, 01, 67
6	0BCA	10	34, 87, 51, AE, 5C, 44, 98
7	0BD1	9	45, 94, 03, EE, F9, B4, 83
8	0BD5	8	97, 40, ED, 46, 98, 40, 07
9	0BDA	7	54, AE, DC, EB, 98, 94, 03, 32
10	0BB2	6	11, 87, 34, AE, 09, C4, 36, 20
11	0BB6	5	28, 3F, ED, 6D, AE, A6, 87, 90
12	0BBB	6	D2, 25, 87, 02, DF, 67, DF, 46, AD
13	0BC2	7	45, 89, DF, 24, 91, 34, E5, 67, DA
14	0BC6	8	13, 2F, 6E, A7, 83, 45, 10, AE
15	0BCB	9	87, 39, 02, 09, 73, ED, EF, 94, 34
16	0BD2	6	67, 45, DF, 20, 64, 30
17	0BD6	7	48, 71, 93, 28, F4, 38, 98
18	0BDB	5	EF, D4, 38, 93, B5, 2E, 9E, DD
19	0BB3	10	46, 93, 56, 02, 83, 47, 10, 88
20	0BB7	8	D3, A7, 23, F4, 65, 19, 0F, 12
21	0BBD	6	57, AF, D6, 7D, 9A, 90, 29, 02, 34
22	0BC3	9	11, 23, DE, 37, 83, CB, 93, 76
23	0BC7	5	23, 5A, 32, D4, 27, 1D, 38, 01, CF
24	0BCC	7	ED, 3D, 4D, 49, 9D, AA, F5, ED, 98
25	0BD3	10	ED, DF, 6A, D7, 98, 02, B3, 36

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется для нахождения начального адреса нового массива по формуле:

$$NewArray = EndOldArray + SS, \quad (2.1)$$

где *NewArray* – адрес начала нового массива;

EndOldArray – адрес окончания начального массива;

SS – смещение.

Рекомендации по использованию регистров микропроцессора:

– *HL* использовать для хранения адреса текущего местоположения массива;

– *DE* – для хранения адреса нового местоположения массива.

2 Выполнить предыдущую задачу 1 при условии, что сначала будут перенесены и записаны все четные элементы, затем нечетные. Выяснить, сколько четных и нечетных элементов содержится в массиве. Исходные данные для выполнения задания приведены в таблице 2.1.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется для нахождения начального адреса нового массива по формуле (2.1).

Рекомендации по использованию регистров микропроцессора:

– *B* использовать для хранения количества четных элементов;

– *C* – для хранения количества нечетных элементов;

– *HL* – для хранения адреса текущего местоположения массива;

– *DE* – для хранения адреса нового местоположения массива.

3 Найти максимальный и минимальный элементы массива. Вычислить их сумму и разность. Записать сумму и разность в память с адреса, рассчитанного по формуле (2.1), а максимальный и минимальный элементы – в регистры *B* и *C* соответственно. Исходные данные для выполнения задания приведены в таблице 2.1.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется для нахождения начального адреса нового массива по формуле (2.1).

Рекомендации по использованию регистров микропроцессора:

– *HL* использовать для хранения адреса текущего местоположения

массива;

– *DE* – для хранения адреса нового местоположения массива.

Содержание отчета

Отчет по практической работе выполняется на листах формата А4 согласно требованиям ГОСТ и ЕСКД и должен содержать следующие пункты:

- а) титульный лист с названием работы, дисциплины, кода группы, ФИО исполнителя и преподавателя;
- б) цель и содержание работы;
- в) краткое теоретическое введение;
- г) блок-схемы алгоритмов решения задач;
- д) тексты программ задач на языке Ассемблер;
- е) реализацию программ в электронном виде для возможности демонстрации их работоспособности;
- ж) выводы по результатам выполнения практического задания.

Список литературы, рекомендованный к использованию по данной теме

Основная учебная литература

1. Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд., испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

Дополнительная учебная литература

- 1 Шевкопляс Б.С. Микропроцессорные структуры. Инженерные решения: Справочник. - М.: Радио и связь, 1990 – 512 с.

2 Щелкунов, Н. Н. Микропроцессорные средства и системы / Н. Н. Щелкунов, А. П. Дианов. – М. : Радио и связь, 1989. – 288 с.

3 Коффрон, Дж. Технические средства микропроцессорных систем: практический курс / Дж. Коффрон. Пер. с англ. – М. : Мир, 1983. – 344 с.

4 Шахнов, В. А. Микропроцессоры и микропроцессорные комплекты интегральных микросхем : справочник / В.-Б. Б. Абрайтис, Н. Н. Аверьянов, А. И. Белоус [и др.]: под ред. В. А. Шахнова. – Том 1. – М. : Радио и связь, 1988. – 368 с.

5 Соппа, И. В. Микропроцессоры и микроконтроллеры / И. В. Соппа, А. С. Самардак, В. А. Иванов. – Владивосток : Издательство Дальневосточного университета, 2009. – 211 с.

6 Алексенко, А. Г. Основы микросхемотехники / А. Г. Алексенко. 3-е изд. перераб. и доп. – М. : БИНОМ. Лаборатория знаний, 2009. – 448 с.

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ 4, 5, 6

Тема: Изучение арифметических операций.

Цель работы: закрепление знаний, полученных на лекциях по разделу: «Архитектура микропроцессора».

Знания, приобретаемые студентом в результате освоения темы:

- аппаратная и логическая структуры типового микропроцессора;
- система команд.

Умения, приобретаемые студентом в результате освоения темы:

- разрабатывать алгоритмы работы и программное обеспечение для аппаратной части микропроцессорных систем.

Формируемые компетенции: ОК-6, ОК-7, ОПК-6, ОПК-7, ОПК-8, ОПК-9, ПК-1, ПК-3.

Актуальность темы: тема является актуальной, так как многие современные электронные устройства выполняют обработку данных с различных объектов с целью их хранения или принятия решений (выработки команд управления). Поэтому студент должен знать основные арифметические операции, уметь составлять блок-схемы алгоритмов программ, а так же сами программы на языке Ассемблер.

Теоретическая часть

Основное предназначение арифметических операций – выполнять определенные арифметические действия над числовыми данными: складывать, вычитать, умножать, делить и т. д. Практически вся обработка данных включает ту или иную из названных арифметических операций. Даже в повседневной жизни мы постоянно сталкиваемся с решением элементарных задач, выполнение которых связано с использованием арифметических операций. Банальным примером является расчет стоимости двух товаров.

В электронной технике для поиска данных в массивах при

использовании соответствующего способа адресации необходимо использовать операцию сложения, так как необходимый элемент располагается на заданном смещении относительно базового адреса.

Выполнение операции вычитания необходимо при расчете разницы между сигналом, действующим на объект управления, и сигналом эталонного значения. Эта разность представляет собой ошибку, которую система управления пытается устранить.

Для того, чтобы выяснить разницу в показаниях датчиков часто достаточно выполнить операцию сравнения, которая не предполагает получение численной разницы в показаниях. Достаточно лишь узнать, сигнал с какого датчика больше или меньше или они вовсе равны между собой.

И, наконец, операция деления. С ее помощью можно выяснить: во сколько раз сигнал с данного датчика отличается от сигнала с другого датчика. Это необходимо знать в устройствах, использующих датчики с множителем-делителем принципом действия.

Таким образом, умение выполнять арифметические операции является достаточно важным. Поэтому в данном практическом занятии будут изучены различные арифметические операции, алгоритмы их выполнения, а также их программная реализация на языке Ассемблер.

Оборудование и материалы

В ходе выполнения практической работы может быть использован любой из двух эмуляторов на выбор студента: «Эмулятор работы микропроцессора КР580ВМ80» (далее ПМК – по заголовку главного окна эмулятора), либо «Программная модель стенда УМПК-80 на базе микропроцессора К580ВМ80» (далее УМПК – по заголовку главного окна эмулятора). Первый эмулятор находится в папке *CPU580*, второй – в папке *UMPK80*.

Указания по технике безопасности

Перед проведением практического занятия следует изучить инструкцию по технике безопасности при работе на компьютере. Также следует соблюдать правила и требования, установленные для данного компьютерного класса.

Задания (задачи)

1 Выполнить операцию арифметического сложения двух чисел. Размер каждого числа составляет 2 байта. Исходные данные для выполнения задания приведены в таблице 3.1.

Таблица 3.1 – Исходные данные для задачи 1

Номер варианта	Число <i>A</i>	Число <i>B</i>	Смещение	Вариант расположения чисел	Вариант расположения результата
1	0BC1	369A	5	1	5
2	6A82	AA18	6	2	6
3	7391	6330	7	3	5
4	D007	0386	8	4	6
5	1C85	1639	9	1	5
6	83D7	C185	10	2	6
7	7391	0562	9	3	5
8	8531	F740	8	4	6
9	84DC	01FF	7	1	6
10	5206	9812	6	2	5
11	7401	7FC3	5	3	6

12	4756	4873	6	4	5
13	840A	481B	7	1	6
14	C108	7E33	8	2	5
15	A89C	4309	9	3	6
16	6819	4961	6	4	5
17	B8A9	123F	7	1	6
18	14DF	39CA	5	2	5
19	A001	B213	10	3	6
20	148F	350D	8	4	5
21	7491	6491	6	1	5
22	4638	0194	9	2	6
23	7DF1	4CD5	5	3	5
24	4ABC	2FAD	7	4	6
25	7461	A165	10	1	6

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется сначала для нахождения начального адреса расположения данных:

$$AddData = AddEndOf Prog + SS, \quad (3.1)$$

где $AddData$ – начальный адрес данных;

$AddEndOf Prog$ – адрес окончания программы;

SS – смещение,

а потом – адреса начала записи суммы:

$$AddSum = AddLastData + SS, \quad (3.2)$$

где $AddSum$ – начальный адрес для записи суммы;

$AddLastData$ – конечный адрес данных;

SS – смещение.

Варианты расположения чисел и суммы представлены в таблице 3.2. В

таблице приняты следующие обозначения:

- «мл. A » – младший байт числа A ;
- «ст. A » – старший байт числа A ;
- «мл. B » – младший байт числа B ;
- «ст. B » – старший байт числа B ;
- «мл. S » – младший байт суммы S ;
- «ст. S » – старший байт суммы S .

Таблица 3.2 – Варианты расположения чисел и суммы

Вариант расположения	Адрес «0»	Адрес «1»	Адрес «2»	Адрес «3»
1	«мл. A »	«мл. B »	«ст. A »	«ст. B »
2	«мл. A »	«ст. A »	«мл. B »	«ст. B »
3	«ст. A »	«ст. B »	«мл. A »	«мл. B »
4	«ст. A »	«мл. A »	«ст. B »	«мл. B »
5	«ст. S »	«мл. S »	–	–
6	«мл. S »	«ст. S »	–	–

Если одно из слагаемых или оба равны нулю, программа должна выполнить следующие действия:

- записать в регистр C число «01h», если первое слагаемое равно нулю. Алгоритм вычисления суммы при этом не должен выполняться, но значение суммы должно быть записано по заданному адресу;
- записать в регистр C число «02h», если второе слагаемое равно нулю. Алгоритм вычисления суммы при этом не должен выполняться, но значение суммы должно быть записано по заданному адресу;
- записать в регистр C число «03h», если оба слагаемых равны нулю. Алгоритм вычисления суммы при этом не должен выполняться, но значение суммы должно быть записано по заданному адресу.

В остальных случаях, в регистр C должно быть записано число « FFh ».

Рекомендации по использованию регистров микропроцессора:

- B использовать для хранения количества байтов слагаемых;
- C – для хранения признака значений слагаемых (см. выше);
- HL – для хранения адреса числа A ;
- DE – для хранения адреса числа B .

2 Выполнить операцию арифметического вычитания двух чисел. Размер каждого числа составляет 2 байта. Исходные данные для выполнения задания приведены в таблице 3.1.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется сначала для нахождения начального адреса расположения данных по формуле (3.1), а потом – адреса начала записи разности по формуле (3.2).

Варианты расположения чисел представлены в таблице 3.2, а разности чисел – в таблице 3.3. В таблице приняты следующие обозначения:

- « $Sign$ » – знак результата операции вычитания;
- «мл. D » – младший байт разности D ;
- «ст. D » – старший байт разности D .

Таблица 3.3 – Варианты расположения разности

Вариант расположения	Адрес «0»	Адрес «1»	Адрес «2»
5	« $Sign$ »	«ст. D »	«мл. D »
6	«мл. D »	«ст. D »	« $Sign$ »

Если одно из чисел или оба равны нулю, программа должна выполнить действия, полностью соответствующие пункту задания 1. В остальных случаях, в регистр C должно быть записано число « FFh ».

Рекомендации по использованию регистров микропроцессора:

- B использовать для хранения количества байтов чисел;
- C – для хранения признака значений чисел (см. выше);
- HL – для хранения адреса числа A ;
- DE – для хранения адреса числа B .

3 Выполнить операцию арифметического умножения двух чисел. Размер каждого числа составляет 2 байта. Исходные данные для выполнения задания приведены в таблице 3.1.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется сначала для нахождения начального адреса расположения данных по формуле (3.1), а потом – адреса начала записи произведения по формуле (3.2).

Варианты расположения чисел представлены в таблице 3.2, а произведения чисел – в таблице 3.4. В таблице приняты следующие обозначения:

- «мл. P » – младший байт произведения P ;
- « i . P » – i -байт произведения P ;
- «ст. P » – старший байт произведения P .

Таблица 3.4 – Варианты расположения произведения

Вариант расположения	Адрес «0»	Адрес «1»	Адрес «2»	Адрес «3»
5	«ст. P »	« $i-1$. P »	« $i-2$. P »	«мл. P »
6	«мл. P »	« $i+1$. P »	« $i+2$. P »	«ст. P »

Если одно из чисел или оба равны нулю, программа должна выполнить действия, полностью соответствующие пункту задания 1. В остальных случаях, в регистр C должно быть записано число « FFh ».

Рекомендации по использованию регистров микропроцессора:

- B использовать для хранения количества байтов чисел;

- C – для хранения признака значений чисел (см. выше);
- HL – для хранения адреса числа A ;
- DE – для хранения адреса числа B .

4 Выполнить операцию арифметического деления двух чисел. Размер каждого числа составляет 2 байта. Исходные данные для выполнения задания приведены в таблице 3.1.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется сначала для нахождения начального адреса расположения данных по формуле (3.1), а потом – адреса начала записи частного и остатка по формуле (3.2).

Варианты расположения чисел представлены в таблице 3.2, а частного и остатка – в таблице 3.5. В таблице приняты следующие обозначения:

- «мл. R » – младший байт частного R ;
- «ст. R » – старший байт частного R ;
- «мл. E » – младший байт остатка E ;
- «ст. E » – старший байт остатка E .

Таблица 3.5 – Варианты расположения частного и остатка

Вариант расположения	Адрес «0»	Адрес «1»	Адрес «2»	Адрес «3»
5	«ст. R »	«мл. R »	«ст. E »	«мл. E »
6	«мл. E »	«ст. E »	«мл. R »	«ст. R »

Если одно из чисел или оба равны нулю, программа должна выполнить следующие действия:

- записать в регистр C число «10h», если делимое равно нулю. Алгоритм деления при этом не должен выполняться, но результат деления должен быть записан по требуемому адресу;
- записать в регистр C число «20h», если делитель равен нулю.

Алгоритм деления при этом не должен выполняться, по указанному адресу ничего не должно быть записано;

– записать в регистр C число «30h», если делимое и делитель равны нулю. Алгоритм деления при этом не должен выполняться, по указанному адресу ничего не должно быть записано.

В остальных случаях, в регистр C должно быть записано число «FFh».

Рекомендации по использованию регистров микропроцессора:

- B использовать для хранения количества байтов чисел;
- C – для хранения признака значений чисел (см. выше);
- HL – для хранения адреса числа A ;
- DE – для хранения адреса числа B .

5 Выполнить арифметическую операцию сравнения двух чисел. Размер каждого числа составляет 2 байта. Исходные данные для выполнения задания приведены в таблице 3.1.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется сначала для нахождения начального адреса расположения данных по формуле (3.1), а потом – адреса записи результата сравнения по формуле (3.2).

Варианты расположения чисел представлены в таблице 3.2, а результата сравнения – в таблице 3.6. В таблице приняты следующие обозначения:

- «мл. Big » – младший байт большего числа Big ;
- «ст. Big » – старший байт большего числа Big ;
- «FF» – сравниваемые числа равны.

Вариант расположения «7» одинаков для всех заданий и используется в том случае, если сравниваемые числа равны. По адресу «1» и адресу «0» в этом случае необходимо записать числа «00h». Операцию сравнения необходимо выполнить без вычисления разницы между сравниваемыми числами.

Таблица 3.6 – Варианты расположения результата сравнения

Вариант расположения	Адрес «0»	Адрес «1»	Адрес «2»
5	«мл. <i>Big</i> »	«ст. <i>Big</i> »	«00»
6	«ст. <i>Big</i> »	«мл. <i>Big</i> »	«00»
7	«00»	«00»	«FF»

Если оба числа равны нулю, программа должна записать в регистр *C* число «00h». В остальных случаях, в регистр *C* должно быть записано число «FFh».

Рекомендации по использованию регистров микропроцессора:

- *B* использовать для хранения количества байтов чисел;
- *C* – для хранения признака значений чисел (см. выше);
- *HL* – для хранения адреса числа *A*;
- *DE* – для хранения адреса числа *B*.

Содержание отчета

Отчет по практической работе выполняется на листах формата А4 согласно требованиям ГОСТ и ЕСКД и должен содержать следующие пункты:

- а) титульный лист с названием работы, дисциплины, кода группы, ФИО исполнителя и преподавателя;
- б) цель и содержание работы;
- в) краткое теоретическое введение;
- г) блок-схемы алгоритмов решения задач;
- д) тексты программ задач на языке Ассемблер;
- е) реализацию программ в электронном виде для возможности демонстрации их работоспособности;

ж) выводы по результатам выполнения практического задания.

Список литературы, рекомендованный к использованию по данной теме

Основная учебная литература

1. Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд., испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

Дополнительная учебная литература

1 Шевкопляс Б.С. Микропроцессорные структуры. Инженерные решения: Справочник. - М.: Радио и связь, 1990 – 512 с.

2 Щелкунов, Н. Н. Микропроцессорные средства и системы / Н. Н. Щелкунов, А. П. Дианов. – М. : Радио и связь, 1989. – 288 с.

3 Коффрон, Дж. Технические средства микропроцессорных систем: практический курс / Дж. Коффрон. Пер. с англ. – М. : Мир, 1983. – 344 с.

4 Шахнов, В. А. Микропроцессоры и микропроцессорные комплекты интегральных микросхем : справочник / В.-Б. Б. Абрайтис, Н. Н. Аверьянов, А. И. Белоус [и др.]: под ред. В. А. Шахнова. – Том 1. – М. : Радио и связь, 1988. – 368 с.

5 Соппа, И. В. Микропроцессоры и микроконтроллеры / И. В. Соппа, А. С. Самардак, В. А. Иванов. – Владивосток : Издательство Дальневосточного университета, 2009. – 211 с.

6 Алексенко, А. Г. Основы микросхемотехники / А. Г. Алексенко. 3-е изд. перераб. и доп. – М. : БИНОМ. Лаборатория знаний, 2009. – 448 с.

ПРАКТИЧЕСКАЯ РАБОТА 7, 8

Тема: Изучение методов работы с разрядами и операциями сдвига.

Цель работы: закрепление знаний, полученных на лекциях по разделу: «Архитектура микропроцессора».

Знания, приобретаемые студентом в результате освоения темы:

- аппаратная и логическая структуры типового микропроцессора;
- система команд.

Умения, приобретаемые студентом в результате освоения темы:

- разрабатывать алгоритмы работы и программное обеспечение для аппаратной части микропроцессорных систем.

Формируемые компетенции: ОК-6, ОК-7, ОПК-6, ОПК-7, ОПК-8, ОПК-9, ПК-1, ПК-3.

Актуальность темы: тема является актуальной, так как многие современные электронные устройства используют механизмы прерываний, работы с флагами, управления отдельными битами и т. д. Поэтому студент должен знать основные методы работы с разрядами и операции сдвига, уметь составлять блок-схемы алгоритмов программ, а так же сами программы на языке Ассемблер.

Теоретическая часть

Внутреннее представление данных в компьютере состоит в том, что все они строятся на основе машинных слов (или байтов). Пока мы используем определенные в языке операции, присущие этим типам данных (например, операцию сложения целых переменных или косвенное обращение по указателю), внутреннее представление данных нам безразлично. Но иногда нам приходится «лезть внутрь» и работать с ними на уровне отдельных двоичных разрядов, т. е. с битами. Например:

- состояние датчика включает несколько различных флагов. Каждый

флаг представляет собой бит с заданным номером, расположенным в байте словосостояния. Чтобы узнать состояние бита необходимо обратиться именно к нему, а не ко всему слову в целом;

- если датчиков в системе несколько и они обслуживаются удаленным контроллером, то последний, чтобы не передавать состояние каждого датчика в управляющую ЭВМ отдельно, может «запаковать» биты состояния в байт, что позволит сократить количество операций пересылки. Это называется упаковкой данных или архивированием;

- по результатам проверки разрядов могут быть приняты различные решения: запуск подпрограмм, вызов функций, выполнение операций и т. д.;

- для контроля свободного пространства на диске файловая система часто использует карту памяти – массив байтов, каждый разряд которого определяется состояние отдельного блока: «0» – свободен, «1» – занят и т. д.

Таких примеров можно привести много. Программист может установить, очистить, получить обратный код (дополнение к единице) или проверить разряды, используя логические операции с соответствующими масками. Команды сдвига и получения обратного кода могут оперировать только с аккумулятором, но в то же время для выполнения небольшого числа сдвигов могут использоваться арифметические и логические команды.

Возможны следующие операции с отдельными разрядами аккумулятора:

- установить с помощью операции логическое ИЛИ с единицами в соответствующих позициях;

- очистить с помощью операции логическое И с нулями в соответствующих позициях;

- инвертировать (изменить на обратное значение) с помощью операции ИСКЛЮЧАЮЩЕЕ ИЛИ с единицами в соответствующих позициях;

- проверить (на все нули в проверяемых разрядах) с помощью операции логическое И с единицами в соответствующих позициях.

Единственный способ работы с разрядами в других регистрах или в памяти состоит в пересылке значения в аккумулятор.

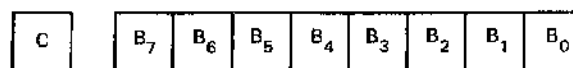
Иногда рационально работать с разрядом «0» какого-либо регистра или ячейки памяти с использованием команды *INR* или *DCR*. Любая из этих команд изменяет значение разряда «0» на противоположное; т. е. как *INR*, так и *DCR* устанавливают разряд «0», если он был равен «0», или очищают его, если он был равен «1». Этот рациональный способ полезен в том случае, когда заданный регистр или ячейка памяти содержит только один одноразрядный флаг.

К операциям работы с разрядами относят также операции сдвига. Команды *RAL* и *RAR* циклически сдвигают аккумулятор вместе с флагом переноса так, как если бы они составляли девятиразрядный регистр. На рисунках 4.1 и 4.2 показано действие команд *RAL* и *RAR*. Как видно из рисунков 4.3 и 4.4, команды *RLC* и *RRC* циклически сдвигают один аккумулятор; разряд, выдвигаемый из конца аккумулятора, появляется как во флаге переноса, так и в разряде с другого конца аккумулятора. Сложение аккумулятора с самим собой, как показано далее, заменяет другие команды сдвига:

– *ADD A* сдвигает логически аккумулятор на одну позицию влево, как показано на рисунке 4.5; при этом разряд «0» очищается;

– *ADC A* циклически сдвигает аккумулятор и флаг переноса влево на одну позицию; *ADC A* отличается от *RAL* только тем, что воздействует на все флаги, в то время как *RAL* изменяет только флаг переноса.

Исходное содержимое флага переноса и аккумулятора



После *RAL* (циклически сдвигать аккумулятор влево через перенос)

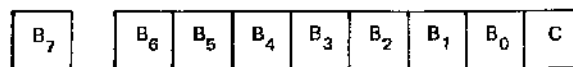
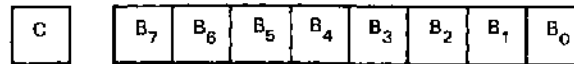


Рисунок 4.1 – Команда *RAL* (циклически сдвигать аккумулятор влево через перенос)

Исходное содержимое флага переноса и аккумулятора



После RAR (циклически сдвигать аккумулятор вправо через перенос)

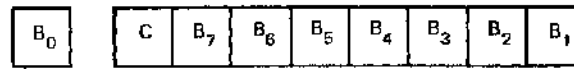
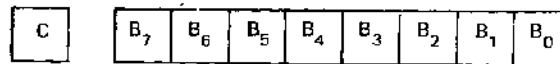


Рисунок 4.2 – Команда *RAR* (циклически сдвигать аккумулятор вправо через перенос)

Исходное содержимое флага переноса и аккумулятора
Флаг переноса Данные



После RLC (циклически сдвигать аккумулятор влево)

Флаг переноса

Данные

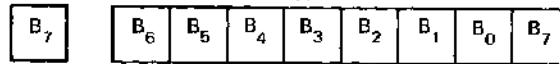
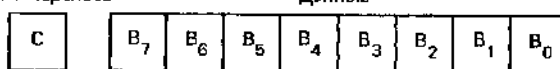


Рисунок 4.3 – Команда *RLC* (циклически сдвигать аккумулятор влево)

Исходное содержимое флага переноса и аккумулятора
Флаг переноса Данные



После RRC (циклически сдвигать аккумулятор вправо)

Флаг переноса

Данные

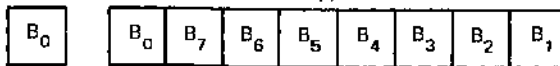
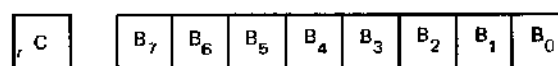


Рисунок 4.4 – Команда *RRC* (циклически сдвигает аккумулятор вправо)

Исходное содержимое флага переноса и аккумулятора



После ADD A (сложить аккумулятор с самим собой)

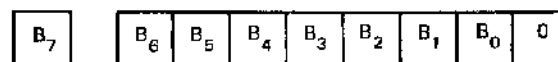


Рисунок 4.5 – Команда *ADD A* (сложить аккумулятор с самим собой)

Необходимо отметить, что команды *RAL*, *ADC A* и *RAR* сохраняют старое значение флага переноса (или в разряде «0»; или в разряде «7»), в то время как *RLC*, *RRC* и *ADD A* уничтожают его.

Арифметический или логический сдвиг можно выполнить, используя

флаг переноса. Команда *STC* устанавливает флаг переноса, в то время как *ANA A* или *ORA A* очищает его без изменения аккумулятора. Регистры или ячейки памяти могут быть сдвинуты с помощью пересылки их содержимого в аккумулятор. Единственный короткий путь выполнения 16-разрядного логического сдвига влево пары регистров *H* – это команда *DAD H*.

Также можно сдвинуть байт данных вправо на одну позицию, сохранив знаковый разряд (разряд «7»). Перемещение, в результате которого сохраняется знаковый разряд, называется расширением знака. Сдвиг, который осуществляется подобным образом, называется арифметическим, так как он сохраняет знак дополнительного кода числа. Поэтому его можно использовать для лечения или нормализации чисел со знаками.

Таким образом, умение выполнять операции с разрядами, включая сдвиг, является достаточно важным. Поэтому в данном практическом занятии будут изучены различные операции указанного типа, алгоритмы их выполнения, а также их программная реализация на языке Ассемблер.

Оборудование и материалы

В ходе выполнения практической работы может быть использован любой из двух эмуляторов на выбор студента: «Эмулятор работы микропроцессора KP580BM80» (далее ПМК – по заголовку главного окна эмулятора), либо «Программная модель стенда УМПК-80 на базе микропроцессора K580BM80» (далее УМПК – по заголовку главного окна эмулятора). Первый эмулятор находится в папке *CPU580*, второй – в папке *UMPK80*.

Указания по технике безопасности

Перед проведением практического занятия следует изучить инструкцию по технике безопасности при работе на компьютере. Также следует соблюдать правила и требования, установленные для данного компьютерного класса.

Задания (задачи)

1 Выполнить установку разрядов. В результате выполнения задания указанный разряд байта устанавливается в «1». Условия выполнения задания:

- а) данные записываются с входного устройства с адресом $05h$;
- б) исходное число помещается в регистр H , результат – в L ;
- в) программа должна выполняться в цикле (постоянно);
- г) данные для обработки студент выбирает самостоятельно;
- д) биты установки записываются в память при написании программы.

Исходные данные для выполнения задания приведены в таблице 4.1.

Таблица 4.1 – Исходные данные для задачи 1

Номер варианта	1	2	3	4	5	6	7	8	9	10	11	12	13
Номера бит	0, 1	1, 2	2, 3	3, 4	4, 5	5, 6	6, 7	0, 2	3, 4	4, 6	1, 3	3, 5	5, 7
Смещение	5	6	7	8	9	8	7	6	5	10	11	12	13

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется для нахождения адреса расположения маски:

$$AddMask = AddEndOf Prog + SS, \quad (4.1)$$

где $AddMask$ – адрес маски;

$AddEndOfProg$ – адрес окончания программы;

SS – смещение.

Продолжение таблицы 4.1

Номер варианта	14	15	16	17	18	19	20	21	22	23	24	25
Номера бит	1, 4	2, 5	3, 6	4, 7	0, 3	1, 4	2, 5	0, 4	1, 5	2, 6	3, 7	1, 6
Смещение	10	9	8	7	14	15	17	6	5	10	11	12

2 Выполнить очистку разрядов. В результате выполнения задания указанный разряд байта устанавливается в «0» (очищается). Условия выполнения задания принять такими же, как в задаче п. 1.

Исходные данные для выполнения задания приведены в таблице 4.1.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется для нахождения адреса расположения маски по формуле (4.1).

3 Выполнить проверку разрядов. В результате выполнения задания пользователю отображается состояние указанных разрядов. Условия выполнения задания принять такими же, как в задаче п. 1, за исключением способа размещения:

- исходное число помещается в регистр H ;
- результат записывается в выходное устройство с адресом $05h$ (светодиоды).

Способ индикации:

- сначала записываются логические единицы во все разряды кроме тех, которые подлежат проверке. В эти разряды записываются логические нули;
- затем все разряды, кроме проверяемых, устанавливаются в «0» и

одновременно выдаются значения проверяемых разрядов;

- переключение режимов осуществляется с частотой, удобной для наблюдения оператору.

Исходные данные для выполнения задания приведены в таблице 4.1.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется для нахождения адреса расположения маски по формуле (4.1).

4 Выполнить выделение поля разрядов. В результате выполнения задания из байта выделяется поле разрядов, и это поле возвращается в младших по значению разрядах. Условия выполнения задания принять такими же, как в задаче п. 1, кроме (д). Дополнительные условия задачи:

- запрос поля, которое выходит за конец байта, вызывает выход программы с полем разрядов вплоть до седьмого. Таким образом, не происходит циклического перехода на начало байта. Если, например, пользователь запрашивает шестиразрядное поле, начинающееся с пятого разряда, то программа возвращает только три разряда (с пятого по седьмой);

- как позиция младшего разряда, так и число разрядов в поле интерпретируются по модулю «8». Таким образом, например, позиция разряда 11 эквивалентна позиции третьего разряда, а поле из 10 разрядов эквивалентно полю из двух разрядов. Число разрядов в поле интерпретируется в диапазоне от одного до восьми, т. е. поле из 16 разрядов эквивалентно полю из восьми разрядов, а не полю из нуля разрядов;

- запрос поля с числом разрядов ноль вызывает выход с результатом, равным нулю.

Исходные данные для выполнения задания приведены в таблице 4.2.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Параметры заданы в десятичном виде и должны записываться самой программой в начале выполнения и храниться:

- ширина поля – в регистре D ;

– поз. МР (позиция младшего разряда) – в регистре *E*.

Таблица 4.2 – Исходные данные для задачи 4

Номер варианта	1	2	3	4	5	6	7	8	9	10	11	12	13
Ширина поля	2	2	2	2	3	3	3	3	3	4	4	4	4
Поз. МР	2	3	4	5	2	3	4	4	6	0	1	2	3

Продолжение таблицы 4.2

Номер варианта	14	15	16	17	18	19	20	21	22	23	24	25
Ширина поля	5	5	5	5	3	4	2	2	3	4	4	5
Поз. МР	0	1	2	3	1	4	1	6	7	4	5	4

5 Выполнить запись поля разрядов. В результате выполнения задания поле разрядов вставляется в байт. Условия выполнения задания принять такими же, как в задаче п. 1, кроме (д). Дополнительные условия задачи:

– при попытке записать поле, которое могло бы выйти за конец байта, записываются только разряды вплоть до седьмого. Таким образом, не происходит циклического перехода на начало байта. Если, например, пользователь пытается записать шестиразрядное поле, начиная с четвертого разряда, то в действительности будут замещены только четыре разряда (разряды с четвертого по седьмой);

– как позиция младшего разряда, так и число разрядов в поле интерпретируются подмодулю «8». Таким образом, например, позиция разряда 11 эквивалентна позиции третьего разряда, а двенадцатиразрядное поле эквивалентно четырехразрядному. Число разрядов в поле

рассматривается в диапазоне от одного до восьми, т. е., например, шестнадцатиразрядное поле эквивалентно восьмиразрядному;

– попытка записать поле шириной ноль разрядов вызывает возврат с результатом, равным нулю.

Исходные данные для выполнения задания приведены в таблице 4.3.

Таблица 4.3 – Исходные данные для задачи 5

Номер варианта	1	2	3	4	5	6	7	8	9	10	11	12	13
Ширина поля	2	2	2	2	2	2	3	3	3	3	3	3	4
Поз. МР	1	2	3	4	5	6	1	2	3	4	5	6	1
Поле разрядов	$2h$	$3h$	$1h$	$0h$	$2h$	$3h$	$5h$	$6h$	$4h$	$5h$	$7h$	$2h$	$7h$

Продолжение таблицы 4.3

Номер варианта	14	15	16	17	18	19	20	21	22	23	24	25
Ширина поля	4	4	4	5	5	5	5	5	2	3	4	4
Поз. МР	2	3	4	0	1	2	3	4	7	7	5	6
Поле разрядов	Ah	Bh	Fh	$7h$	Bh	$1Bh$	$10h$	$13h$	$1h$	$5h$	Ch	Dh

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Параметры заданы в десятичном виде (кроме поля разрядов, которое задано в шестнадцатеричном виде) и должны записываться самой программой в начале выполнения и храниться:

– ширина поля – в регистре D ;

- поз. МР (позиция младшего разряда) – в регистре E ;
- поле разрядов – в регистре C .

6 Выполнить арифметический сдвиг вправо чисел с повышенной точностью. В результате выполнения задания многобайтный операнд сдвигается арифметически вправо на заданное число разрядов. Последний разряд, сдвинутый из самой правой позиции, попадает во флаг переноса. Условия выполнения задания:

- а) операнд записывается в память до запуска программы в следующем порядке: младший байт занимает наименьший адрес;
- б) длина операнда в байтах записывается в регистр B программой после ее запуска;
- в) число сдвигов записывается в регистр C программой после запуска;
- г) результат сохраняется на месте исходных данных в том же порядке: младший байт занимает наименьший адрес;
- д) данные для обработки студент выбирает самостоятельно.

Дополнительные условия задачи:

- если длина операнда равна нулю, то осуществляется немедленный выход из программы, при этом операнд остается без изменения, а флаг переноса очищается;
- если число сдвигов равно нулю, то осуществляется немедленный выход из программы, при этом операнд остается без изменения, а флаг переноса очищается.

Исходные данные для выполнения задания приведены в таблице 4.4.

Смещение SS выбирается из следующих условий:

$$SS = \begin{cases} 4 \cdot n, & n \leq 3; \\ 2 \cdot n, & 3 < n \leq 6; \\ n, & n > 6 \end{cases} \quad (4.2)$$

где n – номер варианта.

Таблица 4.4 – Исходные данные для задачи 6

Номер варианта	1	2	3	4	5	6	7	8	9	10	11	12	13
Длина операнда, байт	3	3	4	4	5	5	6	6	7	7	8	8	7
Число сдвигов	20	19	18	17	13	12	9	10	7	8	3	4	6

Продолжение таблицы 4.4

Номер варианта	14	15	16	17	18	19	20	21	22	23	24	25
Длина операнда, байт	7	6	6	5	5	4	4	3	3	9	9	9
Число сдвигов	5	8	11	11	14	16	19	17	18	3	4	5

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется для нахождения адреса расположения младшего байта данных по формуле (4.1).

7 Выполнить логический сдвиг чисел с повышенной точностью. В результате выполнения задания многобайтный операнд сдвигается логически влево (вправо) на заданное число разрядов. Последний разряд, сдвинутый из самой левой (правой) позиции, попадает во флаг переноса. Условия выполнения задания, включая дополнительные, принять такими же, как в задаче п. 6. Разряды, от которых производится сдвиг, должны заполняться нулями.

Исходные данные для выполнения задания приведены в таблице 4.4.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Выбор направления логического сдвига:

- для нечетных номеров вариантов необходимо выполнять сдвиг влево;
- для четных – вправо.

Смещение рассчитывается по формуле (4.2) и используется для нахождения адреса расположения младшего байта данных по формуле (4.1).

8 Выполнить циклический сдвиг чисел с повышенной точностью. В результате выполнения задания многобайтный операнд сдвигается вправо (влево) на заданное число разрядов так, как если бы самый старший и самый младший по значению разряды были связаны. Последний разряд, сдвинутый из самой правой (левой) позиции, попадает во флаг переноса. Условия выполнения задания, включая дополнительные, принять такими же, как в задаче п. 6.

Исходные данные для выполнения задания приведены в таблице 4.4.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Выбор направления циклического сдвига:

- для нечетных номеров вариантов необходимо выполнять сдвиг влево;
- для четных – вправо.

Смещение рассчитывается по формуле (4.2) и используется для нахождения адреса расположения младшего байта данных по формуле (4.1).

Содержание отчета

Отчет по практической работе выполняется на листах формата А4 согласно требованиям ГОСТ и ЕСКД и должен содержать следующие пункты:

- а) титульный лист с названием работы, дисциплины, кода группы, ФИО исполнителя и преподавателя;
- б) цель и содержание работы;

- в) краткое теоретическое введение;
- г) блок-схемы алгоритмов решения задач;
- д) тексты программ задач на языке Ассемблер;
- е) реализацию программ в электронном виде для возможности демонстрации их работоспособности;
- ж) выводы по результатам выполнения практического задания.

Список литературы, рекомендованный к использованию по данной теме

Основная учебная литература

1 Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд., испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

Дополнительная учебная литература

1 Шевкопляс Б.С. Микропроцессорные структуры. Инженерные решения: Справочник. - М.: Радио и связь, 1990 – 512 с.

2 Щелкунов, Н. Н. Микропроцессорные средства и системы / Н. Н. Щелкунов, А. П. Дианов. – М. : Радио и связь, 1989. – 288 с.

3 Коффрон, Дж. Технические средства микропроцессорных систем: практический курс / Дж. Коффрон. Пер. с англ. – М. : Мир, 1983. – 344 с.

4 Шахнов, В. А. Микропроцессоры и микропроцессорные комплекты интегральных микросхем : справочник / В.-Б. Б. Абрайтис, Н. Н. Аверьянов, А. И. Белоус [и др.]: под ред. В. А. Шахнова. – Том 1. – М. : Радио и связь, 1988. – 368 с.

5 Соппа, И. В. Микропроцессоры и микроконтроллеры / И. В. Соппа, А. С. Самардак, В. А. Иванов. – Владивосток : Издательство Дальневосточного университета, 2009. – 211 с.

6 Алексенко, А. Г. Основы микросхемотехники / А. Г. Алексенко. 3-е изд. перераб. и доп. – М. : БИНОМ. Лаборатория знаний, 2009. – 448 с.

Введение

Целью дисциплины является формирование набора требуемых компетенций будущего бакалавра по направлению подготовки 11.03.04 Электроника и наноэлектроника.

Задачи дисциплины: ознакомить студентов с основами микропроцессорной техники, принципами программирования микропроцессоров, возможностью использования микропроцессоров и устройств на их основе для решения различных прикладных задач, системой команд, организацией интерфейсов связи с внешним оборудованием.

Выполнение практических работ способствует получению знаний и умений, согласно заявленным для дисциплины компетенциям, на базе теоретического материала, изучаемого на лекционных занятиях.

Содержание

Практическое занятие 1. Разработка ядра одноплатного микроконтроллера	5
Практическое занятие 2. Разработка модуля памяти для одноплатного микроконтроллера. Разработка схемы дешифратора адреса для модуля памяти	32
Практическое занятие 3. Разработка портов ввода/вывода для одноплатного микроконтроллера на основе регистров. Разработка портов ввода/вывода для одноплатного микроконтроллера на основе программируемого периферийного адаптера. Разработка схемы дешифратора адреса для портов ввода/вывода	52
Практическое занятие 4. Расчет электрического интерфейса одноплатного микроконтроллера	87

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ 1

Тема: Разработка ядра одноплатного микроконтроллера.

Цель работы: закрепление знаний, полученных на лекциях по разделу: «Проектирование микропроцессорных систем». В ходе выполнения заданий практического занятия решаются вопросы: организации включения в состав микропроцессорной системы однокристального микропроцессора, тактирования его работы, организации цепей сброса, а также внутренних магистралей микроконтроллера.

Знания, приобретаемые студентом в результате освоения темы:

- классификация микропроцессоров и их основные характеристики;
- аппаратная и логическая структуры типовых микропроцессоров;
- режимы адресации и система команд;
- способы организации ввода/вывода в микропроцессорных системах.

Умения, приобретаемые студентом в результате освоения темы:

- разрабатывать микропроцессорные системы для решения задач управления, автоматизации, сбора и обработки данных и других целей.

Формируемые компетенции: ОК-6, ОК-7, ОПК-3, ОПК-4, ОПК-6, ОПК-7, ОПК-8, ОПК-9, ПК-1, ПК-3, ПК-5, ПК-6.

Актуальность темы: тема является актуальной, так как многие современные устройства построены на основе микропроцессоров, что позволяет с их помощью решать достаточно сложные задачи обработки данных, управления, мониторинга и т. д. Поэтому студент должен знать структуру микропроцессорной системы, уметь синтезировать ядро одноплатного микроконтроллера для решения различных задач.

Теоретическая часть

Архитектура однокристального 8-разрядного n-МОП микропроцессора i8080 была рассмотрена в курсе лекций по дисциплине ОМТ. Схемотехни-

ческие и архитектурные особенности в сочетании с n-канальной технологией позволили уменьшить время выполнения команд типов регистр-регистр до 1,6 мкс, регистр-память до 2,8 мкс при тактовой частоте 2,5 МГц, что соответствует быстродействию 500 тыс. операций в секунду.

Для работы i8080 применяется три источника питания:

- минус 5 В (ток потребления менее 1 мА);
- плюс 5 В (ток потребления менее 70 мА);
- плюс 12 В (ток потребления менее 50 мА).

Кристалл МП, содержащий около 5000 транзисторов, помещен в корпус с двухрядным расположением 40 выводов и шагом 2,5 мм между ними. Синхронизация МП осуществляется от двухфазного внешнего генератора, амплитуда тактовых импульсов которого составляет 12 В.

На рисунке 1.1 приведена базовая структурная схема МС, построенной на базе МП i8080. В системе используется общая для хранения программ и данных основная память емкостью 64 К байт и подсистема ВВ емкостью 256 байт. Двухнаправленная 8-разрядная шина данных DAT, 16-разрядная шина адреса ADR и шина управления CB служат для организации побайтового обмена между отдельными модулями системы.

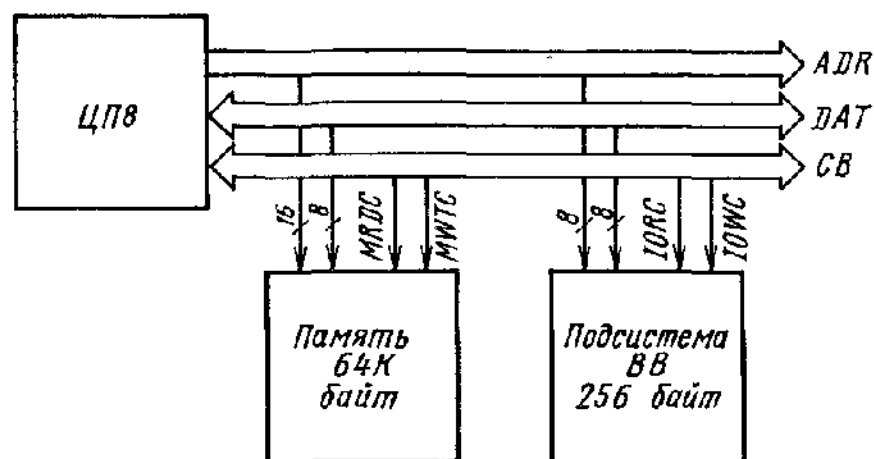


Рисунок 1.1 – Базовая структурная схема МС на основе МП i8080

Схема однокристалного 8-разрядного МП i8080 приведена на рисунке 1.2, а на рисунке 1.3 дано его условно-графическое обозначение (УГО).

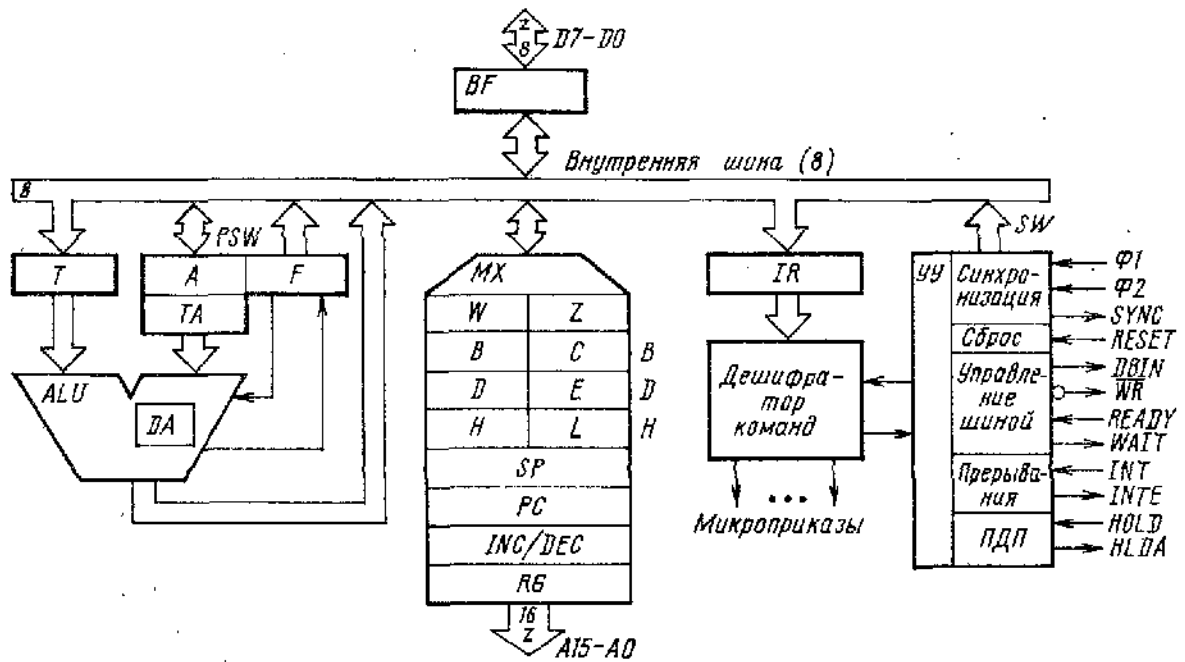


Рисунок 1.2 – Структурная схема МП i8080

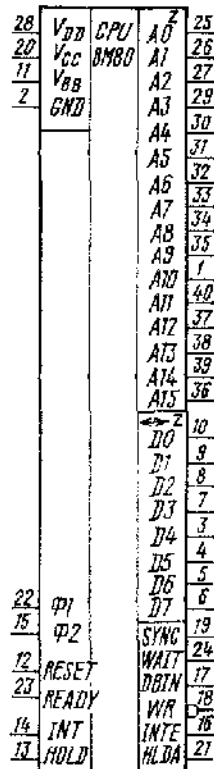


Рисунок 1.3 – УГО МП i8080

Схема содержит АЛУ, блок регистров, устройство управления и буфер данных. Обмен информацией между составными частями схемы осуществляется с помощью 8-разрядной внутренней шины. По шине передаются команды, адреса, данные, а также информация SW (Status Word) о состоянии процессора в текущем машинном цикле.

В состав интерфейса МП входят трехстабильная 16-разрядная шина адреса $A_{15}, \dots, A_0$, трехстабильная двунаправленная 8-разрядная мультиплексированная шина состояния/данных $D_7, \dots, D_0$, две линии двухфазной синхронизации Φ_1, Φ_2 и десять линий управления, из которых четыре входных, а шесть выходных. Каждая линия управления имеет свое функциональное назначение:

- RESET – начальная установка;
- SYNC – сигнал синхронизации по машинным циклам;
- DBIN – строб ввода данных;
- $\overline{WR}$ – строб вывода данных;
- READY – сигнал готовности периферийных модулей к обмену;
- WAIT – сигнал ожидания готовности;
- INTE – разрешение прерывания;
- INT – запрос векторного прерывания;
- HOLD – запрос доступа к магистрали;
- HLDA – подтверждение доступа к магистрали.

Все действия МП синхронизированы вложенными друг в друга циклами трех уровней: командными, машинными и микротактами.

Командный цикл состоит из нескольких (от 1 до 5) машинных циклов, обозначаемых $M_1, \dots, M_5$. Каждый машинный цикл представляет собой цикл обращения к системной магистрали: выборку очередного байта команды или данных из памяти, запись в память, ввод или вывод данных. Исключение составляет команда DAD (сложить HL с парой регистров). Она содержит три машинных цикла, однако только в первом цикле M_1 происходит обращение к

памяти, тогда как два других отводятся для исполнения команды без обращения к магистрали. Цикл M1 всегда первый и возможно единственный цикл, который реализует фазу выборки команды. Его называют также циклом FETCH. В начале каждого машинного цикла МП вырабатывает сигнал SYNC. Реализация машинного цикла возлагается на устройство управления (УУ).

Машинные циклы выполняются по микротактам, определяемым как интервал времени между двумя соседними фронтами импульсов фазы Φ_1 . При частоте импульсов Φ_1 , равной 2 МГц, длительность микротакта составляет 500 нс. Один машинный цикл включает от 3 до 5 микротактов $T_1, \dots, T_5$. В течение одного микротакта исполняется один микроприказ. Обычно цикл M1 длится от 4 до 5 тактов, циклы M2, ..., M5 содержат 3 такта. Сигнал SYNC выдается в первом микротакте каждого машинного цикла.

Для нормального функционирования МС недостаточно управляющих сигналов. Так, в данном наборе сигналов нельзя отличить циклы обращения к памяти от циклов ВВ. Расширение числа управляющих сигналов выполнено с помощью специального 8-разрядного слова состояния SW (Status Word), которое выдается через шину данных в первом такте T_1 каждого машинного цикла. Если сигналы управления и состояния используются в основном для управления процессом исполнения машинного цикла, то сигналы SW кодируют тип машинного цикла. Последние определяют вид информации и направление ее движения через шину данных. Согласно условию

$$\text{SYNC} * \Phi_1 = 1$$

слово состояния SW фиксируется во внешнем регистре и используется для управления МС на уровне машинных циклов.

В состав SW входят следующие сигналы:

- D0 (INTA) – подтверждение прерывания. Выделяет машинные циклы обслуживания системы прерываний;
- D1 ($\overline{WO}$) Запись/вывод. Свидетельствует о выдаче в данном цикле

данных из МП;

– D2 (STACK) – стек. Сигнал активен, если выполняется обращение к стеку, т. е. $addr = (SP)$;

– D3 (HLTA) – подтверждение останова. Информировывает о переходе МП в данный машинный цикл в состояние HALT;

– D4 (OUT) Вывод. Свидетельствует об обращении к портам вывода. Адрес порта установлен на младшей и старшей половинах шины адреса;

– D5 (M1) – цикл M1. Обозначает первый машинный цикл каждого командного цикла;

– D6 (INP) – ввод. Свидетельствует об обращении к портам ввода. Адрес порта установлен на младшей и старшей половинах шины адреса;

– D7 (MEMR) – чтение памяти. Выделяет циклы чтения данных из памяти.

Всего возможно 11 наборов сигналов SW, разбивающих множество машинных циклов на 11 типов (таблица 1.1).

Таблица 1.1

Тип	Мнемоника	Функциональное назначение
1	FETCH	Цикл M1 приема первого байта команды в регистр команд IR
2	MEMORY_READ	Цикл чтения данных из памяти по адресу, определяемому PC, BC, DE, HL или addr16
3	MEMORY_WRITE	Цикл записи данных в память по адресу, определяемому BC, DE, HL или addr16
4	STACK_READ	Цикл чтения из стека, чтение памяти по адресу, определяемому SP
5	STACK_WRITE	Цикл записи в стек, запись в память по адресу,

		определяемому SP
6	INPUT	Цикл ввода данных из порта в аккумулятор A
7	OUTPUT	Цикл вывода данных из аккумулятора в порт
8	INTERRUPT_M1	Первый цикл подтверждения прерывания
9	HALT	Цикл останова МП
10	HALT_INTERRUPT	Первый цикл подтверждения прерывания в состоянии HALT
11	INTERRUPT	Второй и третий циклы подтверждения прерывания

Описание содержания типовых циклов обращения к магистрали представлено в таблице 1.2.

Зная содержание допустимого множества машинных циклов, нетрудно построить их последовательности, реализующие ту или иную команду МП. Машинные циклы удобно представлять в формате, показанном на рисунке 1.4.

Таблица 1.2

Слово состояния	FETCH	MEMORY_READ	MEMORY_WRITE	STACK_READ	
	1	2	3	4	
D0 (INTA)	0	0	0	0	
D1 (WO)	1	1	0	1	
D2 (STACK)	0	0	0	1	
D3 (HLTA)	0	0	0	0	
D4 (OUT)	0	0	0	0	
D5 (M1)	1	0	0	0	
D6 (INP)	0	0	0	0	
D7 (MEMR)	1	1	0	1	
Команды шины управления	MEMR	MEMR	MEMW	MEMR	

Продолжение таблицы 1.2

STACK_WRITE	INPUT	OUTPUT	INTERRUPT_M1	HALT	HALT_INTERRUPT	INTERRUPT
5	6	7	8	9	10	11
0	0	0	1	0	1	1
0	1	0	1	1	1	1
1	0	0	0	0	0	0
0	0	0	0	1	1	0
0	0	1	0	0	0	0
0	0	0	1	0	1	0
0	1	0	0	0	0	0
0	0	0	0	1	0	0
MEMW	I/OR	I/OW	INTA	—	INTA	INTA

Номер
цикла

Тип
цикла

Состояние
AB

Состояние
DB

Рисунок 1.4 – Формат машинных циклов МП

Во время выполнения каждого машинного цикла МП принимает или выдает информацию по шине данных. Направление передачи совпадает со стробами DBIN и $\overline{WR}$. В соответствии с этим все типы циклов можно разбить на циклы чтения и записи, связанные с вводом и выводом информации из МП, как показано на рисунке 1.5.

Циклы чтения

- 1 FETCH
- 2 MEMORY_READ
- 4 STACK_READ
- 6 INPUT
- 8 INTERRUPT_M1
- 10 HALT_INTERRUPT
- 11 INTERRUPT

Циклы записи

- 3 MEMORY_WRITE
- 5 STACK_WRITE
- 7 OUTPUT

Рисунок 1.5 – Организация циклов в группы

Временные диаграммы циклов приведены на рисунке 1.6.

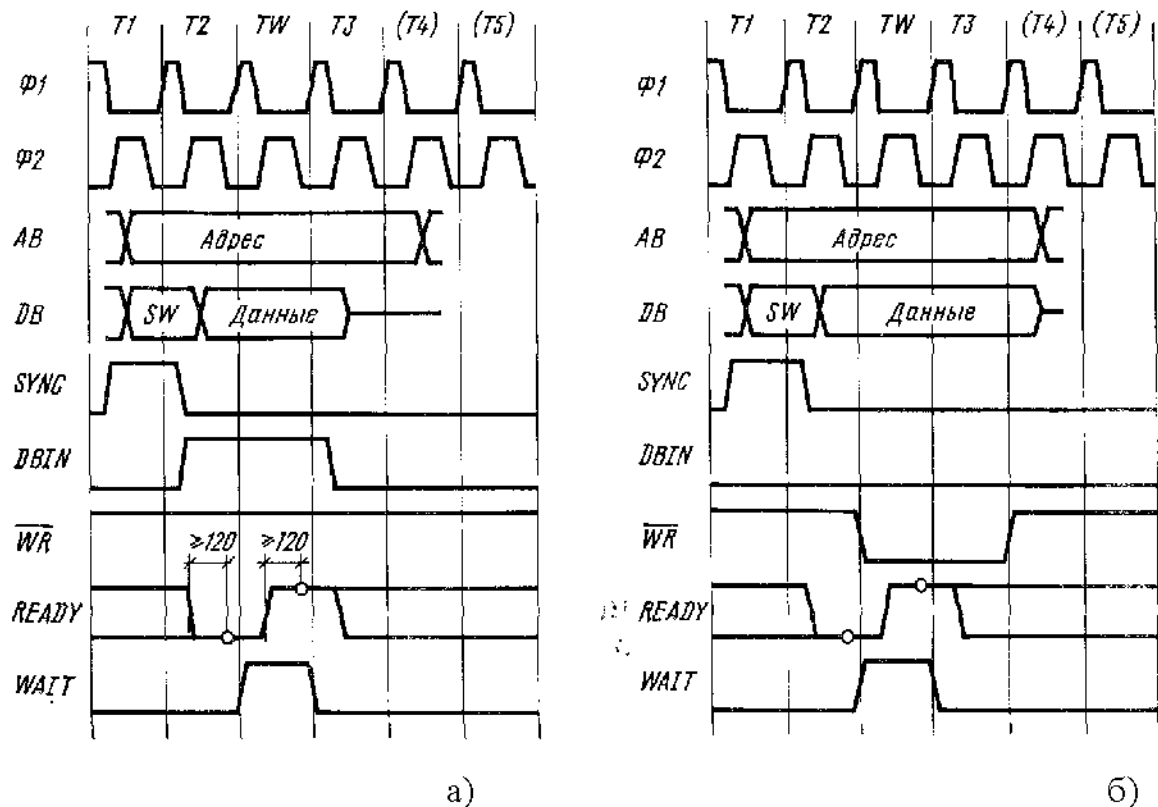


Рисунок 1.6 – Временные диаграммы циклов чтения (а) и записи (б) микропроцессора i8080

В такте T2 по срезу $\Phi 2$ выполняется проверка на активность сигнала READY. В случае его активности ($READY = 1$) МП из такта T2 переходит к такту T3 обмена данными, реализуя синхронный протокол BB. Если же быстродействия внешнего модуля MC недостаточно для синхронного обмена с МП за один такт, он может задержать момент перехода к такту T3 на целое число тактов TW. Для этого достаточно установить в 0 сигнал READY до момента его первой проверки и удерживать в этом состоянии то время, которое необходимо модулю для подготовки к обмену. В случае $READY = 0$ МП из такта T2 переходит в режим ожидания WAIT, о чем свидетельствует появление высокого уровня напряжения на линии WAIT. В данном режиме МП генерирует циклы TW ожидания сигнала READY и реализует асинхронный протокол обмена с внешними модулями MC. Продолжительность режима WAIT определяется длительностью действия

напряжения низкого уровня на линии READY, которая контролируется в каждом такте TW по срезу Ф2.

При обнаружении высокого уровня напряжения сигнала READY МП из такта TW переходит к выполняющему обмен такту T3. При этом сигнал WAIT устанавливается в 0. Для нормальной работы МП установку сигнала READY в 0 или 1 рекомендуется производить более чем за 120 нс до среза Ф2 и удерживать по крайней мере до его проверки.

Микропроцессор имеет несложные встроенные средства обслуживания запросов на прерывания с тремя управляющими сигналами:

- INT – запрос на прерывание;
- INTE – разрешение прерывания;
- SW.INTA – подтверждение прерывания.

В МП предусмотрены простейшие средства для управления захватом шины. Процедура захвата шины инициализируется внешними средствами установкой сигнала HOLD. Микропроцессор реагирует на этот сигнал по срезу Ф2. По фронту Ф1 в такте T3 при вводе данных или в такте, следующем за T3, при выводе данных МП формирует сигнал HLDA, а по фронту Ф2 в этом же такте шина данных переводится в высокоомное z-состояние. Шина адреса всегда переводится в z-состояние в такте, следующем за T3.

Перезапуск МП осуществляется сигналом RESET. Временные диаграммы перезапуска приведены на рисунке 1.7.

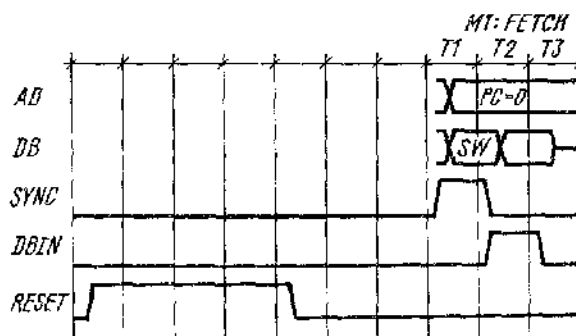


Рисунок 1.7 – Временные диаграммы перезапуска МП i8080

Рассмотрим пример разработки ядра одноплатного микроконтроллера на основе микропроцессора i8080.

Интерфейс однокристалльного МП, как правило, не обеспечивает его непосредственного соединения с системной магистралью по ряду причин. Во-первых, состав шин и логика управления обменом интерфейса МП и системной магистрали отличаются. Во-вторых, однокристалльные МП имеют малые токовую и емкостную нагрузки. Это объясняется ограничениями, накладываемыми на максимальную рассеиваемую мощность кристалла (обычно 1,5 – 2 Вт). Кроме того, выходные буферы МП i8080 обеспечивают токовую нагрузку I_{OL} до 1,6 мА и емкостную C_L до 100 пФ, что недостаточно для больших систем.

Таким образом, для построения законченного модуля ЦП требуются дополнительные схемы. Важнейшими среди них являются:

- генератор тактовых импульсов (ГТИ);
- системный контроллер;
- буферные регистры;
- шинные формирователи.

Генератор тактовых импульсов.

Для МП i8080 используется специализированный генератор тактовых импульсов, выполненный в виде микросхемы. Имеется отечественный аналог – КР580ГФ24. Структура ГТИ и его условное обозначение приведены на рисунке 1.8, временные диаграммы работы схемы – на рисунке 1.9.

Выводы XTAL1 и XTAL2 служат для подключения кварцевого резонатора, а вывод TANK – для выбора его гармоники. Выход генератора буферизирован и выведен на линию OSC так, чтобы другие устройства МС могли его использовать. Основой схемы является делитель на 9, поэтому частота кварцевого резонатора должна быть в 9 раз больше, чем частота следования импульсов Φ_1 , Φ_2 . Частота кварцевого резонатора определяется быстродействием МС и лежит в пределах от 4,5 до 22,5 МГц. При $OSC = 22,5$

МГц период следования синхроимпульсов $\Phi 1$ и $\Phi 2$ (длительность микротакта) 400 нс, что соответствует нижней границе диапазона, допустимого для МП i8080.

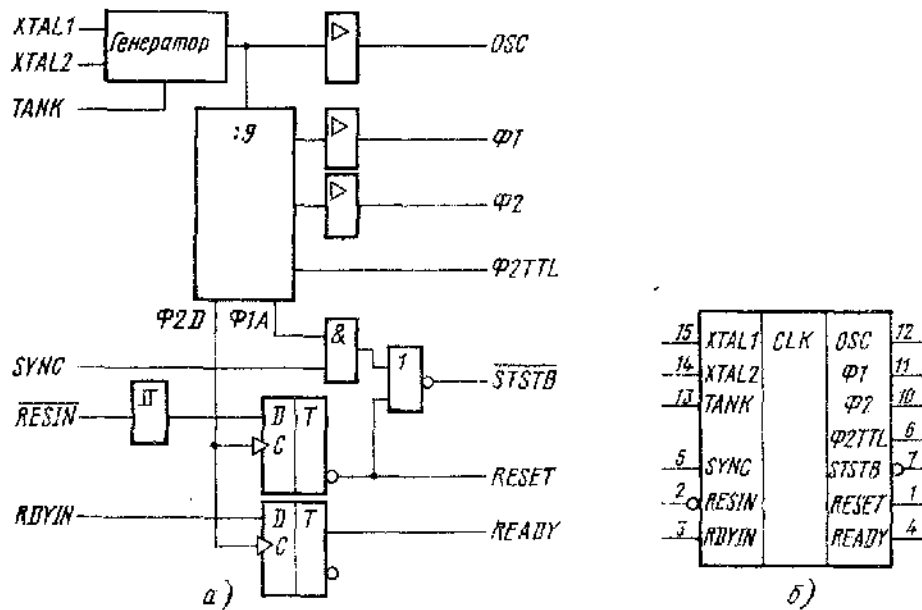


Рисунок 1.8 – Генератор тактовых импульсов KP580ГФ24:

а – структурная схема; б – условно-графическое обозначение

В состав схемы KP580ГФ24 входят также логические цепи для генерации строба $\overline{STSTB}$ (Status Strobe), синхронизации сигнала RDYIN (Ready Input) и формирования мощного выхода RESET, служащего для аппаратного сброса МС в исходное состояние.

Системный контроллер и формирователь шины.

Для МП i8080 используется специализированная микросхема в качестве системного контроллера и формирователя шины данных. Имеется ее отечественный аналог – KP580BK28/38. Схема формирует базовый набор управляющих стробов магистрали типа И41 и обеспечивает двунанравленную буферизацию шины данных МП от основной памяти и устройств ВВ. В БИС использована биполярная ТТЛШ-технология, гарантирующая небольшие задержки и высокую нагрузочную способность.

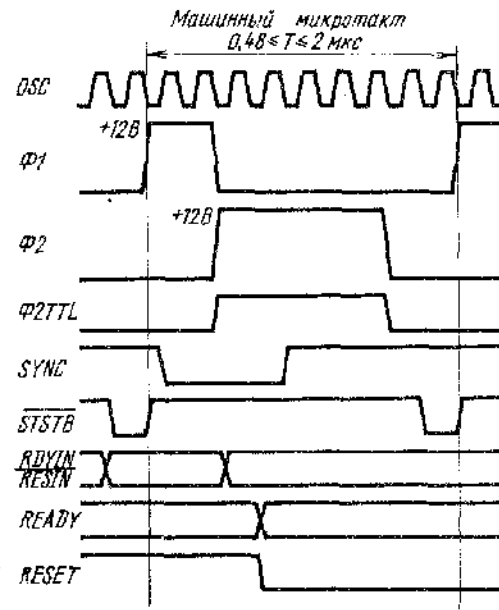


Рисунок 1.9 – Временные диаграммы работы генератора тактовых импульсов КР580ГФ24

Структурная схема системного контроллера КР580ВК28/ВК38 и его условное графическое обозначение приведены на рисунке 1.10. Двухнаправленный 8-разрядный шинный формирователь обеспечивает выход DB7,..., DB0 со стороны системной магистрали с током нагрузки до 10 мА и емкостью нагрузки до 100 пФ, а также изолирует шину данных МП D7,..., D0 от системной. Задержка, вносимая формирователем в шину данных, не превышает 40 нс. Формирователь выполнен по схеме с тремя состояниями.

В состав контроллера входит регистр-защелка, который по стробу $\overline{STSTB}$ фиксирует слово состояния SW, выдаваемое МП в начале каждого машинного цикла. Слово состояния определяет тип текущего машинного цикла в соответствии с таблицей 1.2, в зависимости от которого логическая схема контроллера формирует один из пяти управляющих стробов системной магистрали: $\overline{MEMR}$, $\overline{MEMW}$, $\overline{I/OR}$, $\overline{I/OW}$, $\overline{INTA}$, также указанных в таблице 1.2. Временные диаграммы их генерации приведены на рисунке 1.11

и зависят от длительности стробов $\overline{DBIN}$ и $\overline{WR}$ микропроцессора. Системный контроллер KP580BK38 (BK38) в отличие от KP580BK28 (BK28) формирует опережающие стробы $\overline{MEMW}$ и $\overline{I/OW}$.

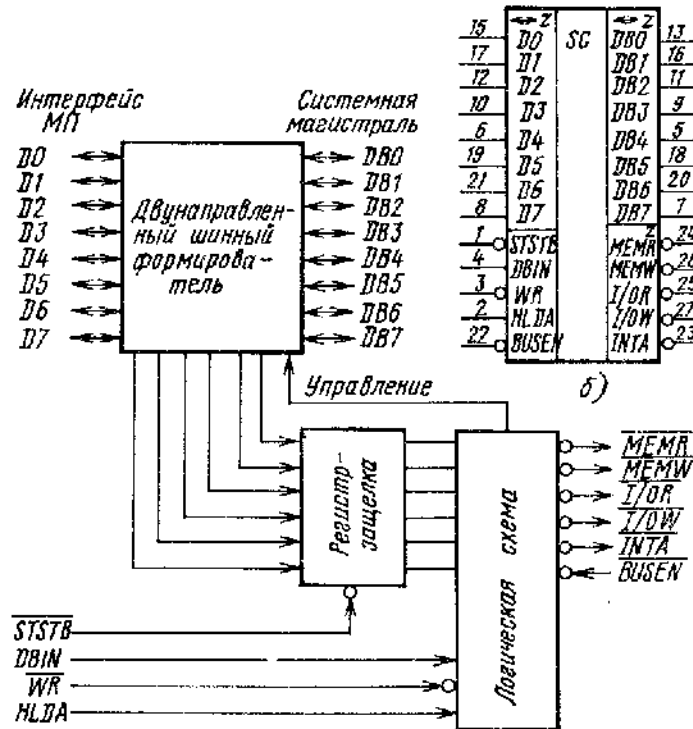


Рисунок 1.10 – Системный контроллер KP580BK28/38

Строб $\overline{INTA}$ обычно используется для выбора порта вектора прерывания, изолированного от пространства ВВ. В МС, где требуется только один вектор прерываний, устройство KP580BK28/BK38 может автоматически в необходимый момент времени выдать команду RST7 на шину данных D7,..., D0 без каких-либо дополнительных логических схем. Для этого выход $\overline{INTA}$ следует соединить с источником питания плюс 12 В через резистор сопротивлением 1 кОм.

Трехстабильные выходные буферы шины данных и управляющих сигналов открываются асинхронно входным сигналом $\overline{BUSEN}$. При $\overline{BUSEN} = 1$ буферы находятся в состоянии с высоким выходным сопротивлением.

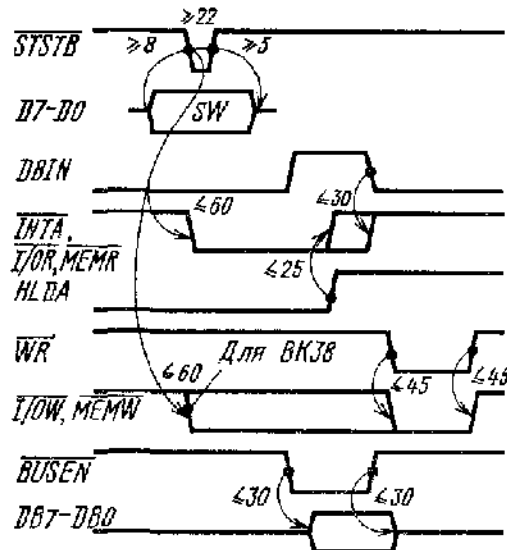


Рисунок 1.11 – Временные диаграммы работы системного контроллера KP580BK28/38

Буферный регистр.

Буферный регистр KP580ИР82/ИР83 выполнен по биполярной технологии и используется в качестве 8-разрядного фиксатора или буфера. На рисунке 1.12 представлены схема и условное обозначение регистра KP580ИР82, в отличие от которого регистр KP580ИР83 имеет инверсную выходную шину.

Основой схемы является 8-разрядный регистр-защелка со статическим синхровходом STB (Strobe). Запись данных в регистр разрешена при $STB = 1$. В противном случае регистр находится в режиме хранения. На выходе регистра имеется трехстабильный буфер, управляемый сигналом $\overline{OE}$ (Output Enable). Буфер обеспечивает выходной ток до 32 мА и емкость нагрузки до 300 пФ. Если управляющий сигнал $\overline{OE}$ активен, то данные регистра передаются на выход микросхемы. При $\overline{OE} = 1$ выходной буфер закрыт и находится в высокоомном состоянии. Временные диаграммы работы регистров представлены на рисунке 1.13.

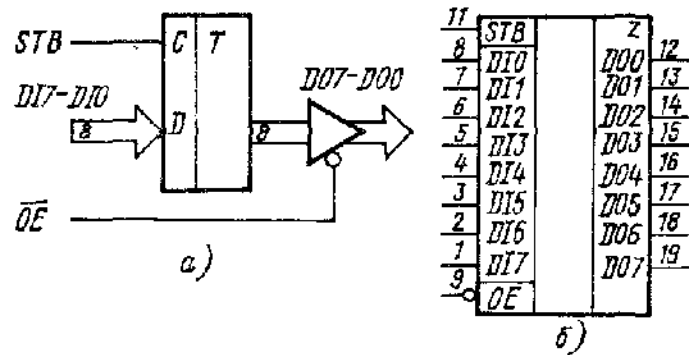


Рисунок 1.12 – Буферный регистр КР580ИР82:

а – структурная схема; б – условно-графическое обозначение

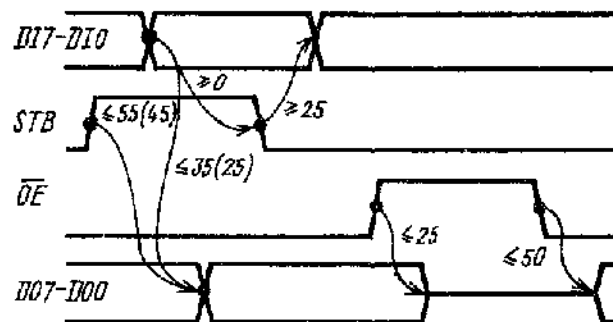


Рисунок 1.13 – Временные диаграммы работы регистров КР580ИР82/ИР83

Шинный формирователь.

Шинный формирователь КР580ВА86/ВА87 – биполярная микросхема, предназначенная для реализации 8-разрядных одно- и двунаправленных буферных схем с тремя состояниями на выходе. На рисунке 1.14 приведены структурная схема и условное графическое обозначение БИС КР580ВА86. Формирователь КР580ВА87 отличается тем, что имеет инвертирующие буферы.

Трехстабильные буферные схемы выбираются только при низком уровне напряжения на линии $\overline{OE}$. При этом, если на входе T (Transmitter) высокий уровень напряжения, то открывается буфер для передачи из канала A в B . В противном случае осуществляется передача в обратном

направлении. Временные диаграммы работы буфера представлены на рисунке 1.15.

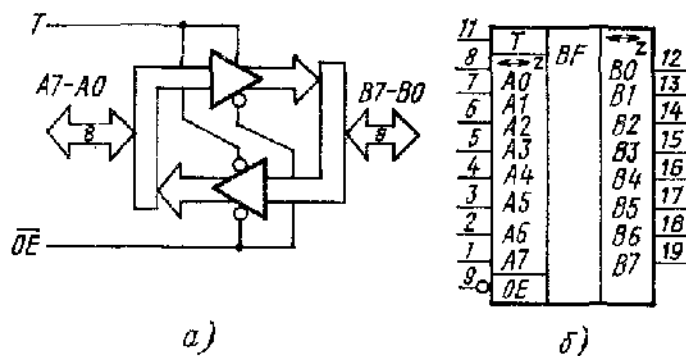


Рисунок 1.14 – Шинный формирователь KP580BA86:

а – структурная схема; б – условно-графическое обозначение

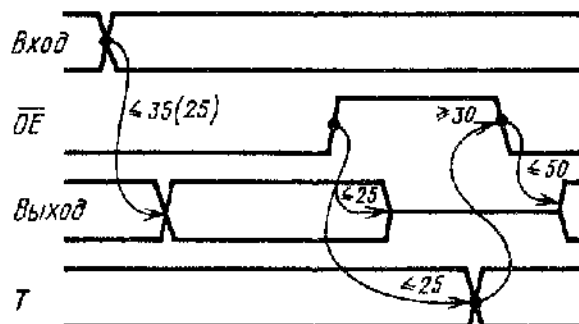


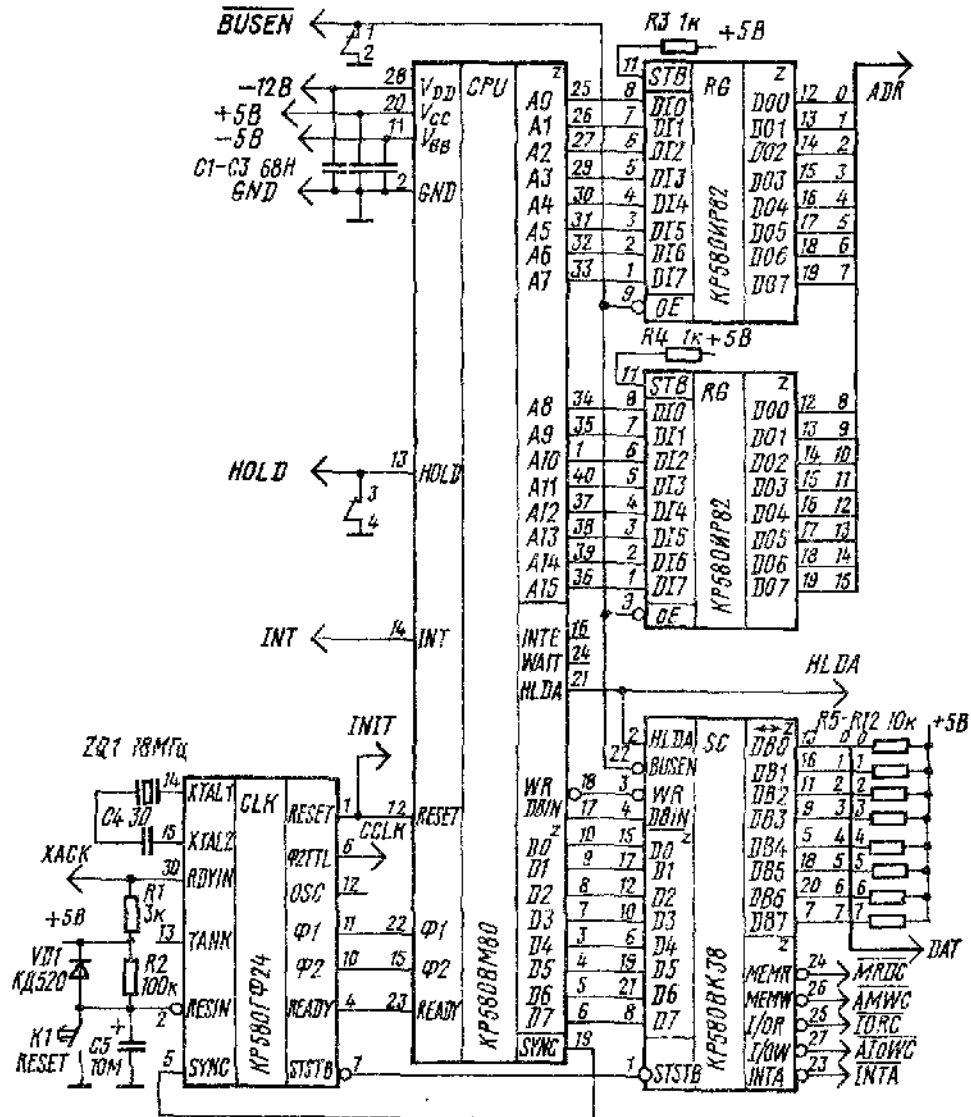
Рисунок 1.15 – Временные диаграммы работы формирователей KP580BA86/BA87

Восьмиразрядные каналы А и В шинного формирователя не эквивалентны. Так, если со стороны канала А ток нагрузки не должен превышать 16 мА, а емкость нагрузки 100 пФ, то со стороны канала В обеспечивается более высокая нагрузочная способность: максимальный ток нагрузки 32 мА, емкость нагрузки не более 300 пФ.

Построение центрального процессора.

С учетом проведенного выше анализа, а также содержания предыдущего практического занятия, на данном этапе можно построить схему центрального процессорного ядра.

Схема ЦП на базе БИС серии КР580 приведена на рисунке 1.16.



входу $\overline{RESIN}$ ГТИ подключать RC-цепочку. Наличие гистерезиса на входе $\overline{RESIN}$ (см. рисунок 1.8, а) гарантирует быстрое переключение схемы в момент достижения порога переключения. Постоянная времени RC-цепочки рассчитывается из требований, накладываемых на длительность сигнала RESET.

Сопротивление резистора R2 рекомендовано фирмой-производителем МП и составляет 100 кОм. Резистор предназначен для формирования зарядного тока конденсатора C5. Емкость конденсатора также рекомендована и составляет величину 10 мкФ. Диод VD1 служит для быстрого разряда конденсатора при отключении напряжения питания. Кнопка K1 служит для осуществления ручного сброса МП.

Зарядный ток конденсатора в момент включения напряжения питания составляет величину:

$$I_{C5} = I_{R2} = \frac{U_{ПИТ.}}{R2}. \quad (1.1)$$

Мощность, рассеиваемая на резисторе R2, рассчитывается по формуле:

$$P_R = I_R^2 \cdot R. \quad (1.2)$$

По результатам расчетов выбирается резистор R2 необходимого тип номинала.

Рабочее напряжение конденсатора C5 рассчитывается по формуле:

$$U_{РАБ.} = 1,5 \cdot U_{ПИТ.}. \quad (1.3)$$

По результатам расчетов выбирается конденсатор необходимого тип номинала.

Диод должен быть рассчитан на прямой ток, не менее:

$$I_{VD,МАКС.} \geq 1,3 \cdot I_{VD}. \quad (1.4)$$

Диод должен быть рассчитан на обратное напряжение $U_{ОБР.}$, не менее:

$$U_{ОБР.} = 1,5 \cdot U_{ПИТ.}. \quad (1.5)$$

Первое требование исходит из применяемого резистора для

ограничения тока заряда конденсатора. Второе требование – это напряжение питания цепи сброса.

По результатам расчетов выбирается диод необходимого типа, имеющий следующие электрические параметры:

- прямой ток через диод;
- обратное напряжение на диоде.

Требования к электрическим параметрам кнопки:

- максимальный ток через контакт рассчитывается по формуле:

$$I_{\text{МАКС.КОМ.}} \geq 1,3 \cdot I_{\text{КОМ.}}; \quad (1.6)$$

- максимальное рабочее напряжение рассчитывается по формуле:

$$U_{\text{МАКС.КОМ.}} \geq 1,5 \cdot U_{\text{ПИТ.}} \quad (1.7)$$

Кнопка должна быть без фиксации и минимальных габаритов. По результатам расчетов выбирается кнопка, имеющая следующие параметры:

- максимальный коммутируемый ток;
- максимальное коммутируемое напряжение.

Период следования синхросигналов $\Phi 1$, $\Phi 2$ микропроцессора i8080 определяется кварцевым резонатором ZQ1. При выборе и установке кварцевого резонатора следует обеспечить минимальное сопротивление внешней цепи генератора XTAL2 – XTAL1. Со стороны кварцевого резонатора генератор представляет индуктивный элемент, приводящий к снижению рабочей частоты. Для компенсации данного эффекта рекомендуется последовательно с кварцевым резонатором со стороны вывода XTAL2 включить конденсатор небольшой емкости. Для кварцевого резонатора частотой 18 МГц емкость от 15 до 30 пФ. При повышении частоты может потребоваться уменьшение емкости конденсатора.

Рабочее напряжение конденсатора C4 рассчитывается по формуле (1.3). По результатам расчетов выбирается конденсатор необходимого типа номинала.

Кроме синхросигналов $\Phi 1$, $\Phi 2$ генератор формирует также системный

сигнал CCLK (ток нагрузки I_{OL} до 15 мА) и сигнал системного сброса INIT (I_{OL} до 15 мА). Сигнал INIT предназначен для начальной установки системы в исходное состояние. Он реализуется при наличии сигнала, формируемого RC-цепочкой на входе $\overline{RESIN}$ генератора. Применяемые в схеме на рисунке 1.16 элементы обеспечивают длительность сигнала сброса 0,35 с.

Сигнал подтверждения ХАСК используется при работе с медленными модулями памяти и ВВ. Он может быть задействован также для организации ожидания ЦП некоторого внешнего события. Микропроцессор i8080 будет переведен в состояние ожидания WAIT, если сигнал ХАСК окажется неактивным в момент первой его проверки (начало второй трети такта T2).

В задании данного курсового проекта не сказано, какие внешние устройства будут подключаться. Поэтому будем разрабатывать более простую систему, ориентированную на работу с быстрыми модулями памяти и ВВ. Для этого на вход RDYIN подадим напряжение высокого уровня. Требуемый потенциал необходимо подавать через ограничивающий резистор. Сопротивление резистора R1 рекомендовано фирмой-производителем МП и составляет 3 кОм. Мощность, рассеиваемая на резисторе, рассчитывается по формуле (1.2) и далее производится выбор тип номинала резистора.

В схеме на рисунке 1.16 использован системный контроллер типа KP580BK38. От контроллера KP580BK28 он отличается тем, что формирует упрежденные стробы $\overline{I/O\overline{W}}$ и $\overline{MEM\overline{W}}$ (см. рисунок 2.11), рассчитанные на периферийные приборы второго поколения.

Как показано на рисунке 1.11, все командные стробы KP580BK38 связаны со стробом STSTB и задержаны относительно его начала на интервал от 20 до 60 нс. Так как STSTB опережает на T/3 фазу Ф2 такта T2, то до момента первой проверки сигнала ХАСК еще остается $4T/9 - 60$ нс. При T = 400 нс это время примерно равно 120 нс, что вполне достаточно для сброса сигнала ХАСК. Временные диаграммы работы ЦП приведены на

рисунке 1.17.

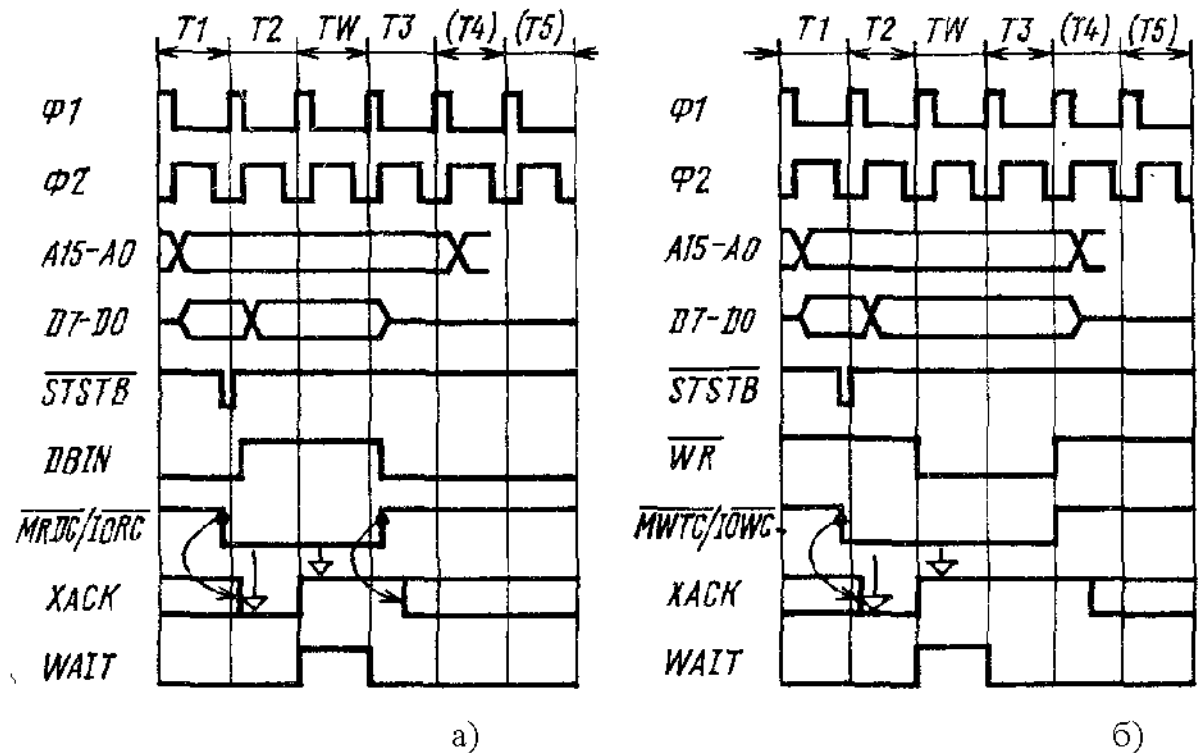


Рисунок 1.17 – Временные диаграммы циклов чтения (а) и записи (б) центрального процессора

Таким образом, применение контроллера KP580BK28 возможно либо в системах с синхронным обменом, либо в системах асинхронного доступа с инверсной линией XACK. В MC, допускающих оба протокола обмена, рекомендуется использовать микросхему KP580BK38. Во всех случаях нагрузочная способность командных линий $\overline{MRDC}$, $\overline{MWTC}$, $\overline{IORC}$ и $\overline{IOWC}$ следующая: I_{OL} до 10 мА, C_L линии до 100 пФ. Допустимый ток I_{OL} и емкость C_L линии $\overline{INTA}$ равны 5 мА и 100 пФ соответственно.

Системный контроллер обеспечивает также двунаправленную буферизацию шины данных МП без инверсии, доводя I_{OL} до 10 мА и C_L до 100 пФ. Дополнительная задержка в шине данных около 30 нс.

Удобно, но не обязательно системную шину данных DAT7,..., DAT0 подключать к источнику питания плюс 5 В через резисторы 10 кОм. Это

практически не скажется на нагрузочной способности шины. С другой стороны, чтение команды из несуществующей области памяти приведет к выполнению команды RST 7, это может быть использовано в отладочных и диагностических целях.

Адресная шина МП i8080 обеспечивает ток нагрузки $I_{OL} = 1,8$ мА и емкость нагрузки C_L до 100 пФ. Этой нагрузочной способности достаточно для построения одноплатного МК закрытого типа. В открытых для расширения МС требуется дополнительная буферизация адресной шины, которая может быть выполнена с помощью двух буферных регистров КР580ИР82 без инверсии. Буферизация адреса позволяет увеличить максимальные значения I_{OL} до 32 мА и C_L до 300 пФ. Вместе с этим в шине появляется дополнительная задержка 35 нс.

Возможна реализация адресного буфера на других микросхемах, например КР580ИР83, КР580ВА86/ВА87, К589ИР12, К589АП16/АП26 и т. д. Они имеют другие нагрузочные и временные характеристики и могут приводить к инверсии адреса.

Три сигнала ЦП – запрос на захват шины HOLD, подтверждение захвата HLDA ($T_{OL} = 1,8$ мА) и разрешение шины BUSEN — служат для организации доступа к магистрали со стороны других активных модулей системы. Частным случаем такого модуля является устройство с прямым доступом к памяти, управляемое БИС КР580ВТ57. Для запуска ЦП в системах, которые не имеют других активных модулей кроме ЦП, необходимо установить переключки 1-2, 3-4.

Вывод HOLD подсоединяем к общему проводу схемы, что предотвратит переход МП в режим прямого доступа в память, так как в данном курсовом проекте указанный режим не используется.

Вывод INT подсоединим к общему проводу схемы, что предотвратит возможность прерывания во время нормального выполнения программы, так как в данном курсовом проекте режим прерываний не используется.

В результате выполнения данного пункта было разработано ядро одноплатного микроконтроллера на основе процессора i8080.

Оборудование и материалы

Практическая работа выполняется с использованием ПК и любого программного обеспечения с функциями САПР для выполнения схемы электрической принципиальной ядра одноплатного микроконтроллера.

Указания по технике безопасности

Перед проведением практического занятия следует изучить инструкцию по технике безопасности при работе на компьютере. Также следует соблюдать правила и требования, установленные для данного компьютерного класса.

Задания (задачи)

1 Разработать ядро одноплатного микроконтроллера на основе микропроцессора i8085. Для решения поставленной задачи необходимо выполнить следующее:

- привести общие характеристика микропроцессора;
- разработать систему синхронизации;
- разработать системный контроллер и формирователь шины;
- разработать схему буферизации адресной шины;
- разработать схему буферизации шины данных;
- разработать общую схему центрального процессора;
- выполнить электрический расчет необходимых компонентов и провести их обоснованный выбор.

2 Разработать ядро одноплатного микроконтроллера на основе микропроцессора Z80. Для решения поставленной задачи необходимо выполнить те же задания, что и в п. 1.

3 Разработать ядро одноплатного микроконтроллера на основе микропроцессора i8086 в минимальном режиме. Для решения поставленной задачи необходимо выполнить те же задания, что и в п. 1.

4 Разработать ядро одноплатного микроконтроллера на основе микропроцессора i8086 в максимальном режиме. Для решения поставленной задачи необходимо выполнить те же задания, что и в п. 1.

5 Разработать ядро одноплатного микроконтроллера на основе микропроцессора i8088 в минимальном режиме. Для решения поставленной задачи необходимо выполнить те же задания, что и в п. 1.

6 Разработать ядро одноплатного микроконтроллера на основе микропроцессора i8088 в максимальном режиме. Для решения поставленной задачи необходимо выполнить те же задания, что и в п. 1.

Выбор типа задачи для каждого студента выполняется по варианту, указанному в таблице 1.2. Номер варианта определяет преподаватель.

Таблица 1.2

№ варианта	1	2	3	4	5	6	7	8	9
Номер задачи	3	2	1	6	5	4	6	2	3

Продолжение таблицы 1.2

№ варианта	10	11	12	13	14	15	16	17	18
Номер задачи	1	4	5	3	2	6	5	1	2

Содержание отчета

Отчет по практическому занятию выполняется на листах формат А4

согласно требованиям ГОСТ и ЕСКД и должен содержать следующие пункты:

- а) титульный лист с названием работы, дисциплины, кода группы, ФИО исполнителя и преподавателя;
- б) цель и содержание работы;
- в) краткое теоретическое введение;
- г) условия, ход выполнения и решения задач;
- д) общую схему электрическую принципиальную ядра одноплатного микроконтроллера;
- е) выводы по результатам выполнения практического задания.

Список литературы, рекомендованный к использованию по данной теме

Основная учебная литература

1 Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд., испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

Дополнительная учебная литература

1 Бродин В.Б., Калинин А.В. Системы на микроконтроллерах и БИС программируемой логики – М.: Издательство ЭКОМ, 2002. – 400 с.

2 Боборыкин А. В., Липовицкий Г.П. и др. Однокристалльные микроЭВМ. – М.: МИКАП, 1994 – 400с.

3 Сташин В. В., Урусов А. В., Мологонцева О. Ф. Проектирование цифровых устройств на однокристалльных микроконтроллерах – М.: Энергоатомиздат, 1990 – 224с.

4 Шевкопляс Б.С. Микропроцессорные структуры. Инженерные решения: Справочник. - М.: Радио и связь, 1990 – 512 с.

5 Щелкунов, Н. Н. Микропроцессорные средства и системы / Н. Н. Щелкунов, А. П. Дианов. – М. : Радио и связь, 1989. – 288 с.

6 Коффрон, Дж. Технические средства микропроцессорных систем: практический курс / Дж. Коффрон. Пер. с англ. – М. : Мир, 1983. – 344 с.

7 Шахнов, В. А. Микропроцессоры и микропроцессорные комплекты интегральных микросхем : справочник / В.-Б. Б. Абрайтис, Н. Н. Аверьянов, А. И. Белоус [и др.]: под ред. В. А. Шахнова. – Том 1. – М. : Радио и связь, 1988. – 368 с.

8 Соппа, И. В. Микропроцессоры и микроконтроллеры / И. В. Соппа, А. С. Самардак, В. А. Иванов. – Владивосток : Издательство Дальневосточного университета, 2009. – 211 с.

9 Алексенко, А. Г. Основы микросхемотехники / А. Г. Алексенко. 3-е изд. перераб. и доп. – М. : БИНОМ. Лаборатория знаний, 2009. – 448 с.

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ 2

Тема: Разработка модуля памяти для одноплатного микроконтроллера.
Разработка схемы дешифратора адреса для модуля памяти.

Цель работы: закрепление знаний, полученных на лекциях по разделу: «Проектирование микропроцессорных систем». В ходе выполнения заданий практического занятия необходимо:

- обосновать тип используемой памяти;
- осуществить выбор БИС запоминающих устройств и схемы их подключения к магистралям микропроцессорного ядра;
- на основе временных диаграмм пояснить работу БИС запоминающих устройств в составе разрабатываемого одноплатного микроконтроллера;
- распределить адреса между компонентами памяти микроконтроллера;
- разработать схему дешифратора адреса согласно варианту задания;
- разработать карту распределения памяти микроконтроллера.

Знания, приобретаемые студентом в результате освоения темы:

- классификация микропроцессоров и их основные характеристики;
- аппаратная и логическая структуры типовых микропроцессоров;
- режимы адресации и система команд;
- способы организации ввода/вывода в микропроцессорных системах.

Умения, приобретаемые студентом в результате освоения темы:

- разрабатывать микропроцессорные системы для решения задач управления, автоматизации, сбора и обработки данных и других целей.

Формируемые компетенции: ОК-6, ОК-7, ОПК-3, ОПК-4, ОПК-6, ОПК-7, ОПК-8, ОПК-9, ПК-1, ПК-3, ПК-5, ПК-6.

Актуальность темы: тема является актуальной, так как любое микропроцессорное устройство использует различные виды памяти, которые имеют различные интерфейсы и протоколы обмена. Поэтому студент должен знать основные виды памяти, уметь синтезировать модуль памяти и дешифратор адреса для него.

Теоретическая часть

Для хранения небольших массивов кодовых слов (информации) обычно используются регистры. Для хранения больших массивов (информации) в таком случае потребуется большое количество микросхем регистров. Поэтому во втором случае используется особый класс устройств – запоминающие устройства (ЗУ). Запоминающие устройства могут хранить не только информацию, но и коды программ, результаты промежуточных вычислений и т. д. Таким образом, ЗУ имеет достаточно широкое применение.

Классификация может быть расширена дополнительными элементами:

а) по типу элементов памяти:

- 1) на биполярных транзисторах различного типа логики;
- 2) на полевых транзисторах различного типа;

б) по способу размещения и поиска информации в информационном поле:

1) адресные (поиск осуществляется по номеру ячейки памяти):

- с произвольным доступом (поиск по новому адресу не зависит от предыстории);

- с последовательным доступом (прежде чем попасть в нужную ячейку памяти, необходимо пройти предыдущие ячейки, причем информация из них считываться не будет);

2) безадресные (номера ячейки нет):

- типа очереди (первым вошел – первым вышел, т. е. в какой последовательности производилась запись, в такой производится и считывание);

- стековые (запись производится с первой по пятую ячейки, а считывание – в обратном порядке: с пятой по первую);

- ассоциативные (поиск происходит по значению

определенного признака, которые характеризует информацию).

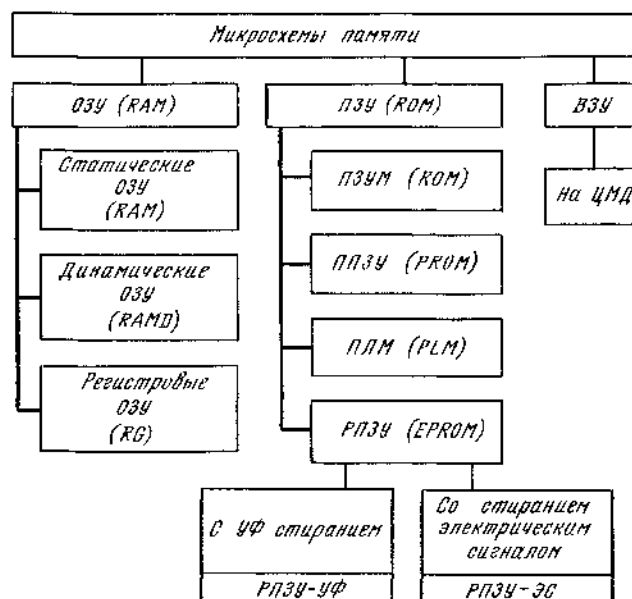


Рисунок 2.1 – Классификация ЗУ

Оперативное ЗУ предназначено для хранения переменной информации, оно допускает изменение своего содержимого в ходе выполнения процессором вычислительных операций с данными. Это значит, что процессор может выбрать (режим считывания) из ОЗУ код команды и данные и после обработки поместить в ОЗУ (режим записи) полученный результат. Причем возможно размещение в ОЗУ новых данных на местах прежних, которые в этом случае перестают существовать. Таким образом, ОЗУ может работать в режимах записи, считывания и хранения информации.

Постоянное ЗУ содержит информацию, которая не должна изменяться в ходе выполнения процессором программы. Такую информацию составляют стандартные подпрограммы, табличные данные, коды физических констант и постоянных коэффициентов и т. п. Эта информация заносится в ПЗУ предварительно, например, путем пережигания легкоплавких металлических перемычек в структуре ПЗУ, и в ходе работы процессора может только считываться. Таким образом, ПЗУ работает в режимах хранения и считывания.

Функциональные возможности ОЗУ шире, чем ПЗУ: ОЗУ может работать в качестве ПЗУ, т. е. в режиме многократного считывания однократно записанной информации, а ПЗУ в качестве ОЗУ использовано быть не может, так как не позволяет изменить однократно занесенную в него информацию. Разновидность перепрограммируемых ПЗУ, которая допускает перезапись информации также не может заменить ОЗУ, т. к. имеет ограниченное количество циклов записи.

В свою очередь, ПЗУ обладает преимуществом перед ОЗУ в свойстве сохранять информацию при сбоях и отключении питания – энергонезависимость. Оперативное ЗУ является энергозависимым, так как информация, записанная в ОЗУ, утрачивается при сбоях питания.

Для обеспечения надежной работы ЭВМ при отказах питания нередко ПЗУ используют и в качестве памяти программ. В таком случае программа заносится в ПЗУ предварительно и уже не может быть заменена в данном ПЗУ другой программой. Очевидно, использовать ПЗУ таким образом целесообразно прежде всего в специализированных автоматических устройствах, работающих по постоянной программе.

Основной составной частью любого ЗУ является элемент памяти (ЭП), который способен сохранить 1 бит информации. Совокупность элементов памяти называется ячейкой памяти и может иметь различную размерность: 4 бита (полубайт), 8 бит (1 байт), 16 бит (2 байта или слово), 32 бита (4 байта или двойное слово) и т. д.

Основные параметры ЗУ:

- информационная емкость – число единиц информации в битах или байтах, которое может одновременно храниться в данном ЗУ;
- быстродействие – характеризуется несколькими временными параметрами, в частности, временем цикла записи, считывания, выборки;
- энергопотребление – потребляемая ЗУ мощность от источника питания. В зависимости от режима работы ЗУ и потребляемая мощность

имеет различные значения. Для режима обращения, а тем более – записи, потребляемая мощность всегда больше, чем для режима хранения;

– электрические параметры, характерные для любой цифровой интегральной микросхемы.

Информационная емкость зависит как от числа ячеек памяти, так и размерности этих ячеек. Обращение к ячейкам памяти осуществляется по так называемым адресным линиям, причем оба этих параметра находятся в однозначной зависимости:

$$N = 2^n,$$

где N – количество ячеек памяти;

n – количество адресных линий.

Таким образом, информационная емкость ЗУ может быть описана формулой вида $N \times D$, где D – разрядность ячейки памяти. Необходимо помнить, что, например:

$$2^{10} = 1024,$$

это количество ячеек памяти при 10 адресных линиях. Эту величину принято обозначать как 1 К. Соответственно, при 11 линиях связи количество ячеек памяти будет составлять 2048 или 2 К.

Как было сказано ранее, размерность всего ЗУ зависит и от количества ЭП в ЯП. Так, если разрядность ЯП равна восьми при 10 адресных линиях, то информационная емкость ЗУ составит:

$$2^{10} \times 8 = 1024 \times 8 = 8192,$$

что соответствует 1 Кбайт или 8192 бита. Наиболее употребительным при многоразрядной организации данных является первое обозначение емкости.

Структура ОЗУ и ПЗУ как функциональных узлов приведена на рисунке 2.2.

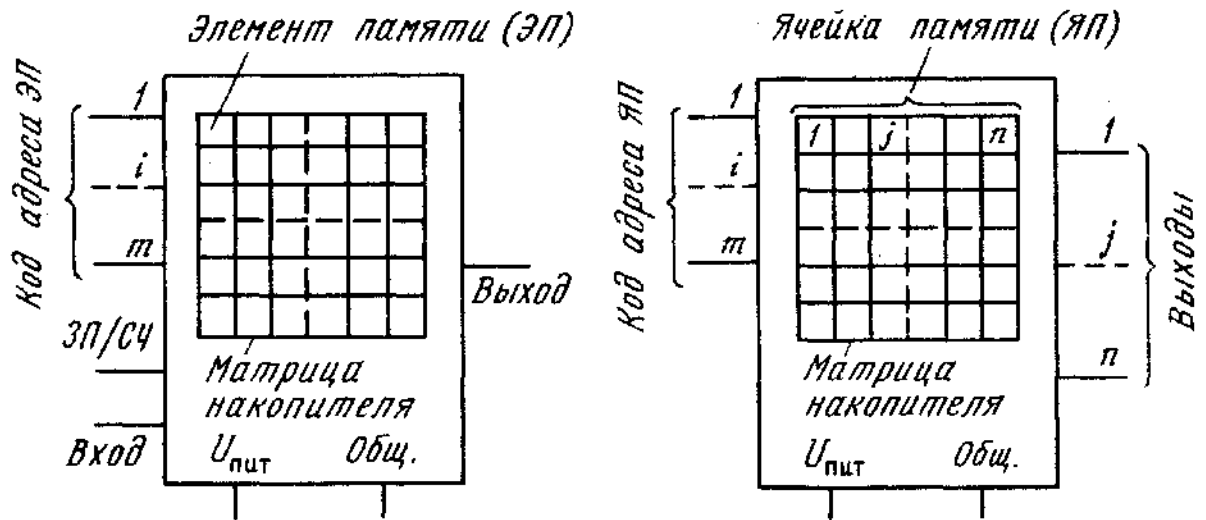


Рисунок 2.2 – Микросхема памяти как функциональный узел:

- а) ОЗУ;
- б) ПЗУ

В ОЗУ статического типа каждый ЭП представляет собой статический триггер, который может быть реализован на основе биполярных или МДП-транзисторов. Это зависит от технологии изготовления микросхем. На рисунке 2.3 показана ячейка памяти микросхемы К561РУ2, изготовленная по КМДП-технологии.

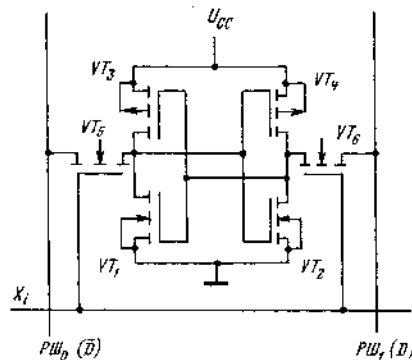


Рисунок 2.3 – Статический элемент памяти на триггере микросхемы К561РУ2

Для сохранения информации в триггере необходим источник питания, т. е. триггер рассматриваемого типа является энергозависимым. При наличии

Для изготовления микросхем динамических ОЗУ в основном применяют *n*-МДП-технология, которая позволяет повышать быстродействие и уровень интеграции микросхем, обеспечивать малые токи утечки и за этот счет увеличивать время сохранения заряда на запоминающем конденсаторе.

Особенности работы микросхемы можно узнать из [11].

Микросхемы ПЗУ по способу программирования, т. е. занесения в них информации, подразделяют на три группы:

- ПЗУ, однократно программируемые изготовителем по способу заказного фотошаблона (маски), масочные ПЗУ (ПЗУМ, ROM);
- ПЗУ, однократно программируемые пользователем по способу пережигания плавких перемычек на кристалле (ПДЗУ, PROM);
- ПЗУ, многократно программируемые пользователем, репрограммируемые ПЗУ (РПЗУ, EPROM).

Общим свойством всех микросхем ПЗУ являются их многоразрядная (словарная) организация, режим считывания как основной режим работы и энергонезависимость. Вместе с тем у них есть и существенные различия в способе программирования, режимах считывания, в обращении с ними при применении.

Наиболее оптимальным вариантом для данной практической работы является использование ПЗУ именно репрограммируемого типа, элементы памяти которых представлены на рисунке 2.5.

Основная отличительная особенность микросхем РПЗУ заключается в их способности к многократному (от 100 до 10 тыс.) перепрограммированию самим пользователем. Это свойство микросхем обеспечено применением ЭП со свойствами управляемых «перемычек», функции которых выполняют транзисторы со структурой МНОП (металл Al-нитрид кремния Si_3N_4 – окисел кремния SiO_2 – полупроводник Si) и транзисторы *n*-МОП с плавающим затвором (ПЗ) с использованием механизма лавинной инжекции заряда

ЛИЗМОП.

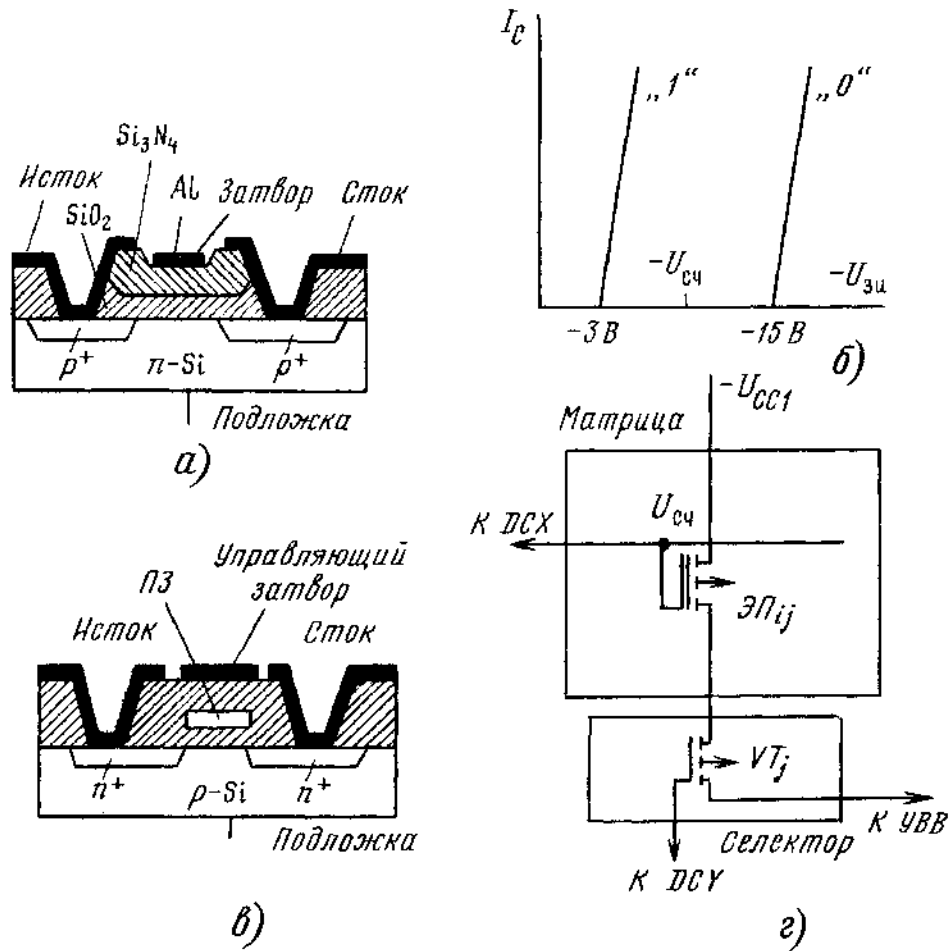


Рисунок 2.5 – Элементы памяти РПЗУ: а – типа МНОП; б – передаточная характеристика; в – типа ЛИЗМОП с двойным затвором; г – расположение ЭП в матрице микросхемы

Более подробное описание см. в [11].

Рассмотрим пример разработки модуля памяти для одноплатного микроконтроллера и разработку схемы дешифратора адреса для модуля памяти.

Выбор микросхем памяти для одноплатного микроконтроллера.

Наиболее удачным является случай, когда требуемый объем памяти содержится в одном корпусе. Но часто бывают случаи, когда нет микросхемы с требуемым объемом памяти. Именно такой случай, как наиболее сложный, будет рассмотрен далее в качестве примера.

Пусть, согласно заданию, необходимо разработать модуль памяти для одноплатного микроконтроллера, состоящий из следующих компонентов:

- ОЗУ – 16 Кбайт;
- ПЗУ – 4 Кбайт.

Рассмотрим требования к микросхемам памяти:

- согласованность по электрическим и временным параметрам с микропроцессором *i8080*;
- разрядность шины данных – 8 бит;
- информационная емкость ПЗУ – 4К x 8 бит;
- информационная емкость ОЗУ – 16К x 8 бит;
- для микросхем ПЗУ достаточно режима чтения;
- возможность перевода ИМС в режим с высокоомным выходным состоянием.

ПЗУ лучше выбрать с возможностью перепрограммирования, чтобы можно было менять программное обеспечение одноплатного микроконтроллера. При анализе микросхем ПЗУ отечественного производства было выявлено, что нет микросхемы емкостью 4К x 8 бит. Выходов из указанной ситуации два:

- реализовать блок ПЗУ необходимого объема на двух микросхемах объемом по 2К x 8 бит;
- реализовать блок ПЗУ необходимого объема на одной микросхеме объемом 8К x 8 бит.

В первом случае потребуется больше места на плате МК для размещения двух корпусов микросхем. И потребление от источника питания будет выше. Поэтому второй случай является более оптимальным. Старший разряд адресной шины микросхемы не будет использоваться. Его необходимо подключить к нулевому проводу источника питания.

Выбрана ИМС репрограммируемого ПЗУ со стиранием ультрафиолетовым излучением – К573РФ4, условно-графическое

обозначение которой приведено на рисунке 2.6. Программирование в разрабатываемой системе использоваться не будет, поэтому сигнал на входе разрешения программирования должен быть всегда в пассивном состоянии – уровень напряжения источника питания. Требуемый потенциал необходимо подавать через ограничивающий резистор. Сопротивление резистора рекомендовано фирмой-производителем ПЗУ и составляет 100 кОм. Мощность, рассеиваемая на резисторе, рассчитывается по формуле (2.1), где в качестве тока через резистор принят ток допустимый для ПЗУ:

$$P_R = I_R^2 \cdot R. \quad (2.1)$$

По результатам расчетов выбирается резистор необходимого тип номинала.

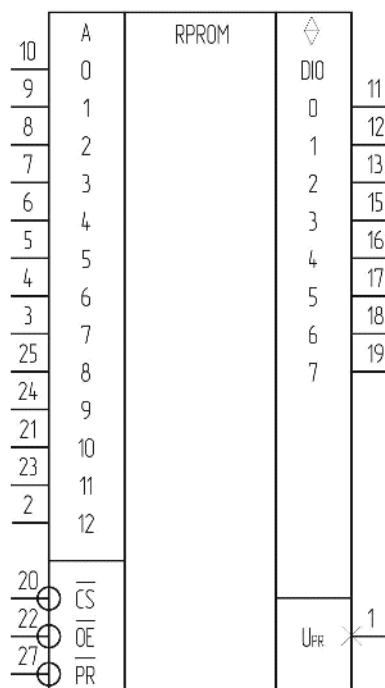


Рисунок 2.6 – Условно-графическое обозначение ИМС K573РФ4

Вывод для подачи напряжения программирования останется не подключенным.

Поскольку объем реализуемого ОЗУ достаточно мал, то оптимальным является его реализация на ОЗУ статического типа. Такие микросхемы обладают достаточным быстродействием для МП i8080. Кроме того,

статическое ОЗУ требует гораздо меньших аппаратных затрат по включению в схему одноплатного микроконтроллера.

При анализе микросхем ОЗУ отечественного производства было выявлено, что нет микросхемы емкостью 16К x 8 бит. Выходов из указанной ситуации только один: реализовать блок ОЗУ необходимого объема на двух микросхемах объемом по 8К x 8 бит, поскольку нет микросхемы ОЗУ объемом 32К x 8 бит.

Выбрана ИМС ОЗУ статического типа КР537РУ17, условно-графическое обозначение которой приведено на рисунке 2.7.

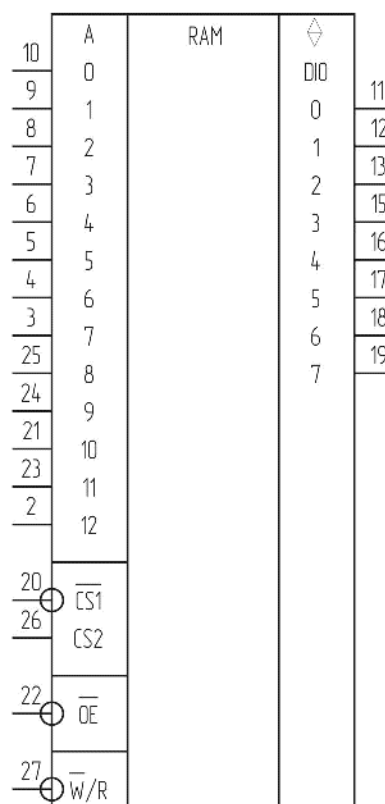


Рисунок 2.7 – Условно-графическое обозначение ИМС КР537РУ17

Составление карты распределения памяти.

Основная память МП i8080 рассматривается как линейный массив, состоящий из 64К байт. Формируемый микропроцессором 16-разрядный адрес дает ему возможность адресовать любой байт памяти. Слова в памяти хранятся в двух соседних байтах. В байте с младшим адресом хранится

младшая половина слова, а в байте со следующим адресом – старшая. Адресом слова служит адрес его младшего байта.

Поскольку все адресное пространство i8080 составляет 64 Кбайта, то с учетом выбранных микросхем ПЗУ и ОЗУ удобнее разделить его на 16 сегментов по 4 Кбайта.

С нулевого адреса рекомендуется размещать ПЗУ, а далее – ОЗУ. Распределение адресного пространства памяти для разрабатываемого одноплатного микроконтроллера приведено в таблице 2.1.

Таблица 2.1 – Распределение адресного пространства памяти

Диапазон адресов	Микросхема памяти	Общий объем
0000h – 0FFFh	ПЗУ 1	4 Кбайт
1000h – 1FFFh	Не используется	4 Кбайт
2000h – 3FFFh	ОЗУ 1	16 Кбайт
4000h – 5FFFh	ОЗУ 2	

Разработка схемы дешифратора адреса.

Блок памяти МПС должен содержать дешифратор старших адресов, который и определяет конфигурацию адресного пространства памяти микросистемы. Для дешифрации трех микросхем памяти достаточно двух адресных линий A13, A14 (т. к. двухразрядное число в двоичной системе счисления может быть представлено четырьмя наборами). Старший разряд A15 будет использован для выбора блока памяти. Работа дешифратора будет разрешена при низком уровне сигнала A15, так что блок памяти занимает адреса в диапазоне от 0000h до 5FFFh (24 Кбайта), из которых занято только 20 Кбайт, что соответствует карте памяти, приведенной в таблице 2.1.

Дешифратор адресов можно построить следующим образом:

- с использованием микросхем логических элементов;
- с использованием микросхем дешифраторов;

– с использованием микросхем ЗУ.

Наиболее оптимальным в данном случае является второй способ, но при этом к нему предъявляются следующие требования:

- дешифратор должен иметь дополнительные входы разрешения работы или включения кристалла;
- выходы дешифратора должны быть инверсными, чтобы обеспечивать активный уровень низкого уровня, что необходимо для входов выборки кристаллов микросхем памяти.

Данным требованиям удовлетворяет микросхема К555ИД7, представляющая собой дешифратор «с 3 на 8» с инверсными выходами. Условно-графическое обозначение приведено на рисунке 2.8. ИМС имеет три входа разрешения работы, причем один из них инверсный.

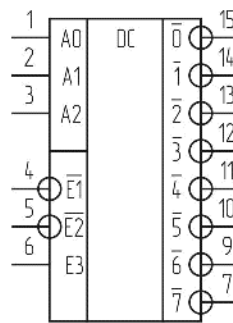


Рисунок 2.8 – Условно-графическое обозначение ИМС К555ИД7

Управление режимами работы модуля памяти (чтение и запись) в микропроцессорном ядре на базе i8080 осуществляется сигналами $\overline{MRDC}$ и $\overline{AMWC}$, которые формирует системный контроллер КР580ВК38.

В результате проведенного анализа получена схема электрическая дешифратора адреса, приведенная на рисунке 2.9.

Уровень логического нуля на линии A15 разрешает работу дешифратора блока памяти.

Сигнал $\overline{MRDC}$ разрешает чтение микросхем ОЗУ и ПЗУ, а сигнал $\overline{AMWC}$ – запись в микросхемы ОЗУ.

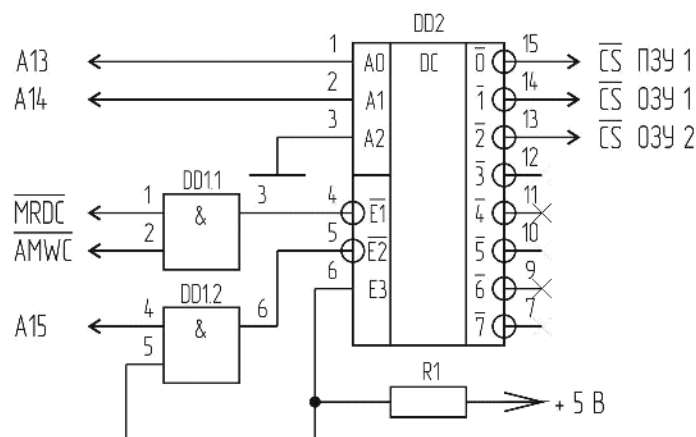


Рисунок 2.9 – Схема электрическая дешифратора адреса области памяти одноплатного микроконтроллера

В качестве логических элементов выбраны:

– ИМС элементов «И» – 1533ЛИ1.

Выбор серии обоснован тем, что ИМС дешифратора и ИМС памяти имеют одно и то же напряжение питания. В пределах этих типов логик микросхемы согласованы между собой по электрическим параметрам.

Потенциал логической единицы необходимо подавать через ограничивающий резистор. Сопротивление резистора рекомендовано фирмой-производителем дешифратора и составляет 100 кОм. Мощность, рассеиваемая на резисторе, рассчитывается по формуле (2.1), где в качестве тока через резистор принят ток допустимый ток дешифратор.

По результатам расчетов выбирается резистор необходимого тип номинала.

Разработка схемы электрической принципиальной модуля памяти одноплатного микроконтроллера.

В результате проведенного анализа и расчетов разработана схема электрическая принципиальная памяти одноплатного микроконтроллера, приведенная на рисунке 2.10.

Алгоритм выбора тип номиналов резисторов R3 и R4 для ИМС ОЗУ аналогичен рассуждениям, использованным при выборе резисторов R1 и R2

для ИМС дешифратора и ИМС ПЗУ.

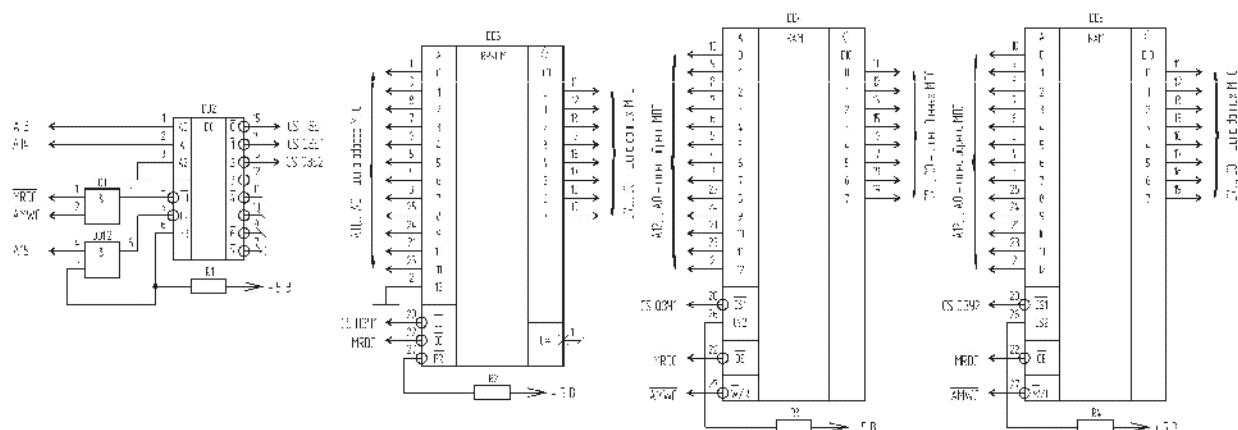


Рисунок 2.10 – Схема электрическая принципиальная памяти одноплатного микроконтроллера

В результате выполнения данного пункта был разработан модуль памяти одноплатного микроконтроллера на основе процессора i8080.

Оборудование и материалы

Практическая работа выполняется с использованием ПК и любого программного обеспечения с функциями САПР для выполнения схемы электрической принципиальной памяти одноплатного микроконтроллера.

Указания по технике безопасности

Перед проведением практического занятия следует изучить инструкцию по технике безопасности при работе на компьютере. Также следует соблюдать правила и требования, установленные для данного компьютерного класса.

Задания (задачи)

1 Разработать модуль памяти, состоящий из микросхем ОЗУ, для чего необходимо:

- выбрать задание из таблицы 2.2 в соответствии с вариантом, номер которого задается преподавателем;
- провести анализ требований и выбрать одну или несколько ИМС ОЗУ в соответствии с требованиями задания;
- разработать схему включения ОЗУ в модуле памяти;
- предъявить требования к управляющим сигналам, а также к линиям адресной шины и шины данных.

Таблица 2.2

№ варианта	1	2	3	4	5	6	7	8	9
Объем памяти, Кбайт	10	6	8	4	2	32	16	64	48

Продолжение таблицы 2.2

№ варианта	10	11	12	13	14	15	16	17	18
Объем памяти, Кбайт	40	20	24	36	56	12	34	14	18

2 Разработать модуль памяти, состоящий из микросхем ПЗУ, для чего необходимо:

- выбрать задание из таблицы 2.2 в соответствии с вариантом, номер которого задается преподавателем;
- провести анализ требований и выбрать одну или несколько ИМС ПЗУ в соответствии с требованиями задания;
- разработать схему включения ПЗУ в модуле памяти;
- предъявить требования к управляющим сигналам, а также к линиям адресной шины и шины данных.

3 Разработать карту распределения памяти, для чего необходимо:

- провести анализ микропроцессора, который был задан вариантом в практическом задании 1;
- выполнить разработку карты распределения памяти в соответствии с ранее заданным объемом ОЗУ и ПЗУ.

4 Разработать схему дешифратора адреса, для чего необходимо:

- провести анализ микропроцессора, который был задан вариантом в практическом задании 1;
- провести анализ разработанного ранее модуля памяти;
- провести анализ разработанной ранее карты распределения памяти;
- выбрать элементную базу реализации дешифратора;
- разработать схему дешифратора адреса;
- предъявить требования к управляющим сигналам, а также к линиям адресной шины и шины данных.

5 Разработать схему электрическую принципиальную модуля памяти одноплатного микроконтроллера, для чего необходимо:

- провести анализ разработанного ранее модуля памяти;
- провести анализ разработанной ранее карты распределения памяти;
- провести анализ разработанной схемы дешифратора адреса;
- разработать схему электрическую принципиальную модуля памяти одноплатного микроконтроллера;
- провести расчет электрических параметров основных и дополнительных компонентов, необходимых для функционирования схемы. Выбрать необходимые компоненты, пользуясь справочной литературой.

Содержание отчета

Отчет по практическому занятию выполняется на листах формат А4 согласно требованиям ГОСТ и ЕСКД и должен содержать следующие

пункты:

- а) титульный лист с названием работы, дисциплины, кода группы, ФИО исполнителя и преподавателя;
- б) цель и содержание работы;
- в) краткое теоретическое введение;
- г) условия, ход выполнения и решения задач;
- д) общую схему электрическую принципиальную модуля памяти одноплатного микроконтроллера;
- е) выводы по результатам выполнения практического задания.

Список литературы, рекомендованный к использованию по данной теме

Основная учебная литература

1 Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд., испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

Дополнительная учебная литература

1 Бродин В.Б., Калинин А.В. Системы на микроконтроллерах и БИС программируемой логики – М.: Издательство ЭКОМ, 2002. – 400 с.

2 Боборыкин А. В., Липовицкий Г.П. и др. Однокристалльные микроЭВМ. – М.: МИКАП, 1994 – 400с.

3 Сташин В. В., Урусов А. В., Мологонцева О. Ф. Проектирование цифровых устройств на однокристалльных микроконтроллерах – М.: Энергоатомиздат, 1990 – 224с.

4 Шевкопляс Б.С. Микропроцессорные структуры. Инженерные решения: Справочник. - М.: Радио и связь, 1990 – 512 с.

5 Щелкунов, Н. Н. Микропроцессорные средства и системы /

Н. Н. Щелкунов, А. П. Дианов. – М. : Радио и связь, 1989. – 288 с.

6 Коффрон, Дж. Технические средства микропроцессорных систем: практический курс / Дж. Коффрон. Пер. с англ. – М. : Мир, 1983. – 344 с.

7 Шахнов, В. А. Микропроцессоры и микропроцессорные комплекты интегральных микросхем : справочник / В.-Б. Б. Абрайтис, Н. Н. Аверьянов, А. И. Белоус [и др.]: под ред. В. А. Шахнова. – Том 1. – М. : Радио и связь, 1988. – 368 с.

8 Соппа, И. В. Микропроцессоры и микроконтроллеры / И. В. Соппа, А. С. Самардак, В. А. Иванов. – Владивосток : Издательство Дальневосточного университета, 2009. – 211 с.

9 Алексенко, А. Г. Основы микросхемотехники / А. Г. Алексенко. 3-е изд. перераб. и доп. – М. : БИНОМ. Лаборатория знаний, 2009. – 448 с.

10 Петровский И.И. и др. Логические ИС К1533, К1554: Справочник / В двух частях. - М.: ТОО “БИНОМ”, 1993 – 500 с.

11 Лебедев О.Н. Микросхемы памяти и их применение. – М.: Радио и связь, 1990 – 303 с.

12 Якубовский, С. В. Аналоговые и цифровые интегральные микросхемы : справочное пособие / С. В. Якубовский, Н. А. Барканов, Л. И. Ниссельсон [и др.]; под ред. С. В. Якубовского. – 2-е изд., перераб. и доп. – М. : Радио и связь, 1984. – 432 с.

13 Цифровые интегральные микросхемы : справочник / М. И. Богданович, И. Н. Грель, С. А. Дубина [и др.]. – 2-е изд., перераб. и доп. – Минск : Беларусь, Польша, 1996. – 605с.

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ 3

Тема: Разработка портов ввода/вывода для одноплатного микроконтроллера на основе регистров. Разработка портов ввода/вывода для одноплатного микроконтроллера на основе программируемого периферийного адаптера. Разработка схемы дешифратора адреса для портов ввода/вывода.

Цель работы: закрепление знаний, полученных на лекциях по разделу: «Проектирование микропроцессорных систем». В ходе выполнения заданий практического занятия необходимо:

- провести анализ типа исполнения порта ввода/вывода и его разрядности;
- осуществить выбор основных ИМС для реализации порта ввода/вывода;
- разработать схемы их подключения к магистралям микропроцессорного ядра;
- осуществить выбор дополнительных ИМС для реализации порта ввода/вывода;
- на основе временных диаграмм пояснить работу схемы электрической принципиальной разработанного порта в составе разрабатываемого одноплатного микроконтроллера;
- разработать схему дешифратора адреса согласно варианту задания.

Знания, приобретаемые студентом в результате освоения темы:

- классификация микропроцессоров и их основные характеристики;
- аппаратная и логическая структуры типовых микропроцессоров;
- режимы адресации и система команд;
- способы организации ввода/вывода в микропроцессорных системах.

Умения, приобретаемые студентом в результате освоения темы:

- разрабатывать микропроцессорные системы для решения задач управления, автоматизации, сбора и обработки данных и других целей.

Формируемые компетенции: ОК-6, ОК-7, ОПК-3, ОПК-4, ОПК-6, ОПК-7, ОПК-8, ОПК-9, ПК-1, ПК-3, ПК-5, ПК-6.

Актуальность темы: тема является актуальной, так как все микропроцессорные устройства имеют какой-либо интерфейс связи с внешними устройствами. Поэтому студент должен знать основные виды портов ввода/вывода, уметь синтезировать модуль ввода/вывода и дешифратор адреса для него.

Теоретическая часть

Подсистема ввода/вывода (ВВ) – один из трех компонентов МС, ответственный за связь с периферийными устройствами (ПУ), а в некоторых случаях и за организацию самих ПУ.

С точки зрения программиста, работающего на командном уровне, систему ВВ можно представить в виде пространства ВВ *IOSEG* и ряда команд ВВ, имеющих к нему доступ. Пространство *IOSEG* обычно организовано в виде набора *n*-разрядных ячеек – портов, каждый из которых может быть адресован независимо от других. Типичная разрядность порта для большинства МС равна восьми. В некоторых системах допускается объединение двух и более элементарных портов в многобайтовые, доступ к которым реализуется как к единому целому.

Подобно пространству памяти пространство ВВ, как правило, линейно упорядочено. Каждому элементарному порту поставлен в соответствие его адрес. Множество адресов занимает целочисленный диапазон от 0 до 2^P-1 . Правила организации и адресации многобайтовых портов могут соответствовать аналогичным правилам размещения и адресации многобайтовых объектов в памяти МС.

В качестве примера рассмотрим систему ВВ для МС на базе МП *i8080*. Пространство *IOSEG* такой системы организовано в виде 256 8-разрядных

портов ВВ с линейно упорядоченными адресами от 0 до 2^8-1 . В распоряжение пользователя предоставляются две команды ВВ, содержащиеся в системе команд:

IN port ; A ← port

OUT port ; port $\leftarrow$ A

В наиболее простом виде процедура ввода или вывода выполняется независимо от состояния ПУ. Такой вид обмена назван прямым или безусловным. Процедуры прямого ВВ в чистом виде возможны только при управлении всегда готовыми к обмену простейшими ПУ. К тому же они являются составными элементами более сложных процедур программно-управляемого обмена, к числу которых относится условный ВВ.

Простейшая процедура ввода информации из подсистемы ВВ связана с чтением отдельного порта пространства *IOSEG*. С этой целью физические линии подсистемы ВВ упаковываются в байтовые и многобайтовые наборы, состояние каждого из которых может быть передано на шину данных МС во время цикла чтения порта ВВ. Физическое управление вводом состояния конкретного набора лучше всего осуществлять с помощью шинного формирователя с тремя состояниями, открываемого стробом $\overline{IORC}$ при условии его выбора. Выбор или адресация формирователей возлагается на специальную декодирующую схему, называемую логикой выборки кристаллов, которая, в свою очередь, управляется сигналами шины адреса и при необходимости шины управления, ответственными за кодирование размера передаваемых данных: байта, слова, двойного слова.

Запись данных в порт соответствует их выводу в подсистему ВВ. В простейшем варианте порта вывода данные фиксируются в регистре по стробу $\overline{IOWC}$ или $\overline{AIOWC}$ при условии его выбора. Роль адресации регистра выполняет логика выборки кристаллов. При использовании строба $\overline{IOWC}$ данные могут фиксироваться его низким уровнем напряжения, при стробе $\overline{AIOWC}$ они должны восприниматься только по его фронту.

Быстродействия шинных формирователей и регистровых схем обычно достаточно, чтобы поддержать обмен с максимальной для ЦП скоростью (синхронный режим работы системной магистрали).

Для поддержки двустороннего ВВ очень широко применяется псевдодвунаправленный порт (рисунок 3.1).

В данной схеме к выходам *DO* подключены буферы с открытым коллектором, допускающие объединение с линиями ввода внешних данных по схеме «монтажное ИЛИ». При этом нагрузочные резисторы могут быть встроены в порт. Установка на шине порта вывода напряжения высокого уровня обеспечивает ввод внешних данных без каких-либо искажений. При необходимости некоторые линии ввода могут быть выборочно замаскированы установлением на соответствующих им линиях *DO* напряжения низкого уровня. Допускается в качестве входных линий использовать лишь часть порта, тогда как оставшаяся часть может выполнять роль выходной шины. В целом схема, приведенная на рисунке 3.1, достаточно универсальна, что объясняет ее широкое распространение, например, в однокристальных микроЭВМ. Она и будет реализована в данном примере.

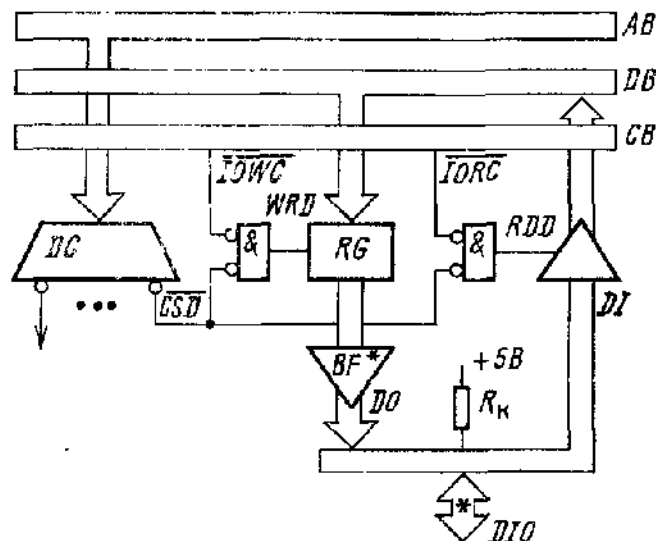


Рисунок 3.1 – Схема псевдодвунаправленного порта

Рассмотрим пример разработки:

- портов ввода/вывода для одноплатного микроконтроллера на основе регистров;
- портов ввода/вывода для одноплатного микроконтроллера на основе программируемого периферийного адаптера;
- схемы дешифратора адреса для портов ввода/вывода.

Выбор ИМС для реализации порта ввода/вывода. Разработка схемы электрической принципиальной подключения к магистралям микропроцессорного ядра.

МП, рассматриваемый в примерах практических занятий, – *i8080*. Он имеет восьмиразрядную шину данных. Наиболее просто реализуется порт ввода/вывода именно такой разрядности. Рассмотрим далее более сложный случай, когда необходимо организовать порт разрядностью 16 бит.

Наиболее просто реализуется порт на специализированной микросхеме комплекта *i8080* – KP580BB55/BB55A, режим работы которой устанавливается программным путем. Более интересный и сложный случай представляет реализация порта на многорежимном буферном регистре (МБР). Оба варианта реализации будут рассмотрены далее.

Разработка порта ввода/вывода на основе микросхем многорежимного буферного регистра.

Пусть, согласно заданию, необходимо разработать порт ввода/вывода для одноплатного микроконтроллера, имеющий следующие характеристики:

- параллельный порт ввода/вывода – регистровый;
- способ подключения порта ввода/вывода к системной магистрали – внешнее устройство;
- разрядность порта – 16 бит;
- базовый адрес порта ввода/вывода – $27h$.

В качестве элементной базы порта необходимо использовать микросхему регистра. Обращение к регистрам порта должно осуществляться

специальными командами работы с внешними устройствами – $\overline{IORC}$ и $\overline{IOWC}$ ($\overline{AIOWC}$).

Т. к. данные, передаваемые/читаемые из устройства ввода/вывода являются 16-разрядными, а шина данных МП *i8080* является 8-разрядной, то работу с таким портом необходимо осуществлять побайтно. Согласно требованию самого МП в линейной памяти микросистемы слова должны располагаться в двух соседних байтах. Старшая часть слова занимает байт со старшим адресом, а младшая – байт с младшим адресом. При этом адрес младшего байта служим адресом слова. Эту особенность необходимо учесть при построении дешифратора адреса порта: сначала необходимо считывать младший байт, потом старший байт. То же относится и к операции записи в порт. Таким образом, регистры параллельного порта ввода/вывода должны быть 8-разрядными и их необходимо два, работающих на ввод данных, и два – на вывод данных.

Указанным требованиям соответствует широко используемая схема псевдодвунаправленного порта, приведенная ранее на рисунке 3.1.

Если в качестве буфера ввода также использовать регистр, а также применить микросхемы регистров, имеющих выводы, способные переключаться в высокоомное состояние, то схема порта значительно упрощается: необходимо два 8-разрядных регистра для вывода 16-разрядных данных и два 8-разрядных регистра для их ввода.

В схеме на рисунке 3.1 *DC* – это дешифратор адреса, включающий порт в работу только тогда, когда на шине адреса МПС установлен адрес именно данного порта.

Требования к микросхемам регистров:

- согласованность по электрическим и временным параметрам с микропроцессором *i8080*;
- разрядность каждого регистра – 8 бит;
- возможность перевода выходов в высокоомное состояние.

Указанным требованиям соответствует микросхема МБР КР580ИР82 (см. рисунок 1.12 практического занятия № 1), использованная ранее для буферизации шины адреса МП. Временные диаграммы работы МБР показаны в том же практическом занятии (см. рисунок 1.13).

Нагрузочная способность регистра составляет величины до:

- I_{OL} , равного 32 мА;
- C_L , равного 300 пФ,

что необходимо учесть при подключении к разрабатываемому параллельному порту внешнего устройства.

Сначала разработаем часть порта, отвечающую за вывод данных из МПС во внешнее устройство, для чего рассмотрим временные диаграммы цикла записи МП *i8080* в порт (см. рисунок 1.17 практического занятия № 1).

Запись слова в порт будет производиться в два этапа:

- запись младшего байта;
- запись старшего байта.

Каждый байт должен быть «защелкнут» в своем собственном регистре вывода. Так как информация на линиях порта будет верна только после того, как будет записан второй (старший) байт данных, то необходимо запретить формирование выходных данных регистров до этого момента. Кроме того, порт вывода должен самостоятельно разделять приходящие младший и старший байты и записывать их в соответствующие регистры.

Анализируя диаграммы можно сделать вывод, что используя сигналы $\overline{IORC}$ и $\overline{IOWC}$, можно выполнять переключение между регистрами младшего и старшего байтов порта ввода/вывода. Чтобы исключить влияние этого сигнала на данный порт, в случае команды обращения к другим портам или в иных ситуациях, лучше использовать сигнал CS (разрешение работы) данного порта. При записи младшего байта этот сигнал будет сформирован один раз, при записи старшего – второй. В качестве устройства для хранения количества поступлений сигнала CS лучше всего использовать T -триггер.

Рисунок 3.2 – Схема электрическая принципиальная блока формирования команд записи в порт (а) и порта вывода (б)

Временные диаграммы, поясняющие работу схемы, приведены на рисунке 3.3.

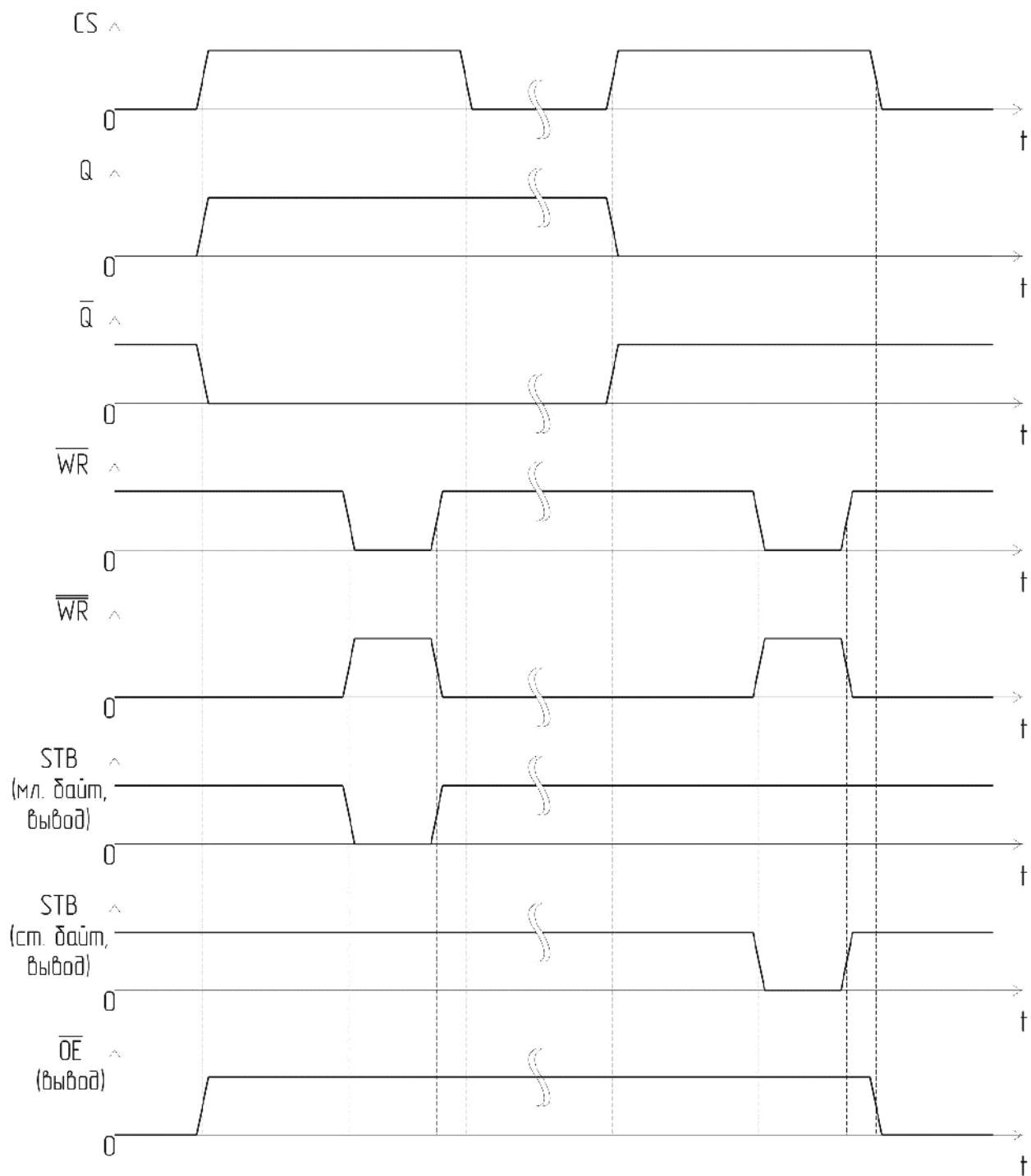


Рисунок 3.3 – Временные диаграммы работы блока формирования команд записи в порт и порта вывода

Для реализации устройства выбраны следующие микросхемы:

- ИМС триггера – 1533ТМ2;
- ИМС элементов «И-НЕ» – 1533ЛА4;
- ИМС инверторов – КР1533ЛН1;
- ИМС элементов «ИЛИ» – К555ЛЛ1.

ИМС 1533ТМ2 представляет собой D -триггер, включенный по схеме T -триггера. Вход $\bar{S}$ не используется в работе, поэтому на него подан пассивный сигнал через ограничивающий резистор. Расчет его параметров полностью аналогичен расчету такого же по функции резистора для ИМС ОЗУ. Выбирается резистор соответствующего тип номинала.

Вход триггера $\bar{R}$ предназначен для формирования сигнала сброса. При включении МПС необходимо выполнить установку триггера в начальное нулевое состояние во избежание нарушения его работы. Проще всего это сделать с помощью цепи сброса самого МП, что достигается подачей на вход триггера импульса с системы сброса микропроцессора (см. рисунок 1.16 практического занятия № 1).

Буферные элементы необходимы для организации псевдодвунаправленного порта, как было указано ранее (см. рисунок 3.1). Требования к буферным элементам:

- выходы должны быть типа «открытый коллектор», допускающие объединение с линиями ввода по схеме «монтажное ИЛИ». При этом нагрузочные резисторы будут встроенными в порт. Установка на шине порта вывода напряжения высокого уровня обеспечит ввод внешних данных без каких-либо искажений;
- согласованность по электрическим и временным параметрам с микропроцессором *i8080*;
- разрядность – 16 бит.

Выбрана ИМС *SN74ALS760* фирмы *Texas Instruments*, состоящая из восьми буферных элементов в одном корпусе с выходом типа «открытый коллектор» и возможностью перевода в третье высокоомное состояние.

Перевод в третье состояние будет осуществляться тем же сигналом, что и регистры порта вывода.

Для реализации псевдодвунаправленного порта и нормального функционирования ИМС *SN74ALS760* требуются подтягивающие резисторы. Для обеспечения высокой нагрузочной способности их сопротивление необходимо выбрать так, чтобы обеспечить выходной допустимый ток буферных элементов. Он составляет значение 24 мА, что несколько ниже, чем давали регистры порта, но вполне достаточно.

Сопротивление каждого резистора может быть рассчитано по закону Ома:

$$R = \frac{U_{\text{ПИТ.}}}{I_{\text{ДОП. БУФ.}}} . \quad (3.1)$$

Мощность, рассеиваемая на каждом резисторе, рассчитывается по формуле (1.2) (см. практическое занятие № 1).

По результатам расчетов выбираются резисторы необходимого тип номинала.

Далее разработаем часть порта, отвечающую за ввод данных из внешнего устройства в МПС. Чтение слова в МПС будет производиться в два этапа: сначала будет считываться младший байт, затем – старший. Каждый байт должен быть «защелкнут» в своем собственном регистре ввода, что можно сделать одновременно. Кроме того, порт ввода должен самостоятельно разделять младший и старший байты и передавать их на шину данных МПС. Для реализации такого алгоритма обратимся к временным диаграммам цикла чтения МП *i8080* из порта (см. рисунок 1.17 практического занятия № 1).

Анализируя диаграммы можно сделать вывод, что сигналы $\overline{IORC}$, $\overline{IOWC}$ и *CS* данного порта можно использовать так же, как и в части порта вывода. В качестве устройства для хранения количества поступлений сигнала *CS* лучше всего использовать *T*-триггер. Поскольку чтение и запись данных

выполняются разными сигналами, то один и тот же триггер будет использован как для порта вывода, так и порта ввода.

Разрешение чтения на шину данных МПС, согласно диаграмме рисунка 1.17, а, необходимо производить по сигналу чтения $DBIN$ (сигнал берется непосредственно с выхода $DBIN$ МП i8080).

Формировать сигнал фиксации данных линий порта в регистрах можно, используя сигнал $\overline{OE}_{ВЫВ}$ (см. рисунок 3.3). При первом обращении к порту данные с линий внешнего устройства будут «защелкнуты» в регистрах ввода.

Формировать сигналы чтения данных из регистров ввода на шину данных МПС можно лишь при совпадении условия активности сигналов:

- CS с дешифратора;
- выбора младшего (старшего) байта слова данных;
- команды $DBIN$.

Данное условие можно реализовать с помощью логического элемента «И».

В результате проведенного анализа разработана схема электрическая принципиальная блока формирования команд чтения из порта ввода на шину данных МПС, приведенная на рисунке 3.4. Временные диаграммы, поясняющие работу схемы, приведены на рисунке 3.5.

Для реализации устройства выбраны следующие микросхемы:

- ИМС элементов «И-НЕ» – 1533ЛА4.

Подача сигнала $\overline{OE}_{ВЫВ}$ на буферные элементы порта вывода переводит их в высокоомное состояние, что позволяет считывать данные линий внешнего устройства без каких-либо искажений.

Разработка порта ввода/вывода на основе микросхемы программируемого периферийного адаптера KP580BB55.

Пусть согласно заданию, необходимо разработать порт ввода/вывода для одноплатного микроконтроллера, имеющий следующие характеристики:

- реализация параллельного порта ввода/вывода – KP580BB55;

- способ подключения порта ввода/вывода к системной магистрали – внешнее устройство;
- разрядность порта – 16 бит;
- базовый адрес порта ввода/вывода – $38h$.

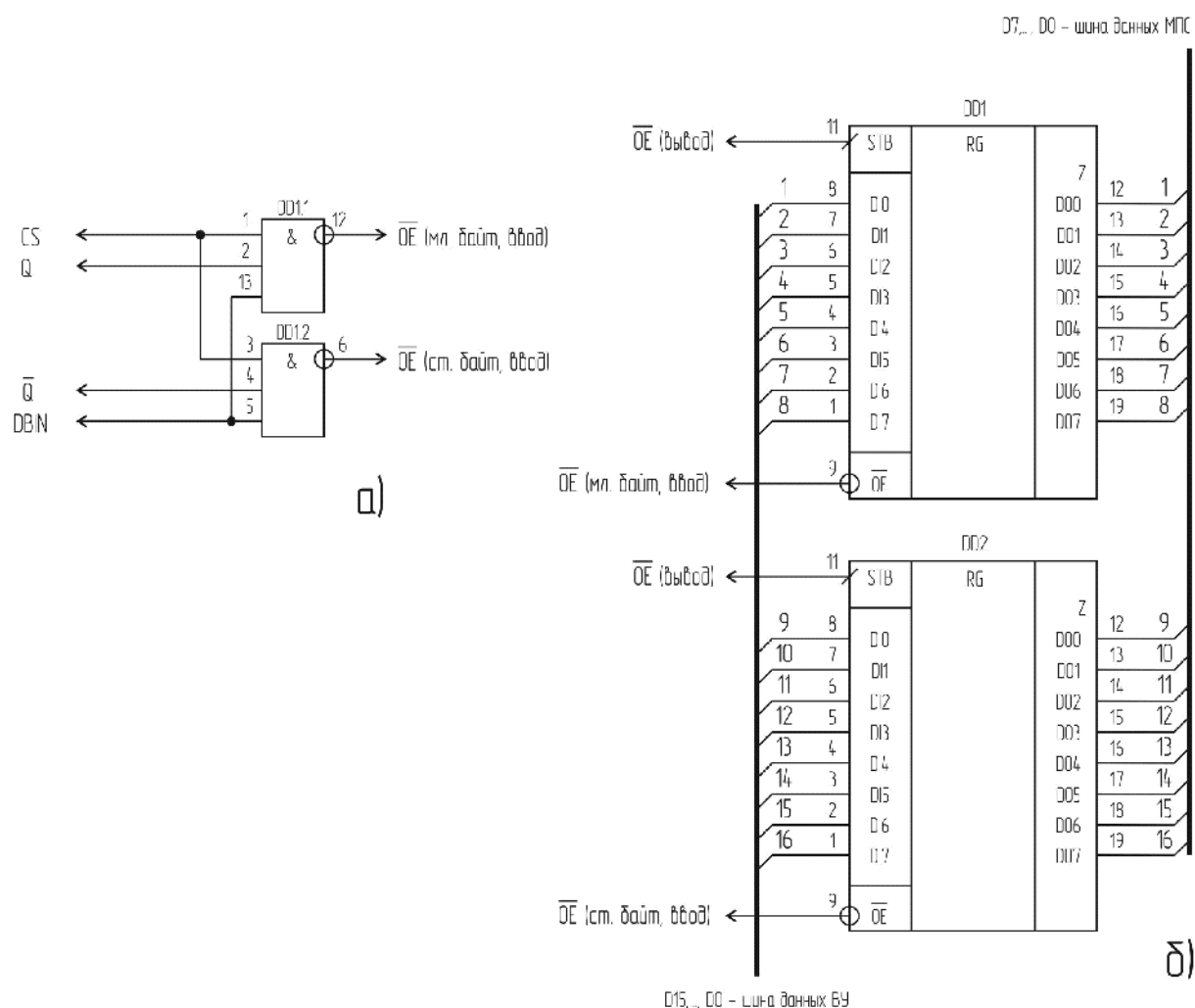


Рисунок 3.4 – Схема электрическая принципиальная блока формирования команд чтения из порта ввода на шину данных МПС (а) и порта ввода (б)

Программируемый периферийный адаптер (ППА) применяется в случаях, когда логика обмена заранее не известна или характеристики процедур обмена во время работы должны меняться. В состав ППА входят

программно-доступные регистры управляющих слов CW , которые и определяют режим работы адаптера. Программный доступ к управляющему регистру со стороны ЦП дает возможность оперативно управлять работой ППА и изменять характеристики интерфейса.

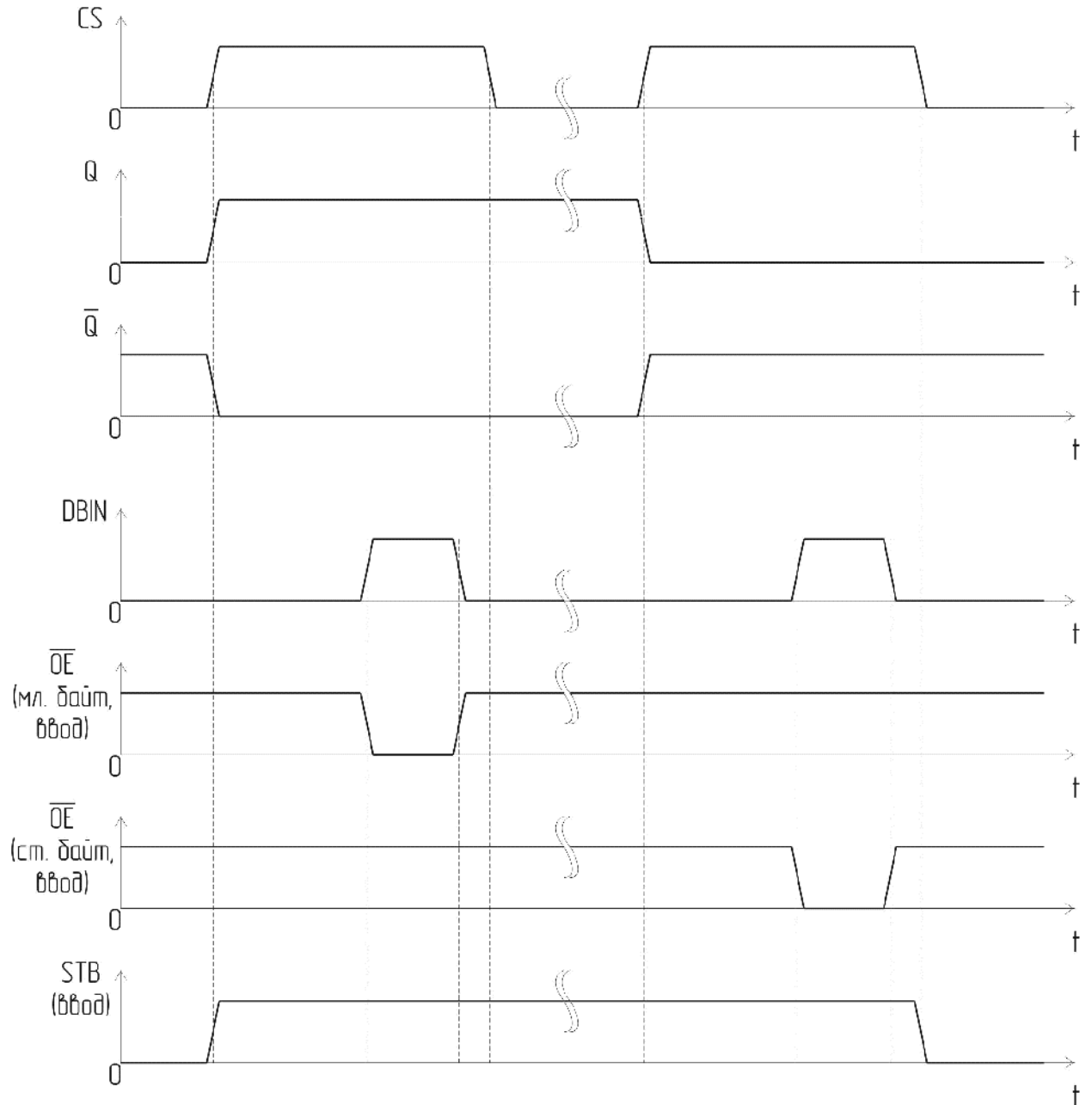


Рисунок 3.5 – Временные диаграммы работы блока формирования команд чтения из порта ввода на шину данных МПС

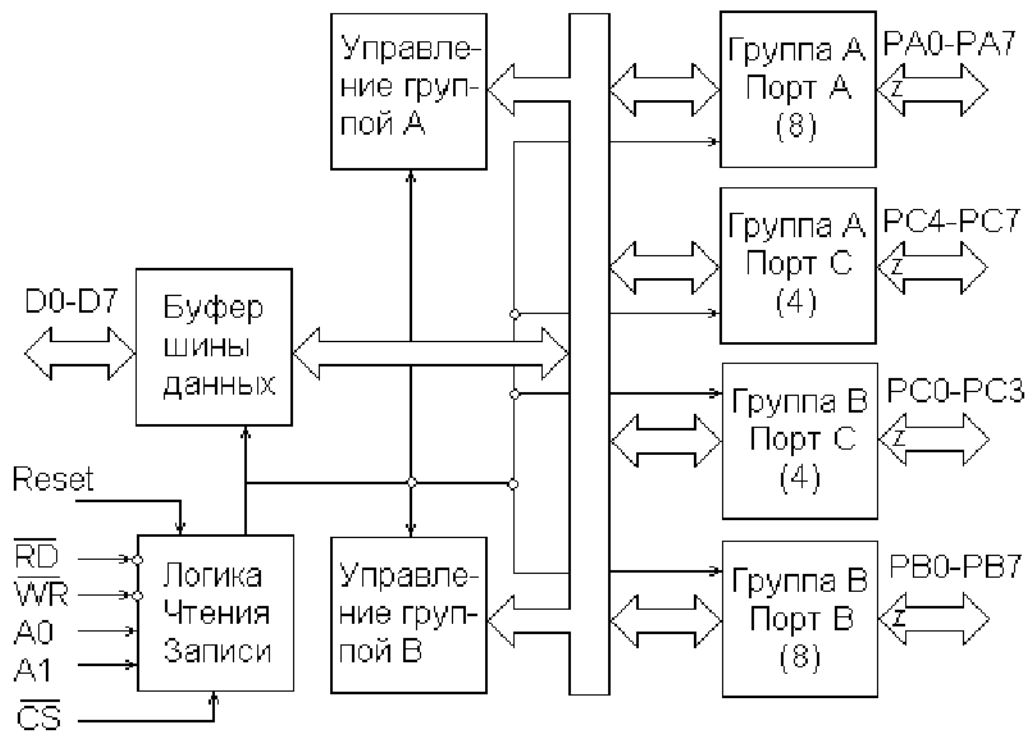
К числу ППА относят и ИМС КР580ВВ55 – это однокристальное

программируемое устройство параллельного ВВ информации произвольного формата. В состав осуществляемых им процедур входит параллельный обмен данными с квитированием или без него как в режиме программного управления, так и по прерываниям. При этом обеспечивается организация не только однонаправленного, но и двунаправленного ВВ. Определение и переопределение типа выполняется программными методами с помощью специализированных процедур инициализации.

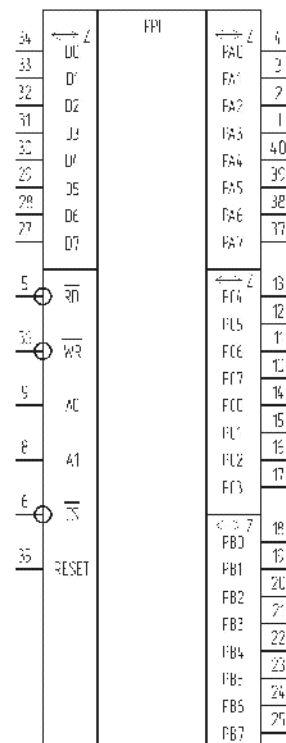
В состав ВВ55 (рисунок 3.6) входят три двунаправленных 8-разрядных порта, разделенных на две группы, два устройства управления группами портов и периферийная логика для согласования с системной шиной. Порты содержат буферные регистры и шинные формирователи с тремя состояниями. Схема управления содержит регистр управляющего слова CW , доступный только для записи.

Обмен информацией между МП и внутренними регистрами ВВ55 осуществляется через двунаправленный шинный формирователь и управляется сигналами $\overline{CS}$, $A0$, $A1$, $\overline{RD}$, $\overline{WR}$, согласно таблице 3.1. Адресные сигналы $A0$, $A1$ выбирают один из внутренних регистров, стробы $\overline{RD}$ и $\overline{WR}$ управляют направлением передачи, а сигнал $\overline{CS}$ определяет доступ к микросхеме.

Вход $RESET$ служит для аппаратного сброса микросхемы в исходное состояние. Все регистры ВВ55, включая регистр управляющего слова CW , устанавливаются в состояние логического «0». Сброс CW соответствует переводу всех портов в режим прямого ввода без квитирования.



а)



б)

Рисунок 3.6 – KP580BB55:

а – структурная схема;

б – условно-графическое обозначение

Таблица 3.1 – Сигналы управления ВВ55 и выполняемые операции

$A1$	$A0$	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	Операция
0	0	0	1	0	Чтение порта A
0	1	0	1	0	Чтение порта B
1	0	0	1	0	Чтение порта C
1	1	0	1	0	Недопустимо
0	0	1	0	0	Запись в порт A
0	1	1	0	0	Запись в порт B
1	0	1	0	0	Запись в порт C
1	1	1	0	0	Запись управляющего слова
x	x	1	1	0	Нет операции
x	x	x	x	1	Нет операции

Дальнейшая настройка микросхемы ВВ55 осуществляется программно с помощью управляющего слова *MS (Mode Selection)*, которое назначает режим работы каждому порту. Эти режимы могут быть изменены в любое время.

Адаптер поддерживает три режима работы портов:

- «режим 0» – однонаправленный ВВ без квитирования (применим к любому из трех портов);
- «режим 1» – однонаправленный ввод-вывод с квитированием (применим к портам A и B);
- «режим 2» – двунаправленный ВВ (допускается только для порта A).

Для разрабатываемого одноплатного микроконтроллера выбран «режим 0». В данном режиме осуществляется прямой однонаправленный ВВ через любой из трех портов без каких-либо сигналов сопровождения. Данные вводятся или выводятся через выбранный канал в соответствии с временными диаграммами, приведенными на рисунке 3.7. В данном режиме

интерфейс можно представить как набор параллельных линий ВВ, организованных в две байтовые и две 4-разрядные шины, причем каждая может быть применена либо для ввода, либо для вывода независимо от других. Входная информация адаптером не запоминается и читается при низком уровне напряжения сигнала на входе $\overline{RD}$. Выходная информация защелкивается в выходной буферный регистр выбранного порта по срезу системного сигнала $\overline{WR}$ и остается на выходе порта до нового цикла вывода или изменения режима.

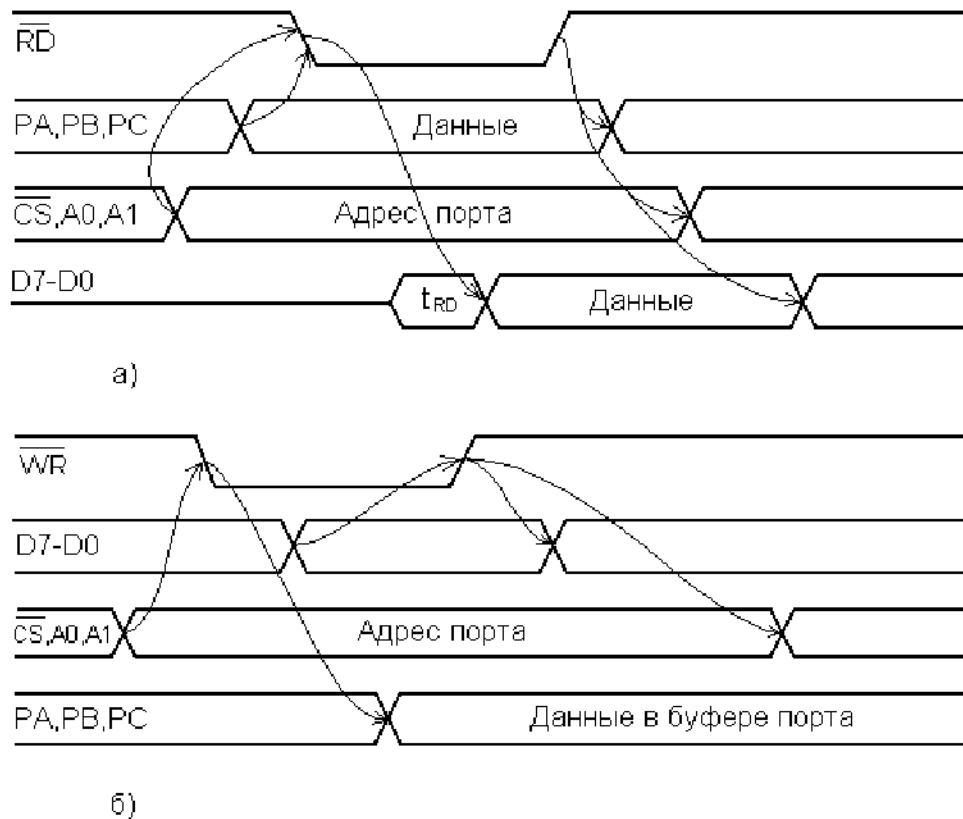


Рисунок 3.7 – Временные диаграммы работы BB55:

- а – ввод в «режиме 0»;
- б – вывод в «режиме 0»

Порты A , B и C для работы в «режиме 0» программируются независимо друг от друга. Выбраны порты A и B для организации требуемого параллельного интерфейса. Они будут настраиваться в «режиме 0» либо оба

на ввод, либо оба на вывод, что позволит организовать обмен данными с 16-разрядным ВУ.

Обращение к порту будет осуществляться специальными командами работы с внешними устройствами – $\overline{IORC}$ и $\overline{AIOWC}$ (при разработке процессорного ядра должен был быть выбран системный контроллер ВК38. Это же было сказано при разработке модуля ввода/вывода на основе регистров МБР).

Согласно требованию самого МП в линейной памяти микросистемы слова должны располагаться в двух соседних байтах. Старшая часть слова занимает байт со старшим адресом, а младшая – байт с младшим адресом. При этом адрес младшего байта служим адресом слова. Эту особенность необходимо учесть при построении схемы порта: сначала необходимо считывать младший байт, выбирая порт A , потом старший байт, выбирая порт B . То же относится и к операции записи в порт $ВВ$.

Обмен данными с ВУ должен производиться через одни и те же линии. Это позволяет сделать сам ППА ВВ55. Учитывая его особенности при приеме данных, передаче и сбросе, адаптер будет работать так:

- после сброса все порты установятся в «режим 0» на ввод;
- при выводе слова оно будет защелкиваться побайтно в выходных буферных регистрах, сначала порта A , затем порта B ;
- при вводе слова сначала будет считываться информация с порта A , затем из порта B .

Возможна ситуация, когда ВВ55 настроен на вывод и формирует на своих выходах заданное слово. В этот момент ВУ устанавливает на шину данных свое слово. Это может вывести из строя выходы порта ППА, если значения сигналов будут различными. Чтобы этого избежать, необходимо применить буферизацию шины, позволяющую разделить потоки данных на разъеме ВУ: информацию на вывод пропускать через одно устройство, информацию на ввод – через другое. Такое разделение можно осуществить с

помощью буферных элементов. Кроме того, применение последних позволит повысить нагрузочную способность порта ВВ МПС. Буфер для вывода информации должен иметь выходы типа «открытый коллектор», допускающие объединение с линиями ввода по схеме «монтажное ИЛИ». При этом нагрузочные резисторы будут встроенными в порт. Установка на шине порта вывода напряжения высокого уровня обеспечит ввод внешних данных без каких-либо искажений.

Также буферы должны иметь возможность установки их выходов в высокоомное состояние, что позволит включать только буферы вывода при записи в ВУ, или включать буферы ввода при чтении из ВУ.

Разделение по направлению удобнее всего выполнить, используя сигналы программирования самого ВВ55. С учетом таблицы 3.1, можно сделать вывод, что комбинация сигналов

$$A1 = A0 = 1$$

выполняет установку режима работы каждого порта адаптера. Чтобы не происходило срабатывания логики управления порта ВВ, необходимо отслеживать состояние сигнала включения, который будет выдаваться дешифратором выбора УВВ (см. рисунок 3.1).

Разделение по режимам чтение/запись удобно проводить с помощью сигналов системного контроллера: $\overline{IORC}$ – чтение порта ВВ и $\overline{AIOWC}$ – запись в порт ВВ.

Фиксацию сигналов буферами проще всего выполнить с помощью устройств, способных запоминать информацию при поданном напряжении питания. Наиболее оптимальным является использование триггера.

В результате проведенного анализа получена схема электрическая принципиальная порта ввода/вывода на КР580ВВ55 для работы с 16-разрядным внешним устройством, приведенная на рисунке 3.8.

Только режим записи управляющего слова в ВВ55 сигналами $A1$ и $A0$, при наличии сигнала включения порта $\overline{CS}$, позволит пройти сигналу

$\overline{AIOWC}$ на триггеры управления буферами. В каждом плече один триггер отслеживает состояние бита $D4$, а второй – бита $D1$ шины данных, по которой при инициализации BB55 передается управляющее слово MS , содержащее информацию о настройке ввода (вывода) портов A и B . Также отслеживается состояние бита $D7$. Этот бит равный логической «1» соответствует передаче MS . «Защелкнутые» в триггерах состояния используются для управления буферами шины данных для BB55, согласно таблице 3.2. Временные диаграммы работы порта ввода/вывода МПС приведены на рисунке 3.9.

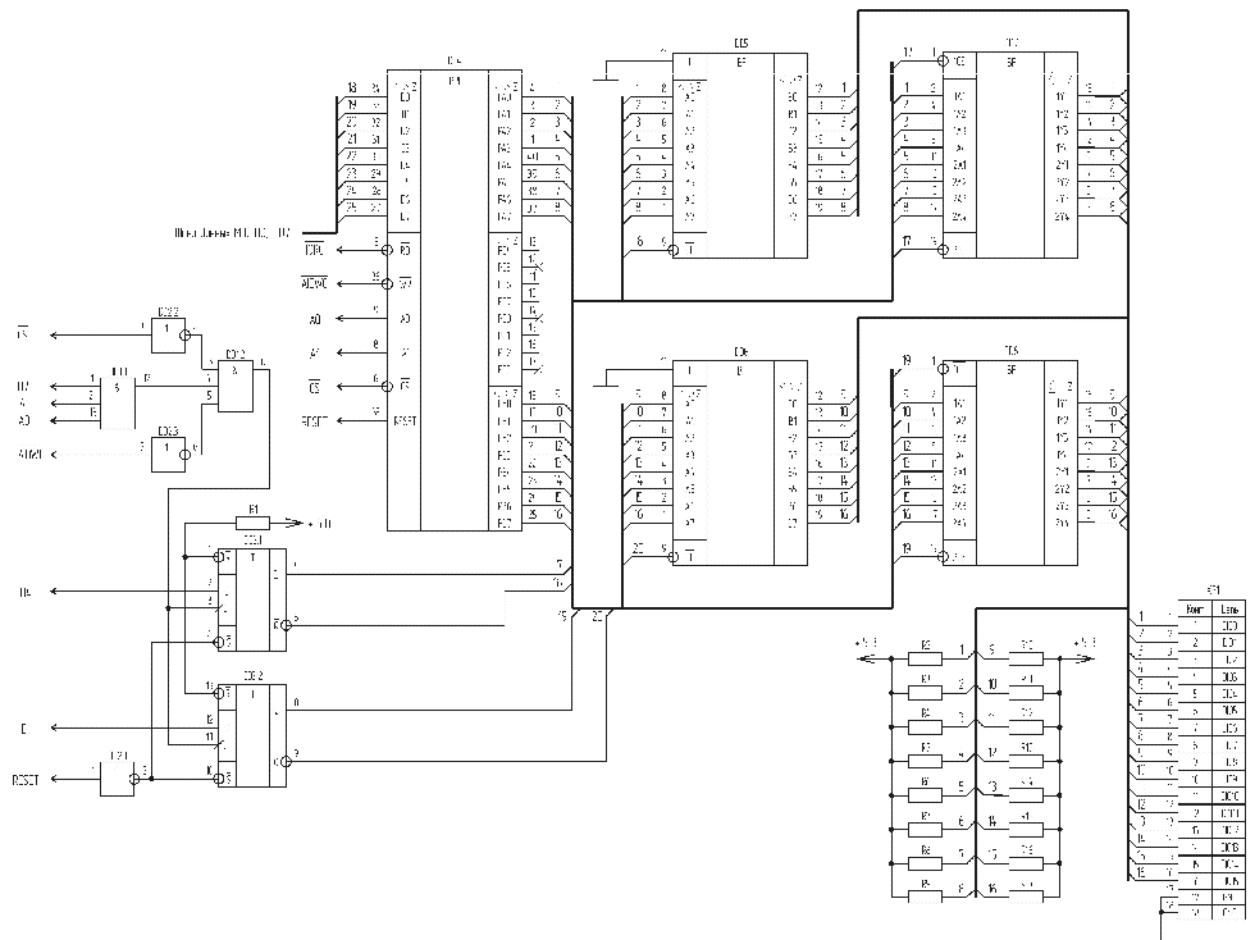


Рисунок 3.8 – Схема электрическая принципиальная порта ввода/вывода на KP580BB55 для работы с внешним 16-разрядным устройством

Таблица 3.2 – Управление буферами шины данных для BB55

$\overline{AIOWC}$	$A1$	$A0$	Управление каналом данных порта A		Управление каналом данных порта B	
			$\overline{OE}_{буф. вв.}$	$\overline{OE}_{буф. выв.}$	$\overline{OE}_{буф. вв.}$	$\overline{OE}_{буф. выв.}$
↓	0	0	Хранение предыдущего состояния			
↓	0	1				
↓	1	0				
↓	1	1	$\overline{D4}$	$D4$	$\overline{D1}$	$D1$
1	0	0	Хранение предыдущего состояния			
1	0	1				
1	1	0				
1	1	1				

Если биты $D4$ и $D1$ равны логической «1», то будут включены буферы вывода данных обоих портов BB55. Если биты $D4$ и $D1$ равны логическому «0», то будут включены буферы ввода данных обоих портов BB55.

Для реализации устройства управления выбраны следующие микросхемы:

- ИМС триггера – 1533ТМ2;
- ИМС элементов «И» – К555ЛИЗ;
- ИМС элементов «НЕ» – КР1533ЛН1.

ИМС 1533ТМ2 представляет собой D -триггер. Вход $\overline{S}$ используется для установки триггера в состояние логической «1» по прямому выходу при сбросе МП. Это обеспечивает включение буферов ввода данных и выключение буферов вывода, что соответствует режиму BB55 после сигнала системного сброса. Сигнал формируется микропроцессором, одновременно поступая и на вход сброса ППА.

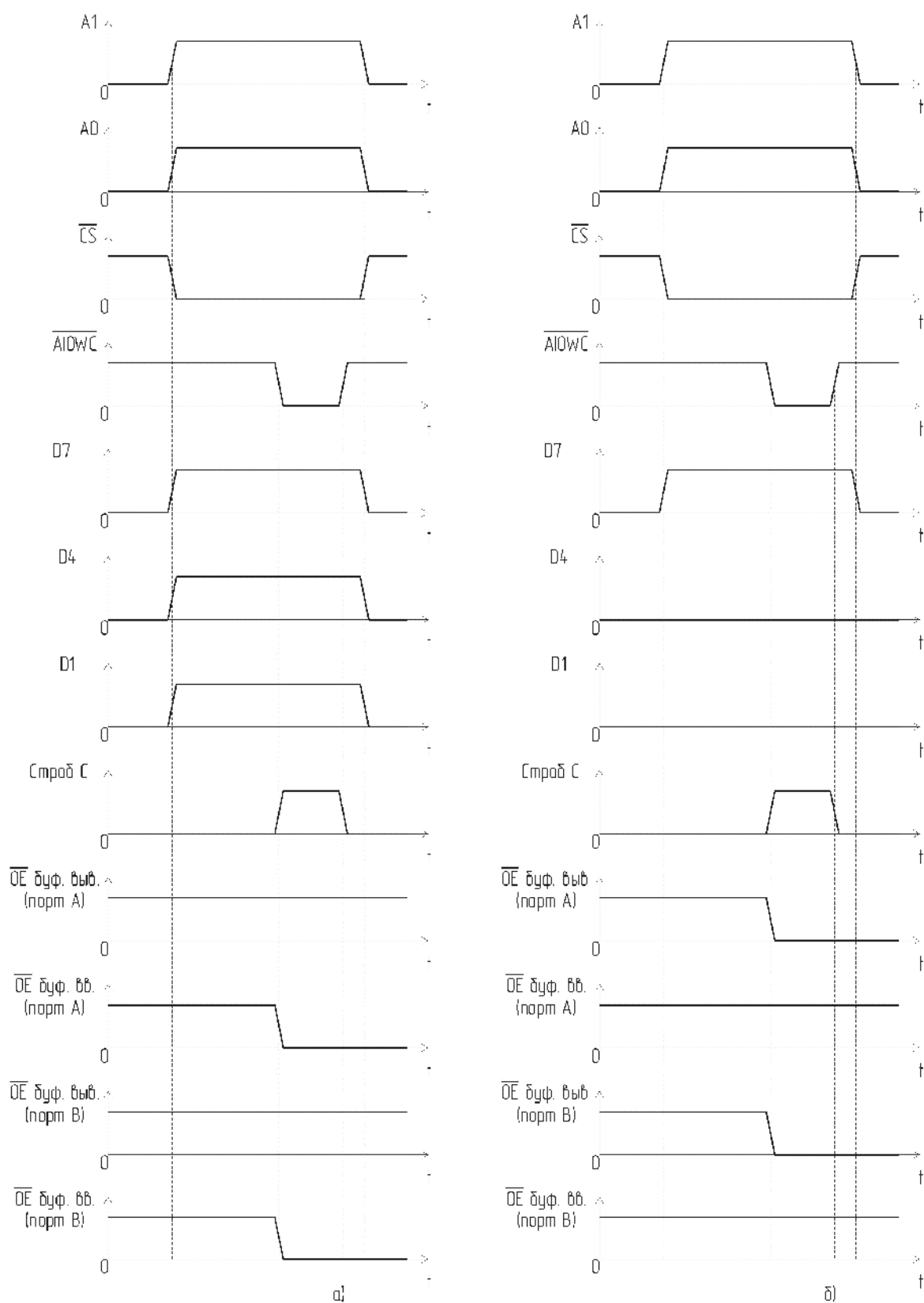


Рисунок 3.9 – Временные диаграммы работы порта ввода/вывода МПС:

а – включение режима ввода;

б – включение режима вывода

Вход триггера $\bar{R}$ не используется в работе, поэтому на него подан пассивный сигнал через ограничивающий резистор. Сопротивление резистора рекомендовано фирмой-производителем триггера и составляет 100 кОм. Мощность, рассеиваемая на резисторе, рассчитывается по формуле (3.1) (см. практическое занятие №2), где в качестве тока через резистор принят ток допустимый для триггера.

Мощность, рассеиваемая на каждом резисторе, рассчитывается по формуле (1.2) (см. практическое занятие № 1).

По результатам расчетов выбираются резисторы необходимого тип номинала.

О необходимости буферных элементов было сказано ранее. Требования к буферным элементам:

- у буферов вывода выходы должны быть типа «открытый коллектор», допускающие объединение с линиями ввода по схеме «монтажное ИЛИ». При этом нагрузочные резисторы будут встроенными в порт. Установка на шине порта вывода напряжения высокого уровня обеспечит ввод внешних данных без каких-либо искажений;

- для буферов ввода достаточно возможности перевода в высокоомное состояние;

- согласованность по электрическим и временным параметрам с микропроцессором *i8085*;

- разрядность – 16 бит.

Для буферов вывода выбрана ИМС импортной элементной базы – *SN74ALS760* фирмы *Texas Instruments*, состоящей из восьми буферных элементов в одном корпусе с выходом типа «открытый коллектор» и возможностью перевода в третье высокоомное состояние.

Для реализации псевдодвунаправленного порта и нормального функционирования ИМС *SN74ALS760* требуются подтягивающие резисторы. Для обеспечения высокой нагрузочной способности их сопротивление

необходимо выбрать так, чтобы обеспечить выходной допустимый ток буферных элементов. Он составляет значение 24 мА.

Сопротивление резистора может быть рассчитано по закону Ома (см. формулу (3.1)).

Мощность, рассеиваемая на каждом резисторе, рассчитывается по формуле (1.2) (см. практическое занятие № 1).

По результатам расчетов выбираются резисторы необходимого тип номинала.

Для буферов ввода выбрана ИМС отечественной элементной базы – КР580ВА86, рассмотренная ранее (см. рисунок 1.14 практического занятия № 1). Эти буферы все время работают на ввод, т. е. передачу информации от канала *B* к каналу *A*. Для этого на управляющий вход *T* подано напряжение логического «0».

Для соединения с внешним устройством необходим разъем, имеющий 18 контактов.

Разработка схемы дешифратора адреса.

Дешифратор адреса для УВВ первого примера.

Согласно заданию первого примера базовый адрес порта ввода/вывода равен $27h$.

Дешифратор адресов можно построить следующим образом:

- с использованием микросхем логических элементов;
- с использованием микросхем дешифраторов;
- с использованием микросхем ЗУ.

Необходимо отслеживать состояние 16 линий адресной шины. Таким образом, наиболее оптимально стоит дешифратор на основе логических элементов. Логика работы дешифратора приведена в таблице 3.3.

Дешифратор адреса порта ввода/вывода должен работать только тогда, когда МПС обращается к УВВ. Для этого необходимо следить за состоянием сигналов $\overline{IORC}$ и $\overline{IOWC}$ системного контроллера микропроцессора. В состав

контроллера входит регистр-защелка, который по стробу $STSTB$ фиксирует слово состояния SW , выдаваемое МП в начале каждого машинного цикла. Слово состояния определяет тип текущего машинного цикла, в зависимости от которого логическая схема контроллера формирует один из пяти управляющих стробов системной магистрали: $\overline{MEMR}$, $\overline{MEMW}$, $\overline{I/OR}$, $\overline{I/OW}$, $\overline{INTA}$ и т. д.

Временные диаграммы их генерации приведены на рисунке 1.11 (см. практическое занятие № 1) и зависят от длительности стробов $DBIN$ и $\overline{WR}$ микропроцессора. Системный контроллер КР580ВК38 (ВК38) в отличие от КР580ВК28 (ВК28) формирует опережающие стробы $\overline{MEMW}$ и $\overline{I/OW}$, что для данной задачи и необходимо использовать.

В результате проведенного анализа получена схема электрическая принципиальная дешифратора адреса порта ввода/вывода, приведенная на рисунке 3.10.

Таблица 3.3 – Логика работы дешифратора адреса порта ввода/вывода

Адресная шина																Адрес порта	Выход CS
A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0		
0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	$27h$	1
Все остальные наборы																	0

Сигналом CS , равным логической «1», разрешается работа порта ввода/вывода.

Переход любого из сигналов $\overline{IORC}$, $\overline{IOWC}$ в состояние логического «0» означает обращение МП к порту ввода/вывода. Любой из этих сигналов, совместно с адресом порта, разрешит работу данного порта, т. е. формирование сигнала CS . Логика формирования сигналов приведена в таблице 3.4.

В качестве логических элементов выбраны:

- ИМС инвертора – КР1533ЛН1;
- ИМС элементов «ИЛИ-НЕ» – К531ЛЕ7;
- ИМС элементов «И-НЕ» – 1533ЛА4. Выбрана ИМС с тремя входами, т. к. в предыдущих схемах остались неиспользованными два таких элемента.

Таблица 3.4 – Логика использования сигналов $\overline{IORC}$, $\overline{IOWC}$

$\overline{IORC}$	$\overline{IOWC}$	Состояние линии управления
0	0	Такой комбинации не существует
0	1	Работа с портом (запись в порт)
1	0	Работа с портом (чтение порта)
1	1	Запрет работы с портом

Потенциал логической единицы необходимо подавать через ограничивающий резистор. Сопротивление резистора рекомендовано фирмой-производителем ИМС логических элементов, либо рассчитывается с использованием закона Ома по формуле (4.1). Мощность, рассеиваемая на резисторе, рассчитывается по формуле (1.2) (см. практическое занятие № 1).

По результатам расчетов выбирается резистор необходимого тип номинала.

Дешифратор адреса для УВВ второго примера.

Согласно заданию второго примера базовый адрес порта ввода/вывода равен $38h$. Все рассуждения о выборе элементной базы для реализации дешифратора рассмотрены выше.

В данном случае необходимо отслеживать состояние восьми линий адресной шины. Таким образом, наиболее оптимально стоит дешифратор на основе логических элементов. Логика работы дешифратора приведена в таблице 3.5.

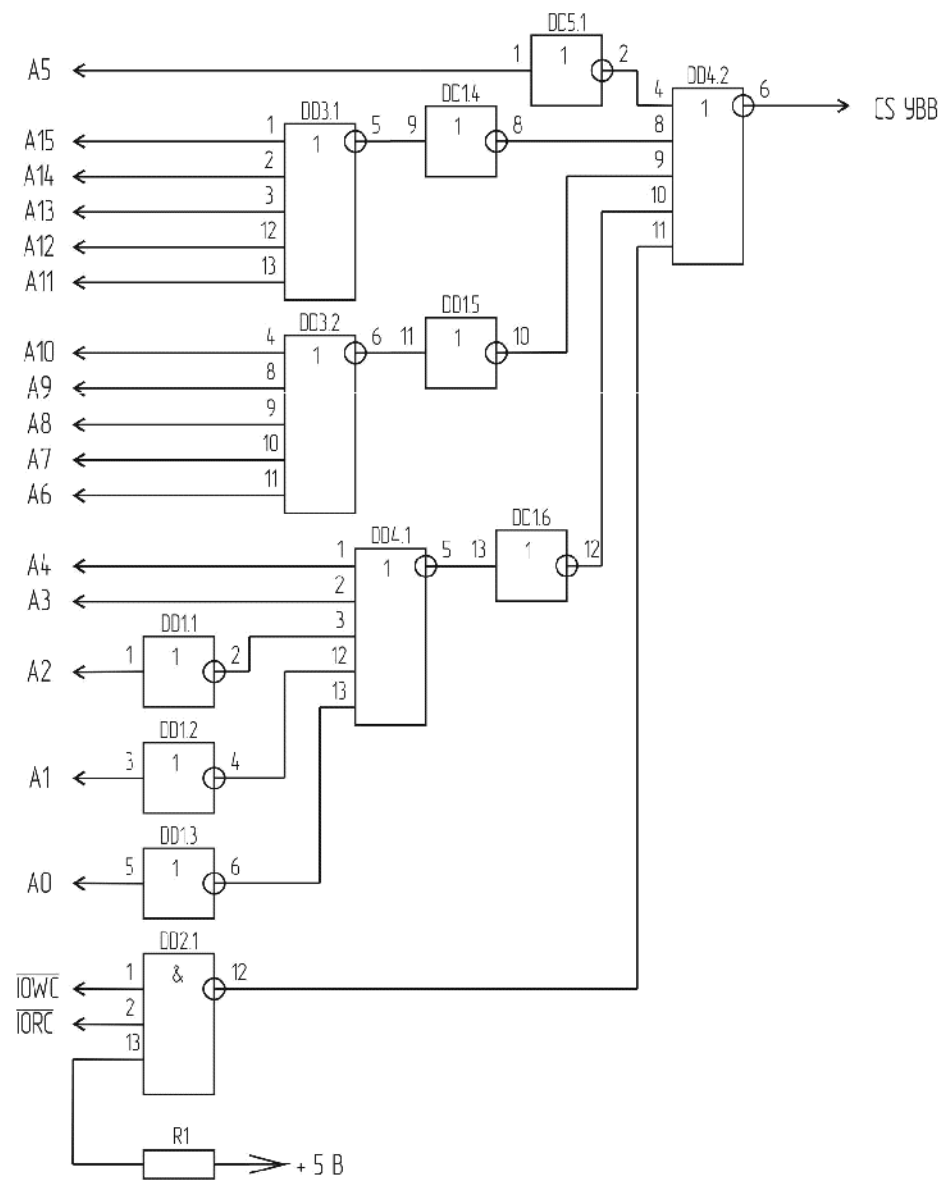


Рисунок 3.10 – Схема электрическая принципиальная дешифратора адреса порта ввода/вывода

Таблица 3.5 – Логика работы дешифратора выбора YBB

Адресная шина								Адрес порта	Выход CS
A7	A6	A5	A4	A3	A2	A1	A0		
0	0	1	1	1	0	0	0	38h	0
Любые остальные наборы									1

Дешифратор должен работать только тогда, когда МПС обращается к

УВВ. Поскольку МП имеет изолированный ввод/вывод, то обращение к УВВ возможно только по командам $\overline{IORC}$ и $\overline{AIOWC}$. При отсутствии этих команд, даже при совпадении адреса на адресной шине МП с адресом порта, состояние ВВ55 не изменится.

В результате проведенного анализа получена схема электрическая принципиальная дешифратора выбора УВВ, приведенная на рисунке 3.11.

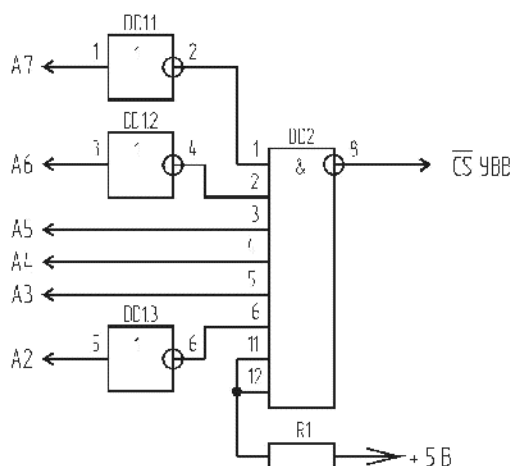


Рисунок 3.11 – Схема электрическая принципиальная дешифратора выбора УВВ

Сигналом $\overline{CS}$, равным логическому «0», разрешается работа порта ввода/вывода. В качестве логических элементов выбраны:

- ИМС инвертора – КР1533ЛН1;
- ИМС элементов «И-НЕ» – КР1533ЛА2.

Неиспользуемые входы ИМС КР1533ЛА2 необходимо подключить так, чтобы не изменить логику работы устройства, т. е. к логической «1». Этот необходимо подавать через ограничивающий резистор. Сопротивление резистора рекомендовано фирмой-производителем ИМС и составляет 100 кОм.

Мощность, рассеиваемая на резисторе, рассчитывается по формуле (3.1) (см. практическое занятие № 2), где в качестве тока через резистор принят ток допустимый для триггера.

Мощность, рассеиваемая на каждом резисторе, рассчитывается по формуле (1.2) (см. практическое занятие № 1).

По результатам расчетов выбираются резисторы необходимого тип номинала.

Разработка схемы электрической принципиальной порта ввода/вывода параллельных цифровых данных одноплатного микроконтроллера.

В результате проведенного анализа и расчетов разработаны схемы электрические принципиальные портов ввода/вывода параллельных цифровых данных одноплатного микроконтроллера, полученные путем объединения синтезированных ранее отдельных блоков, и приведенные на рисунках 3.12 (на МБР) и 3.13 (на KP580BB55).

Рисунок 3.12 – Схема электрическая принципиальная порта ввода/вывода параллельных цифровых данных одноплатного микроконтроллера (на МБР)

Рисунок 3.13 – Схема электрическая принципиальная порта ввода/вывода параллельных цифровых данных одноплатного микроконтроллера (на KP580BB55)

В результате выполнения данного пункта был разработан модуль порта ввода/вывода параллельных цифровых данных одноплатного микроконтроллера на основе процессора *i8080*.

Оборудование и материалы

Практическая работа выполняется с использованием ПК и любого программного обеспечения с функциями САПР для выполнения схемы

электрической принципиальной модуля ввода/вывода одноплатного микроконтроллера.

Указания по технике безопасности

Перед проведением практического занятия следует изучить инструкцию по технике безопасности при работе на компьютере. Также следует соблюдать правила и требования, установленные для данного компьютерного класса.

Задания (задачи)

1 Разработать порт ввода/вывода для одноплатного микроконтроллера на основе регистров, для чего необходимо:

- выбрать задание из таблицы 3.6 в соответствии с вариантом, номер которого задается преподавателем;
- провести анализ требований и выбрать одну или несколько ИМС регистров в соответствии с требованиями задания;
- разработать схему управления регистрами для осуществления операции записи данных в порт;
- разработать схему управления регистрами для осуществления операции чтения данных из порта;
- разработать схему дешифратора адреса для порта ввода/вывода;
- выбрать элементную базу реализации схем управления и дешифратора адреса;
- разработать схему электрическую принципиальную порта ввода/вывода для одноплатного микроконтроллера на основе регистров;
- предъявить требования к управляющим сигналам, а также к линиям адресной шины и шины данных;

- провести расчет электрических параметров основных и дополнительных компонентов, необходимых для функционирования схемы. Выбрать необходимые компоненты, пользуясь справочной литературой;
- начертить временные диаграммы работы порта ввода/вывода.

Таблица 3.6

№ варианта	1	2	3	4	5	6	7	8	9
Тип подключения	П	ВУ	П	ВУ	П	ВУ	П	ВУ	П
Разрядность порта	8	16	8	8	8	8	8	16	8
Базовый адрес	C75E3h	38h	FB3Ch	4C23h	D645Fh	CD75h	CB4D2h	38h	D934Fh

Продолжение таблицы 3.6

№ варианта	10	11	12	13	14	15	16	17	18
Тип подключения	ВУ	П	ВУ	П	ВУ	П	ВУ	П	ВУ
Разрядность порта	16	8	8	8	16	8	8	8	16
Базовый адрес	12h	BCA08h	792Dh	F8420h	38h	EA5F0h	9DA3h	DA2Bh	64h

2 Разработать порт ввода/вывода для одноплатного микроконтроллера на основе программируемого периферийного адаптера (ППА), для чего необходимо:

- выбрать задание из таблицы 3.6 в соответствии с вариантом, номер которого задается преподавателем;
- разработать схему управления ППА для осуществления операции записи данных в порт;
- разработать схему управления ППА для осуществления операции чтения данных из порта;
- разработать схему дешифратора адреса для порта ввода/вывода;
- выбрать элементную базу реализации схем управления и дешифратора адреса;

- разработать схему электрическую принципиальную порта ввода/вывода для одноплатного микроконтроллера на основе ППА;
- предъявить требования к управляющим сигналам, а также к линиям адресной шины и шины данных;
- провести расчет электрических параметров основных и дополнительных компонентов, необходимых для функционирования схемы. Выбрать необходимые компоненты, пользуясь справочной литературой;
- начертить временные диаграммы работы порта ввода/вывода.

Содержание отчета

Отчет по практическому занятию выполняется на листах формат А4 согласно требованиям ГОСТ и ЕСКД и должен содержать следующие пункты:

- а) титульный лист с названием работы, дисциплины, кода группы, ФИО исполнителя и преподавателя;
- б) цель и содержание работы;
- в) краткое теоретическое введение;
- г) условия, ход выполнения и решения задач;
- д) общую схему электрическую принципиальную модуля памяти одноплатного микроконтроллера;
- е) временные диаграммы работы порта ввода/вывода;
- ж) выводы по результатам выполнения практического задания.

Список литературы, рекомендованный к использованию по данной теме

Основная учебная литература

1 Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд.,

испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

Дополнительная учебная литература

1 Бродин В.Б., Калинин А.В. Системы на микроконтроллерах и БИС программируемой логики – М.: Издательство ЭКОМ, 2002. – 400 с.

2 Боборыкин А. В., Липовицкий Г.П. и др. Однокристалльные микроЭВМ. – М.: МИКАП, 1994 – 400с.

3 Сташин В. В., Урусов А. В., Мологонцева О. Ф. Проектирование цифровых устройств на однокристалльных микроконтроллерах – М.: Энергоатомиздат, 1990 – 224с.

4 Шевкопляс Б.С. Микропроцессорные структуры. Инженерные решения: Справочник. - М.: Радио и связь, 1990 – 512 с.

5 Щелкунов, Н. Н. Микропроцессорные средства и системы / Н. Н. Щелкунов, А. П. Дианов. – М. : Радио и связь, 1989. – 288 с.

6 Коффрон, Дж. Технические средства микропроцессорных систем: практический курс / Дж. Коффрон. Пер. с англ. – М. : Мир, 1983. – 344 с.

7 Шахнов, В. А. Микропроцессоры и микропроцессорные комплекты интегральных микросхем : справочник / В.-Б. Б. Абрайтис, Н. Н. Аверьянов, А. И. Белоус [и др.]: под ред. В. А. Шахнова. – Том 1. – М. : Радио и связь, 1988. – 368 с.

8 Соппа, И. В. Микропроцессоры и микроконтроллеры / И. В. Соппа, А. С. Самардак, В. А. Иванов. – Владивосток : Издательство Дальневосточного университета, 2009. – 211 с.

9 Алексенко, А. Г. Основы микросхемотехники / А. Г. Алексенко. 3-е изд. перераб. и доп. – М. : БИНОМ. Лаборатория знаний, 2009. – 448 с.

10 Петровский И.И. и др. Логические ИС К1533, К1554: Справочник / В двух частях. - М.: ТОО “БИНОМ”, 1993 – 500 с.

11 Лебедев О.Н. Микросхемы памяти и их применение. – М.: Радио и

связь, 1990 – 303 с.

12 Якубовский, С. В. Аналоговые и цифровые интегральные микросхемы : справочное пособие / С. В. Якубовский, Н. А. Барканов, Л. И. Ниссельсон [и др.]; под ред. С. В. Якубовского. – 2-е изд., перераб. и доп. – М. : Радио и связь, 1984. – 432 с.

13 Цифровые интегральные микросхемы : справочник / М. И. Богданович, И. Н. Грель, С. А. Дубина [и др.]. – 2-е изд., перераб. и доп. – Минск : Беларусь, Полымя, 1996. – 605с.

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ 4

Тема: Расчет электрического интерфейса одноплатного микроконтроллера.

Цель работы: закрепление знаний, полученных на лекциях по разделу: «Проектирование микропроцессорных систем». В ходе выполнения заданий практического занятия необходимо:

- рассчитать общий потребляемый ток от источника питания (от источников, если их несколько);
- рассчитать полную потребляемую от источника (источников) мощность и предъявить к нему требования по электрическим параметрам;
- выбрать разъемы для соединения разработанного одноплатного микроконтроллера с источником (источниками) питания;
- выбрать разъемы для соединения разработанного одноплатного микроконтроллера с внешним (внешними) устройствами.

Знания, приобретаемые студентом в результате освоения темы:

- классификация микропроцессоров и их основные характеристики.

Умения, приобретаемые студентом в результате освоения темы:

- разрабатывать микропроцессорные системы для решения задач управления, автоматизации, сбора и обработки данных и других целей.

Формируемые компетенции: ОК-6, ОК-7, ОПК-3, ОПК-4, ОПК-6, ОПК-7, ОПК-8, ОПК-9, ПК-1, ПК-3, ПК-5, ПК-6.

Актуальность темы: тема является актуальной, так как для работы любого микропроцессорного устройства необходим источник питания. Также, для разработки печатной платы микропроцессорной системы необходимо знать: ток, потребляемый каждым компонентом и суммарный, рабочее напряжение и иные электрические параметры. Поэтому студент должен знать методику расчета электрического интерфейса одноплатного микроконтроллера.

Теоретическая часть

Источник питания – устройство, предназначенное для обеспечения различных устройств электрическим питанием.

Различают первичные и вторичные источники питания.

К первичным относят преобразователи различных видов энергии в электрическую, примером может служить аккумулятор, преобразующий химическую энергию в электрическую.

Вторичные источники сами не генерируют электроэнергию, а служат лишь для её преобразования с целью обеспечения требуемых параметров (напряжения, тока, пульсаций напряжения и т. п.).

Разумеется, для такого устройства, как одноплатный микроконтроллер единственным оптимальным решением будет использование вторичного источника питания с обязательной стабилизацией напряжения и тока.

Рассмотрим пример расчета электрического интерфейса одноплатного микроконтроллера.

Расчет полной потребляемой от источника питания мощности. Формирование требований к источнику питания.

В зависимости от используемого микропроцессора и компонентов его типовой схемы включения (системный контроллер, генератор, блок памяти, блок ввода/вывода и т. д.) может потребоваться источник, формирующий несколько напряжений питания, включая и отрицательные. Такой источник называют многоканальным.

Для предъявления требований к источнику питания необходимо:

- провести анализ напряжений, используемых для компонентов одноплатного микроконтроллера;
- рассчитать ток, потребляемый по каждому каналу напряжения (данные для расчетов – справочные данные использованных в схеме компонентов);

- зная потребляемый ток и напряжения каналов, рассчитать полную потребляемую от источника питания мощность.

Рассмотрим пример.

Пусть разрабатываемый одноплатный микроконтроллер выполнен на основе процессора *i8085*. МП использует источник питания с напряжением плюс 5 В. Его периферия имеет такое же напряжение питания. Рассчитаем полный потребляемый системой ток:

- микропроцессор *i8085* – 170 мА;
- регистры шины адреса КР580ИР82 – 2 штуки по 160 мА;
- шинный формирователь КР580ВА86 – 160 мА;
- цепь сброса МП – $0,5 \cdot 10^{-3}$ А;
- вход *READY* – $2 \cdot 10^{-3}$ А;
- системный контроллер (КР1533ЛН1 и 1533ЛЕ1) – 3 и 3,8 мА

соответственно;

- микросхема ПЗУ К573РФ7 – 100 мА;
- микросхемы ОЗУ КР537РУ9 – 65 мА;
- микросхема дешифратора адреса модуля памяти 1533ИД7 – 10 мА;
- ограничивающий резистор для дешифратора – $0,1 \cdot 10^{-3}$ мА;
- элемент логики «И» 1533ЛИ1 для дешифратора – 1 мА;
- дешифратор адреса УВВ (КР1533ЛА2 и КР1533ЛН1) – 0,9 и 3,8 мА

соответственно;

- триггер УВВ 1533ТМ2 – 40 мА;
- элементы «И» УВВ К555ЛИЗ – 6,6 мА;
- ограничивающий резистор для элемента «И» – 0,1 мА;
- элементы «НЕ» УВВ КР1533ЛН1 – 3,8 мА;
- буферные ИМС КР580ВА86 – 2 штуки по 160 мА;
- буферные элементы УВВ *SN74ALS760* – 2 штуки по 19 мА;
- подтягивающие резисторы УВВ – 16 штук по 24 мА,

что в сумме составляет 1567,6 мА.

Полная потребляемая мощность рассчитывается по известной формуле:

$$P_{\text{полн.}} = P_1 + P_2 + \dots + P_n = U_{\text{ип1}} \cdot I_{\text{ип1}} + U_{\text{ип2}} \cdot I_{\text{ип2}} + \dots + U_{\text{ипn}} \cdot I_{\text{ипn}},$$

где P_n – мощность потребления по каналу n , Вт;

$U_{\text{ипn}}$ – напряжение питания канала n источника питания, В;

$I_{\text{ипn}}$ – ток потребления по каналу n , А.

Таким образом, источник питания для рассматриваемого в примере одноплатного микроконтроллера должен обеспечивать стабилизированное выходное напряжение питания 5 В при токе нагрузки, не менее 1567,6 мА.

Выбор электрических соединителей для подключения блока питания.

Обычно многоканальные источники выполняются конструктивно в одном корпусе. Соединение с потребителем осуществляется кабелями с использованием разборных соединений – разъемов.

Соответственно, печатная плата одноплатного микроконтроллера также должна иметь разъем, для подключения источника питания.

В рассмотренном ранее примере для соединения с источником питания необходимо предусмотреть разъем, который должен содержать две клеммы и удовлетворять следующим требованиям:

- максимальный ток через контакт рассчитывается по формуле (1.6) (см. практическое занятие № 1):

$$I_{\text{макс.конт}} = 1,3 \cdot 1,6 = 2,08 \text{ А};$$

- максимальное рабочее напряжение рассчитывается по формуле (1.7) (см. практическое занятие № 1):

$$U_{\text{макс.конт}} = 1,5 \cdot 5 = 7,5 \text{ В.}$$

Выбран разъем типа CWF-2 (вилка на плату, закрытая, прямая), имеющий параметры:

Рабочее напряжение, В 250

Максимальный ток через контакт, А 3

При монтаже микросхем на печатных платах необходимо установить вблизи разъема конденсаторы из расчета не менее 0,1 мкФ на одну ИМС, исключающих низкочастотные помехи, и по одному керамическому конденсатору на группу микросхем числом не более 10 из расчета 0,002 мкФ на одну ИС, с целью устранения высокочастотных помех.

Рабочее напряжение для конденсаторов рассчитывается по формуле (1.3) (см. практическое занятие № 1).

По результатам расчетов выбираются электролитический и керамические конденсаторы (например, типа К53-1 и К10-17-1А соответственно).

Схема электрическая принципиальная подключения приведена на рисунке 4.1.

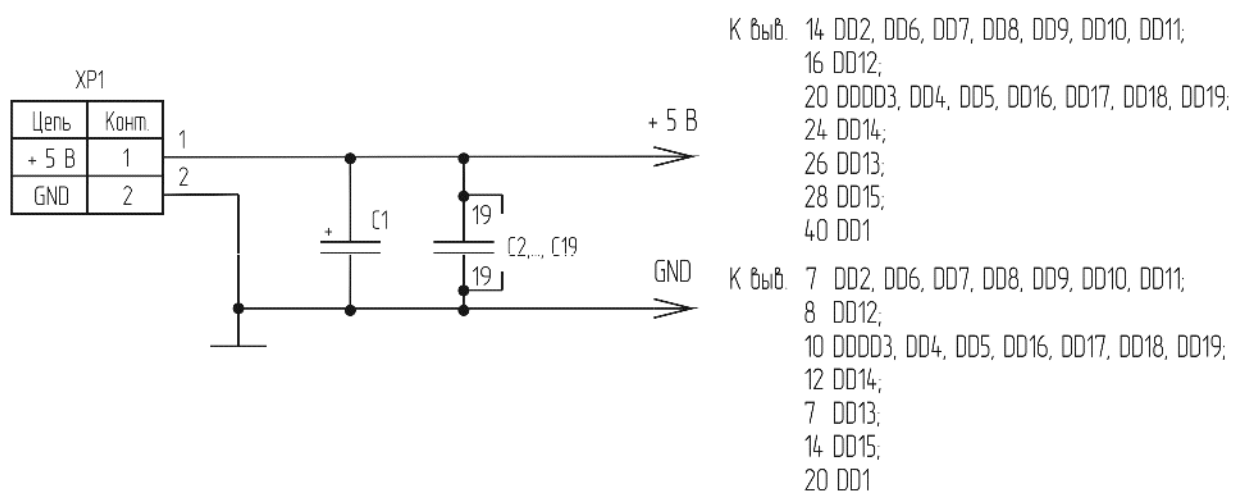


Рисунок 4.1 – Схема электрическая принципиальная

На рисунке схематически показан разъем, а также конденсаторы: электролитический и керамические. Количество конденсаторов приведено к примеру и зависит от количества микросхем в конкретном задании.

Шины питания «+5 В» и «GND» подключены к выводам микросхем. На рисунке показан пример. К каким выводам и каких микросхем они подключаются, зависит от конкретного задания.

Выбор электрических соединителей для подключения внешних устройств.

Любые внешние устройства располагаются вне платы разработанного одноплатного микроконтроллера. Поэтому для их соединения также необходимо предусматривать разъемы.

Пусть согласно заданию (см. практическую работу № 3), необходимо разработать порт ввода/вывода для одноплатного микроконтроллера, имеющий следующие характеристики:

- реализация параллельного порта ввода/вывода – KP580VB55;
- способ подключения порта ввода/вывода к системной магистрали – внешнее устройство;
- разрядность порта – 16 бит;
- базовый адрес порта ввода/вывода – $38h$.

Для соединения с внешним устройством необходим разъем, имеющий 18 контактов, к которому предъявляются следующие требования:

- максимальный ток через контакт (1.6) (см. практическое занятие № 1):

$$I_{\text{МАКС.КОНТ}} = 1,3 \cdot 32 \cdot 10^{-3} = 41,6 \cdot 10^{-3} \text{ А};$$

- максимальное рабочее напряжение (1.7) (см. практическое занятие № 1):

$$U_{\text{МАКС.КОНТ}} = 1,5 \cdot 5 = 7,5 \text{ В.}$$

Данные для расчетов взяты из примера практического занятия № 3.

В качестве тока через контакт выбрано максимальное значение тока KP580BA86. В качестве напряжения на контакте выбрано напряжение питания МП i8085.

Выбран миниатюрный разъем типа PLS-18, имеющий параметры:

Максимальное рабочее напряжение, В 500

Максимальный ток через контакт, А 1

Схема электрическая принципиальная подключения приведена на

рисунке 3.8 (см. практическое занятие № 3).

В результате проведенного анализа и расчетов выполнено:

- рассчитан потребляемый ток от источника питания;
- предъявлены требования к источнику питания;
- выбраны электрические соединители для подключения к источнику питания;
- выбраны электрические соединители для подключения внешних устройств.

Оборудование и материалы

Практическая работа выполняется с использованием ПК и любого программного обеспечения с функциями САПР для выполнения схемы электрической принципиальной одноплатного микроконтроллера.

Указания по технике безопасности

Перед проведением практического занятия следует изучить инструкцию по технике безопасности при работе на компьютере. Также следует соблюдать правила и требования, установленные для данного компьютерного класса.

Задания (задачи)

1 Рассчитать общий потребляемый ток от источника питания (от источников, если их несколько), для чего необходимо:

- провести анализ напряжений, используемых для компонентов одноплатного микроконтроллера;
- рассчитать ток, потребляемый по каждому каналу напряжения

(данные для расчетов – справочные данные использованных в схеме компонентов);

- зная потребляемый ток и напряжения каналов, рассчитать полную потребляемую от источника (источников) питания мощность.

2 Выбрать разъемы для соединения разработанного одноплатного микроконтроллера с источником (источниками) питания, для чего необходимо:

- рассчитать максимальный ток через контакт;
- рассчитать максимальное рабочее напряжение;
- провести расчеты, необходимые для выбора элементов фильтра по цепи (цепям) источника питания;

- выбрать необходимые компоненты, пользуясь справочной литературой;

- разработать схему электрическую принципиальную подключения.

3 Выбрать разъемы для соединения разработанного одноплатного микроконтроллера с внешним (внешними) устройствами, для чего необходимо:

- рассчитать максимальный ток через контакт;
- рассчитать максимальное рабочее напряжение;
- выбрать необходимые компоненты, пользуясь справочной литературой;

- разработать схему электрическую принципиальную подключения.

Задания выполняются по вариантам. Исходными данными являются схемотехнические решения, полученные в результате выполнения заданий практических занятий № 1 – 3.

Содержание отчета

Отчет по практическому занятию выполняется на листах формат А4

согласно требованиям ГОСТ и ЕСКД и должен содержать следующие пункты:

- а) титульный лист с названием работы, дисциплины, кода группы, ФИО исполнителя и преподавателя;
- б) цель и содержание работы;
- в) краткое теоретическое введение;
- г) условия, ход выполнения и решения задач;
- д) общую схему электрическую принципиальную одноплатного микроконтроллера;
- е) выводы по результатам выполнения практического задания.

Список литературы, рекомендованный к использованию по данной теме

Основная учебная литература

1 Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд., испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

Дополнительная учебная литература

1 Щелкунов, Н. Н. Микропроцессорные средства и системы / Н. Н. Щелкунов, А. П. Дианов. – М. : Радио и связь, 1989. – 288 с.

2 Шахнов, В. А. Микропроцессоры и микропроцессорные комплекты интегральных микросхем : справочник / В.-Б. Б. Абрайтис, Н. Н. Аверьянов, А. И. Белоус [и др.]: под ред. В. А. Шахнова. – Том 1. – М. : Радио и связь, 1988. – 368 с.

3 Петровский И.И. и др. Логические ИС К1533, К1554: Справочник / В двух частях. - М.: ТОО “БИНОМ”, 1993 – 500 с.

4 Лебедев О.Н. Микросхемы памяти и их применение. – М.: Радио и

связь, 1990 – 303 с.

5 Якубовский, С. В. Аналоговые и цифровые интегральные микросхемы : справочное пособие / С. В. Якубовский, Н. А. Барканов, Л. И. Ниссельсон [и др.]; под ред. С. В. Якубовского. – 2-е изд., перераб. и доп. – М. : Радио и связь, 1984. – 432 с.

6 Цифровые интегральные микросхемы : справочник / М. И. Богданович, И. Н. Грель, С. А. Дубина [и др.]. – 2-е изд., перераб. и доп. – Минск : Беларусь, Полымя, 1996. – 605с.

ОСНОВЫ МИКРОПРОЦЕССОРНОЙ ТЕХНИКИ**Методические указания по выполнению практических работ**

по направлению подготовки

11.03.04 Электроника и нанoeлектроника

Квалификация выпускника

Бакалавр

Часть 2

Составитель: Лапин А.А.

Редактор:

Подписано в печать _____			
Формат _____	Усл. п. л. – _____	Уч.-изд. л. – _____.	
Бумага _____.	Печать _____.	Заказ _____	Тираж _____ экз.
ФГАОУ ВО «Северо-Кавказский федеральный университет»			
355029, г. Ставрополь, пр. Кулакова, 2			

Издательство Северо-Кавказского федерального университета
Отпечатано в типографии СКФУ

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего
образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

по выполнению лабораторных работ

по дисциплине «Основы микропроцессорной техники»

для студентов направления подготовки

11.03.04 Электроника и нанoeлектроника

Направленность (профиль)

Интеллектуальные программируемые системы и устройства

Ставрополь 2026 г.

Печатается по решению
Учебно-методического совета
Северо-Кавказского
федерального университета

УДК
ББК

Рецензенты: Валюхов Д.П.

Лапин А.А. **Основы микропроцессорной техники:** методические указания по выполнению лабораторных работ. – Ставрополь: Изд-во СКФУ, 20___. – 155 с.

Настоящие методические указания являются руководством по выполнению лабораторных работ по дисциплине «Основы микропроцессорной техники» для студентов очной формы обучения направления подготовки 11.03.04 Электроника и наноэлектроника. В методических указаниях приведены методика выполнения работ, требования к содержанию отчета, его форме, литература.

Методические указания составлены в соответствии с требованиями ФГОС ВО по направлению подготовки 11.03.04 Электроника и наноэлектроника, рабочим учебным планом и программой дисциплины «Основы микропроцессорной техники».

УДК
ББК

© ФГАОУ ВО «Северо-Кавказский
федеральный университет», 20__

Введение

Целью дисциплины является формирование набора требуемых компетенций будущего бакалавра по направлению подготовки 11.03.04 Электроника и нанoeлектроника.

Задачи дисциплины: ознакомить студентов с основами микропроцессорной техники, принципами программирования микропроцессоров, возможностью использования микропроцессоров и устройств на их основе для решения различных прикладных задач, системой команд, организацией интерфейсов связи с внешним оборудованием.

Выполнение лабораторных работ способствует получению навыков, согласно заявленным для дисциплины компетенциям, на базе теоретического материала, изучаемого на лекционных занятиях.

Содержание

Лабораторная работа 1. Изучение учебного микропроцессорного комплекта	5
Лабораторная работа 2. Исследование записи и выполнения простых программ	22
Лабораторная работа 3. Исследование операций ввода-вывода и изучение маскирования данных, организации условных переходов	33
Лабораторная работа 4. Исследование работы подпрограмм и методов использования стека	48
Лабораторная работа 5. Исследование арифметических операций	65
Лабораторная работа 6. Исследование работы устройств ввода и вывода информации	85
Лабораторная работа 7. Исследование методов моделирования функций алгебры логики	108
Лабораторная работа 8. Исследование работы средств параллельного ввода-вывода	122
Лабораторная работа 9. Исследование работы средств последовательного ввода-вывода	139

ЛАБОРАТОРНАЯ РАБОТА 1

Тема: Изучение учебного микропроцессорного комплекта.

Цель работы: ознакомление со структурой учебного микропроцессорного комплекта (УМК), картой памяти, органами управления и режимами работы.

Формируемые компетенции: ОК-6, ОК-7, ОПК-9, ПК-1, ПК-3.

Теоретическая часть

Учебная микро-ЭВМ (или учебный микропроцессорный комплект) предназначена для знакомства с особенностями построения микро-ЭВМ на микропроцессоре с фиксированным набором команд и может быть использована для исследования методов программирования и работы БИС, входящих в микропроцессорный комплект серии K580. Микро-ЭВМ может применяться как управляющая ЭВМ при создании и исследовании работы систем управления различными объектами. Она является легко осваиваемым и удобным средством для отладки относительно небольших (до 0,5 Кбайт) программ пользователя. Открытая конструкция и наличие большого количества средств индикации позволяют наглядно исследовать процесс преобразования и передачи информации в микро-ЭВМ.

В задачу теоретического обоснования не входит детальное описание УМК с точки зрения его программной реализации, специфики функционирования в различных операционных системах. Цель его – иллюстрация структуры и возможностей реально работающей микропроцессорной системы, ориентированной на определенный круг задач.

Оборудование и материалы

Микро-ЭВМ представляет собой виртуальное устройство, выполненное в виде программного обеспечения для персонального компьютера. Назначение устройства – эмуляция существующей учебной микро-ЭВМ на микропроцессорном комплекте серии K580.

В ходе выполнения лабораторных работ по дисциплине «Основы микропроцессорной техники» будут использоваться два эмулятора: «Эмулятор работы микропроцессора KP580BM80» (далее ПМК – по заголовку главного окна эмулятора) и «Программная модель стенда УМПК-80 на базе микропроцессора K580BM80» (далее УМПК – по заголовку главного окна эмулятора). Первый эмулятор находится в папке *CPU580*, второй – в папке *UMPK80*.

При входе в первый эмулятор открывается окно ПМК KP580BM80, в котором изображена структура «стенда» (рисунок 1.1).

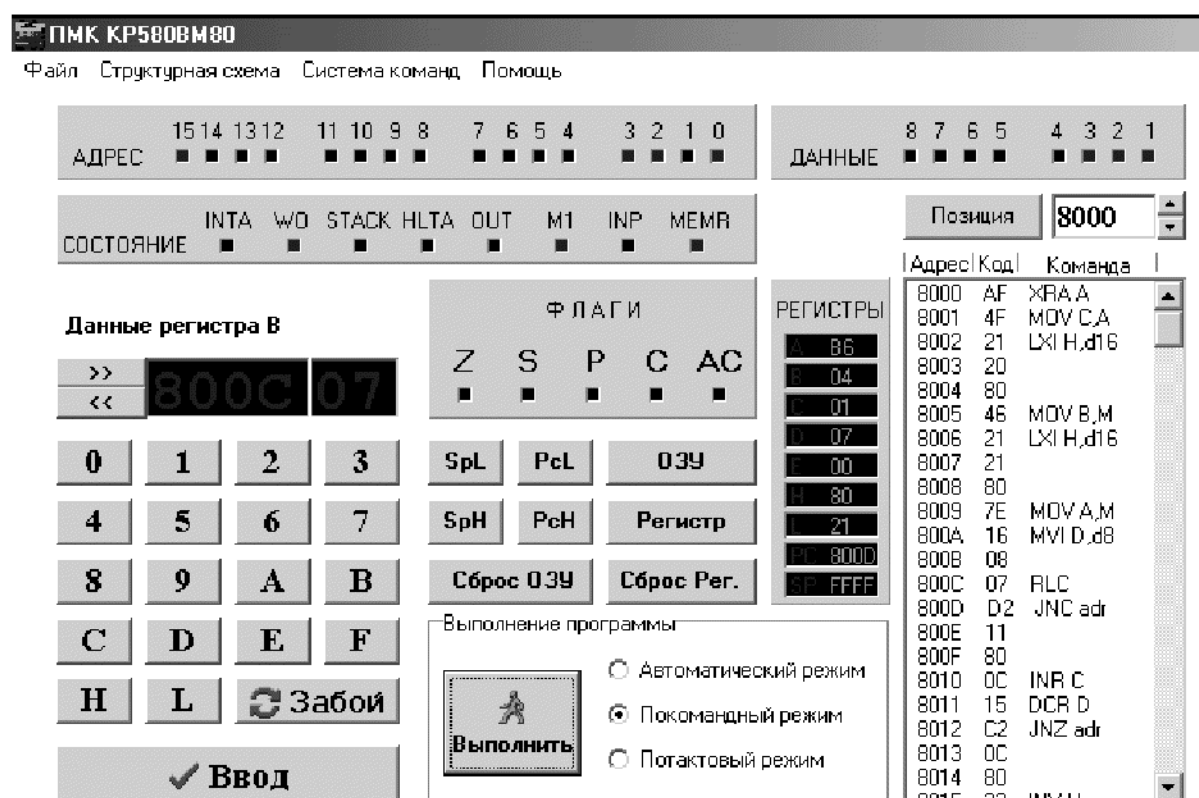


Рисунок 1.1 – Главное окно эмулятора ПМК KP580BM80

В верхней части окна указываются режимы работы: «Файл», «Структурная схема», «Система команд», «Помощь».

Строкой ниже указывается адрес (состояние шины адреса в шестнадцатеричной системе счисления) и данные (состояние шины данных в шестнадцатеричной системе счисления).

Далее показано окно, в котором указывается адрес ОЗУ и его содержимое, или номер регистра и его содержимое.

В окне флагов записывается состояние флагов Z, S, P, C, AC на данный момент выполнения команды процессором.

Для ввода информации используется клавиатура, которая подразделяется на цифровую: от 0 до F и режимную: SpL, SpH, PcL, PcH , ОЗУ, Сброс ОЗУ, Регистр, Сброс Рег.

Для ввода информации ОЗУ необходимо нажать кнопку «ОЗУ», затем кнопку «Ввод». После этого ввести в данную ячейку код команды или данные.

Для ввода информации в регистры требуется нажать клавишу «Регистр», клавишу «Ввод», а затем на цифровой клавиатуре произвести ввод данных.

Клавиши SpL, SpH и PcL, PcH позволяют записывать адреса в указатель стека и счетчик команд.

При вводе любой информации необходимо выполнить нажатия левой кнопкой мыши на клавишу «Ввод».

В правой части окна эмулятора расположены регистры процессора:

- A – аккумулятор;
- B, C, D, E, H, L – регистры общего назначения (РОН);
- PC – программный счетчик;
- SP – указатель стека.

Указатель стека указывает вершину стека $FFFF$.

В области окна «Выполнение программы» задается режим:

- автоматический режим. При нажатии этой кнопки программа выполняется до команды *HLT*;
- покомандный режим. При нажатии этой кнопки реализуется весь цикл исполнения текущей команды;
- потактовый режим. При каждом нажатии кнопки «Выполнить» осуществляется выполнение одного такта работы процессора.

В правой части окна эмулятора расположено окно «Позиция» адресного пространства ОЗУ, в котором производится набор программы: адрес команды, код, ассемблерный код команды. «Позиция» указывает область адресного пространства ОЗУ, в котором прокруткой «>>» или «<<» можно указывать начало требуемого адресного пространства.

При указании курсором мыши позиции «Структурная схема» раскрывается следующее окно, в котором показана структурная схема микропроцессора (рисунок 1.2).

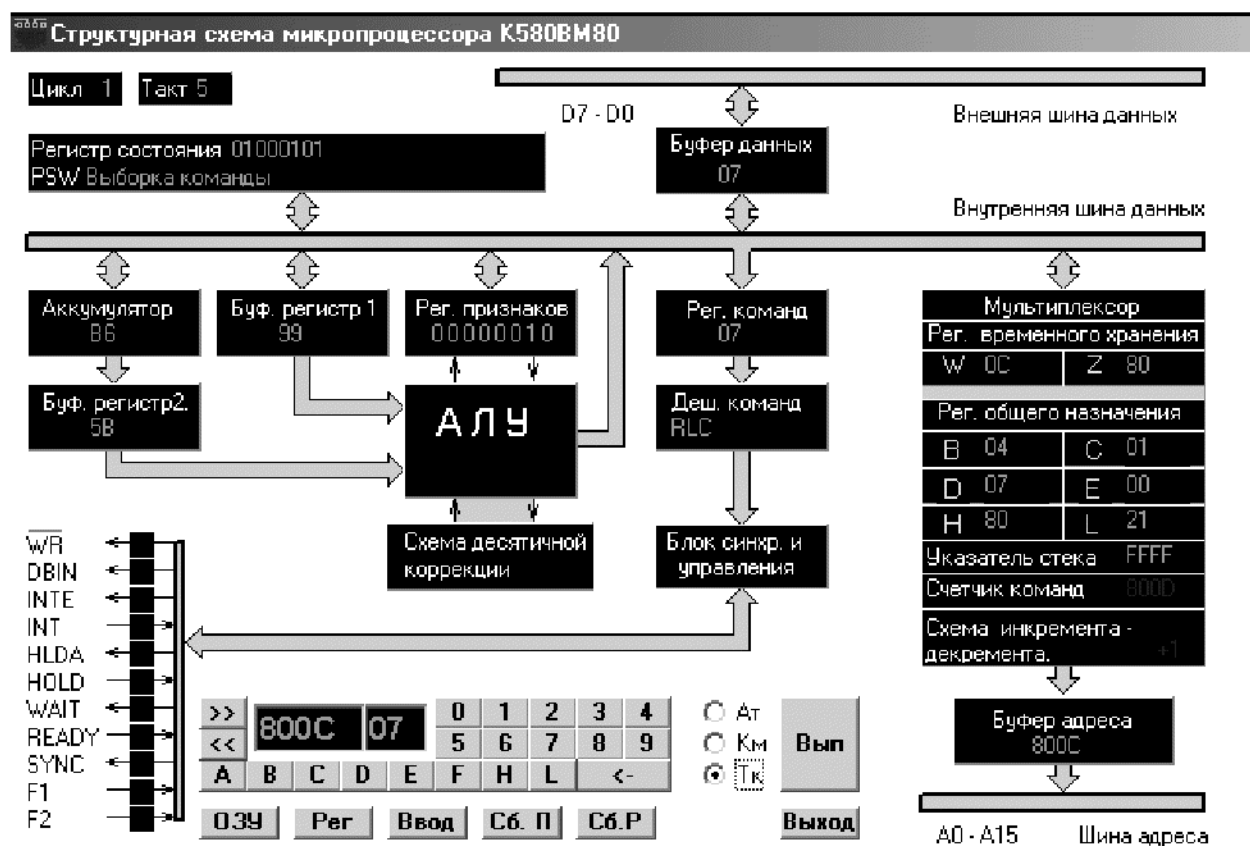


Рисунок 1.2 – Структурная схема микропроцессора K580BM80

Подробное описание структуры приведено в файле справки эмулятора.

Для организации режимов работы, как и в предыдущем окне, в данном расположена клавиатура ввода информации, аналогичная клавиатуре предыдущего окна, и соответствующие режимные клавиши, а также клавиши режима работы.

В потактовом режиме можно исследовать выполнение любой команды. При каждом нажатии кнопки «Вып.» реализуется очередной такт работы процессора. При этом фиксируется на экране все состояния его регистров, номер такта, номер цикла, состояния управляющих сигналов, регистр состояния и *PSW*.

При нажатии левой кнопки мыши в окне «ПМК КР580ВМ80» на позицию «Система команд», раскрывается окно «Команды» в котором представлена таблица системы команд. Код команды определяется по матрице в шестнадцатеричной системе счисления по столбцу и строке таблицы.

В окне команд можно получить «помощь» по назначению команды и правилам ее выполнения, указав эту команду курсором мыши и щелкнув ее два раза левой кнопкой.

Полное описание правил работы с эмулятором ПМК приведено в файле справки, вызываемом в главном окне при нажатии кнопки «Помощь», а также в прилагаемой к эмулятору документации «Описание эмулятора». Файл находится в той же папке, что и файл запуска эмулятора. Важным является приложение Б файла документации, так как в нем указаны особенности поведения эмулятора, которые обязательно необходимо учитывать, во избежание некорректного выполнения кода составленной программы.

При входе во второй эмулятор открывается окно эмулятора «Модель лабораторного стенда УМПК-80», в котором изображена структура «стенда» (рисунок 1.3). Основные элементы главного окна показаны на рисунке 1.4.

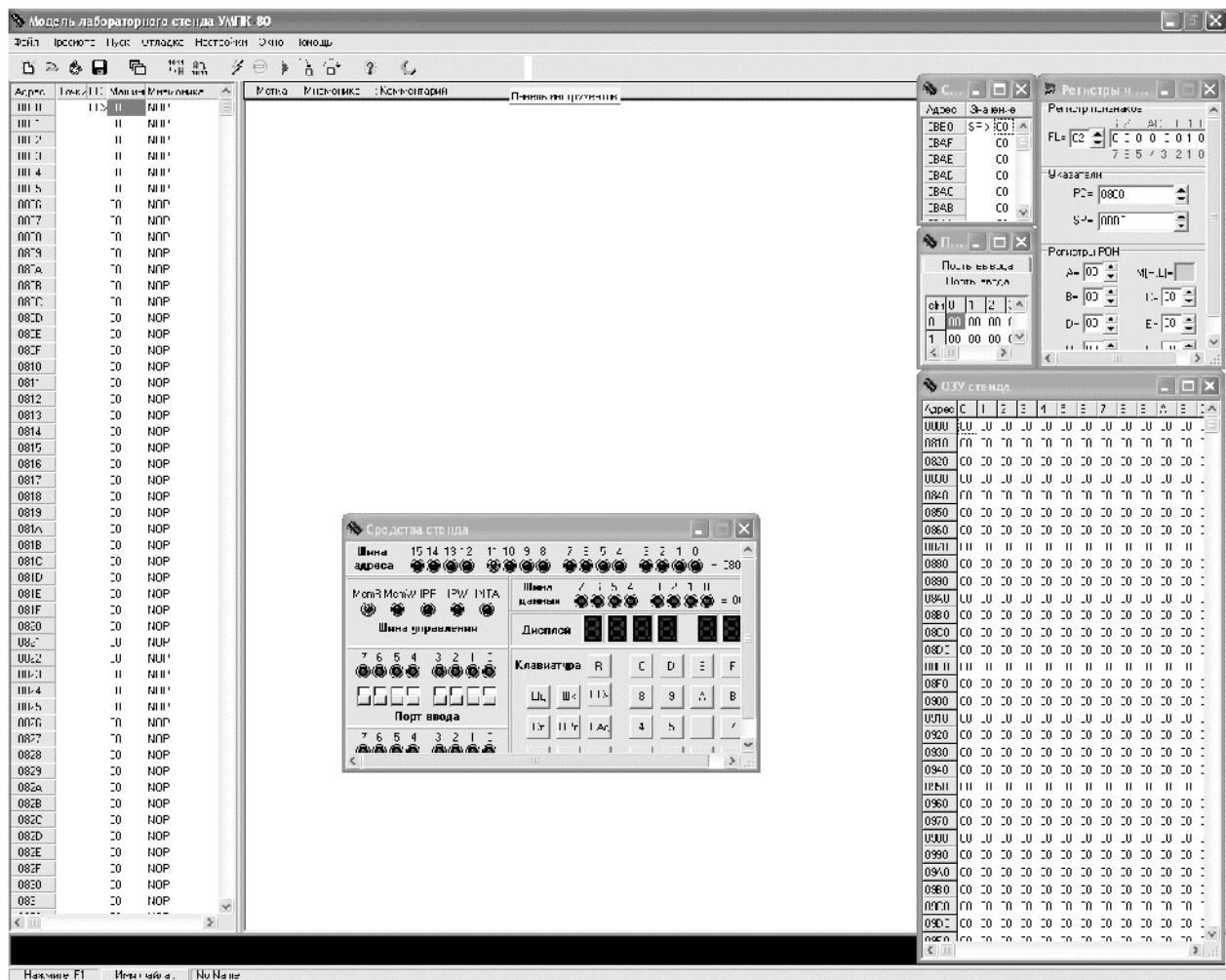


Рисунок 1.3 – Главное окно эмулятора «Модель лабораторного стенда УМПК-80»

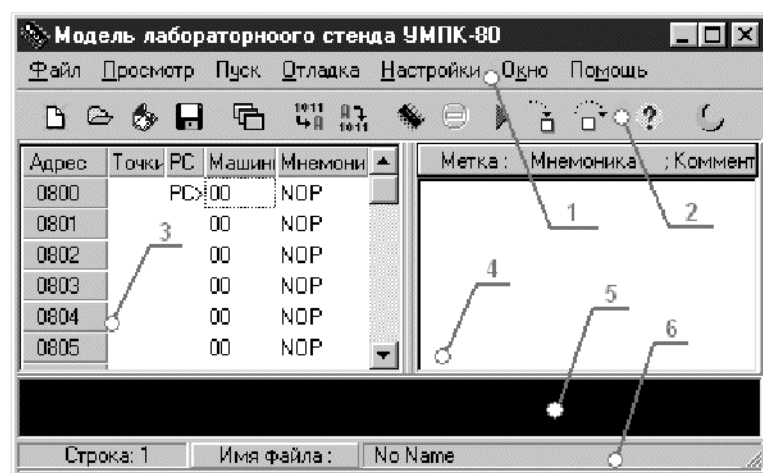


Рисунок 1.4 – Основные элементы главного окна эмулятора УМПК

В состав окна входит:

- 1 – Меню модели;
- 2 – Панель инструментов;
- 3 – Секция машинных кодов;
- 4 – Секция мнемокодов;
- 5 – Секция сообщений;
- 6 – Строка состояния.

Поскольку эмулятор имеет достаточно подробный файл справки по функциям, управлению, назначению кнопок управления и т. д., то в данном методическом указании эта информация не приводится. Файл справки можно вызвать нажатием правой кнопки мыши на кнопке «Помощь» Меню модели. Так же файл находится в той же папке, что и файл запуска эмулятора.

В обоих эмуляторах микро-ЭВМ реализована адресация с помощью карты памяти: каждому физическому устройству макета ставится в соответствие адрес, и обращение к нему при таком типе адресации аналогично обращению к ячейке памяти с использованием всего набора команд МП БИС. На рисунке 1.5 приведена карта памяти микро-ЭВМ.

Биты в старшем байте адреса								Адреса		
15	14	13	12	11	10	9	8			
0	0	0	0	0	0	0	0	0000	Управляющая программа	2 КБ
0	0	0	0	0	1	1	1	07FF	Демонстрационные программы ПЗУ	
0	0	0	0	1	0	0	0	0800	Область адресов для записи исследуемых программ	1 КБ
								0AFF		
								0B00		
								0BB0		
								0BB1		
								0BFF	Область адресов для записи данных управляющей программы ОЗУ	
0	0	0	0	1	1	1	1	0FFF		
0	0	0	1	0	0	0	0	1000	Управление защитой первых 314 адресов ОЗУ от записи УПР	
0	0	0	1	0	1	1	1	17FF		
0	0	0	1	1	0	0	0	1800	Входной регистр чтения клавиатуры РгЧК	
0	0	0	1	1	1	1	1	1FFF		
0	0	1	0	0	0	0	0	2000	Входной регистр РгВВ	
0	0	1	0	0	1	1	1	27FF		
0	0	1	0	1	0	0	0	2800	Регистр сканирования дисплея и клавиатуры РгСк	
0	0	1	0	1	1	1	1	2FFF		
0	0	1	1	0	0	0	0	3000	Выходной регистр РгВыв	
0	0	1	1	0	1	1	1	37FF		
0	0	1	1	1	0	0	0	3800	Регистр сегментов дисплея РгСг	
0	0	1	1	1	1	1	1	3FFF		
1	0	0	0	0	0	0	0	8000	Регистр звукового выхода СхВЗС	
1	0	0	0	0	0	0	1	8001	Неиспользуемые адреса	
1	1	1	1	1	1	1	1	FFFF		

Рисунок 1.5 – Карта памяти эмулируемой микро-ЭВМ

На рисунке 1.6 приведена схема возможных переходов от одного режима работы микро-ЭВМ к другому.

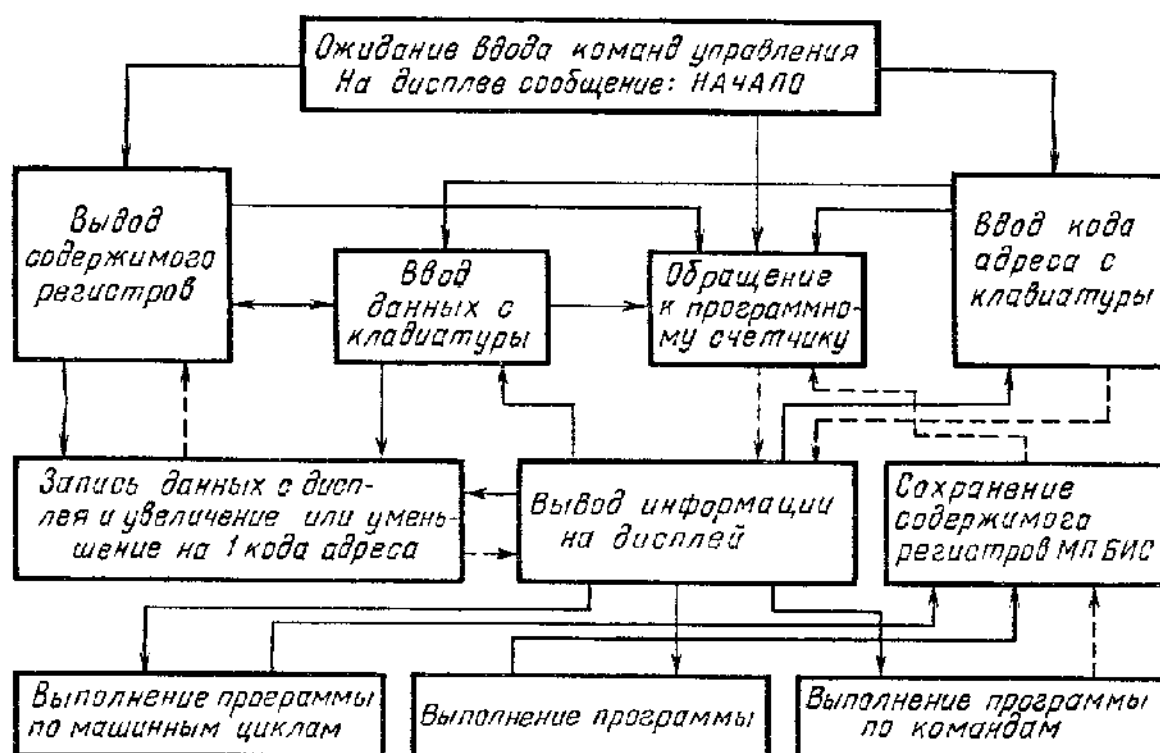


Рисунок 1.6 – Схема перехода к различным режимам работы в микро-ЭВМ

В режиме ожидания ввода команд управления микро-ЭВМ может реагировать на нажатие кнопок: отыскание адреса, отыскание регистра и программный счетчик. Все последующие возможные переходы от режима к режиму можно проследить по рисунку, где сплошными линиями показаны возможные пути изменения режимов работы по командам пользователя, вводимым с клавиатуры управления, а штриховыми – изменения в состояниях микро-ЭВМ, вызванные выполнением соответствующих подпрограмм команд управления.

С помощью этих эмуляторов необходимо выполнить задания данной и следующих лабораторных работ.

Указания по технике безопасности

Перед проведением лабораторной работы следует изучить инструкцию по технике безопасности при работе на компьютере. Также следует соблюдать правила и требования, установленные для данного компьютерного класса.

Задания

1 Задание на подготовку к выполнению экспериментальной части работы.

1.1 Ознакомиться с описанием учебной микро-ЭВМ.

1.2 Ознакомиться с типовой минимальной структурой микро-ЭВМ, методами организации магистралей, подключения памяти и внешних устройств к магистралям.

1.3 Изучить алгоритм работы управляющей программы и возможные режимы работы по рисунку 1.6.

1.4 Рассмотреть работу МП БИС при включении и состоянии магистралей микро-ЭВМ при его выполнении.

1.5 Изучить внутренние регистры МП БИС K580BM80 и временные диаграммы выполнения команд.

2 Задание на выполнение экспериментальной части работы.

2.1 Исследовать порядок запуска эмулятора ПМК, для чего в папке *CPU580* запустить файл *CPU580.EXE*. Состояние Главного окна эмулятора будет таким, как на рисунке 1.7. Индикаторы шин адреса и данных, состояния, флагов будут выключены. Состояния регистров будут «00», кроме *SP* – «FFFF» и *PC* – «0000». В поле команды указано «Начните работу с кнопок «ОЗУ» или «Регистр»». Поля «Позиция» и «Адрес – Код – Команда» будут пустыми.

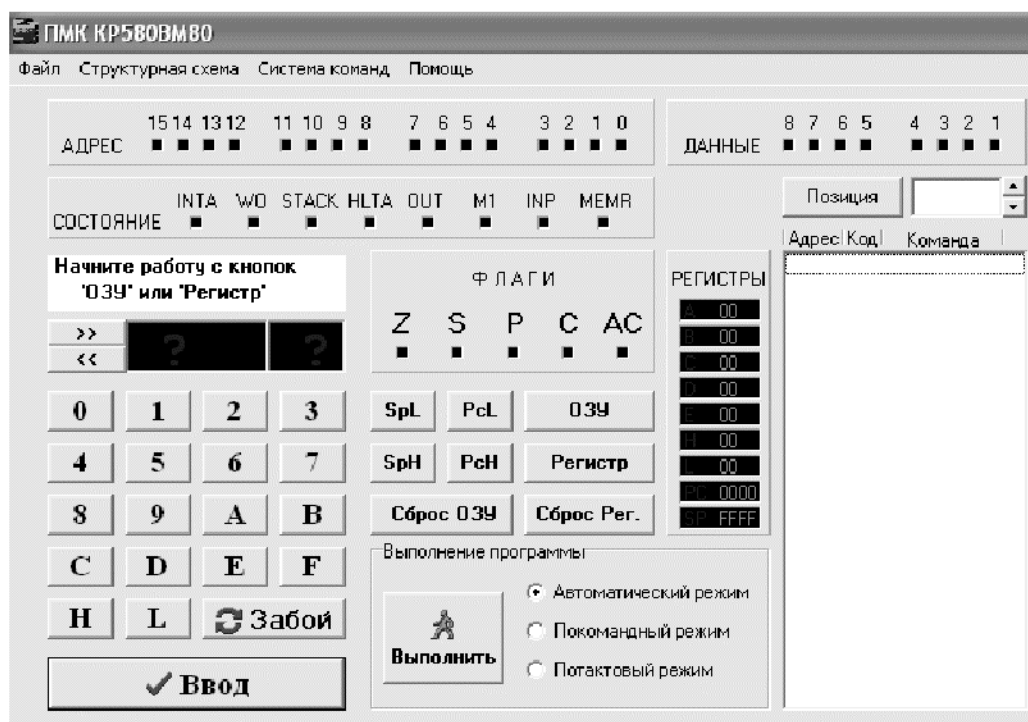


Рисунок 1.7 – Состояние Главного окна эмулятора ПМК после запуска

Из любого места управляющей программы можно вернуть эмулятор к этому начальному состоянию нажатием кнопки «Сброс ОЗУ» (здесь и далее для этого эмулятора необходимо иметь ввиду, что кнопки нажимаются только при наведении курсора мыши и нажатием ее левой кнопки. В результате все ячейки ОЗУ будут обнулены и, следовательно, будет стерта вся программа пользователя, записанная ранее в ОЗУ.

2.2 Исследовать содержимое памяти эмулятора ПМК, для чего необходимо выполнить следующее:

- нажать на кнопку «ОЗУ» при этом на дисплее появится команда «Введите номер ячейки ОЗУ»;
- последовательно нажать кнопки «0», «8», «0», «0». Каждая цифра записывается в разряды адресного дисплея, начиная со старшего разряда. При введении четвертой цифры эмулятор выведет на дисплей число, записанное по этому адресу. В данном случае в разрядах дисплея данных появится число «00». Над дисплеем также будет надпись «Ячейка 0800 не

содержит данных...». И это верно, т. к. при выключении эмулятора из ОЗУ стираются все записанные во время сеанса программы и данные. С помощью кнопок «>>» и «<<» можно перемещаться в адресном поле ОЗУ. Убедиться, что по следующим адресам ячейки ОЗУ также не содержат данных. Кнопка «Забой» выполняет стирание одного, двух или трех разрядов неверно введенного адреса. После введения всех четырех разрядов адреса исправить ситуацию можно только нажатием кнопки «ОЗУ» и введением правильного адреса;

- выполнить команду «Сброс ОЗУ». С помощью команд «Файл» и «Открыть» выбрать тестовый файл *sodergimoe.580* из папки *CPU580*. Посмотреть содержимое ОЗУ по произвольному адресу можно, если нажать кнопку «ОЗУ» и с помощью клавиатуры указать требуемый адрес (например, «0801»). Над дисплеями высветится: «Данные ячейки 0801», а по выбранному адресу на дисплее данных будут показаны записанные данные.

Таким образом, использование этого режима позволяет проверить содержимое всех ячеек памяти эмулируемой микро-ЭВМ.

2.3 Исследовать режим записи в память эмулятора ПМК, для чего необходимо выполнить следующее:

- запустить файл *CPU580.EXE* или выполнить команду «Сброс ОЗУ»;
- нажать на кнопку «ОЗУ» и ввести адрес «0800». Далее нажать кнопки для ввода данных, например, «0» и «1». Каждая цифра записывается в разряды дисплея данных, начиная со старшего разряда. При введении второй цифры эмулятор выведет на дисплей число, записанное по этому адресу. Для записи данных по выбранному адресу в ОЗУ эмулятора необходимо нажать кнопку «Ввод». Сохраненные данные сразу же отобразятся в поле «Адрес – Код – Команда». Поле «Позиция» примет значение «0800». Эмулятор автоматически увеличит адрес ОЗУ на единицу, что отобразится на дисплее адреса. По адресу «0801» необходимо ввести следующие данные, причем поле адреса будет изменяться уже автоматически. Ввести данные, например

«1» и «0», завершив операцию нажатием кнопки «Ввод». С помощью кнопок «>>» и «<<» можно перемещаться в адресном поле ОЗУ. Для коррекции неверно введенных данных нужно указанными кнопками выбрать адрес ячейки ОЗУ и, нажав кнопки с необходимыми данными, подтвердить запись кнопкой «Ввод». Кнопка «Забой» выполняет стирание одного или двух разрядов неверно введенных данных до нажатия кнопки «Ввод»;

- выполнить команду «Сброс ОЗУ». Начиная с адреса «0800» записать последовательно дату и год своего рождения в формате «xx yy zzzz» (где xx – число рождения; yy – месяц рождения; zzzz – год рождения. Например, 08 11 1990). Выполнив команды «Файл» и «Сохранить как...», сохранить полученный код ОЗУ в файл под именем *data1*;

- перезаписать дату и год своего рождения в формате «zzzz xx yy» и сохранить в файл под именем *data2*.

Таким образом, использование этого режима позволяет записать данные в память эмулятора, а также произвести их модификацию.

2.4 Исследовать режим записи в программно-доступные регистры эмулятора ПМК, для чего необходимо выполнить следующее:

- запустить файл *CPU580.EXE* или выполнить команду «Сброс Рег.». Состояния регистров будут «00», кроме *SP* – «FFFF» («0000» – после команды сброса) и *PC* – «0000»;

- нажать на кнопку «Регистр». Над дисплеем появится надпись «Введите имя регистра». Выбрать на клавиатуре имя необходимого регистра, например, «4». В поле адреса высветится имя выбранного регистра, а над дисплеем – строка «Регистр 4 не содержит данных...». Данные в регистр записываются аналогично данным памяти ОЗУ. Записать в регистр 4 число «01», подтвердив запись нажатием кнопки «Ввод». Далее на дисплее адреса эмулятор автоматически покажет следующий регистр 5. Можно ввести в него необходимые данные, а можно оставить без изменений и перейти к следующему регистру кнопками «>>», «<<» (или «Регистр» и «имя»);

– доступ к регистрам *PC* и *SP* осуществляется только после нажатия кнопки «Регистр». Так как эти регистры 16-разрядные, то допустим отдельный доступ как к младшим байтам «*PcL*» и «*SpL*» регистров, так и к старшим «*PcH*» и «*SpH*»;

– выполнить команду «Сброс Рег.». Записать в регистры с *A* по *L* число *Xh*, рассчитанное по формуле:

$$Xh = ih + Nh,$$

где *ih* – порядковый номер регистра;

Nh – номер студента в списке журнала преподавателя.

Порядковые номера регистров принять равными: *A=01h*, *B=02h*,... Индекс *h* указывает, что число должно быть представлено в шестнадцатеричной системе счисления. Для подтверждения выполнения задания необходимо воспользоваться функцией «*PrtScr*»;

– перезаписать содержимое регистров, приняв порядковые номера регистров: *L=01h*, *H=02h*,... Для подтверждения выполнения задания необходимо воспользоваться функцией «*PrtScr*».

Таким образом, использование этого режима позволяет записать данные в программно-доступные регистры эмулятора, а также произвести их модификацию.

2.5 Исследовать режимы запуска и выполнения программ на эмуляторе ПМК, для чего необходимо выполнить следующее:

– запустить файл *CPU580.EXE* или выполнить команды «Сброс ОЗУ», «Сброс Рег.». С помощью команд «Файл» и «Открыть» выбрать тестовый файл *zapusk.580* из папки *CPU580*. Установить начальный адрес выполнения команды (в данном примере – «0800»). Нажать кнопку «Сброс Рег.». В поле «Выполнение программы» выбрать «Покомандный режим». Каждое нажатие кнопки «Выполнить» будет приводить к выполнению текущей команды программы, в регистре *PC* будет автоматически устанавливаться адрес следующей выполняемой команды, а в регистрах МП будут производиться

соответствующие программному коду операции. Если программа длинная, то возможен переход с «Покомандного режима» в «Автоматический режим». Для этого достаточно выбрать этот режим и нажать «Выполнить». Программа будет выполнена полностью. Аналогично можно перейти в «Автоматический режим» из «Потактового режима»;

- вновь загрузить файл *zapusk.580* и выполнить программу в «Покомандном режиме». Об окончании программы свидетельствует отсутствие увеличения счетчика команд при нажатиях кнопки «Выполнить», а также код «76» команды останова программы. Проанализировать изменения состояний шины адреса, данных, линий состояния, флагов и содержимого регистров МП. Для подтверждения выполнения задания необходимо воспользоваться функцией «*PrtScr*»;

- вновь загрузить файл *zapusk.580* и выполнить две команды программы в «Покомандном режиме», затем, переключившись в «Автоматический режим», завершить выполнение программы. Для подтверждения выполнения задания необходимо воспользоваться функцией «*PrtScr*»;

- вновь загрузить файл *zapusk.580* и выполнить программу в «Автоматическом режиме». Для подтверждения выполнения задания необходимо воспользоваться функцией «*PrtScr*»;

- вновь загрузить файл *zapusk.580*. Переключившись в режим «Структурная схема», выполнить программу в «Пошаговом режиме». Проанализировать изменения по структуре МП. Для подтверждения выполнения задания необходимо воспользоваться функцией «*PrtScr*»;

- вновь загрузить файл *zapusk.580*. Переключившись в режим «Структурная схема», выполнить программу в «Потактовом режиме». Проанализировать изменения по структуре МП. Для подтверждения выполнения задания необходимо воспользоваться функцией «*PrtScr*».

Таким образом, использование этого режима позволяет исследовать режимы запуска и выполнения программ, а также поведение МП в ходе выполнения команд программы.

2.6 Исследовать порядок запуска эмулятора УМПК, для чего в папке *UMPK80* запустить файл *K580BM80.EXE*. Используя файл справки, провести исследования эмулятора УМПК, аналогичные пункту 2.1.

2.7 Исследовать содержимое памяти эмулятора УМПК, аналогично пункту 2.2, используя файл справки эмулятора. Использовать тестовый файл *sodergimoe.cod* из папки *UMPK80*.

2.8 Исследовать режим записи в память эмулятора УМПК, аналогично пункту 2.3, используя файл справки эмулятора. Необходимо учесть, что запись и изменение данных в памяти эмулятора возможны как в «Главном окне» в секции «Машинных кодов», так и в «Дополнительном окне» – «ОЗУ стенда». Применить оба способа.

2.9 Исследовать режим записи в программно-доступные регистры эмулятора УМПК, аналогично пункту 2.4, используя файл справки эмулятора.

2.10 Исследовать режимы запуска и выполнения программ на эмуляторе УМПК, аналогично пункту 2.5, используя файл справки эмулятора. Использовать тестовый файл *zapusk.cod* из папки *UMPK80*.

3 Задание на обработку экспериментальных данных.

Используя результаты выполнения основных операций с эмуляторами, выявить:

- основные моменты в работе эмуляторов, удобство использования различных функций;
- разницу в управлении;
- разницу в выполнении основных операций;
- достоинства и недостатки каждого эмулятора.

Содержание отчета

Отчет по лабораторной работе выполняется на листах формат А4 согласно требований ГОСТ и ЕСКД и должен содержать следующие пункты:

- а) титульный лист с названием работы, дисциплины, кода группы, ФИО исполнителя и преподавателя;
- б) цель и содержание работы;
- в) краткое описание эмуляторов;
- г) результаты выполнения пунктов 2.1,..., 2.5, подтвержденные снимками экрана в необходимых ситуациях, созданными согласно заданию файлами, демонстрацией работоспособности эмулятора по заданию;
- д) результаты выполнения пунктов 2.6,..., 2.10, с описанием основных действий, исследованных с помощью файла справки эмулятора УМПК. Подтверждение: снимками экрана в необходимых ситуациях, созданными согласно заданию файлами, демонстрацией работоспособности эмулятора по заданию;
- е) результаты анализа по пункту 3;
- ж) выводы по результатам работы.

Контрольные вопросы

- 1 Изобразить структуру учебной микро-ЭВМ.
- 2 Изобразить карту памяти микро-ЭВМ.
- 3 Какие адреса памяти относятся к ПЗУ и ОЗУ?
- 4 Пояснить назначение отдельных элементов карты памяти.
- 5 Какие существуют режимы работы микро-ЭВМ?
- 6 Как проверить содержимое ячеек памяти ОЗУ?
- 7 Как записать данные в ячейки памяти ОЗУ?
- 8 Как изменить данные, записанные в ячейках памяти ОЗУ?

- 9 Как записать данные в программно-доступные регистры?
- 10 Как изменить данные, записанные в программно-доступные регистры?
- 11 Какие существуют режимы запуска программы?
- 12 Возможен ли переход от одного режима запуска к другому?
- 13 Описать функции заданной преподавателем клавиши управления эмулятора.

Список литературы, рекомендованный к использованию по данной теме

Основная учебная литература

1 Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд., испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

Дополнительная учебная литература

1 Щелкунов, Н. Н. Микропроцессорные средства и системы / Н. Н. Щелкунов, А. П. Дианов. – М. : Радио и связь, 1989. – 288 с.

2 Коффрон, Дж. Технические средства микропроцессорных систем: практический курс / Дж. Коффрон. Пер. с англ. – М. : Мир, 1983. – 344 с.

3 Алексенко, А. Г. Основы микросхемотехники / А. Г. Алексенко. 3-е изд. перераб. и доп. – М. : БИНОМ. Лаборатория знаний, 2009. – 448 с.

ЛАБОРАТОРНАЯ РАБОТА 2

Тема: Исследование записи и выполнения простых программ.

Цель работы: исследование выполнения отдельных команд и простых программ; использование различных методов адресации в программах; запись программ.

Формируемые компетенции: ОК-6, ОК-7, ОПК-6, ОПК-8, ОПК-9, ПК-3.

Теоретическая часть

Микропроцессор БИС КР580ИК80 имеет фиксированный набор команд. Время выполнения команды определяется процессом получения, декодирования и выполнения команды. Это время можно представить состоящим из ряда временных интервалов. Наиболее короткий временной интервал, равный периоду синхросигналов МП БИС, называется машинным тактом. Время, необходимое для извлечения 1 байт информации из памяти или внешнего устройства или выполнения команды, определяемой одним машинным словом, называется машинным циклом. Машинный цикл для МП БИС может включать в себя от трех до пяти машинных тактов. В зависимости от вида команды время выполнения может состоять из от одного до пяти машинных циклов. Для МП БИС имеется 10 различных типов машинных циклов: извлечение кода команды (цикл M_1), чтение данных из памяти, запись данных в память, извлечение данных из стека, запись данных в стек, ввод данных из внешнего устройства, запись данных во внешнее устройство, цикл обслуживания прерывания, останов, обслуживание прерывания в режиме останова. Первым машинным циклом при извлечении любой команды является цикл M_1 .

На каждом машинном МП БИС проверяет состояние сигнала «Готов» на своем входе. Нулевой сигнал на этом входе приостанавливает нормальную

работу МП БИС, при этом на магистралях микро-ЭВМ присутствует вся информация, передаваемая на рассматриваемом машинном цикле. В учебной микро-ЭВМ это используется для исследования выполнения команд по машинным тактам. В этом режиме информация на магистралях микро-ЭВМ отображается светодиодами состояния.

Программа записывается в микро-ЭВМ в последовательных ячейках памяти.

Рассмотрим простейшую программу, которая извлекает число из адреса памяти 0B00, инвертирует его и записывает результат в адрес памяти 0B01.

Мнемокод	Комментарий
<i>LDA 0B00</i>	получить число из адреса 0B00
<i>CMA</i>	инвертировать число
<i>STA 0B01</i>	записать результат по адресу 0B01
<i>RST1</i>	прервать выполнения программы

Рисунок 2.1 – Программа чтения числа из памяти, его инверсии и записи в память (на языке Ассемблер)

При записи программ все числа представляются в шестнадцатеричной системе счисления.

Для записи программы в память микро-ЭВМ необходимо перевести мнемокоды команд в машинные коды. Команды в программе могут быть одно-, двух- или трех- байтные и должны в памяти занимать соответственно один, два или три адреса (рисунок 2.2).

Адрес	Число	Комментарий
0800	3A	код команды <i>LDA</i>
0801	00	младший байт адреса
0802	0B	старший байт адреса

0803	<i>2F</i>	код команды <i>CMA</i>
0804	<i>32</i>	код команды <i>STA</i>
0805	<i>01</i>	младший байт адреса
0806	<i>0B</i>	старший байт адреса
0807	<i>CF</i>	код команды <i>RST1</i>

Рисунок 2.2 – Программа чтения числа из памяти, его инверсии и записи в память (размещение по адресам памяти)

Предварительную запись программ удобно проводить в более компактной форме. В программе указывается начальный адрес каждой команды и при этом понимается, что в зависимости от длины (одно-, двух- или трехбайтная) команды в памяти будут занимать от одной до трех последовательных ячеек. При такой записи в левом столбце указываются лишь адреса команд в программе. Это позволяет сократить объем при описании программ и сделать более простым их анализ (рисунок 2.3).

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	<i>3A 000B</i>		<i>LDA 0B00</i>	получить число из адреса <i>0B00</i>
0803	<i>2F</i>		<i>CMA</i>	инвертировать число
0804	<i>32 010B</i>		<i>STA 0B01</i>	записать результат по адресу <i>0B01</i>
0807	<i>CF</i>		<i>RST1</i>	прервать выполнения программы

Рисунок 2.3 – Программа чтения числа из памяти, его инверсии и записи в память (общий вид записи)

Здесь используется прямой способ адресации («Программа 1»).

Рассмотрим аналогичную программу с использованием косвенного способа адресации – «Программа 2» (рисунок 2.4).

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	21	000B	<i>LXIH 0B00</i>	записать в регистры <i>H, L</i> 0B00
0803	7E		<i>MOVA, M</i>	получить число из адреса, указанного в регистрах <i>H, L</i>
0804	2F		<i>CMA</i>	инвертировать число в аккумуляторе
0805	23		<i>INXH</i>	увеличить на 1 число в регистрах <i>H, L</i>
0806	77		<i>MOV M, A</i>	записать число из аккумулятора по адресу, указанному в <i>H, L</i>
0807	CF		<i>RST1</i>	прервать выполнения программы

Рисунок 2.4 – Пример программы с косвенной адресацией

Оборудование и материалы

В ходе выполнения лабораторной работы будут использоваться два эмулятора: «Эмулятор работы микропроцессора КР580ВМ80» (далее ПМК – по заголовку главного окна эмулятора) и «Программная модель стенда УМПК-80 на базе микропроцессора К580ВМ80» (далее УМПК – по заголовку главного окна эмулятора). Первый эмулятор находится в папке *CPU580*, второй – в папке *UMPK80*.

Указания по технике безопасности

Перед проведением лабораторной работы следует изучить инструкцию по технике безопасности при работе на компьютере. Также следует

соблюдать правила и требования, установленные для данного компьютерного класса.

Задания

1 Задание на подготовку к выполнению экспериментальной части работы.

1.1 Ознакомьтесь с языком программирования и структурой команд МП БИС KP580BM80.

1.2 Изучите режимы работы и временные диаграммы процесса выполнения команд МП БИС KP580BM80.

1.3 Изучите методы программирования на языке ассемблера и в машинных кодах для МП БИС KP580BM80.

1.4 Рассмотрите правила выполнения команд *INR A (3C)*, *DCRA (3D)*, *ADD A (87)*, *ANA A (A7)*, *ORA A (B7)*, *CMP A (BF)*, *DAA (27)*.

1.5 Рассмотрите результат выполнения «Программы 1» с прямой адресацией при записи по адресу 0803 команд, приведенных в п. 1.4 задания. Результаты выполнения программы при различных командах, записанных в программе с прямой адресацией (рисунок 2.3) по адресу 0803, занести в таблицу 2.1.

Таблица 2.1 – Результат выполнения программы с прямой адресацией при различных командах, записанных по адресу 0803

Число, записанное по адресу 0B00	Команда, записанная по адресу 0803	Число, записанное по адресу 0B01

1.6 Видоизмените и запишите программу примера с косвенной адресацией «Программу 2» (рисунок 2.4) так, чтобы при ее выполнении исследуемое число первоначально записывалось по адресу 0B00.

1.7 Разработайте программы:

а) увеличения на пять числа, записанного по адресу 0B00, и записи результата по адресу 0B01 («Программа 3»);

б) сложения чисел, записанных по адресам 0B00 и 0BA0, и записи результата по адресу 0B01 («Программа 4»);

в) сравнения чисел в адресах 0B00 и 0B01 и записи большего из них в регистр B («Программа 5»).

2 Задание на выполнение экспериментальной части работы.

2.1 Исследовать «Программу 1», для чего выполнить следующее:

- ввести в микро-ЭВМ «Программу 1»;
- записать по адресу 0B00 исследуемое число (задает преподаватель);
- осуществить пуск «Программы 1» с адреса 0800. Проверить результат выполнения программы путем исследования числа, записанного по адресу 0B01;

- исследовать процесс выполнения программы по командам. После выполнения каждой команды проанализировать содержание всех программно-доступных регистров МП БИС;

- исследовать процесс выполнения команд в «Программе 1» по машинным циклам. Обратить внимание на последовательность передачи и преобразования информации в микро-ЭВМ при выполнении каждой команды;

- заменяя в «Программе 1» команду *CMA* на команды *INR A* (3C), *DCR A* (3D), *ADD A* (87), *ANA A* (A7), *ORA A* (B7), *CMP A* (BF), *DAA* (27), исследовать результат выполнения указанных команд по числу, записанному по адресу 0B01. Проверить таблицу 2.1, заполненную при выполнении домашнего задания.

Задание необходимо выполнить в обоих эмуляторах. Исследование выполнения программы по командам и машинным циклам проводить в эмуляторе ПМК, открыв «Структурную схему микропроцессора K580BM80». Результат выполнения каждого шага сопровождать снимками экрана.

2.2 Исследовать «Программу 2», для чего выполнить следующее:

- ввести в микро-ЭВМ «Программу 2»;
- записать по адресу 0B00 исследуемое число (задает преподаватель);
- осуществить пуск программы с адреса 0800. Проверить результат выполнения программы по числу, записанному по адресу 0B01;
- исследовать процесс выполнения команды *MOV A, M* по машинным циклам;
- ввести и исследовать выполнение микро-ЭВМ видоизмененной «Программы 2», позволяющей первоначально записывать исследуемое число по адресу 0B00.

Задание необходимо выполнить в обоих эмуляторах. Исследование выполнения команды *MOV A, M* по машинным циклам проводить в эмуляторе ПМК, открыв «Структурную схему микропроцессора K580BM80». Результат выполнения каждого шага сопровождать снимками экрана.

2.3 Исследовать «Программу 3», для чего выполнить следующее:

- ввести в микро-ЭВМ «Программу 3», разработанную при домашней подготовке;
- записать по адресу 0B00 исследуемое число (задает преподаватель);
- осуществить пуск программы с адреса 0800. Проверить результат выполнения программы по числу, записанному по адресу 0B01.

Задание необходимо выполнить в обоих эмуляторах. Результат выполнения сопровождать снимками экрана.

2.4 Исследовать «Программу 4», для чего выполнить следующее:

- ввести в микро-ЭВМ «Программу 4», разработанную при домашней подготовке;

- записать по адресам 0B00 и 0BA0 исследуемые числа (задает преподаватель). Необходимо выполнить два варианта, когда сумма двух чисел не превышает восьмиразрядное число и когда будет превышать, о чем необходимо предупредить преподавателя;

- осуществить пуск программы с адреса 0800. Проверить результат выполнения программы по числу, записанному по адресу 0B01;

- видоизменить и исследовать «Программу 4» для случая, когда сумма двух чисел будет превышать восьмиразрядное число.

Задание необходимо выполнить в обоих эмуляторах. Результат выполнения сопровождать снимками экрана.

2.5 Исследовать «Программу 5», для чего выполнить следующее:

- ввести в микро-ЭВМ «Программу 5», разработанную при домашней подготовке;

- записать по адресам 0B00 и 0B01 исследуемые числа (задает преподаватель);

- осуществить пуск программы с адреса 0800. Проверить результат выполнения программы по числу, записанному в регистре B.

Задание необходимо выполнить в обоих эмуляторах. Результат выполнения сопровождать снимками экрана.

3 Задание на обработку экспериментальных данных.

3.1 Используя результаты выполнения п. 2.1 сделать выводы:

- о влиянии команд на заданное число;

- о влиянии команд на состояние флагов микропроцессора.

3.2 Сравнить таблицу 2.1, полученную при выполнении домашнего задания и при выполнении п. 2.1. В случае несовпадения объяснить причину.

3.3 Сделать выводы об операции сложения по п. 2.4, когда сумма двух чисел не превышает восьмиразрядное число и когда превышает. Сделать выводы о влиянии на состояние флагов микропроцессора.

Содержание отчета

Отчет по лабораторной работе выполняется на листах формат А4 согласно требований ГОСТ и ЕСКД и должен содержать следующие пункты:

- а) титульный лист с названием работы, дисциплины, кода группы, ФИО исполнителя и преподавателя;
- б) цель и содержание работы;
- в) машинные коды «Программы 1» и «Программы 2»;
- г) заполненную таблицу 2.1 для случаев выполнения «Программы 1» при использовании команд, представленных в п. 1.4 задания на подготовку к выполнению экспериментальной части работы;
- д) видоизмененную «Программу 2», записанную в машинных кодах, позволяющую первоначально записывать исследуемое число по адресу 0B00;
- е) разработанные в процессе домашней подготовки «Программу 3», «Программу 4» и «Программу 5», записанные в машинных кодах;
- ж) результаты исследований работы программ по п. 2.3, 2.4, 2.5 заданий.
- з) результаты выполнения по пункту 3;
- и) выводы по результатам работы.

Контрольные вопросы

- 1 За сколько машинных тактов выполняется каждая команда в «Программах 1 и 2»?
- 2 Укажите различия в способах адресации, используемых в микро-ЭВМ, построенной на основе микропроцессора серии K580.
- 3 Укажите все возможные способы адресации, используемые при составлении программы по п. 1.7.

4 При выполнении каких команд, приведенных в п. 1.4, задействуются разряды регистра состояния МП БИС?

5 Сформулируйте правила выполнения МП БИС команд, приведенных в п. 1.4.

6 Изобразите временные диаграммы процесса выполнения микро- ЭВМ следующих команд:

– $LDA <A_2> <A_1>$;

– CMA ;

– $STA <A_2> <A_1>$;

– $MOVM, A$;

– $MOV A, M$;

– $INXM$.

Список литературы, рекомендованный к использованию по данной теме

Основная учебная литература

1 Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд., испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

Дополнительная учебная литература

1 Шевкопляс Б.С. Микропроцессорные структуры. Инженерные решения: Справочник. - М.: Радио и связь, 1990 – 512 с.

2 Щелкунов, Н. Н. Микропроцессорные средства и системы / Н. Н. Щелкунов, А. П. Дианов. – М. : Радио и связь, 1989. – 288 с.

3 Коффрон, Дж. Технические средства микропроцессорных систем: практический курс / Дж. Коффрон. Пер. с англ. – М. : Мир, 1983. – 344 с.

4 Шахнов, В. А. Микропроцессоры и микропроцессорные комплекты

интегральных микросхем : справочник / В.-Б. Б. Абрайтис, Н. Н. Аверьянов, А. И. Белоус [и др.]: под ред. В. А. Шахнова. – Том 1. – М. : Радио и связь, 1988. – 368 с.

5 Соппа, И. В. Микропроцессоры и микроконтроллеры / И. В. Соппа, А. С. Самардак, В. А. Иванов. – Владивосток : Издательство Дальневосточного университета, 2009. – 211 с.

6 Алексенко, А. Г. Основы микросхемотехники / А. Г. Алексенко. 3-е изд. перераб. и доп. – М. : БИНОМ. Лаборатория знаний, 2009. – 448 с.

ЛАБОРАТОРНАЯ РАБОТА 3

Тема: Исследование операций ввода-вывода и изучение маскирования данных, организации условных переходов.

Цель работы: исследование методов подключения и организации обмена информацией с простейшими устройствами ввода-вывода. Изучение программных способов маскирования данных и организации условных переходов в микро-ЭВМ.

Формируемые компетенции: ОК-6, ОК-7, ОПК-6, ОПК-8, ОПК-9, ПК-1, ПК-3.

Теоретическая часть

К командам ввода-вывода МП БИС КР580ВМ80 относятся команды $IN < A_1 >$ и $OUT < A_1 >$. При выполнении команды $IN < A_1 >$ микро-ЭВМ считывает число из входного устройства с адресом (A_1) (A_1) и записывает его в аккумулятор. При выполнении команды $OUT < A_1 >$ МП БИС записывает число из аккумулятора в выходное устройство с адресом (A_1) (A_1). Так как адрес устройства указывается в одном байте, то с помощью этих команд микро-ЭВМ может обмениваться информацией не более чем с 256 внешними устройствами.

В качестве простейших устройств ввода-вывода могут использоваться восьмиразрядные регистры (например, многорежимный буферный регистр (МБР) К589ИР12). Обмен данными между микро-ЭВМ и внешними устройствами может вызываться как в определенных местах в программе, так и по сигналам прерывания. В последнем случае подпрограмма обмена данными с внешним устройством будет вызываться за счет перевода микро-ЭВМ в режим обслуживания прерывания. Схемы подключения МБР К589ИР12 при использовании его в качестве устройства ввода-вывода и

организации обмена информацией как по сигналам, формируемым микро-ЭВМ, так и по сигналам запросов прерывания приведены на рисунке 3.1.

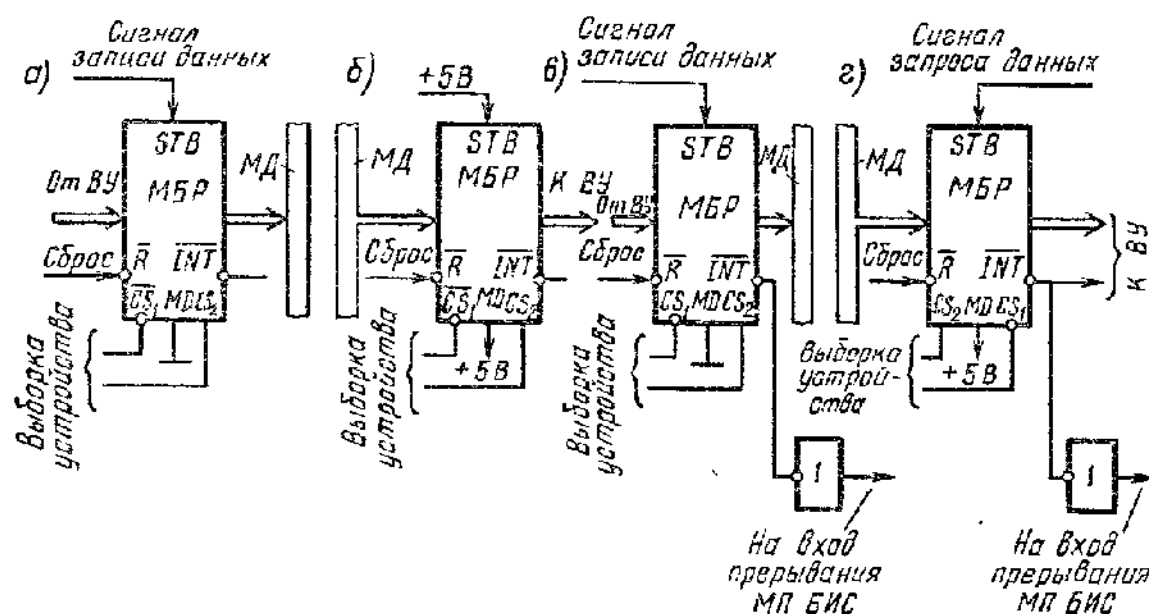


Рисунок 3.1 – Использование схемы многорежимного буферного регистра (МБР) К589ИР12 в качестве устройства ввода-вывода при различных способах обмена информацией

На рисунке 3.1, а) и б) показаны схемы подключения МБР при работе его в качестве устройства ввода-вывода и осуществлении обмена информацией с ним по сигналам, формируемым микро-ЭВМ.

На рисунке 3.1, в) и г) приведены схемы подключения МБР к микро-ЭВМ для обмена информацией по сигналам прерывания. В этом случае внешнее устройство записывает данные во входное устройство (рисунок 3.1, в) по сигналу, подаваемому на вход STB многорежимного буферного регистра. Этим же сигналом формируется сигнал прерывания на выходе $\overline{INT}$ схемы, подаваемый на вход прерывания МП БИС. Вызванная подпрограмма обслуживания прерывания заставляет микро-ЭВМ обратиться ко входному устройству для получения данных. Схема подключения МБР и микро-ЭВМ для вывода данных во внешнее устройство приведена на

рисунке 3.1, г. При поступлении сигнала запроса от внешнего устройства (ВУ) на вход *STB* многорежимный буферный регистр вырабатывает сигнал низкого уровня на выходе $\overline{INT}$, который может подаваться на вход прерывания МП БИС. Подпрограмма обслуживания этого прерывания записывает данные в МБР по сигналам выборки. Этими же сигналами сбрасывается внутренний триггер запроса прерывания МБР, что приводит к появлению сигнала единичного уровня на выходе $\overline{INT}$, который сообщает внешнему устройству о приеме данных от микро-ЭВМ в МБР.

В качестве устройств ввода-вывода могут применяться и более сложные схемы, например программируемое устройство ввода-вывода информации в параллельном коде (КР580ИК55). Схема подключения к микро-ЭВМ входного устройства, выполненного на базе МБР К589ИР12 (*DD1*) с переключателями, приведена на рисунке 3.2, а. При замкнутом переключателе на вход регистра подается «0», а при разомкнутом – «1». Переключатели используются для имитации передачи данных от внешнего устройства. К регистру можно подключить светодиоды (*HL1*, ..., *HL8*) для индикации чисел, записанных в нем. На рисунке 3.2, б приведена схема подключения выходного устройства микро-ЭВМ, построенная на базе схемы К589ИР12 (*DD1*). Светодиоды *HL1*, ..., *HL8* указывают число, записанное в выходном устройстве.

Простейшая программа («Программа 1») перезаписи числа с входного устройства с адресом *05h* (переключатели *S1*, ..., *S8* рисунка 3.2, а) в выходное устройство с адресом *05h* (светодиоды *HL1*, ..., *HL8* рисунка 3.2, б) имеет вид, показанный на рисунке 3.3.

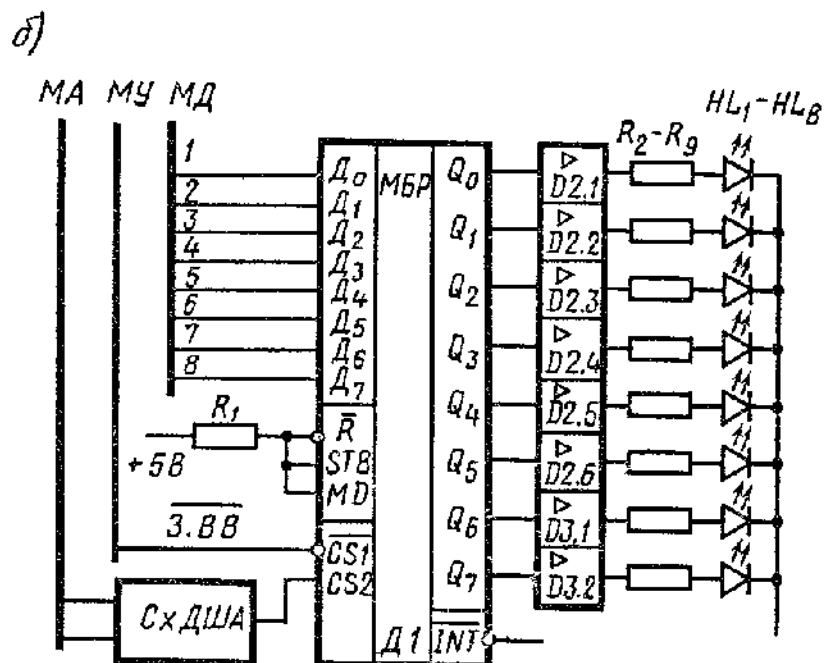
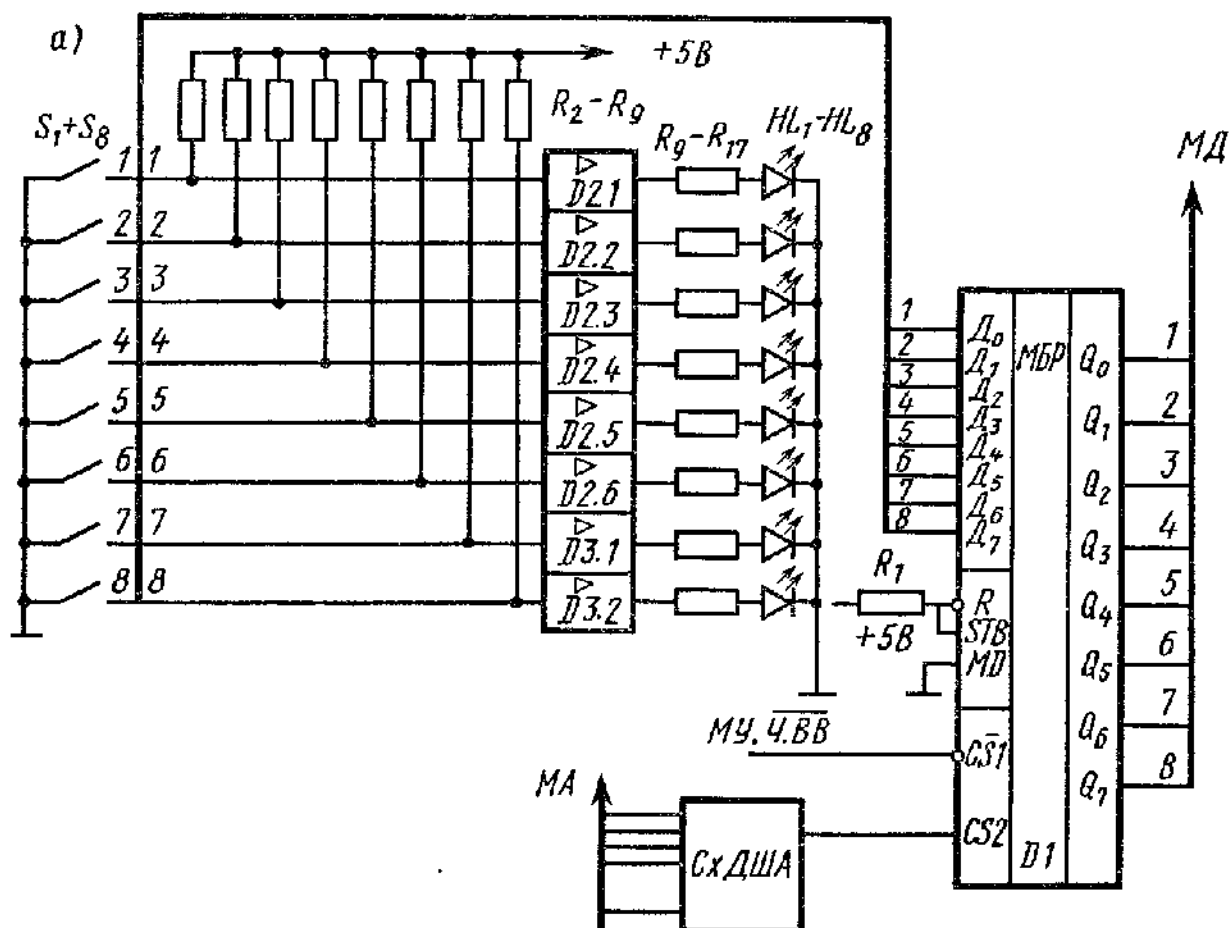


Рисунок 3.2 – Схемы электрические принципиальные подключения входного (а) и выходного (б) устройств к микро-ЭВМ

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	<i>DB 05</i>	<i>CNT</i>	<i>IN 05</i>	записать число из входного устройства с адресом <i>05h</i> в аккумулятор
0802	<i>D3 05</i>		<i>OUT 05</i>	записать число из аккумулятора в выходное устройство с адресом <i>05h</i>
0804	<i>C3 0008</i>		<i>JMP CNT</i>	перейти на метку <i>CNT</i>

Рисунок 3.3 – «Программа 1» перезаписи числа с входного устройства с адресом *05h* в выходное устройство с адресом *05h*

Организация условных переходов в микро-ЭВМ осуществляется с помощью регистра признаков МП БИС.

Регистр признаков имеет пять разрядов, каждый из которых устанавливается по определенному правилу в соответствии с выполнением МП БИС последней команды. Этими разрядами являются:

а) разряд переполнения *C* – *CARRY*. В него записывается «1», если при выполнении арифметической команды или команды сдвига было переполнение аккумулятора, в противном случае в разряд записывается «0»;

б) разряд знака *S* – *SIGN*. В него записывается «1», если при выполнении арифметической или логической команды в старшем, седьмом, разряде аккумулятора записана «1», в противном случае в разряд записывается «0»;

в) разряд нулевого результата *Z* – *ZERO*. В него записывается «1», если при выполнении арифметической или логической команды во всех разрядах числа в аккумуляторе имеются «0», в противном случае в разряд записывается «0»;

г) дополнительный разряд переполнения $AC - AUX. CARRY$. В него записывается «1», если при выполнении команд в аккумуляторе возникает единица переноса из третьего разряда числа;

д) разряд четности $P - PARITY$. В него записывается «1», если при выполнении команды количество единиц в разрядах аккумулятора будет четным.

Во многих случаях при выполнении программ необходимо проверять или изменять (маскировать) состояние одного или нескольких разрядов числа в аккумуляторе. Это можно осуществить с помощью следующих операций:

а) логического умножения числа в аккумуляторе и маски, которое очищает разряд числа, если в соответствующем разряде маски будет записан «0», и не изменяет его, если в разряде маски записана «1»;

б) логического сложения числа в аккумуляторе и маски, которое устанавливает разряд числа в «1», если в таком же разряде маски будет записана «1», и не изменяет его, если в этом разряде записан «0»;

в) логического «исключающего ИЛИ» числа в аккумуляторе и маски, которое инвертирует содержание разряда числа, если в соответствующем разряде маски записана «1», и не изменяет его, если в этом разряде записан «0».

Проведение логических операций возможно также с содержимым аккумулятора и внутренними регистрами МП БИС. В этом случае команды будут однобайтные. При выполнении всех логических команд задействуются разряды Z, S, P, AC регистра признаков (в разряд C записывается «0»). Это позволяет проверять состояние любого разряда числа и выполнять условные переходы в программах. В «Программе 2» выполняется маскирование отдельных разрядов числа (рисунок 3.4), записанного во входном устройстве по адресу $05h$. Программа помещает результат маскирования в выходное устройство по адресу $05h$.

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	<i>DB 05</i>	<i>CNT</i>	<i>IN 05</i>	записать число из входного устройства с адресом <i>05h</i> в аккумулятор
0802	<i>E6 20</i>		<i>ANI 20</i>	выполнение логической операции
0804	<i>D3 05</i>		<i>OUT 05</i>	записать число из аккумулятора в выходное устройство с адресом <i>05h</i>
0807	<i>C3 0008</i>		<i>JMP CNT</i>	перейти на метку <i>CNT</i>

Рисунок 3.4 – «Программа 2» маскирования отдельных разрядов числа

Условные переходы организуются в программах с помощью команд условных переходов. При выполнении этих команд МП БИС проверяет состояние соответствующего разряда регистра состояния. Если при проверке состояния разряда регистра состояния условие не подтверждается, то выполняется следующая по порядку команда программы. Все команды условных переходов – трехбайтные: первый байт содержит код команды, второй и третий байты – адрес передачи управления. Таким образом, команды условных переходов позволяют строить ветвящиеся алгоритмы и в зависимости от текущего значения результата выполнения программы переходить на различные участки программы.

«Программа 3» (рисунок 3.5) предназначена для определения «1» в пятом разряде числа, записанного во входном устройстве. Программа использует маскирование числа и условный переход.

В представленных ранее программах имел место лишь один цикл, в котором работала микро-ЭВМ. Программа ожидания появления «1» во втором и пятом разрядах числа, записанного во входном устройстве («Программа 4»), содержит два цикла, и приведена на рисунке 3.6.

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	<i>DB 05</i>	<i>WAIT</i>	<i>IN 05</i>	записать число из входного устройства с адресом <i>05h</i> в аккумулятор
0802	<i>E6 20</i>		<i>ANI 20</i>	проверить состояние пятого разряда числа
0804	<i>CA 0008</i>		<i>JZ WAIT</i>	перейти на метку <i>WAIT</i> , если в пятом разряде был «0» ($Z = 1$)
0807	<i>CF</i>		<i>RTS1</i>	окончить выполнение программы

Рисунок 3.5 – «Программа 3» определения «1» в пятом разряде числа с условным переходом

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	<i>DB 05</i>	<i>WAIT1</i>	<i>IN 05</i>	записать число из входного устройства с адресом <i>05h</i> в аккумулятор
0802	<i>E6 04</i>		<i>ANI 04</i>	выключен ли второй переключатель?
0804	<i>CA 0008</i>		<i>JZ WAIT1</i>	если нет, продолжить <i>WAIT1</i>
0807	<i>3E FF</i>		<i>MVI A, FF</i>	если да, включить светодиоды
0809	<i>D3 05</i>		<i>OUT 05</i>	выходного регистра
080B	<i>DB 05</i>	<i>WAIT2</i>	<i>IN 05</i>	записать число из входного устройства с адресом <i>05h</i> в аккумулятор
080D	<i>E6 20</i>		<i>ANI 20</i>	выключен ли пятый

			переключатель?
080F	CA 0B08	JZ WAIT2	если нет, продолжить WAIT2
0812	3E 00	MVI A, 00	если да, выключить светодиоды
8014	D3 05	OUT 05	выходного регистра
0816	C3 0008	JMP WAIT1	повторить программу

Рисунок 3.6 – «Программа 4» ожидания появления «1» во втором и пятом разрядах числа

Оборудование и материалы

В ходе выполнения лабораторной работы будет использоваться эмулятор «Программная модель стенда УМПК-80 на базе микропроцессора K580BM80» (далее УМПК – по заголовку главного окна эмулятора). Эмулятор находится в папке *UMPK80*.

Указания по технике безопасности

Перед проведением лабораторной работы следует изучить инструкцию по технике безопасности при работе на компьютере. Также следует соблюдать правила и требования, установленные для данного компьютерного класса.

Задания

1 Задание на подготовку к выполнению экспериментальной части работы.

1.1 Ознакомьтесь со схемой многорежимного буферного регистра и схемой программируемого устройства ввода-вывода информации в

параллельном коде KP580BB55.

1.2 Изучите способы организации обмена информацией между микро-ЭВМ и внешними устройствами. Рассмотрите схемы подключения устройств ввода-вывода данных при различных способах обмена.

1.3 Ознакомьтесь с командами ввода-вывода МП БИС KP580BM80, а также временными диаграммами их выполнения.

1.4 Изобразите схемы подключения к микро-ЭВМ устройств ввода-вывода, используемых при проведении лабораторной работы.

1.5 Изучите группу логических команд и команд условной передачи управления.

1.6 Ознакомьтесь с разрядами регистра признаков МП БИС и правилами записи в них «1».

1.7 Ознакомьтесь с программами «Программа 1», «Программа 2», «Программа 3», «Программа 4».

1.8 Самостоятельно разработайте программы:

а) включения светодиодов выходного устройства, если число, записанное во входном устройстве, больше трех;

б) включения светодиодов выходного устройства, если число, записанное во входном устройстве, больше трех, но меньше восьми.

1.9 Видоизмените «Программу 4» так, чтобы микро-ЭВМ реагировала на «0» в пятом разряде при записанных «1» во всех остальных разрядах.

2 Задание на выполнение экспериментальной части работы.

2.1 Исследовать «Программу 1», для чего выполнить следующее:

– ввести в микро-ЭВМ «Программу 1»;

– осуществить пуск программы;

– убедиться, что при выполнении программы микро-ЭВМ постоянно переписывает данные с входного устройства в выходное. Для этого с помощью переключателей входного устройства изменить числа, записанные в нем. Информация о числах в устройствах ввода-вывода будет отображаться

светодиодами;

- установить заданное преподавателем число на входном устройстве.

Результат выполнения представить в виде снимка экрана.

2.2 Исследовать «Программу 2», для чего выполнить следующее:

- ввести в микро-ЭВМ «Программу 2»;
- осуществить пуск программы;
- исследовать результат ее выполнения по числу, записанному в выходное устройство. Входное число и «маску» задает преподаватель.

Результат выполнения представить в виде снимка экрана;

– заменяя в «Программе 2» двухбайтную команду *ANI <D>* на однобайтные *ANA A*, *XRA A*, *ORA A*, исследовать результат их выполнения по числу, записанному в выходном устройстве. Входное число задает преподаватель. Результат выполнения представить в виде снимка экрана.

2.3 Исследовать «Программу 3», для чего выполнить следующее:

- ввести в микро-ЭВМ «Программу 3»;
- осуществить пуск программы;
- убедиться, что при ее выполнении микро-ЭВМ реагирует лишь на те числа во входном устройстве, которые содержат «1» в пятом разряде. После окончания выполнения программы (выполнения в программе команды *RST1*) в разряде *Z* регистра состояния должен быть записан «0». Результат выполнения представить в виде снимка экрана;

– исследовать видоизмененную «Программу 3», позволяющую микро-ЭВМ реагировать на «0» в пятом разряде при записанных «1» во всех остальных разрядах. Результат выполнения представить в виде снимка экрана;

– установить заданные преподавателем условия для исследования видоизмененной «Программы 3». Результат выполнения представить в виде снимка экрана.

2.4 Исследовать «Программу 4», для чего выполнить следующее:

- ввести в микро-ЭВМ «Программу 4»;
- осуществить пуск программы;
- убедиться, что при наличии «1» лишь во втором разряде числа входного регистра светодиода выходного регистра включены и микро-ЭВМ работает в цикле *WAIT2* ожидания появления единицы в пятом разряде числа. Результат выполнения представить в виде снимка экрана;

- записать «0» во второй разряд входного устройства при включенных светодиодах выходного устройства. Записать «1» в пятый разряд входного устройства и убедиться, что светодиоды выходного устройства выключаются и микро-ЭВМ находится при выполнении цикла *WAIT1* программы. Результат выполнения представить в виде снимка экрана;

- установить «1» одновременно во втором и пятом разрядах числа во входном устройстве и проверить, что микро-ЭВМ последовательно выполняет оба цикла (*WAIT1* и *WAIT2*) программы. Результат выполнения представить в виде снимка экрана;

- установить заданные преподавателем условия для исследования «Программы 4». Результат выполнения представить в виде снимка экрана.

2.5 Исследовать программы, разработанные в п. 1.8, для чего выполнить следующее:

- ввести в микро-ЭВМ каждую программу;
- осуществить пуск программы;
- убедиться, что светодиоды выходного устройства включаются только при условии, что число, записанное во входном устройстве больше трех (для задания п. 1.8, а). Результат выполнения представить в виде снимка экрана;

- убедиться, что светодиоды выходного устройства включаются только при условии, что число, записанное во входном устройстве больше трех, но меньше восьми (для задания п. 1.8, б). Результат выполнения представить в виде снимка экрана;

- установить заданные преподавателем условия для исследования

программ п. 1.8. Результаты выполнения представить в виде снимка экрана.

3 Задание на обработку экспериментальных данных.

3.1 Используя результаты выполнения п. 2.1 привести фрагмент машинного кода «Программы 1», необходимый для установки условий, заданных преподавателем.

3.2 Используя результаты выполнения п. 2.2 привести фрагмент машинного кода «Программы 2», необходимый для установки условий, заданных преподавателем.

3.3 Используя результаты выполнения п. 2.2 объяснить действие команд *ANA A*, *XRA A*, *ORA A*, установленных в «Программе 2» вместо команды *ANI <D>*.

3.4 Используя результаты выполнения п. 2.3:

- привести фрагмент машинного кода «Программы 3», необходимый для выполнения п. 1.9;

- привести фрагмент машинного кода «Программы 3», необходимый для установки условий, заданных преподавателем.

3.5 Используя результаты выполнения п. 2.4 привести фрагмент машинного кода «Программы 4», необходимый для установки условий, заданных преподавателем.

3.6 Используя результаты выполнения п. 2.5 привести фрагменты машинного кода программ, разработанных по п. 1.8, необходимые для установки условий, заданных преподавателем.

Содержание отчета

Отчет по лабораторной работе выполняется на листах формат А4 согласно требований ГОСТ и ЕСКД и должен содержать следующие пункты:

- а) титульный лист с названием работы, дисциплины, кода группы, ФИО исполнителя и преподавателя;

- б) цель и содержание работы;
- в) машинные коды «Программы 1», «Программы 2», «Программы 3» и «Программы 4»;
- г) машинный код программ, разработанных по пунктам 1.8 и 1.9;
- д) результаты исследований работы программ по п. 2.1,..., 2.5 заданий;
- е) результаты выполнения по пункту 3;
- ж) выводы по результатам работы.

Контрольные вопросы

- 1 С помощью каких команд микро-ЭВМ может осуществлять ввод-вывод информации?
- 2 За сколько машинных тактов осуществляется ввод-вывод данных по командам $IN < A_1 >$ и $OUT < A_1 >$?
- 3 Приведите схемы подключения дешифраторов адреса и внешних устройств при организации обмена информацией с внешними устройствами с помощью различных команд.
- 4 Укажите достоинства и недостатки различных методов адресации к внешним устройствам.
- 5 При выполнении каких команд, приведенных в «Программе 2», задействуются разряды регистра состояния МП БИС?
- 6 По каким условиям записывается «1» в каждый из разрядов регистра состояния МП БИС?
- 7 Перечислите виды логических операций, выполняемые МП БИС.
- 8 Перечислите режимы работы программируемого устройства ввода-вывода информации в параллельном коде КР580ВВ55.
- 9 Укажите различия в управляющих сигналах схем К539ИР12 и КР580ВВ55 при использовании их в режиме ввода-вывода информации в микро-ЭВМ.

10 Приведите различные варианты подключения внешних устройств к микро-ЭВМ с помощью схем K589ИР12 и КР580ВВ55.

11 Рассмотрите возможные способы организации обмена информацией между двумя микро-ЭВМ в параллельном коде с помощью схемы КР580ВВ55.

Список литературы, рекомендованный к использованию по данной теме

Основная учебная литература

1 Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд., испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

Дополнительная учебная литература

1 Шевкопляс Б.С. Микропроцессорные структуры. Инженерные решения: Справочник. - М.: Радио и связь, 1990 – 512 с.

2 Щелкунов, Н. Н. Микропроцессорные средства и системы / Н. Н. Щелкунов, А. П. Дианов. – М. : Радио и связь, 1989. – 288 с.

3 Коффрон, Дж. Технические средства микропроцессорных систем: практический курс / Дж. Коффрон. Пер. с англ. – М. : Мир, 1983. – 344 с.

4 Шахнов, В. А. Микропроцессоры и микропроцессорные комплекты интегральных микросхем : справочник / В.-Б. Б. Абрайтис, Н. Н. Аверьянов, А. И. Белоус [и др.]: под ред. В. А. Шахнова. – Том 1. – М. : Радио и связь, 1988. – 368 с.

5 Алексенко, А. Г. Основы микросхемотехники / А. Г. Алексенко. 3-е изд. перераб. и доп. – М. : БИНОМ. Лаборатория знаний, 2009. – 448 с.

ЛАБОРАТОРНАЯ РАБОТА 4

Тема: Исследование работы подпрограмм и методов использования стека.

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

Формируемые компетенции: ОК-6, ОК-7, ОПК-6, ОПК-8, ОПК-9, ПК-1, ПК-3.

Теоретическая часть

Память микро-ЭВМ, построенной на основе МП серии K580, может иметь не более 65536 однобайтных ячеек. Учитывая ограниченные возможности памяти при разработке программ, нужно стараться сделать их как можно короче. С этой целью часть программы, которая неоднократно повторяется, или программа, которая часто используется, могут быть оформлены в виде подпрограмм – последовательностей команд, выполнение которых может быть вызвано из любого места программы любое количество раз. Процесс передачи управления к подпрограмме называется ее вызовом. Данные и адреса, требуемые для работы подпрограммы, называются входными параметрами. Результаты работы подпрограммы, передаваемые по окончании ее работы в основную программу, называются выходными параметрами.

Для вызова подпрограммы и возврата из них используются команды *CALL* <A2> <A1> и *RET*.

Команда *CALL* <A2> <A1> загружает в программный счетчик МП БИС содержимое байтов <A2> <A1>, записанных в последующих двух адресах памяти после адреса, где записан код команды *CALL* (CD). Содержимое байта <A2> записывается в младший байт *PCL* программного счетчика, а

третий байт $\langle A1 \rangle$ команды – в старший байт *PCN* программного счетчика, при этом МП БИС автоматически сохраняет в стеке адрес основной программы, к которому она будет обращаться после выполнения подпрограммы.

Стек – специально организованная область ОЗУ, задействованная в микро-ЭВМ для временного сохранения данных или адресов. Число, записанное в стек последним, извлекается из него первым.

Команда *RET* (*C9*) помещает в программный счетчик последнее записанное на данный момент в стеке число. После этого выполнение программы будет осуществляться с этого адреса. Любая подпрограмма должна кончаться командой *RET*.

Автоматическое сохранение и восстановление адреса основной программы при выполнении подпрограмм позволяет сделать подпрограммы вложенными, т. е. осуществить вызов одной подпрограммы из другой. Уровень вложенности для данной микро-ЭВМ определяется лишь размером стека.

Существуют также команды условного вызова подпрограммы и возврата из них. Они позволяют вызвать подпрограмму и возвратиться из нее по определенному состоянию заданных разрядов регистра признаков (аналогично командам условных переходов). Все команды условного вызова подпрограммы – трехбайтные, во втором и третьем байтах сообщается начальный адрес подпрограммы. Команды вызова подпрограмм и возврата из них используют стек и внутренний регистр МП БИС *SP* (*STACK POINTER*) для адресации к стеку.

Помимо команд вызова подпрограмм и возврата из них со стеком можно обмениваться информацией с помощью команд *PUSH* $\langle R \rangle$ (записать в стек содержание обозначенного регистра МП БИС) и *POP* $\langle R \rangle$ (записать данные из стека в обозначенный регистр МП БИС). Эти команды являются однобайтными, и в них содержится указание пары регистров МП БИС.

При записи в стек содержимого пары регистров или программного счетчика по адресу $SP-1$ записывается содержимое старшего регистра из указанной пары или старший байт PCH программного счетчика, а по адресу $SP-2$ в стек записывается содержимое младшего регистра из указанной пары младшего байта PCL программного счетчика.

При записи из стека данных в пару регистров или программный счетчик в младший регистр пары или PCL записывается число из адреса, указанного в указателе стека SP , а в старший регистр пары или PCH – число, записанное по адресу $SP+1$. В результате выполнения команды содержимое указателя стека SP увеличивается на два. Данные в памяти не изменяются, а лишь происходит их чтение и увеличение содержимого SP .

Таким образом, при записи данных адреса стека растут от больших к меньшим, а указатель стека SP всегда содержит последний адрес стека, в котором записано число.

При разработке программ необходимо первоначально назначать область стека, записывая в SP адрес с помощью команды $LXI SP <A2> <A1>$ или команды $SPHL$.

Все операции со стеком должны быть сбалансированы, т. е. каждая подпрограмма должна содержать равное количество команд $PUSH <R>$ и $POP <R>$ и оканчиваться командой RET . В противном случае выполнение команды RET в конце подпрограммы приведет к записи в программный счетчик случайного числа из стека. Адрес возврата в основную программу будет потерян и нарушится последовательность ее выполнения.

Как правило, в начале каждой программы сохраняют в стеке содержимое всех задействованных при ее выполнении регистров с помощью команд $PUSH <R>$. В конце подпрограммы восстановление содержимого регистров осуществляется с помощью команд $POP <R>$ и в обратной последовательности по отношению к их записи в стек.

Обычно в виде подпрограмм записываются многократно используемые фрагменты программ, например подпрограмма выдачи звукового сигнала, подпрограмма обслуживания клавиатуры и дисплея и т. д.

Рассмотрим простейшие подпрограммы. Алгоритм работы простой подпрограммы временной задержки приведен на рисунке 4.1.

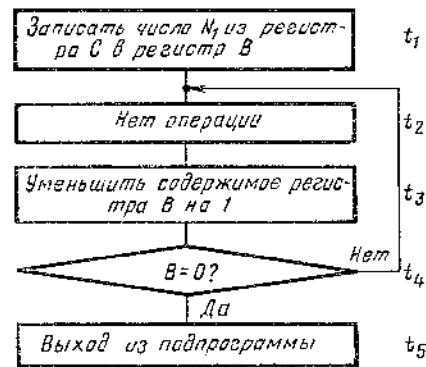


Рисунок 4.1 – Алгоритм простой подпрограммы временной задержки

Общее время задержки вычисляется по формуле:

$$T_D = t_1 + (t_2 + t_3 + t_4) \cdot N_1 + t_5, \quad (4.1)$$

где N_1 – число, первоначально записанное в счетчике.

В качестве счетчика выбран регистр B , в который записывается число N_1 из регистра C .

Команда *NOP* нужна для увеличения времени выполнения цикла, а следовательно, и общей задержки. Вместо команды *NOP* может быть записана любая последовательность команд, выполнение которых не изменяет содержимого регистров микропроцессора. Время записи числа N_1 в регистр B и возврата из подпрограммы $t_1 + t_5$ фиксировано и в цикл не входит. Минимальная задержка для приведенной подпрограммы определяется при N_1 , равном «01», и вычисляется по формуле:

$$T_{D\ MIN} = t_1 + t_2 + t_3 + t_4 + t_5. \quad (4.2)$$

Максимальная задержка имеет место при N_1 , равном «00», и вычисляется по формуле:

$$T_{D\ MAX} = t_1 + (t_2 + t_3 + t_4) \cdot 256 + t_5. \quad (4.3)$$

Подпрограмма *DLY* («Подпрограмма 1», показанная на рисунке 4.2) представляет подпрограмму временной задержки, записанную в соответствии с алгоритмом, представленным на рисунке 4.1.

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0B30	41		<i>MOV B, C</i>	записать число из регистра <i>C</i> в регистр <i>B</i>
0B31	00	<i>DLY</i>	<i>NOP</i>	нет операции
0B32	05		<i>DCR B</i>	уменьшить число в регистре <i>B</i> на 1
0B33	<i>C2 310B</i>		<i>JNZ DLY</i>	если число, записанное в регистре <i>B</i> , не равно «0», то идти на <i>DLY</i>
0B36	<i>C9</i>		<i>RET</i>	

Рисунок 4.2 – «Подпрограмма 1»

Рассмотрим пример использования подпрограммы временной задержки при организации световых эффектов. Световые эффекты могут формироваться простейшей схемой выходного устройства микро-ЭВМ (см. рисунок 3.2, б) лабораторной работы 3), на вход которой с заданной частотой записываются по очереди «0» и «1». Будем считать, что устройство формирования световых эффектов имеет адрес 05*h*. Схема алгоритма работы подпрограммы световых эффектов («Подпрограмма 2») приведена на рисунке 4.3, а текст «Подпрограммы 2» – на рисунке 4.4.

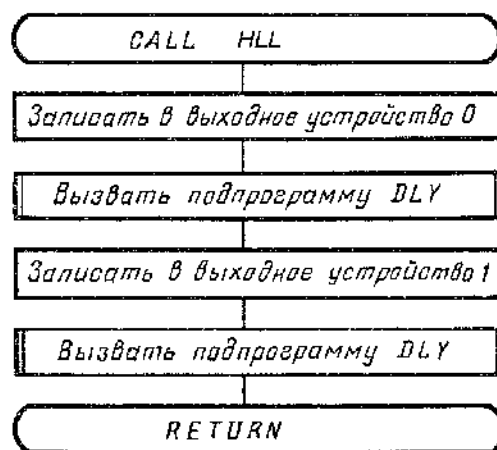


Рисунок 4.3 – Алгоритм подпрограммы световых эффектов

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0B50	AF	HLL	XRA A	очистить аккумулятор
0B51	D3 05		OUT HL	записать «00» в выходное устройство
0B53	CD 300B		CALL DLY	вызвать подпрограмму задержки
0B56	2F		CMA	записать код в аккумулятор
0B57	D3 05		OUT HL	записать код в выходное устройство
0B59	CD 300B		CALL DLY	вызвать подпрограмму задержки
0B5C	C9		RET	

Рисунок 4.4 – «Подпрограмма 2»

Программа *MAIN* («Программа 1») представляет программу генерации световых эффектов с частотой, задаваемой числом с входного устройства микро-ЭВМ (см. рисунок 3.2, а) лабораторной работы 3).

Текст программы приведен на рисунке 4.5.

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	<i>DB 05</i>		<i>IN 05</i>	записать число из входного регистра в аккумулятор
0802	<i>4F</i>		<i>MOV C, A</i>	записать число в регистр <i>C</i>
0803	<i>CD 500B</i>		<i>CALL HLL</i>	вызвать подпрограмму <i>HLL</i>
0806	<i>C3 0008</i>		<i>JMP MAIN</i>	продолжать

Рисунок 4.5 – «Программа 1»

При использовании подпрограмм за счет изменения входных параметров можно влиять на конечный результат выполнения подпрограмм. Рассмотрим это на примере подпрограммы регулируемой временной задержки («Подпрограмма 3»), показанной на рисунке 4.6.

Программа состоит из двух подпрограмм: *DELB* (адрес *0B77*) и *DELA* (адрес *0B70*). Подпрограмма *DELB* осуществляет регулируемую временную задержку, и входным параметром ее является двухбайтное число, записанное в паре регистров (*B*, *C*). Это число и определяет длительность задержки в миллисекундах. Частным случаем подпрограммы *DELB* является подпрограмма *DELA*, осуществляющая фиксированную задержку в 1 мс и не имеющая входных параметров. Таким образом, если необходимо иметь в программе фиксированную задержку в 1 мс, то можно обращаться с помощью команды *CALL* к подпрограмме *DELA*. При необходимости получения определенной заданной временной задержки в программе необходимо записать соответствующее число в регистры *B* и *C*, а затем вызвать подпрограмму *DELB*.

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0B70	C5	DELA	PUSH B	сохранить содержимое регистров B и C в стеке
0B71	01 0100		LXI B, 0001	установить длительность 1 мс
0B74	C3 780B		JMP DEL1	
0B77	C5	DELB	PUSH B	сохранить содержимое регистров B и C в стеке
0B78	F5	DEL1	PUSH PSW	сохранить содержимое регистров PSW в стеке
0B79	AF		XRA A	очистить аккумулятор
0B7A	D5		PUSH D	сохранить содержимое регистров D и E в стеке
0B7B	16 67	DEL2	MVI D, TIME	загрузить счетчик 1 мс задержки
0B7D	15	DEL3	DCR D	уменьшить содержимое счетчика на 1
0B7E	C2 7D0B		JNZ DEL3	если не «0», продолжать
0B81	0B		DCX B	уменьшить содержимое счетчика длительности на 1
0B82	B8		CMP B	
0B83	C2 7B0B		JNZ DEL2	
0B86	B9		CMP C	
0B87	C2 7B0B		JNZ DEL2	если не «0», продолжать
0B8A	D1		POP D	восстановить содержимое регистров D, E
0B8B	F1		POP PSW	то же с PSW
0B8C	C1		POP B	то же с B, C

Рисунок 4.6 – «Подпрограмма 3»

Оборудование и материалы

В ходе выполнения лабораторной работы будут использоваться два эмулятора: «Эмулятор работы микропроцессора КР580ВМ80» (далее ПМК – по заголовку главного окна эмулятора) и «Программная модель стенда УМПК-80 на базе микропроцессора К580ВМ80» (далее УМПК – по заголовку главного окна эмулятора). Первый эмулятор находится в папке *CPU580*, второй – в папке *UMPK80*.

Указания по технике безопасности

Перед проведением лабораторной работы следует изучить инструкцию по технике безопасности при работе на компьютере. Также следует соблюдать правила и требования, установленные для данного компьютерного класса.

Задания

1 Задание на подготовку к выполнению экспериментальной части работы.

1.1 Изучите временные диаграммы выполнения микро-ЭВМ команд *CALL <A2> <A1>* и *RET*.

1.2 Ознакомьтесь с командами вызова и возврата их подпрограммы по условию для МП БИС.

1.3 Определите, при каких числах, записанных в регистре *B*,

«Подпрограмма 1» будет осуществлять минимальное и максимальное время задержки. Определите эти времена, если машинный такт равен 1 мкс.

1.4 Разработайте подпрограмму 5- и 10-секундной временной задержки.

1.5 Составьте программу, последовательно включающую светодиоды выходного устройства на время соответственно 10 и 5 с. При разработке программы необходимо воспользоваться подпрограммами, разработанными в п. 1.4 задания.

1.6 Определите для «Подпрограммы 3», при каких числах в регистрах *B* и *C* будет максимальное и минимальное время задержки. Определите эти времена, если машинный такт равен 1 мкс.

1.7 Изучите «Программу 2» (рисунок 4.7), определяющую, какой из восьми переключателей входного устройства (адрес *05h*) микро-ЭВМ установлен в положение «0»:

- а) составьте алгоритм работы «Программы 2»;
- б) запишите подпрограмму, используемую в «Программе 2»;
- в) установите, в каком регистре МП БИС содержится информация о номере переключателя входного устройства, установленном в «0»;
- г) определите, как будет работать «Программа 2», если на входном устройстве будут установлены в положение «0» не один, а несколько переключателей.

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	31 <i>B00B</i>		<i>LXI SP,</i> <i>0BB0</i>	записать в указатель стека <i>SP</i> адрес <i>0BB0</i>
0803	<i>DB 20</i>	<i>WAITC</i>	<i>IN 05</i>	получить число из входного устройства
0805	<i>FE FF</i>		<i>CPI FF</i>	содержит ли какой-либо разряд

			число «0»?
0807	CA 0308	JZ WAITC	если нет, то ждать
080A	CD 1008	CALL IDSW	если да, то вызвать подпрограмму
			определения разряда, в котором
			записан «0»
080D	CF	RST1	окончить выполнение программы
0810	06 FF	IDSW MVI B, FF	записать в регистр B число FF
0812	04	SRCH INR B	увеличить содержимое регистра B
			на 1
0813	0F	RRC	сдвинуть число в аккумуляторе
			вправо
0814	DA 1208	JC SRCH	если C=1, то продолжать
0817	C9	RET	возврат из подпрограммы

Рисунок 4.7 – «Программа 2»

2 Задание на выполнение экспериментальной части работы.

2.1 Исследовать процесс выполнения команд вызова и возврата из подпрограммы, а также команд работы со стеком, для чего выполнить следующее:

- ввести в микро-ЭВМ «Подпрограмму 4», показанную на рисунке 4.8;
- выполнить «Подпрограмму 4» по командам, используя режим выполнения программы по командам. После каждой команды проверить содержимое всех регистров МП БИС;
- выполнить команды *CALL STDY*, *PUSH PSW*, *POP*, *RET* по машинным циклам, для чего ввести исходный код подпрограммы в эмуляторе ПМК и включить режим «Структурная схема». Указанный эмулятор используется только для данного подпункта;
- заменить в «Подпрограмме 4» команду *POP PSW* на команду *NOP* и

проследить, как будет выполняться «Подпрограмма 4». Объяснить произошедшие изменения.

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	31 B00B		<i>LXI SP, 0BB0</i>	записать в указатель стека <i>SP</i> адрес 0BB0
0803	CD 0708		<i>CALL STDY</i>	вызвать подпрограмму <i>STDY</i>
0806	CF		<i>RST1</i>	остановить выполнение подпрограммы
0807	F5	<i>STDY</i>	<i>PUSH PSW</i>	записать слово состояния ПМ БИС в стек
0808	C5		<i>PUSH B</i>	записать содержимое регистров <i>B</i> , <i>C</i> в стек
0809	D5		<i>PUSH D</i>	записать содержимое регистров <i>D</i> , <i>E</i> в стек
080A	E5		<i>PUSH H</i>	записать содержимое регистров <i>H</i> , <i>L</i> в стек
080B	3E 05		<i>MVI A, 05</i>	записать в аккумулятор число 05h
080D	47		<i>MOV B, A</i>	записать число из <i>A</i> в регистр <i>B</i>
080E	87		<i>ADD A</i>	удвоить содержимое аккумулятора
080F	5F		<i>MOV E, A</i>	записать число из <i>A</i> в регистр <i>E</i>
0810	67		<i>MOV H, A</i>	записать число из <i>A</i> в регистр <i>H</i>
0811	E1		<i>POP H</i>	записать числа из стека в регистры <i>H</i> , <i>L</i>
0812	D1		<i>POP D</i>	записать числа из стека в регистры <i>D</i> , <i>E</i>
0813	C1		<i>POP B</i>	записать числа из стека в регистры <i>B</i> , <i>C</i>

0814	<i>F1</i>	<i>POP PSW</i>	записать слово состояния из стека в МП БИС
0815	<i>C9</i>	<i>RET</i>	возврат из подпрограммы

Рисунок 4.8 – «Подпрограмма 4»

Результаты выполнения представить в виде снимков экрана.

2.2 Исследовать подпрограмму временной задержки на примере работы программы световых эффектов («Программа 1»), для чего выполнить следующее:

- записать полный текст «Программы 1» для управления световыми эффектами с учетом «Подпрограммы 1» и «Подпрограммы 2». Ввести в микро-ЭВМ написанную программу;
- установить на входном устройстве число «01»;
- осуществить пуск программы с адреса 0800;
- проследить за изменением частоты включений, формируемых микро-ЭВМ в процессе выполнения программы, увеличивая число, записанное во входном устройстве. Числа задает преподаватель;
- сделать выводы.

2.3 Исследовать программу регулируемой временной задержки на примере программы, последовательно включающей и выключающей светодиоды выходного устройства на время соответственно 10 и 5 с, для чего выполнить следующее:

- ввести в микро-ЭВМ программу, разработанную в п. 1.5;
- осуществить пуск программы и проверить ее выполнение ;
- изменить числа, записанные в регистрах *B*, *C*, в подпрограмме регулируемой временной задержки («Подпрограмма 3»). Проверить возможность изменения длительности задержки. Числа для задержек задает преподаватель;

- сделать выводы.

2.4 Исследовать «Программу 2», для чего выполнить следующее:

- ввести «Программу 2» в микро-ЭВМ;
- установить на входном устройстве переключатели соответственно числу FF ;
- осуществить пуск программы. Убедиться, что микро-ЭВМ будет находиться в режиме ожидания появления «0» в любом разряде входного устройства;

- установить «0» с помощью переключателей в любом из разрядов входного устройства (задает преподаватель). Проверить содержимое регистров МП БИС после окончания выполнения «Программы 2»;

- осуществить повторный запуск программы при наличии нулей в двух разрядах входного устройства (задает преподаватель). Выяснить, какое число будет записано в регистре В МП БИС после окончания выполнения программы;

- сделать выводы.

3 Задание на обработку экспериментальных данных.

3.1 Используя результаты выполнения п. 2.2 привести фрагмент машинного кода «Программы 1», необходимый для установки условий, заданных преподавателем.

3.2 Используя результаты выполнения п. 2.3:

- привести фрагмент машинного кода программы по п. 1.5, необходимый для установки условий, заданных преподавателем;
- привести фрагмент машинного кода «Подпрограммы 3», необходимый для установки условий, заданных преподавателем.

3.3 Используя результаты выполнения п. 2.4 привести фрагменты машинного кода «Программы 2», необходимые для установки условий, заданных преподавателем.

Содержание отчета

Отчет по лабораторной работе выполняется на листах формат А4 согласно требований ГОСТ и ЕСКД и должен содержать следующие пункты:

- а) титульный лист с названием работы, дисциплины, кода группы, ФИО исполнителя и преподавателя;
- б) цель и содержание работы;
- в) временные диаграммы выполнения команд *CALL <A2> <A1>, RET*;
- г) расчеты по пунктам 1.3 и 1.6;
- д) машинные коды подпрограммы и программы, разработанных по пунктам 1.4 и 1.5;
- е) алгоритм работы по п. 1.7;
- ж) машинный код подпрограммы из «Программы 2» согласно п. 1.7;
- з) ответы на вопросы п. 1.7;
- и) снимки экрана по п. 2.1;
- к) полный текст программы по п. 2.2;
- л) результаты исследований работы программ и подпрограмм по п. 2.2,..., 2.4 заданий;
- м) выводы по результатам работы.

Контрольные вопросы

1 Укажите количество машинных тактов выполнения команды *CALL <A2> <A1>*.

2 В какой последовательности записывается и считывается из стека содержимое аккумулятора и регистра признаков МП БИС при выполнении команд *PUSH PSW* и *POP PSW*?

3 С помощью каких команд можно задать или переобозначить область памяти, отведенную под стек?

- 4 Укажите порядок выполнения микро-ЭВМ команды *RET*.
- 5 Сравните процесс выполнения микро-ЭВМ команд *CALL* и *RST*.
- 6 Как нужно изменить «Подпрограмму 3», чтобы временная задержка, получаемая в результате ее выполнения, определялась числами, записанными по адресам *0B00* и *0B01*?
- 7 В какой последовательности сохраняется и извлекается содержимое регистров МП БИС в подпрограммах?
- 8 Какой минимальный адрес будет записан в указатель стека *SP* МП БИС при выполнении «Подпрограммы 4»?
- 9 Как будет выполняться «Подпрограмма 4», если вместо команды *POP B* в ней будет записана команда *NOP*?
- 10 Какое максимальное время задержки может обеспечивать «Подпрограмма 1», если длительность машинного такта равна 1 мкс?
- 11 Какое максимальное и минимальное время задержки может обеспечить «Подпрограмма 3», если длительность машинного такта равна 1 мкс?
- 12 Как увеличить время задержки с использованием «Подпрограммы 3»?

Список литературы, рекомендованный к использованию по данной теме

Основная учебная литература

- 1 Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд., испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

Дополнительная учебная литература

- 1 Шевкопляс Б.С. Микропроцессорные структуры. Инженерные

решения: Справочник. - М.: Радио и связь, 1990 – 512 с.

2 Щелкунов, Н. Н. Микропроцессорные средства и системы / Н. Н. Щелкунов, А. П. Дианов. – М. : Радио и связь, 1989. – 288 с.

3 Коффрон, Дж. Технические средства микропроцессорных систем: практический курс / Дж. Коффрон. Пер. с англ. – М. : Мир, 1983. – 344 с.

4 Шахнов, В. А. Микропроцессоры и микропроцессорные комплекты интегральных микросхем : справочник / В.-Б. Б. Абрайтис, Н. Н. Аверьянов, А. И. Белоус [и др.]: под ред. В. А. Шахнова. – Том 1. – М. : Радио и связь, 1988. – 368 с.

5 Соппа, И. В. Микропроцессоры и микроконтроллеры / И. В. Соппа, А. С. Самардак, В. А. Иванов. – Владивосток : Издательство Дальневосточного университета, 2009. – 211 с.

6 Алексенко, А. Г. Основы микросхемотехники / А. Г. Алексенко. 3-е изд. перераб. и доп. – М. : БИНОМ. Лаборатория знаний, 2009. – 448 с.

ЛАБОРАТОРНАЯ РАБОТА 5

Тема: Исследование арифметических операций.

Цель работы: изучение способов организации и исследование программ выполнения арифметических операций.

Формируемые компетенции: ОК-6, ОК-7, ОПК-6, ОПК-8, ОПК-9, ПК-3.

Теоретическая часть

Из двух способов определения чисел с фиксированной и плавающей точкой первый получил наибольшее распространение при программировании микро-ЭВМ на МП БИС KP580BM80. Это связано с отсутствием специальных команд, позволяющих работать МП БИС с числами с плавающей точкой. В свою очередь, двоичное восьмиразрядное число с фиксированной точкой можно представить как двоичное число со знаком, имеющее значение от минус 128_{10} до плюс 127_{10} . При этом отрицательные числа представляются в дополнительном коде, а старший, седьмой, разряд числа используется как знаковый. Такое представление чисел не позволяет выполнять арифметические операции с использованием переноса при сложении и заема при вычитании.

Число с фиксированной точкой можно представлять также двоичными числами без знака, имеющими значения от 0 до 255_{10} .

Для МП БИС KP580BM80 можно представлять такие числа в виде двоично-десятичного числа – BINARY-CODED-DECIMAL (BCD), при котором каждый байт рассматривается как два полубайта – две тетрады, каждая из которых кодирует десятичную цифру. Такое представление позволяет закодировать в 1 байт числа от 0 до 99_{10} .

Проведение арифметических операций сложения, вычитания, умножения, деления, вычисления специальных функций рассмотрим на примерах соответствующих подпрограмм.

Программа сложения массива однобайтных чисел с получением двухбайтного результата – подпрограмма *MAIN* («Программа 1»), приведенная на рисунке 5.1.

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	21 000B	<i>MAIN</i>	<i>LXI H, 0B00</i>	записать в регистры <i>H, L</i> адрес первого слагаемого
0803	06 05		<i>MVI B, 05</i>	загрузить в регистр <i>B</i> количество слагаемых
0805	<i>CD 0908</i>		<i>CALL ADDB</i>	вызвать подпрограмму сложения
0808	<i>CF</i>		<i>RST1</i>	прервать выполнение программы
0809	<i>AF</i>	<i>ADDB</i>	<i>XRA A</i>	очистить аккумулятор
080A	<i>4F</i>		<i>MOV C, A</i>	очистить счетчик переносов
080B	86	<i>CNT</i>	<i>ADD M</i>	прибавить к содержимому аккумулятора число из массива слагаемых
080C	<i>D2 1008</i>		<i>JNC TRM</i>	если переноса нет, то идти на <i>TRM</i>
080F	<i>0C</i>		<i>INR C</i>	Увеличить содержимое регистра <i>C</i> на 1
0810	23	<i>TRM</i>	<i>INX H</i>	указать на следующий адрес слагаемого
0811	05		<i>DCR B</i>	уменьшить содержимое счетчика слагаемых
0812	<i>C2 0B08</i>		<i>JNZ CNT</i>	если не все слагаемые, то идти на

Рисунок 5.1 – «Программа 1»

Слагаемые должны быть расположены в последовательных адресах памяти. Входными параметрами подпрограммы *ADDB* являются адрес первого слагаемого, записанный в регистрах *H*, *L*, и число слагаемых, записанное в регистре *B*. Выходным параметром программы *MAIN* является сумма, старший байт которой записан в регистре *C*, а младший – в аккумуляторе *A*. Алгоритм программы сложения состоит в том, что после прибавления каждого элемента массива определяется переполнение аккумулятора (разряд *C*), и если это имеет место, то содержимое регистра *C* увеличивается на 1. Таким образом, за счет суммирования единиц переноса получается старший байт суммы (рисунок 5.2).

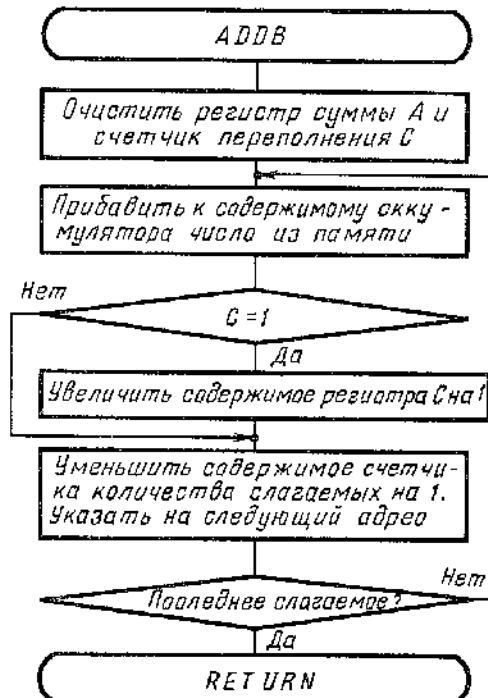


Рисунок 5.2 – Алгоритм подпрограммы сложения

Микро-ЭВМ может проводить арифметические операции с числами с двойной или большей длиной машинного слова. Так как МП БИС имеет 8-разрядное АЛУ, то операции с такими числами должны проводиться по байтам начиная с младших байтов. Так, операция сложения чисел $17F5$ и 3411 будет проводиться так, как показано на рисунке 5.3.

Старший байт	Флаг C	Младший байт	Числа
00101111		11110101	$17F5$
+		+	+
00110100		00010001	3411
+1	$\leftarrow 1$		
01001100		00000110	$4C06$

Рисунок 5.3 – Выполнение операции сложения двухбайтных чисел

Операция вычитания чисел $6F5C$ и $13C5$ будет осуществляться так, как показано на рисунке 5.4.

Старший байт	Флаг C	Младший байт	Числа
01101111		01011100	$6F5C$
–		–	–
00010011		11000101	$13C5$
–1	$\leftarrow 1$		
01011011		10010111	$5B97$

Рисунок 5.4 – Выполнение операции вычитания двухбайтных чисел

Из приведенных примеров видно, что при суммировании (вычитании) младших байтов чисел необходимо применять команду ADD (SUB), а для сложения (вычитания) остальных – команду ADC (SBB), которая будет учитывать состояние разряда регистра C признаков МП БИС.

Программа нахождения разности чисел, имеющих одинаковую длину, – «Программа 2», приведена на рисунке 5.5.

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	<i>CD 0508</i>		<i>CALL 0805</i>	вызов подпрограммы
0803	<i>76</i>		<i>HLT</i>	останов программы
0804	<i>00</i>		<i>NOP</i>	
0805	<i>0E 02</i>		<i>MVI C, 02</i>	загрузка в регистр <i>C</i> длины чисел в байтах
0807	<i>00</i>		<i>NOP</i>	
0808	<i>11 2408</i>		<i>LXI D, 0824</i>	загрузка в регистры <i>D, E</i> адреса младшего байта уменьшаемого
080B	<i>21 3508</i>		<i>LXI H, 0835</i>	загрузка в регистры <i>H, L</i> адреса младшего байта вычитаемого
080E	<i>00</i>		<i>NOP</i>	
080F	<i>AF</i>	<i>SBN</i>	<i>XRA A</i>	очистить аккумулятор и флаг <i>C</i>
0810	<i>1A</i>	<i>CNT</i>	<i>LDAX D</i>	записать в аккумулятор уменьшаемое
0811	<i>9E</i>		<i>SBB M</i>	вычесть из содержимого аккумулятора вычитаемое
0812	<i>77</i>		<i>MOV M, A</i>	записать разность на место вычитаемого
0813	<i>23</i>		<i>INX H</i>	указать на следующий байт вычитаемого
0814	<i>13</i>		<i>INX D</i>	указать на следующий байт уменьшаемого
0815	<i>0D</i>		<i>DCR C</i>	уменьшить содержимое счетчика длины числа
0816	<i>C2 1008</i>		<i>JNZ CNT</i>	если не последний (старший байт числа), то идти на <i>CNT</i>

0819	<i>D0</i>	<i>RNC</i>	если байт старший и результат без заема, то возврат
081A	<i>CD 000B</i>	<i>CALL HLL</i>	если был заем, то включить светодиоды
081D	<i>CF</i>	<i>RST1</i>	прервать выполнение программы

Рисунок 5.5 – «Программа 2»

Входные параметры: регистр *C* – длина чисел (в байтах), регистры *H*, *L* – адрес младшего байта вычитаемого, регистры *D*, *E* – адрес младшего байта уменьшаемого. Каждое из чисел записывается в последовательных адресах памяти начиная с младших байтов. Результат заносится в область памяти, отведенную под вычитаемое. В том случае, если уменьшаемое меньше вычитаемого, будут включаться светодиоды.

Выполнение команд *MOV M, A*; *INX D* не воздействует на разряд *C*. При исследовании программы *SBN* перед началом ее выполнения осуществляется начальная установка всех входных параметров.

В программе *SBN* подпрограмма (*HLL*) начинается с адреса *0B00*. Она позволяет включать светодиоды на время, удобное для наблюдения оператором, например, 3 с и более. После этого светодиоды должны выключиться. Пример выполнения такой подпрограммы приведен в лабораторной работе 4.

Существует несколько алгоритмов умножения чисел. При первом алгоритме умножение можно заменить многократным сложением, например:

$$14 \cdot 3 = 14 + 14 + 14.$$

Существенный недостаток этого способа – значительная длительность процесса вычисления. При втором алгоритме умножение осуществляется в столбец. Этот алгоритм применим и для умножения двоичных чисел, как показано на рисунке 5.6.

$$\begin{array}{r}
 0110 = 6_{10} \\
 0011 = 3_{10} \\
 \hline
 0110 \\
 0110 \\
 0000 \\
 0000 \\
 \hline
 00010010 = 18_{10}
 \end{array}$$

Рисунок 5.6 – Выполнение операции умножения двоичных чисел

При вычислении результата по второму алгоритму (рисунок 5.7) необходимо осуществить многократное суммирование со сдвигом влево множимого при одновременной проверке содержимого разрядов множителя, начиная со стороны его младшего разряда.

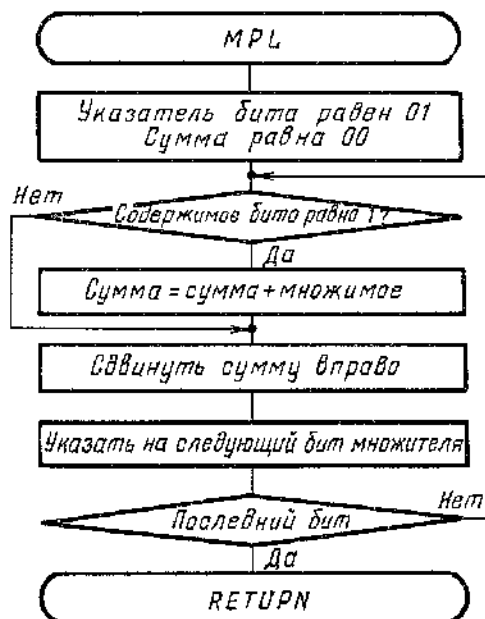


Рисунок 5.7 – Алгоритм подпрограммы умножения

При этом если в очередном разряде множителя записана «1», то множимое прибавляется к сумме и сдвигается влево на один разряд, а если в

разряде записан «0», то произойдет только сдвиг множимого. Сдвиг множимого влево можно заменить сдвигом суммы вправо. По этому алгоритму работает программа умножения двух однобайтных чисел с получением двухбайтного результата («Программа 3»), приведенная на рисунке 5.8. Входные параметры: регистр *D* – множимое, регистр *E* – множитель. Результат перемножения записывается в регистры *B*, *C*.

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	16		<i>MVI D, 06</i>	загрузить в регистр <i>D</i> множимое
0802	1E		<i>MVI E, 03</i>	загрузить в регистр <i>C</i> множитель
0804	CD 0808		<i>CALL 0808</i>	вызов подпрограммы умножения
0807	76		<i>HLT</i>	останов
0808	01 0000	<i>MPL</i>	<i>LXI B, 0000</i>	очистить содержимое регистров <i>B</i> , <i>C</i>
080B	3E 01		<i>MVI A, 01</i>	загрузить в аккумулятор указатель разряда
080D	A7		<i>ANA A</i>	очистить флаг <i>C</i>
080E	F5	<i>MPL1</i>	<i>PUSH PSW</i>	сохранить слово состояния в стеке
080F	A3		<i>ANA E</i>	проверить содержимое очередного разряда множителя
0810	78		<i>MOV A, B</i>	загрузить в аккумулятор старший байт суммы
0811	CA 1508		<i>JZ MPL2</i>	если в очередном разряде «0», то перейти на <i>MPL2</i>
0814	82		<i>ADD D</i>	иначе, прибавить множимое к сумме
0815	1F	<i>MPL2</i>	<i>RAR</i>	сдвинуть число в аккумуляторе вправо (младший бит → <i>C</i>)

0816	47	<i>MOV B, A</i>	сохранить содержимое аккумулятора в регистре <i>B</i>
0817	79	<i>MOV A, C</i>	загрузить в аккумулятор младший байт суммы
0818	1F	<i>RAR</i>	сдвинуть число в аккумуляторе вправо (<i>C</i> → в старший бит)
0819	4F	<i>MOV C, A</i>	сохранить содержимое аккумулятора в регистре <i>C</i>
081A	F1	<i>POP PSW</i>	получить из стека слово состояния
081B	17	<i>RAL</i>	указатель на следующий разряд
081C	D2 0E08	<i>JNC 080E</i>	если разряд не последний, продолжать с <i>MPL1</i>
081F	C9	<i>RET</i>	если последний, то возврат

Рисунок 5.8 – «Программа 3»

Деление двоичных чисел, как и чисел, представленных в любой другой системе счисления, основывается на последовательном вычитании делителя из делимого и остатков от деления. Однако двоичное деление реализуется проще, так как использование только двух цифр (0 и 1) исключает в каждом цикле деления необходимость определения числа делителей, содержащихся в текущем значении делимого или остатка (достаточно только сравнить их).

Схема алгоритма деления двоичных чисел приведена на рисунке 5.9.

Программа *DIVB* построена по этому алгоритму («Программа 4»), приведенная на рисунке 5.10. Входными параметрами этой программы являются делимое (в регистре *E*) и делитель (в регистре *D*); выходными параметрами – частное (в регистре *H*) и остаток (в регистре *C*).

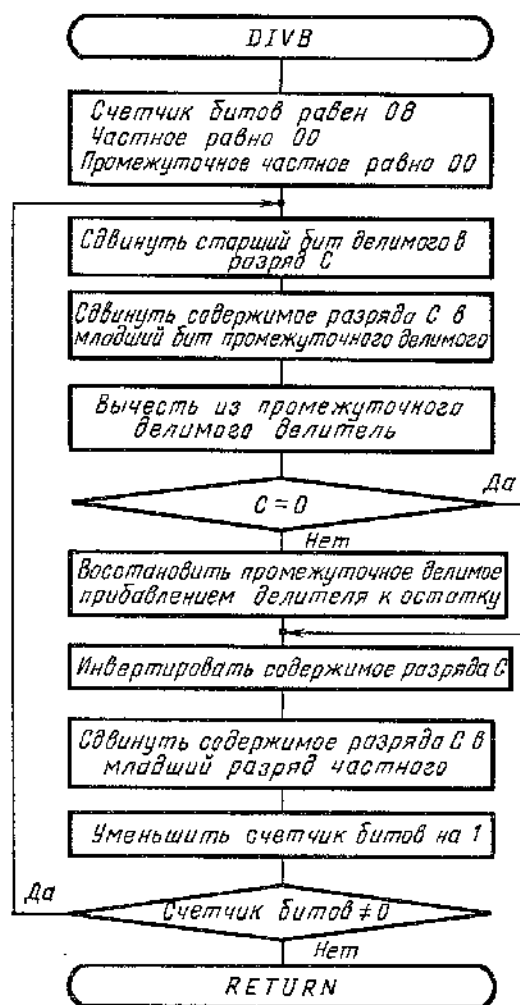


Рисунок 5.9 – Алгоритм подпрограммы деления

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	1E AA		MVI E, AA	загрузить в регистр E делимое
0802	16 33		MVI D, 33	загрузить в регистр D делитель
0804	21 0800	DIVB	LXI H, 0008	загрузить счетчик битов (регистр L) и очистить регистр частного H
0807	0E 00		MVI C, 00	очистить регистр промежуточного делимого
0809	7B	MXTB	MOVA, E	загрузить делимое в аккумулятор
080A	17		RAL	сдвинуть старший бит в разряд C

080B	5F	MOV E, A	вернуть делимое в регистр E
080C	79	MOV A, C	загрузить в аккумулятор промежуточное делимое из регистра C
080D	17	RAL	сдвинуть разряд C в младший бит
080E	92	SUB D	вычесть из содержимого аккумулятора делитель
080F	D2	JNC	если C равен 1, восстановить
		NOADD	содержимое аккумулятора
0812	82	ADD D	
0813	4F	NOADD MOV C, A	возвратить промежуточное данное в регистр C
0814	3F	CMC	инвертировать разряд C
0815	7C	MOV A, H	
0816	17	RAL	сдвинуть разряд C в младший бит регистра частного H
0817	67	MOV H, A	
0818	2D	DCR L	проверены ли все 8 разрядов?
0819	C2	JNZ MXTB	если нет, продолжать
081C	CF	RST1	если да, прервать выполнение программы

Рисунок 5.10 – «Программа 4»

Для вычисления специальных функций ($\sin x$, $\cos x$, $\operatorname{tg} x$, $\ln x$, $\sqrt{x}$) применяются специальные алгоритмы. Например, функции $\sin x$, $\cos x$, $\ln x$ можно вычислить, воспользовавшись их разложением в ряд:

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots;$$

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots;$$

$$\ln(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \dots$$

Для первых двух выражений x может быть любым в радианах, для третьего должно выполняться условие:

$$0 < x \leq 1.$$

Число членов ряда определяется из условия получения требуемой точности.

Для вычисления функции $\sqrt{x}$ с точностью до целых чисел можно применить алгоритм, основанный на том, что квадрат числа можно определить сложением последовательности нечетных чисел, как показано на рисунке 5.11.

Число	Сумма нечетных чисел	Результат
1	1	= 1
2	1+3	= 2 ²
3	1+3+5	= 3 ²
4	1+3+5+7	= 4 ²
5	1+3+5+7+9	= 5 ²

Рисунок 5.11 – Вычисление функции $\sqrt{x}$

Исходя из приведенного примера видно, что какое число необходимо возвести в квадрат, такое же количество последовательных нечетных чисел, начиная с единицы, необходимо сложить.

Вычисление специальных функций по приведенным выражениям занимает длительное время и обеспечивает низкую точность. Это

обусловлено сравнительно небольшой длиной машинного слова и ограниченным быстродействием МП БИС. Поэтому в тех случаях, когда ставятся жесткие требования по быстродействию и точности, применяется вычисление функций с помощью таблиц. Проиллюстрируем этот метод на примере программы вычисления квадрата числа x («Программа 5»), приведенной на рисунке 5.12.

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	<i>DB 05</i>		<i>IN 05</i>	чтение числа x из порта 05
0802	<i>6F</i>		<i>MOV L, A</i>	запись числа x в регистр L
0803	26 00	<i>SQ</i>	<i>MI H, 00</i>	очистить регистр H
0805	11 000A		<i>LXI D,</i> <i>0A00</i>	загрузить начальный адрес таблицы
0808	19		<i>DAD D</i>	получить адрес элемента
0809	66		<i>MOV H, M</i>	загрузить его в регистр H
080A	<i>CF</i>		<i>RST1</i>	прервать выполнение программы
0A00	00	<i>SQTB</i>	00	} таблица квадратов чисел
0A01	01		01	
0A02	04		04	
0A03	09		09	
0A04	10		10	
0A05	19		19	
0A06	24		24	
0A07	31		31	
0A08	40		40	
0A09	51		51	

Рисунок 5.12 – «Программа 5»

Программа *SQ* определяет квадрат чисел от нуля до 10 включительно. Входной параметр программы – число x . Оно записывается в регистр L . Выходной параметр – значение x^2 , записывается в регистр H .

Оборудование и материалы

В ходе выполнения лабораторной работы будет использоваться эмулятор «Программная модель стенда УМПК-80 на базе микропроцессора K580BM80» (далее УМПК – по заголовку главного окна эмулятора). Эмулятор находится в папке *UMPK80*.

Указания по технике безопасности

Перед проведением лабораторной работы следует изучить инструкцию по технике безопасности при работе на компьютере. Также следует соблюдать правила и требования, установленные для данного компьютерного класса.

Задания

1 Задание на подготовку к выполнению экспериментальной части работы.

1.1 Изучите группу арифметических команд МП БИС КР580ВМ80.

1.2 Ознакомьтесь с правилами выполнения команды *DAA* – десятичной коррекции аккумулятора.

1.3 Изучите «Программу 1», «Программу 2», «Программу 3», «Программу 4», «Программу 5», приведенные выше. Рассмотрите результат выполнения каждой программы на конкретных числовых примерах.

1.4 Проведите оценку времени выполнения подпрограмм умножения и программы деления восьмиразрядных чисел.

1.5 Составьте программу для исследования результата перемножения двух чисел на основе «Программы 3».

1.6 Разработайте программу сложения двух восьмиразрядных чисел с получением результата в двоично-десятичном коде (результат должен записываться в выходное устройство) .

1.7 Оцените время выполнения подпрограммы SQ «Программы 5».

2 Задание на выполнение экспериментальной части работы.

2.1 Исследовать программу сложения однобайтных чисел с получением двухбайтного результата, для чего выполнить следующее:

- ввести в микро-ЭВМ «Программу 1»;
- записать в микро-ЭВМ последовательность из пяти чисел, начиная с адреса 0B00. Числа задает преподаватель;
- осуществить пуск программы и проверить ее выполнение по данным, записанным в регистре C и аккумуляторе МП БИС;
- изменить «Программу 1» так, чтобы результат выполнения записывался по адресу, указанному преподавателем. Проверить результат ее выполнения;
- изменить в «Программе 1» указатель количества слагаемых в сумме и выполнить программу заново. Количество слагаемых задает преподаватель;
- сделать выводы и представить результаты выполнения в виде снимков экрана.

2.2 Исследовать программу вычитания двух чисел, имеющих одинаковую длину, для чего выполнить следующее:

- ввести «Программу 2» в микро-ЭВМ;

- записать по адресам, указанным в регистрах D , E и H , L соответственно младшие байты уменьшаемого и вычитаемого. Выбрать при этом уменьшаемое больше, чем вычитаемое;

- записать в регистр C длину числа в байтах. Для первого случая записать в регистр C число «01» (рассматривается вычитание двух восьмиразрядных чисел);

- осуществить пуск «Программы 2» и исследовать результат ее выполнения по числу, записанному по адресу, где записано вычитаемое;

- видоизменить «Программу 2» так, чтобы результат вычисления разности двух 8-разрядных чисел записывался в выходное устройство с адресом «30h»;

- изменить числа, записанные по адресам, указанным в регистрах H , L и D , E , так, чтобы уменьшаемое было меньше вычитаемого;

- осуществить пуск программы и проследить, что при этом получается;

- исследовать в аналогичной последовательности выполнение «Программы 2» и результат вычитания двухбайтных чисел;

- числа для всех примеров задает преподаватель;

- сделать выводы и представить результаты выполнения в виде снимков экрана.

2.3 Исследовать программу умножения двух восьмиразрядных чисел с получением 16-разрядного результата, для чего выполнить следующее:

- ввести в микро-ЭВМ программу для исследования результата перемножения двух чисел, разработанную в п. 1.5;

- осуществить пуск программы и проверить результат перемножения двух чисел по числу, записанному в регистрах B , C . Числа для операций умножения задает преподаватель;

- сделать выводы и представить результаты выполнения в виде снимков экрана.

2.4 Исследовать программу деления двух восьмиразрядных чисел, для чего выполнить следующее:

- ввести в микро-ЭВМ «Программу 4»;
- записать в регистры E , D соответственно делимое и делитель;
- осуществить пуск программы и проверить результат деления двух чисел по содержимому регистров H , C . Числа для операции деления задает преподаватель;
- сделать выводы и представить результаты выполнения в виде снимков экрана.

2.5 Исследовать программу вычисления квадрата числа с помощью таблицы, для чего выполнить следующее:

- ввести в микро-ЭВМ «Программу 5»;
- записать в регистр L число, квадрат которого необходимо вычислить;
- осуществить пуск программы и проверить результат вычисления квадрата числа по содержимому регистра H ;
- видоизменить программу так, чтобы результат вычисления квадрата числа записывался в выходное устройство. Адрес выходного устройства задает преподаватель;
- ввести в «Программу 5» проверку на значение числа, квадрат которого определяется в результате выполнения программы. Если задаваемое число больше 10, то микро-ЭВМ должна указывать на это, например, включением светодиодов выходного устройства;
- осуществить пуск программы и проверить выполнение, задавая сначала число, меньшее 10, затем, большее 10;
- сделать выводы и представить результаты выполнения в виде снимков экрана.

2.6 Исследовать программу сложения двух восьмиразрядных чисел с получением результата в двоично-десятичном коде, для чего выполнить следующее:

– ввести в микро-ЭВМ программу, разработанную при выполнении п. 1.6;

– осуществить пуск программы и проверить результат сложения следующих чисел: $33 + 25$; $38 + 25$; $98 + 25$;

– заменить в разработанной программе операцию сложения на операцию вычитания двух чисел. Проверить, возможно ли осуществлять десятичную коррекцию числа аккумулятора после команды вычитания;

– сделать выводы и представить результаты выполнения в виде снимков экрана.

3 Задание на обработку экспериментальных данных.

3.1 Используя результаты выполнения п. 2.1 привести фрагменты машинного кода «Программы 1», необходимые для установки условий, заданных преподавателем.

3.2 Используя результаты выполнения п. 2.2 привести фрагменты машинного кода «Программы 2», необходимые для установки условий, заданных преподавателем.

3.3 Используя результаты выполнения п. 2.5 привести фрагменты машинного кода «Программы 5», необходимые для установки условий, заданных преподавателем.

3.4 Используя результаты выполнения п. 2.6 привести фрагменты машинного кода написанной программы, необходимые для установки условий, заданных преподавателем.

Содержание отчета

Отчет по лабораторной работе выполняется на листах формат А4 согласно требований ГОСТ и ЕСКД и должен содержать следующие пункты:

а) титульный лист с названием работы, дисциплины, кода группы, ФИО исполнителя и преподавателя;

- б) цель и содержание работы;
- в) полный перечень арифметических команд МП БИС КР580ВМ80;
- г) тексты исследуемых программ – «Программы 1, ..., 5»;
- д) программу сложения двух восьмиразрядных чисел с получением результата в двоично-десятичном коде, разработанного в п. 1.6;
- е) программу для исследования результата перемножения двух восьмиразрядных чисел, разработанную в п. 1.5;
- ж) видоизмененную «Программу 3», исследованную в задании п. 2.2;
- з) видоизмененную «Программу 5», исследованную в задании п. 2.5;
- и) результаты выполнения арифметических операций по всем заданиям.
- к) выводы по результатам работы.

Контрольные вопросы

- 1 Какие команды арифметических операций может выполнять МП БИС КР580ВМ80?
- 2 Сформулируйте правило выполнения МП БИС команды *DAA*.
- 3 После каких команд можно осуществлять десятичную коррекцию числа аккумулятора (выполнять команду *DAA*)?
- 4 Как оценить максимальное время выполнения подпрограммы умножения двух чисел («Программа 3»)?
- 5 Можно ли непосредственно исследовать перемножения двух чисел по «Программе 3»?
- 6 На чем основаны алгоритмы работы подпрограмм умножения и деления чисел?
- 7 В чем преимущество вычисления функций, заданных в виде таблиц?
- 8 Представьте числа от нуля до 20 в двоично-десятичном коде.

9 Укажите возможные способы представления чисел для МП БИС КР580ВМ80.

10 Оцените максимальное время выполнения программы деления двух чисел, если время машинного такта для МП БИС равно 1 мкс.

Список литературы, рекомендованный к использованию по данной теме

Основная учебная литература

1 Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд., испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

Дополнительная учебная литература

1 Шевкопляс Б.С. Микропроцессорные структуры. Инженерные решения: Справочник. - М.: Радио и связь, 1990 – 512 с.

2 Щелкунов, Н. Н. Микропроцессорные средства и системы / Н. Н. Щелкунов, А. П. Дианов. – М. : Радио и связь, 1989. – 288 с.

3 Коффрон, Дж. Технические средства микропроцессорных систем: практический курс / Дж. Коффрон. Пер. с англ. – М. : Мир, 1983. – 344 с.

4 Шахнов, В. А. Микропроцессоры и микропроцессорные комплекты интегральных микросхем : справочник / В.-Б. Б. Абрайтис, Н. Н. Аверьянов, А. И. Белоус [и др.]: под ред. В. А. Шахнова. – Том 1. – М. : Радио и связь, 1988. – 368 с.

5 Соппа, И. В. Микропроцессоры и микроконтроллеры / И. В. Соппа, А. С. Самардак, В. А. Иванов. – Владивосток : Издательство Дальневосточного университета, 2009. – 211 с.

6 Алексенко, А. Г. Основы микросхемотехники / А. Г. Алексенко. 3-е изд. перераб. и доп. – М. : БИНОМ. Лаборатория знаний, 2009. – 448 с.

ЛАБОРАТОРНАЯ РАБОТА 6

Тема: Исследование работы устройств ввода и вывода информации.

Цель работы: изучение аппаратных методов подключения клавиатуры и дисплея к микро-ЭВМ; исследование программных способов ввода и вывода информации.

Формируемые компетенции: ОК-6, ОК-7, ОПК-6, ОПК-8, ОПК-9, ПК-1, ПК-3.

Теоретическая часть

В качестве устройства вывода информации, удобного для восприятия, часто используется дисплей. Рассмотрим метод подключения дисплея, состоящего из шести ячеек (семисегментных индикаторов), представляющих собой восемь светодиодов с общим анодом в одном корпусе. Каждый индикатор (рисунок 6.1, а) имеет семь светодиодов для отображения сегментов цифр, а восьмой светодиод отображает десятичную точку (рисунок 6.1, б). Индикатор может отображать цифры от нуля до девяти, а также некоторые буквы.

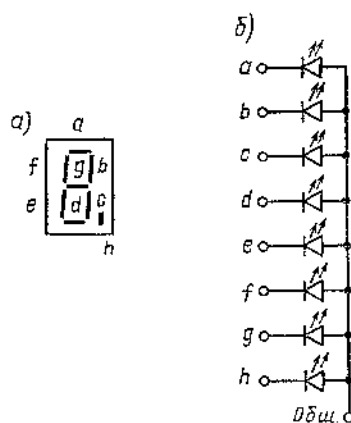


Рисунок 6.1 – Внешний вид (а) и схема семисегментной светодиодной матрицы (б)

Пронумеруем ячейки дисплея, как показано на рисунке 6.2

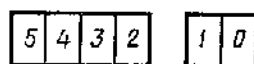


Рисунок 6.2 – Нумерация ячеек дисплея

Для уменьшения схемотехнического обеспечения, необходимого для подключения дисплея к микро-ЭВМ, часто применяют мультиплексный режим работы индикаторов. При этом для вывода на дисплей информации используют два выходных регистра: РгСг (адрес 06_{16} (в старой версии стенда 38_{16})) для записи семисегментного кода и РгСк (адрес 07_{16} (в старой версии стенда 28_{16})) для записи номера индикатора (рисунок 6.3).

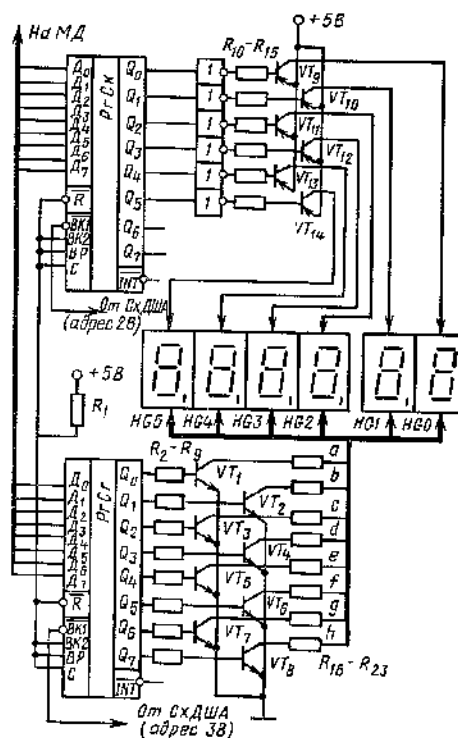


Рисунок 6.3 – Схема подключения дисплея к микро-ЭВМ

Одинаковые сегменты каждой ячейки индикатора связаны общей шиной, которая соединена с одним из транзисторных ключей $VT1, \dots, VT8$ на выходе регистра РгСг. Общие аноды индикатора подключены к одному из транзисторных ключей $VT9, \dots, VT14$ на выходе регистра сканирования РгСк.

Включение индикатора и его сегментов при записи в соответствующие разряды показано в таблице 6.1.

Таблица 6.1 – Включение индикатора и его сегментов

Регистр PгCг								
Номер разряда регистра сегментов дисплея	0	1	2	3	4	5	6	7
Включенный сегмент	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>	<i>f</i>	<i>g</i>	<i>h</i>
Регистр PгCк								
Номер разряда регистра цифр дисплея	0	1	2	3	4	5	6	7
Включаемая цифра дисплея	0	1	2	3	4	5	–	–

Сигналы заведены на регистры PгCг и PгCк так, что при поступлении на вход ВК1 сигнала выборки от дешифратора адреса данные с МД, подключенной по входам $D_0, \dots, D_7$, записываются в регистр и появляются на его выходах $Q_0, \dots, Q_7$. Таким образом, например, при записи в регистр сегментов числа «0000 0110» отпираются транзисторные ключи VT_2 и VT_3 , а при записи в регистр сканирования PгCк числа «0000 0101» отпирается транзисторный ключ VT_{14} и ток проходит по цепи «+5 В – VT_{14} – $HG1$ – b и c – VT_2 и VT_3 – общий провод схемы», при этом на левом индикаторе высветится символ «1». Если теперь с помощью программы высвечивать по очереди все ячейки, записывая их код в регистр PгCг и включая необходимую цифру с помощью регистра PгCк, то при достаточно высокой частоте переключений можно получать устойчивое изображение информации на дисплее.

Рассмотрим простую программу включения сегментов пятой ячейки дисплея с помощью кода, задаваемого с входного устройства микро-ЭВМ («Программа 1»), приведенную на рисунке 6.4.

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	<i>3E 05</i>		<i>MVI A, 05</i>	загрузить в аккумулятор число «0000 0101»
0802	<i>D3 07</i>		<i>OUT SCAN</i>	включить пятый разряд дисплея
0804	<i>IN 05</i>	<i>CNT</i>	<i>IN 05</i>	считать данные из входного устройства
0806	<i>OUT 06</i>		<i>OUT DSP</i>	записать их в регистр сегментов дисплея
0808	<i>C3 0408</i>		<i>JMP CNT</i>	продолжить программу

Рисунок 6.4 – «Программа 1»

«Программа 1» позволяет за счет изменения кода на входном устройстве (адрес 05_{16}) включать различные сегменты пятого индикатора дисплея.

При мультиплексном режиме работы вывод информации на каждый индикатор дисплея выводится микро-ЭВМ последовательно. Цифра или символ на индикаторе высвечивается некоторый промежуток времени, задаваемый подпрограммой задержки. При большой частоте сканирования индикаторов на цифровом дисплее получается устойчивое изображение.

Схема алгоритма программы, обеспечивающей мультиплексный режим работы дисплея, приведена на рисунке 6.5, код «Программы 2» – на рисунке 6.6. Код цифр для вывода на каждую ячейку дисплея хранится в последовательных ячейках памяти с адресами $0B00, \dots, 0B05$. При этом полагается, что код цифры для вывода на нулевой индикатор дисплея записан по адресу $0B00$. Начальный адрес подпрограммы временной задержки $0B77$.

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	01 0500	<i>DSO</i>	<i>LXI B, 0005</i>	загрузить в регистры <i>B, C</i> длительность задержки
0803	<i>AF</i>		<i>XRA A</i>	очистить аккумулятор
0804	21 050B	<i>CNT1</i>	<i>LXI H, 0B05</i>	загрузить адрес первых данных
0807	16 05		<i>MVI D, 05</i>	загрузить в регистр <i>D</i> указатель на пятый разряд дисплея
0809	7A	<i>CNT2</i>	<i>MOV A, D</i>	загрузить указатель в аккумулятор
080A	D3 07		<i>OUT 07</i>	включить нужный разряд
080C	7E		<i>MOV A, M</i>	получить из ОЗУ код данных
080D	D3 06		<i>OUT 06</i>	записать в регистр сегментов
080F	15		<i>DCR D</i>	указать на следующий разряд
0810	00		<i>NOP</i>	
0811	CD 770B		<i>CALL DEL</i>	вызвать подпрограмму задержки
0814	<i>AF</i>		<i>XRA A</i>	очистить аккумулятор
0815	D3 06		<i>OUT 06</i>	выключить индикатор
0817	2B		<i>DCX H</i>	уменьшить указатель адреса
0818	3E FF		<i>MVI A, FF</i>	проверка: все ли сообщение
081A	AA		<i>XRA D</i>	выведено?
081B	C2 0908		<i>JNZ CNT2</i>	если нет, то продолжать
081E	C3 0408		<i>JMP CNT1</i>	если да, то начать сначала
0821	76		<i>HLT</i>	останов

Рисунок 6.6 – «Программа 2»

В «Программе 2» используется вызов подпрограммы регулируемой временной задержки. Рекомендуется использовать «Подпрограмму 3»,

приведенную в лабораторной работе 4, а именно, подпрограмму *DELB*, записанную по адресу *0B77*. Входными данными для нее будет загруженная в регистры *B*, *C* длительность задержки, которую необходимо экспериментально подобрать так, чтобы удобно было читать выводимое на дисплей сообщение.

Перед выполнением «Программы 2», начиная с адреса *0B00*, необходимо ввести следующие данные: *06*, *7F*, *3F*, *40*, *4F*, *37* (указанные числа приведены в шестнадцатеричном виде).

Клавиатура является одним из широко распространенных устройств ввода данных и управляющих воздействий в микро-ЭВМ. С помощью клавиатуры можно вводить программу в ОЗУ, инициировать различные режимы работы микро-ЭВМ (пуск программы с заданного адреса, останов программы, выполнение программы по машинным циклам, вывод содержания регистров МП БИС на дисплей и т. д.).

Во всех случаях при организации ввода информации с клавиатуры в микро-ЭВМ, перед разработчиком ставится ряд задач, к основным из которых можно отнести:

- а) определение факта нажатия клавиши на клавиатуре;
- б) нахождение номера (или кода) нажатой клавиши;
- в) осуществление передачи управления на соответствующую подпрограмму.

Первые две задачи являются специфическими при организации ввода информации с клавиатуры, и именно им будет уделено внимание в данной лабораторной работе. Последняя, как правило, решается программными методами.

Наиболее удобно организовывать клавиатуру в виде матрицы размером $n \times m$, где n и m – соответственно число строк и столбцов. При таком способе организации к микро-ЭВМ можно подключать $n \times m$ клавиш. Сопряжение клавиатуры с микро-ЭВМ производят с помощью устройств ввода-вывода

данных. Для иллюстрации метода сопряжения клавиатуры с микро-ЭВМ рассмотрим клавиатуру 3×3 , представленную на рисунке 6.7 (где СхДША – схема дешифрации адреса, УВВ – устройство ввода, УВыв – устройство вывода). Ряды клавиатуры подключены к трем младшим разрядам устройства вывода УВыв (символьное обозначение адреса *KBDOT*), а столбцы подключены к трем младшим разрядам устройства ввода УВв (символьное обозначение адреса *KBDIN*).

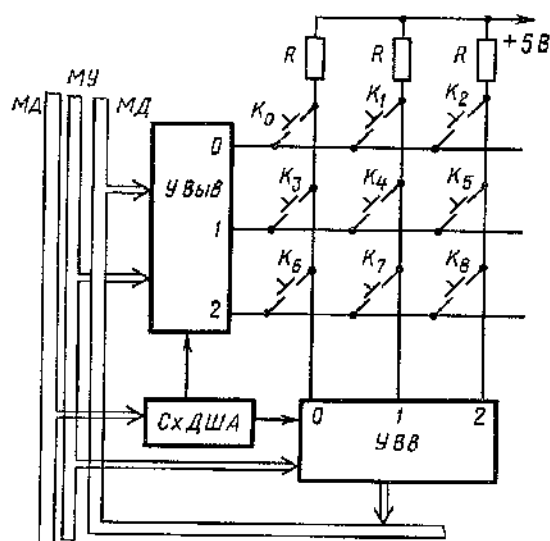


Рисунок 6.7 – Схема подключение клавиатуры к микро-ЭВМ

При программном способе дешифрации нажатой клавиши определение факта нажатия на клавишу может быть осуществлено с помощью такой последовательности операций:

- а) записать нули в разряды выходного устройства;
- б) считать содержание разрядов входного устройства;
- в) повторить снова, если во всех разрядах входного устройства записаны единицы.

Эмулятор «Программная модель стенда УМПК-80 на базе микропроцессора K580BM80» имеет в своем составе клавиатуру из 26 кнопок. Они организованы в матрицу 8×4 , т. е., устройство вывода адресует восемь строк, а устройство ввода считывает информацию с четырех

столбцов.

«Программа 3», написанная в соответствии с приведенным алгоритмом, позволяет определить факт нажатия на одну из клавиш, но не указывает номер (и код) нажатой клавиши (рисунок 6.8).

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	3E 00	WAITK	MVI A, 00	установка сигнала разрешения сканирования всех строк
0802	D3 07		OUT KBDOT	запись разрешения в выходное устройство (адрес 07h)
0804	DB 06		IN KBDIN	получить число с входного устройства (адрес 06h)
0806	E6 0F		ANI 0F	очистить старшую тетраду аккумулятора
0808	FE 0F		CPI 0F	есть ли в младшей тетраде нули?
080A	CA 0008		JZ WAITK	если нет, то ждать
080D	76		HLT	останов, если есть

Рисунок 6.8 – «Программа 3»

Определить номер нажатой клавиши можно с помощью алгоритма, приведенного на рисунке 6.9. Алгоритм основывается на последовательной записи нуля в каждый из рядов матрицы клавиатуры. При наличии нуля в каждом ряду микро-ЭВМ определяет факт нажатия на клавиши, находящиеся в столбцах клавиатуры, принадлежащих анализируемому ряду. Если какая-либо клавиша нажата, то определяется ее номер по номеру разряда, в котором записан ноль. «Программа 4» (рисунок 6.10) реализует описанный выше алгоритм. Номер нажатой клавиши определяется по номеру ряда клавиатуры, в котором обнаружена нажатая клавиша, и номеру разряда

входного устройства, в котором записан ноль.

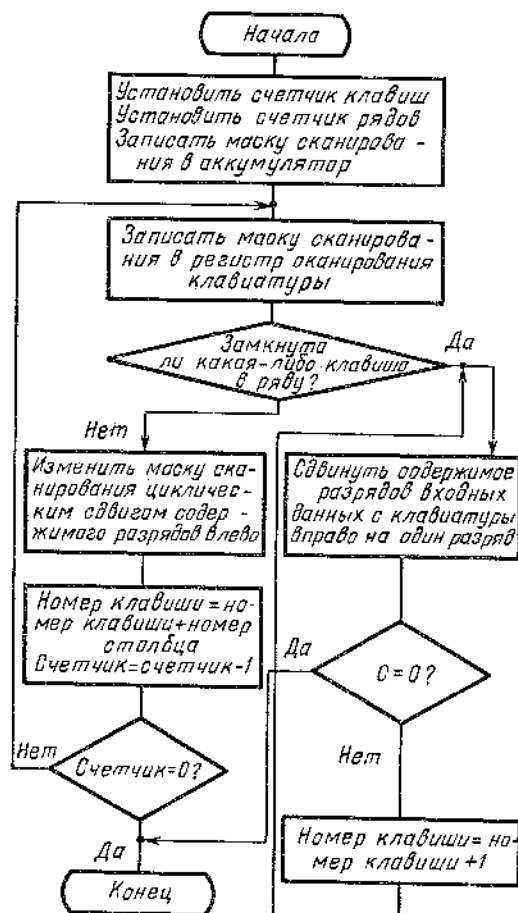


Рисунок 6.9 – Схема алгоритма определения нажатой клавиши при сканировании клавиатуры микро-ЭВМ

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	06 00		<i>MVI B, 00</i>	обнулить счетчик клавиш
0802	0E FE		<i>MVI C, FE</i>	начальная установка маски сканирования рядов
0804	16 08		<i>MVI D, 08</i>	установить счетчик рядов
0806	79	<i>FROW</i>	<i>MOV A, C</i>	записать маску сканирования в аккумулятор
0807	D3 07		<i>OUT KBDOT</i>	записать маску сканирования в выходное устройство (адрес 07h)

0809	07		<i>RLC</i>	изменить маску сканирования
080A	4F		<i>MOV C, A</i>	сохранить маску в регистре <i>C</i>
080B	DB 06		<i>IN KBDIN</i>	получить число с входного устройства (адрес 06h)
080D	E6 0F		<i>ANI 0F</i>	маскировать старшую тетраду
080F	FE 0F		<i>CPI 0F</i>	есть ли в младшей тетраде нули?
0811	C2 1F08		<i>JNZ FCOL</i>	если да, то идти на определение столбца (<i>FCOL</i>)
0814	78		<i>MOV A, B</i>	} если нет, то изменить содержание счетчика номера клавиш
0815	C6 04		<i>ADI 04</i>	
0817	47		<i>MOV B, A</i>	
0818	15		<i>DCR D</i>	уменьшить содержимое счетчика рядов
0819	C2 0608		<i>JNZ FROW</i>	если не последний ряд, то повторить для следующего ряда
081C	C3 2708		<i>JMP DONE</i>	если последний, то идти на окончание
081F	1F	<i>FCOL</i>	<i>RAR</i>	} определение номера строки, в которой записан ноль и определение номера нажатой клавиши
0820	D2 2708		<i>JNC DONE</i>	
0823	04		<i>INR B</i>	
0824	C3 1F08		<i>JMP FCOL</i>	
0827	00	<i>DONE</i>	<i>NOP</i>	метка завершения
0828	76		<i>HLT</i>	останов программы

Рисунок 6.10 – «Программа 4»

«Программа 4» представляет собой наиболее оптимальный алгоритм определения номера нажатой кнопки. К сожалению, в эмуляторе «Программная модель стенда УМПК-80 на базе микропроцессора

К580ВМ80» данный алгоритм реализовать невозможно в силу следующих причин:

- время такта в эмуляторе не соответствует действительному времени реального МП К580ВМ80, следовательно, время выполнения команды в эмуляторе также не соответствует микропроцессорному. Оно составляет в любом случае большую величину и зависит от быстродействия ПК, на котором работает эмулятор. Если во время выполнения отлаживаемой программы открыты все окна эмулятора, то это также увеличивает время такта;

- нажатие клавиши соответствует реальному, то есть при нажатии происходит обработка «дребезга контактов». Всего сигнал клавиши удерживается примерно от 200 до 600 мс, что также зависит от ПК;

- регистр устройства вывода формирует сигнал на своих выходах в течении времени, соответствующего реальной скорости сканирования клавиатуры.

С учетом выше сказанного делаем вывод, что при сканировании клавиатуры (выдача сигналов через устройство вывода) эмулятор в большинстве случаев не успеет определить нажатие кнопки, из-за низкой скорости эмуляции и, при этом, малого времени формирования сигнала от кнопки, соответствующего действительности. Поэтому и не возможна реализация оптимального алгоритма «Программы 4», как было сказано ранее.

Указанные особенности эмулятора требуют видоизменения программы, что и было выполнено в ущерб размеру и скорости работы. Основным элементом является введение цикла дополнительных считываний сигналов от клавиатуры. Текст «Программы 5» для эмулятора приведен на рисунке 6.11. В ней добавлена возможность определения шестнадцатеричного кода нажатой клавиши (по окончании определения код заносится в регистровую пару *HL*), а также возможность циклической работы для последовательного определения кодов всех клавиш.

Адрес	Машин. код	Метка	Мнемокод	Комментарий
0800	06 00	<i>KBD</i>	<i>MVI B, 00</i>	обнулить счетчик клавиш
0802	0E FE		<i>MVI C, FE</i>	начальная установка маски сканирования рядов
0804	16 08		<i>MVI D, 08</i>	установить счетчик рядов
0806	26 AF	<i>FROW</i>	<i>MVI H, AF</i>	установить время ожидания ввода
0808	79	<i>WAIT</i>	<i>MOV A, C</i>	записать маску сканирования в аккумулятор
0809	D3 07		<i>OUT KBDOT</i>	записать маску сканирования в выходное устройство (адрес 07h)
080B	DB 06		<i>IN KBDIN</i>	получить число с входного устройства (адрес 06h)
080D	E6 0F		<i>ANI 0F</i>	маскировать старшую тетраду
080F	FE 0F		<i>CPI 0F</i>	есть ли в младшей тетраде нули?
0811	C2 2608		<i>JNZ FCOL</i>	если да, то идти на определение столбца (<i>FCOL</i>)
0814	25		<i>DCR H</i>	уменьшить время ожидания ввода
0815	C2 0808		<i>JNZ WAIT</i>	если время ожидания не истекло, то ждать еще
0818	79		<i>MOV A, C</i>	загрузить маску в аккумулятор
0819	07		<i>RLC</i>	изменить маску сканирования
081A	4F		<i>MOV C, A</i>	сохранить маску в регистре C
081B	78		<i>MOV A, B</i>	} изменить содержание счетчика номера клавиш
081C	C6 04		<i>ADI 04</i>	
081E	47		<i>MOV B, A</i>	
081F	15		<i>DCR D</i>	уменьшить содержимое счетчика рядов

0820	C2 0608		JNZ FROW	если не последний ряд, то повторить для следующего ряда
0823	C3 0008		JMP KBD	если последний, то начать сканирование заново
0826	6F	FCOL	MOV L, A	записать младший байт кода клавиши в регистр L
0827	C5		PUSH B	сохранение счетчика номера клавиш в стек
0828	1F	CCOL	RAR	} определение номера строки, в которой записан ноль и определение номера нажатой клавиши
0829	D2 3008		JNC ECOL	
082C	04		INR B	
082D	C3 2808		JMP CCOL	
0830	61	ECOL	MOV H, C	записать старший байт кода клавиши в регистр H
0834	CF		RST1	прерывание программы
0835	00		NOP	
083A	C1		POP B	восстановить счетчик номера клавиши из стека
083B	C3 0608		JMP FROW	продолжить сканирование
083E	76		HLT	останов программы

Рисунок 6.11 – «Программа 5»

Часто при работе микро-ЭВМ ее операционная система строится таким образом, что информация, вводимая с клавиатуры, отображается на дисплее. Возможны случаи, когда при нажатии определенной клавиши будут выполнены заданные действия, причем различные, для разных клавиш. Такие режимы работы также будут исследованы в данной лабораторной работе.

Оборудование и материалы

В ходе выполнения лабораторной работы будет использоваться эмулятор «Программная модель стенда УМПК-80 на базе микропроцессора K580BM80» (далее УМПК – по заголовку главного окна эмулятора). Эмулятор находится в папке *UMPK80*.

Указания по технике безопасности

Перед проведением лабораторной работы следует изучить инструкцию по технике безопасности при работе на компьютере. Также следует соблюдать правила и требования, установленные для данного компьютерного класса.

Задания

1 Задание на подготовку к выполнению экспериментальной части работы.

1.1 Ознакомьтесь со схемами подключения клавиатуры и дисплея к микро-ЭВМ, приведенными на рисунках 6.3 и 6.7.

1.2 Изучите принцип мультиплексного вывода информации на дисплей.

1.3 Изучите принцип определения номера нажатой клавиши на клавиатуре.

1.4 Изучите «Программы 1,..., 5».

1.5 Разработайте программу мультиплексного последовательного вывода лишь одной цифры, задаваемой со входного устройства микро-ЭВМ на разные ячейки дисплея. Для задания времени высвечивания цифры на каждой ячейке дисплея используйте подпрограмму временной задержки. Определите, на что будет влиять время задержки.

1.6 Модернизируйте «Программу 5» так, чтобы номер нажатой клавиши отображался на дисплее.

1.7 Модернизируйте «Программу 5» так, чтобы на дисплее отображался тот символ, который показан на самой клавише.

1.8 Разработайте программу передачи управления по заданным адресам в зависимости от нажатой клавиши на клавиатуре. Адреса передачи управления в зависимости от номера клавиши на клавиатуре задаются таблицей. Адрес передачи управления записывается в регистры H, L .

2 Задание на выполнение экспериментальной части работы.

2.1 Исследовать простую программу включения сегментов пятой ячейки дисплея с помощью кода, задаваемого с входного устройства микро-ЭВМ, для чего выполнить следующее:

- ввести в микро-ЭВМ «Программу 1»;
- осуществить пуск программы и проверить ее выполнение, задавая различные значения на входном устройстве;
- меняя число на входном устройстве выявить соответствия кодов числа, записываемого в регистр сегментов дисплея, включаемым сегментам. Полученные данные занести в таблицу 6.1. Проверить, все ли возможные символы указаны в таблице, если нет, то добавить недостающие в таблицу;
- выбрать другой разряд дисплея и вывести в него символ. Разряд и символ задает преподаватель;
- сделать выводы и представить результаты выполнения в виде снимков экрана.

Таблица 6.1 – Соответствие кодов символам

Символ	0	1	2	3	4	5	6	7	8	9	A	b	C	d
Код														

Продолжение таблицы 6.1

Символ	<i>E</i>	<i>F</i>	<i>Space</i>	<i>H</i>	<i>h</i>	<i>L</i>	<i>Y</i>	Б	Г	Е	Н	о	П	п
Код														
Символ	Р	С	У	Ь	–	–	8.	<i>U</i>	<i>u</i>	<i>r</i>	<i>l</i>			
Код														

2.2 Исследовать программу, обеспечивающую мультиплексный режим работы дисплея, для чего выполнить следующее:

- ввести в микро-ЭВМ «Программу 2», подпрограмму регулируемой временной задержки *DELB* и данные, приведенные в описании программы в п. 2 теоретического обоснования;

- осуществить пуск программы и прочитать сообщение, выводимое на дисплей. Длительность задержки необходимо экспериментально подобрать так, чтобы удобно было читать выводимое на дисплей сообщение;

- записать новое сообщение в память эмулятора (задает преподаватель). Изменив способ вывода информации (задает преподаватель), осуществить пуск программы. Убедиться в правильности выводимого сообщения и способа вывода. Для формирования шестнадцатеричных кодов символов сообщения необходимо воспользоваться данными таблицы 6.1;

- сделать выводы и представить результаты выполнения в виде снимков экрана.

2.3 Исследовать программу, разработанную в п. 1.5, для чего выполнить следующее:

- ввести в микро-ЭВМ разработанную программу;

- осуществить пуск программы и проверить, что на дисплей выводится лишь один символ в один разряд в данный момент времени и последовательное включение индикаторов дисплея проходит в требуемом порядке;

- установить в программе время включенного состояния цифры на

каждой ячейке дисплея, заданное преподавателем (время задается подпрограммой задержки);

- проследить изменения в информации, выводимой на дисплей;
- сделать выводы и представить результаты выполнения в виде снимков экрана.

2.4 Исследовать программу определения факта нажатия на одну из клавиш, для чего выполнить следующее:

- ввести в микро-ЭВМ «Программу 3»;
- осуществить пуск программы и проверить ее функционирование. Если клавишу не нажимать, то программа будет постоянно сканировать клавиатуру;
- определить реакцию программы на нажатие клавиши;
- после останова программы выяснить код столбца нажатой клавиши. Нажимая на разные клавиши определить код столбца для каждой (клавишу задает преподаватель);

– видоизменить «Программу 3» так, чтобы при нажатии клавиши она осуществляла вывод кода столбца на светодиоды эмулятора в течение 5 с, затем, выключив светодиоды, продолжила сканирование клавиатуры, ожидая нового нажатия;

- ввести в микро-ЭВМ видоизмененную программу;
- осуществить пуск программы и проверить ее функционирование;
- сделать выводы и представить результаты выполнения в виде снимков экрана.

2.5 Исследовать программу определения шестнадцатеричного кода и номера нажатой клавиши, для чего выполнить следующее:

- ввести в микро-ЭВМ «Программу 5», разработанную с учетом особенностей эмулятора;
- установить в эмуляторе точки останова на адреса «0800» и «0806». Первая точка служит для визуального определения пользователем момента,

когда цикл сканирования начинается заново в случае, если за время сканирования всех строк не было ни одного нажатия клавиши. Вторая точка служит для визуального определения пользователем момента, когда программа закончит сканирование клавиш одной строки и будет готова перейти к следующей. В регистре *C* будет указана текущая маска сканирования строки. После останова на указанных точках пользователь может продолжить выполнение программы, подав команду «Запуск программы» в эмуляторе;

- осуществить пуск программы;

- нажимать по очереди на все клавиши клавиатуры эмулятора (**не рекомендуется нажимать на клавиши с частотой более одного раза в секунду!** Это также связано с особенностями эмулятора). Как только программа обнаружит нажатие клавиши, она определит ее код и номер и остановится. Двухбайтный код будет записан в регистровой паре *H*, *L*, а однобайтный номер – в регистре *B* (оба числа будут представлены в шестнадцатеричном виде). Время ожидания нажатия задается в регистре *H*. По умолчанию установлена величина, достаточная для нажатия всех клавиш за указанное время с частотой одна клавиша в секунду. В зависимости от быстродействия ПК возможно это время придется подобрать опытным путем;

- загрузить программу снова, установить точки останова и осуществить ее запуск. Нажимая на все клавиши последовательно определить код и номер каждой клавиши (**внимание: клавиши «R», «Ст» и «Шц» не нажимать!**). Результаты занести в таблицу 6.2;

- выяснить действия программы в случае, если за время сканирования всех строк не была нажата ни одна клавиша;

- выявить коды и номера, не используемые для клавиш. Записать их в отчете списком;

- сделать выводы и представить результаты выполнения в виде

снимков экрана.

Таблица 6.2 – Шестнадцатеричные коды и номера клавиш клавиатуры эмулятора УМПК-80

Клавиша (символ)									
Код									
Номер									

2.6 Исследовать программу, разработанную в п. 1.6, для чего выполнить следующее:

- ввести в микро-ЭВМ разработанную программу. Рекомендуется использовать точки останова на адресах «0800» и «0806». Программа должна индцировать номер только до момента запуска продолжения выполнения по команде «Запуск программы» в эмуляторе;

- осуществить пуск программы и проверить, что на дисплей выводится номер нажатой клавиши. Сопоставить с данными таблицы 6.2;

- сделать выводы и представить результаты выполнения в виде снимков экрана.

2.7 Исследовать программу, разработанную в п. 1.7, для чего выполнить следующее:

- ввести в микро-ЭВМ разработанную программу. Рекомендуется использовать точки останова на адресах «0800» и «0806». Программа должна индцировать символ нажатой клавиши только до момента запуска продолжения выполнения по команде «Запуск программы» в эмуляторе;

- записать в ОЗУ эмулятора коды, соответствующие символам, изображенным на клавишах. Начальный адрес выбрать «0B00». Использовать символы только для клавиш «0», «1», «2», «3», «4», «5», «6», «7», «8», «9», «A», «B», «C», «D», «E», «F». Для остальных клавиш использовать символ

«У» (клавиша управления). Для неиспользуемых номеров матрицы записать в ОЗУ «00» (**внимание: клавиши «R», «Ст» и «Шц» не нажимать!**);

- осуществить пуск программы и проверить, что на дисплей выводится символ нажатой клавиши;

- сделать выводы и представить результаты выполнения в виде снимков экрана.

5.2.8 Исследовать программу, разработанную в п. 1.8, для чего выполнить следующее:

- ввести в микро-ЭВМ разработанную программу;
- установить соответствие нажатых клавиш и адресов, по которым будет передаваться управление, согласно заданию преподавателя;

- осуществить запуск программы и проверить соответствие адреса, записанного в регистры H , L , номеру нажатой клавиши;

- сделать выводы и представить результаты выполнения в виде снимков экрана.

3 Задание на обработку экспериментальных данных.

3.1 Используя результаты выполнения п. 2.1 привести фрагменты машинного кода «Программы 1», необходимые для установки условий, заданных преподавателем.

3.2 Используя результаты выполнения п. 2.2 привести фрагменты машинного кода «Программы 2», необходимые для установки условий, заданных преподавателем.

3.3 Используя результаты выполнения п. 2.3 привести фрагменты машинного кода написанной программы, необходимые для установки условий, заданных преподавателем.

3.4 Используя результаты выполнения п. 2.4 привести фрагменты машинного кода «Программы 3», необходимые для установки условий, заданных преподавателем.

3.5 Используя результаты выполнения п. 2.5 нарисовать схему

подключения исследуемой клавиатуры ввода данных и управления к микро-ЭВМ. Нанести на схему символы кнопок согласно данным таблицы 6.2.

3.6 Используя результаты выполнения п. 2.6 сопоставить полученные результаты с данными таблицы 6.2.

3.7 Используя результаты выполнения п. 2.8 привести фрагменты машинного кода написанной программы, необходимые для установки условий, заданных преподавателем.

Содержание отчета

Отчет по лабораторной работе выполняется на листах формат А4 согласно требований ГОСТ и ЕСКД и должен содержать следующие пункты:

- а) титульный лист с названием работы, дисциплины, кода группы, ФИО исполнителя и преподавателя;
- б) цель и содержание работы;
- в) схемы подключения исследуемого дисплея;
- г) схему подключения исследуемой клавиатуры;
- д) тексты программ, выполненных по пунктам 1.5,..., 1.8 заданий на подготовку к выполнению экспериментальной части работы;
- е) тексты исследуемых программ – «Программы 1,..., 5»;
- ж) таблицу соответствия кодов символам;
- з) результаты выполнения заданий по пунктам 2.2 и 2.3;
- и) результаты выполнения заданий по п. 2.4 и видоизмененную «Программу 3», исследованную в задании п. 2.4;
- к) шестнадцатеричные коды и номера клавиш клавиатуры эмулятора УМПК-80;
- л) результаты исследований по пунктам 2.6, 2.7 и 2.8;
- м) выводы по результатам работы.

Контрольные вопросы

- 1 Какие коды необходимо записать для вывода на дисплей чисел 1, 2, 3, 4, 5, 6 и т. д.?
- 2 Как следует изменить «Программу 2», чтобы изображение на дисплее начало равномерно перемещаться?
- 3 Как следует изменить «Программу 2» для изменения направления сканирования индикаторов дисплея?
- 4 Какая из клавиш будет определяться нажатой с помощью «Программы 5», если на клавиатуре будут одновременно нажаты:
 - а) клавиши «3» и «5»;
 - б) клавиши «2» и «D»?
- 5 Видоизмените «Программу 5» так, чтобы микро-ЭВМ определяла номер нажатой клавиши в клавиатуре, организованной в виде матрицы 8×8 .
- 6 Видоизмените «Программу 5» так, чтобы номер нажатой клавиши выводился на светодиоды порта вывода эмулятора.

Список литературы, рекомендованный к использованию по данной теме

Основная учебная литература

- 1 Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд., испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

Дополнительная учебная литература

- 1 Петровский И.И. и др. Логические ИС K1533, K1554: Справочник / В двух частях. - М.: ТОО «БИНОМ», 1993 – 500 с.
- 2 Шевкопляс Б.С. Микропроцессорные структуры. Инженерные

решения: Справочник. - М.: Радио и связь, 1990 – 512 с.

3 Щелкунов, Н. Н. Микропроцессорные средства и системы / Н. Н. Щелкунов, А. П. Дианов. – М. : Радио и связь, 1989. – 288 с.

4 Коффрон, Дж. Технические средства микропроцессорных систем: практический курс / Дж. Коффрон. Пер. с англ. – М. : Мир, 1983. – 344 с.

5 Шахнов, В. А. Микропроцессоры и микропроцессорные комплекты интегральных микросхем : справочник / В.-Б. Б. Абрайтис, Н. Н. Аверьянов, А. И. Белоус [и др.]: под ред. В. А. Шахнова. – Том 1. – М. : Радио и связь, 1988. – 368 с.

6 Соппа, И. В. Микропроцессоры и микроконтроллеры / И. В. Соппа, А. С. Самардак, В. А. Иванов. – Владивосток : Издательство Дальневосточного университета, 2009. – 211 с.

7 Алексенко, А. Г. Основы микросхемотехники / А. Г. Алексенко. 3-е изд. перераб. и доп. – М. : БИНОМ. Лаборатория знаний, 2009. – 448 с.

8 Якубовский, С. В. Аналоговые и цифровые интегральные микросхемы : справочное пособие / С. В. Якубовский, Н. А. Барканов, Л. И. Ниссельсон [и др.]; под ред. С. В. Якубовского. – 2-е изд., перераб. и доп. – М. : Радио и связь, 1984. – 432 с.

9 Цифровые интегральные микросхемы : справочник / М. И. Богданович, И. Н. Грель, С. А. Дубина [и др.]. – 2-е изд., перераб. и доп. – Минск : Беларусь, Польша, 1996. – 605с.

ЛАБОРАТОРНАЯ РАБОТА 7

Тема: Исследование методов моделирования функций алгебры логики.

Цель работы: исследование возможности эмуляции функций алгебры логики (ФАЛ) с помощью микропроцессора; исследование метода программной реализации ФАЛ; исследование метода моделирования ФАЛ.

Формируемые компетенции: ОК-6, ОК-7, ОПК-6, ОПК-8, ОПК-9, ПК-3.

Теоретическая часть

В большинстве современных ЭВМ и цифровых устройствах различного назначения обработка информации осуществляется с помощью двоичных чисел, операции над которыми выполняют логические элементы (ЛЭ).

На входы любых логических элементов или цифровых устройств поступают сигналы, называемые логическими переменными, которые могут принимать два значения: логический ноль или логическая единица.

В зависимости от значений этих переменных на выходе будет образован сигнал – выходная логическая функция (ЛФ). ЛФ – это выражение логической зависимости значения логической функции от значения входных логических переменных. Выходная логическая функция также может принимать два значения: логический ноль и логическая единица.

Если соединить между собой несколько одинаковых или различных ЛЭ, то получится более сложная логическая функция. Таким образом, комбинируя определенным образом ЛЭ можно получить сколь угодно сложную логическую функцию или их системы.

Многие из таких устройств, выполненные в виде специальных микросхем, предназначены для выполнения стандартных операций над цифровыми сигналами.

Таким образом, роль ЛЭ и функциональных узлов на их основе достаточно высока. Поэтому важным является умение разрабатывать устройства, как на ЛЭ, так и на специализированных цифровых микросхемах. В микропроцессорных системах часто ЛЭ и цифровые узлы реализуются на программном уровне, эмулируя работу реальных устройств, что доказывает актуальность данной лабораторной работы.

Рассмотрим пример ФАЛ, заданной полной таблицей истинности (таблица 7.1), в которой всем набором входных переменных (x_2, x_1, x_0) ставится в соответствие вектор выходных значений ФАЛ (y_1, y_0) . Соответствующая таблице совершенная дизъюнктивная нормальная форма (СДНФ) имеет следующий вид:

$$\begin{cases} y_1 = \sum(2, 3, 4); \\ y_0 = \sum(1, 2, 5, 7). \end{cases} \quad (7.1)$$

Таблица 7.1 – Пример ФАЛ с тремя входными переменными и двумя выходными функциями

x_2	x_1	x_0	y_1	y_0
0	0	0	0	0
0	0	1	0	1
0	1	0	1	1
0	1	1	1	0
1	0	0	1	0
1	0	1	0	1
1	1	0	0	0
1	1	1	0	1

Метод программной реализации ФАЛ, заданных полной таблицей истинности состоит из следующих этапов:

- в память МПС записывается массив векторов выходных значений ФАЛ для всех 2^n наборов значений входных переменных (в порядке возрастания номера набора);

- считываемое извне значение входных переменных интерпретируется как номер строки в массиве, суммируется с начальным адресом массива и образует адрес строки выходных значений ФАЛ.

Метод моделирования ФАЛ, заданной СДНФ, состоит из следующих этапов:

- в памяти МПС формируется массив входных наборов – термов, на которых ФАЛ принимают значение логической единицы;

- после получения текущего входного набора обслуживающая программа ведет поиск терма, равного текущему входному набору и, в случае обнаружения равного, присваивает ФАЛ значение логической единицы;

- если равенства не будет обнаружено для всех термов в массиве, то ФАЛ присваивается значение логического нуля.

Программная реализация рассмотренных методов будет исследоваться в данной лабораторной работе.

Оборудование и материалы

В ходе выполнения лабораторной работы будет использоваться эмулятор «Программная модель стенда УМПК-80 на базе микропроцессора K580BM80» (далее УМПК – по заголовку главного окна эмулятора). Эмулятор находится в папке *UMPK80*.

Указания по технике безопасности

Перед проведением лабораторной работы следует изучить инструкцию

по технике безопасности при работе на компьютере. Также следует соблюдать правила и требования, установленные для данного компьютерного класса.

Задания

1 Задание на подготовку к выполнению экспериментальной части работы.

Все задания данного пункта выполняются на основе изученного материала по дисциплинам: «Схемотехника» и «Микросхемотехника».

1.1 Записать основные логические операции и нарисовать соответствующие им элементы в виде условно-графических обозначений.

1.2 Записать таблицы истинности, отображающие логику работы каждого ЛЭ.

1.3 Записать логику работы ЛЭ в виде математических выражений.

1.4 Определить соответствие между таблицей истинности и записанными выражениями.

1.5 Разработать программу для исследования метода программной реализации ФАЛ, включающую следующие процедуры:

- процедура ввода значений функций для всех наборов заданной СДНФ;

- процедура чтения записанных значений ФАЛ.

Для ввода значений функций необходимо использовать порт ввода эмулятора (переключатели), имеющий адрес «05h».

Процедура ввода значений ФАЛ в ОЗУ эмулятора:

- установить необходимые значения ФАЛ;
- разрешить запись значений в ОЗУ эмулятора нажатием кнопки управления;

- в регистре *B* должно отображаться количество введенных значений;

– по завершению ввода всех значений программа автоматически должна перейти к процедуре чтения. Содержимое регистра *B* должно сохраняться.

Условия выбора пространства памяти для записи значений:

- запись нулевого набора значений должна быть произведена по первому адресу, определяемому по смещению;
- запись последующих наборов должна производиться строго по возрастанию адреса и по возрастанию номеров самих наборов.

Для вывода значений функций необходимо использовать порт вывода эмулятора (светодиоды), имеющий адрес «05h». Для ввода наборов входных переменных необходимо использовать порт ввода эмулятора (переключатели), имеющий адрес «05h».

Процедура чтения записанных значений ФАЛ:

- установить необходимый набор входных переменных;
- разрешить чтение значений ФАЛ нажатием на кнопку управления;
- проверить корректность установленного набора на входном устройстве. Если набор установлен неправильно, то в регистр *C* должно быть записано значение «EE». Программа ничего не считывает и снова переходит к ожиданию ввода набора входных переменных;
- если входной набор установлен верно, то программа считывает соответствующие ему значения ФАЛ;
- считанное значение индицируется пользователю;
- в регистр *C* записывается номер индицируемого набора значений;
- индикация длится 2 с, после чего ее необходимо выключить.

Программа возвращается к началу процедуры чтения. Значение в регистре *C* сохраняется либо до вывода нового набора значений ФАЛ, либо до ситуации с неправильным вводом набора входных переменных.

Исходные данные приведены в таблице 7.2.

Таблица 7.2 – Исходные данные

Номер варианта	Система СДНФ	Разряды входного устройства	Разряды выходного устройства /ячейка дисплея	Смещение
1	2	3	4	5
1	$\begin{cases} y_1 = \sum(0, 1, 2, 4, 8, 9, 14); \\ y_0 = \sum(2, 3, 4, 5, 15). \end{cases}$	7, 6, 5, 4	1, 0 / 0	10
2	$\begin{cases} y_1 = \sum(2, 3, 5, 7, 8, 12, 14); \\ y_0 = \sum(2, 5, 6, 9, 10, 13, 15). \end{cases}$	6, 5, 4, 3	1, 0 / 1	11
3	$\begin{cases} y_1 = \sum(2, 5, 7, 9, 11, 12, 14); \\ y_0 = \sum(0, 5, 6, 12, 13, 14, 15). \end{cases}$	5, 4, 3, 2	1, 0 / 2	12
4	$\begin{cases} y_1 = \sum(6, 7, 8, 9, 11, 12, 14); \\ y_0 = \sum(1, 2, 3, 4, 11, 13, 15). \end{cases}$	4, 3, 2, 1	1, 0 / 3	13
5	$\begin{cases} y_1 = \sum(0, 1, 2, 4, 6, 7, 14); \\ y_0 = \sum(7, 8, 12, 13, 14, 15). \end{cases}$	3, 2, 1, 0	1, 0 / 4	14
6	$\begin{cases} y_1 = \sum(1, 3, 5, 7, 9, 11, 13); \\ y_0 = \sum(0, 2, 4, 6, 8, 10, 12). \end{cases}$	7, 6, 5, 4	3, 2 / 5	15
7	$\begin{cases} y_1 = \sum(3, 4, 5, 8, 9, 12, 14); \\ y_0 = \sum(0, 1, 5, 10, 14, 15). \end{cases}$	6, 5, 4, 3	3, 2 / 0	16

Продолжение таблицы 7.2

1	2	3	4	5
8	$\begin{cases} y_1 = \sum(0, 1, 5, 7, 9, 12, 14); \\ y_0 = \sum(0, 1, 5, 10, 11, 15). \end{cases}$	5, 4, 3, 2	3, 2 / 1	17
9	$\begin{cases} y_1 = \sum(0, 3, 5, 7, 9, 13, 14); \\ y_0 = \sum(0, 4, 6, 9, 10, 11, 12). \end{cases}$	4, 3, 2, 1	3, 2 / 2	18
10	$\begin{cases} y_1 = \sum(2, 5, 6, 8, 10, 12, 14); \\ y_0 = \sum(5, 7, 10, 11, 14, 15). \end{cases}$	3, 2, 1, 0	3, 2 / 3	19
11	$\begin{cases} y_1 = \sum(0, 1, 2, 3, 4, 5, 6); \\ y_0 = \sum(7, 8, 9, 10, 11, 12). \end{cases}$	7, 6, 5, 4	5, 4 / 4	20
12	$\begin{cases} y_1 = \sum(1, 3, 5, 11, 13); \\ y_0 = \sum(0, 2, 4, 10, 12). \end{cases}$	6, 5, 4, 3	5, 4 / 5	19
13	$\begin{cases} y_1 = \sum(7, 8, 9, 10); \\ y_0 = \sum(2, 5, 6, 9, 10, 13, 15). \end{cases}$	5, 4, 3, 2	5, 4 / 0	18
14	$\begin{cases} y_1 = \sum(0, 1, 7, 9, 10, 11, 12); \\ y_0 = \sum(0, 3, 6, 11, 12, 13, 15). \end{cases}$	4, 3, 2, 1	5, 4 / 1	17
15	$\begin{cases} y_1 = \sum(12, 13, 14, 15); \\ y_0 = \sum(1, 2, 3, 4, 5, 10, 13, 15). \end{cases}$	3, 2, 1, 0	5, 4 / 2	16
16	$\begin{cases} y_1 = \sum(2, 5, 7, 9, 11, 12, 14); \\ y_0 = \sum(0, 5, 6, 12, 13, 14, 15). \end{cases}$	7, 6, 5, 4	7, 6 / 3	15
17	$\begin{cases} y_1 = \sum(0, 1, 8, 9, 11, 15); \\ y_0 = \sum(7, 8, 9, 10). \end{cases}$	6, 5, 4, 3	7, 6 / 4	14
18	$\begin{cases} y_1 = \sum(1, 3, 5, 6, 7, 8); \\ y_0 = \sum(0, 1, 3, 4, 5, 12). \end{cases}$	5, 4, 3, 2	7, 6 / 5	13

Продолжение таблицы 7.2

1	2	3	4	5
19	$\begin{cases} y_1 = \sum(1, 2, 3, 4, 5); \\ y_0 = \sum(1, 3, 5, 7, 8, 11, 13, 15). \end{cases}$	4, 3, 2, 1	7, 6 / 0	12
20	$\begin{cases} y_1 = \sum(2, 10, 11, 12, 14, 15); \\ y_0 = \sum(0, 1, 2, 3, 4, 15). \end{cases}$	3, 2, 1, 0	7, 6 / 1	11

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется сначала для нахождения начального адреса расположения данных:

$$AddData = AddEndOf Prog + SS, \quad (7.2)$$

где $AddData$ – начальный адрес для размещения данных;

$AddEndOf Prog$ – адрес окончания программы;

SS – смещение.

Номера разрядов входного устройства используются только в процедуре чтения. Именно с этих разрядов должен подаваться код входного набора переменных. Программа должна выполнять проверку: если в каких-либо других разрядах присутствует логическая единица, то ввод набора выполнен неверно. Необходимо учитывать, что слева располагается старший разряд, соответствующий старшей переменной x_3 .

Номера разрядов выходного устройства используются только в процедуре чтения. На эти светодиоды необходимо выводить значения ФАЛ. В остальные разряды выходного устройства всегда должны быть записаны сигналы логического нуля. Необходимо учитывать, что слева располагается индикатор старшей функции y_1 .

В качестве кнопки управления можно использовать любую клавишу в диапазоне от «0» до «F» клавиатуры эмулятора. Для определения нажатия на

кнопку рекомендуется использовать алгоритм «Программы 3» лабораторной работы 6, видоизменив ее под условия данной задачи.

1.6 Разработать программу для исследования метода моделирования ФАЛ, заданной СДНФ, включающую следующие процедуры:

- процедура ввода массива наборов входных переменных, на которых ФАЛ принимает значение, равное логической единице;
- процедура поиска набора в массиве и присвоения ФАЛ соответствующего значения.

Для ввода наборов необходимо использовать порт ввода эмулятора (переключатели), имеющий адрес «05h».

Процедура ввода массива наборов в ОЗУ эмулятора:

- установить необходимый набор входных переменных;
- разрешить запись в ОЗУ эмулятора нажатием кнопки управления;
- проверить корректность установленного набора на входном устройстве. Если набор установлен неправильно, то в регистр *C* должно быть записано значение «EE». Программа запись выполнять не будет и снова переходит к ожиданию ввода набора входных переменных;
- если входной набор установлен верно, то программа выполняет его запись;
- в регистре *B* должно отображаться количество введенных наборов;
- по завершению ввода всех наборов программа автоматически должна перейти к процедуре поиска. Содержимое регистра *B* должно сохраняться.

Условия выбора пространства памяти для записи наборов такие же, как и для записи значений ФАЛ (см. п. 1.5.).

Для вывода значения функции необходимо использовать дисплей эмулятора. Для ввода наборов входных переменных необходимо использовать порт ввода эмулятора (переключатели), имеющий адрес «05h».

Процедура поиска набора в массиве и присвоения ФАЛ соответствующего значения:

- установить необходимый набор входных переменных;
- разрешить поиск нажатием на кнопку управления;
- проверить корректность установленного набора на входном устройстве. Если набор установлен неправильно, то в регистр C должно быть записано значение «ЕЕ». Программа поиск выполнять не будет и снова переходит к ожиданию ввода набора входных переменных;
- если входной набор установлен верно, то программа ищет его в записанном массиве. Если набор найден, ФАЛ присваивается значение логической единицы, в противном случае – нуля;
- это значение индицируется пользователю;
- в регистр C записывается номер входного набора;
- индикация длится 2 с, после чего ее необходимо выключить.

Программа возвращается к началу процедуры поиска. Значение в регистре C сохраняется либо до ввода нового набора переменных и вывода значения ФАЛ, либо до ситуации с неправильным вводом набора входных переменных.

Исходные данные приведены в таблице 7.2. Для выполнения задания необходимо использовать только логическую функцию y_0 из заданной системы.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется сначала для нахождения начального адреса расположения наборов по формуле (7.2).

Номера разрядов входного устройства используются как в процедуре записи наборов, так и в процедуре поиска аналогично процедуре чтения (см. п. 1.5).

Как было сказано выше, для вывода значения функции необходимо использовать дисплей эмулятора. Значение ФАЛ будет отображаться либо символом «0», либо «1» в соответствующей заданию ячейке дисплея. Для

управления дисплеем рекомендуется использовать алгоритм «Программы 1» лабораторной работы 6, видоизменив ее под условия данной задачи.

В качестве кнопки управления можно использовать любую клавишу в диапазоне от «0» до «F» клавиатуры эмулятора. Для определения нажатия на кнопку рекомендуется использовать алгоритм «Программы 3» лабораторной работы 6, видоизменив ее под условия данной задачи.

2 Задание на выполнение экспериментальной части работы.

5.2.1 Исследовать программу, разработанную в п. 1.5, для чего выполнить следующее:

- ввести в микро-ЭВМ разработанную программу;
- осуществить пуск программы;
- проверить ее работоспособность в режиме ввода значений функций;
- проверить ее работоспособность в режиме чтения записанных значений;
- проверить реакцию программы на неверный набор входных переменных в режиме чтения;
- сделать выводы и представить результаты выполнения в виде снимков экрана.

2.2 Исследовать программу, разработанную в п. 1.6, для чего выполнить следующее:

- ввести в микро-ЭВМ разработанную программу;
- осуществить пуск программы;
- проверить ее работоспособность в режиме ввода массива наборов;
- проверить ее работоспособность в режиме поиска набора в массиве и присвоения значения ФАЛ;
- проверить реакцию программы на неверный набор входных переменных в режиме ввода массива наборов;
- проверить реакцию программы на неверный набор входных переменных в режиме поиска наборов в массиве;

– сделать выводы и представить результаты выполнения в виде снимков экрана.

3 Задание на обработку экспериментальных данных.

3.1 Используя результаты выполнения п. 2.1 нарисовать блок-схему алгоритма работы разработанной программы.

3.2 Используя результаты выполнения п. 2.1 выполнить описание реакции программы на неверный набор входных переменных.

3.3 Нарисовать схему электрическую принципиальную ФАЛ, которые эмулирует программа по п. 2.1.

3.4 Используя результаты выполнения п. 2.2 нарисовать блок-схему алгоритма работы разработанной программы.

3.5 Используя результаты выполнения п. 2.2 выполнить описание реакции программы на неверный набор входных переменных в обоих режимах.

3.6 Нарисовать схему электрическую принципиальную ФАЛ, которые эмулирует программа по п. 2.2.

Содержание отчета

Отчет по лабораторной работе выполняется на листах формат А4 согласно требований ГОСТ и ЕСКД и должен содержать следующие пункты:

- а) титульный лист с названием работы, дисциплины, кода группы, ФИО исполнителя и преподавателя;
- б) цель и содержание работы;
- в) условно-графические обозначения основных ЛЭ и их таблицы истинности;
- г) блок-схемы алгоритма работы программ по п. 1.5 и 1.6;
- д) схемы электрические принципиальные моделируемых ФАЛ;
- е) тексты программ, выполненных по пунктам 1.5 и 1.6 заданий на

- подготовку к выполнению экспериментальной части работы;
- ж) результаты выполнения заданий по пунктам 2.1 и 2.2;
- з) выводы по результатам работы.

Контрольные вопросы

- 1 Как следует изменить программу п. 1.5, чтобы можно было записывать иное количество значений функций?
- 2 Как следует изменить программу п. 1.5, чтобы можно было записывать функцию, зависящую от иного количества входных переменных?
- 3 Как следует изменить программу п. 1.6, чтобы можно было записывать иное количество значений функций?
- 4 Как следует изменить программу п. 1.6, чтобы можно было записывать функцию, зависящую от иного количества входных переменных?
- 5 Как следует изменить любую из программ, чтобы в качестве управляющей выступала конкретно заданная кнопка?
- 6 Можно ли видоизменить программу п. 1.6 таким образом, чтобы логических функций было две? Если можно, то как? Привести фрагмент кода.
- 7 Как следует видоизменить программу п. 1.5, чтобы значения функции выводились на дисплей? Привести фрагмент кода.
- 8 Можно ли данными методами реализовать дешифратор, мультиплексор? Если можно, то привести фрагменты кодов и пояснения.

Список литературы, рекомендованный к использованию по данной теме

Основная учебная литература

- 1 Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд.,

испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

Дополнительная учебная литература

1 Петровский И.И. и др. Логические ИС К1533, К1554: Справочник / В двух частях. - М.: ТОО “БИНОМ”, 1993 – 500 с.

2 Шевкопляс Б.С. Микропроцессорные структуры. Инженерные решения: Справочник. - М.: Радио и связь, 1990 – 512 с.

3 Щелкунов, Н. Н. Микропроцессорные средства и системы / Н. Н. Щелкунов, А. П. Дианов. – М. : Радио и связь, 1989. – 288 с.

4 Алексенко, А. Г. Основы микросхемотехники / А. Г. Алексенко. 3-е изд. перераб. и доп. – М. : БИНОМ. Лаборатория знаний, 2009. – 448 с.

5 Якубовский, С. В. Аналоговые и цифровые интегральные микросхемы : справочное пособие / С. В. Якубовский, Н. А. Барканов, Л. И. Ниссельсон [и др.]; под ред. С. В. Якубовского. – 2-е изд., перераб. и доп. – М. : Радио и связь, 1984. – 432 с.

6 Цифровые интегральные микросхемы : справочник / М. И. Богданович, И. Н. Грель, С. А. Дубина [и др.]. – 2-е изд., перераб. и доп. – Минск : Беларусь, Польша, 1996. – 605с.

ЛАБОРАТОРНАЯ РАБОТА 8

Тема: Исследование работы средств параллельного ввода-вывода.

Цель работы: исследовать возможности настройки программируемого периферийного адаптера; исследовать работу устройства в «режиме 0», «режиме 1» и «режиме 2»; исследовать возможности ввода-вывода с квити́рованием.

Формируемые компетенции: ОК-6, ОК-7, ОПК-6, ОПК-8, ОПК-9, ПК-1, ПК-3.

Теоретическая часть

Параллельный ввод-вывод (ВВ) – один из наиболее простых и широко распространенных способов обмена информацией с периферийными устройствами, при котором практически не требуется никакого промежуточного преобразования данных. Поэтому параллельный обмен информацией производится непосредственно портами ввода-вывода.

Для построения портов могут быть использованы различные БИС буферных регистров и шинных формирователей. Использование БИС с подобной организацией позволяет реализовывать различные параллельные интерфейсы ВВ. Проектирование интерфейса осуществляется аппаратными средствами за счет предварительной коммутации управляющих входов БИС. Оперативное изменение такого интерфейса практически невозможно, а его функции ограничены. Это обстоятельство сужает область применения подобных устройств, она ограничивается реализацией простейших параллельных интерфейсов с постоянной по времени логикой обмена.

В усложненном интерфейсе, когда логика обмена заранее не известна или характеристики процедур обмена во время работы МС должны меняться, используются программируемые периферийные адаптеры (ППА). В их состав входят программно-доступные регистры управляющих слов *СW*,

которые и определяют режим работы адаптера. Программный доступ к управляющему регистру со стороны микропроцессора дает возможность оперативно управлять работой ППА и изменять характеристики интерфейса.

К числу ППА относят БИС ППА фирмы *Intel* – *i8255* (отечественный аналог КР580ВВ55).

Программируемый периферийный адаптер *i8255* – это однокристальное программируемое устройство параллельного ВВ информации произвольного формата. В состав осуществляемых им процедур входит параллельный обмен данными с квитированием или без него как в режиме программного управления, так и по прерываниям. При этом обеспечивается организация не только однонаправленного, но и двунаправленного ВВ. Определение и переопределение типа интерфейса выполняется программными методами с помощью специальных процедур инициализации.

В состав ППА (рисунок 8.1) входят три двунаправленных восьмиразрядных порта, разбитых на две группы, два устройства управления группами портов и интерфейсная логика для согласования с системной магистралью. Схема управления содержит регистр управляющего слова *CW*, который доступен только для записи, чтение *CW* не допускается.

Обмен информацией между МП и внутренними регистрами ППА осуществляется через двунаправленный шинный формирователь и управляется сигналами $\overline{CS}$, $A0$, $A1$, $\overline{RD}$ и $\overline{WR}$ согласно таблице 8.1. Сигнал $\overline{CS}$ необходим для выбора кристалла.

Вход *RESET* служит для аппаратного сброса БИС в исходное состояние. Все внутренние регистры ППА, включая регистр управляющего слова *CW*, устанавливаются в «0». Сброс *CW* соответствует переводу всех портов в режим прямого ввода без квитирования.

В таблице 8.1 приняты следующие обозначения:

- х – любое значение (лог. «1» или лог. «0»);
- *D* – шина данных.

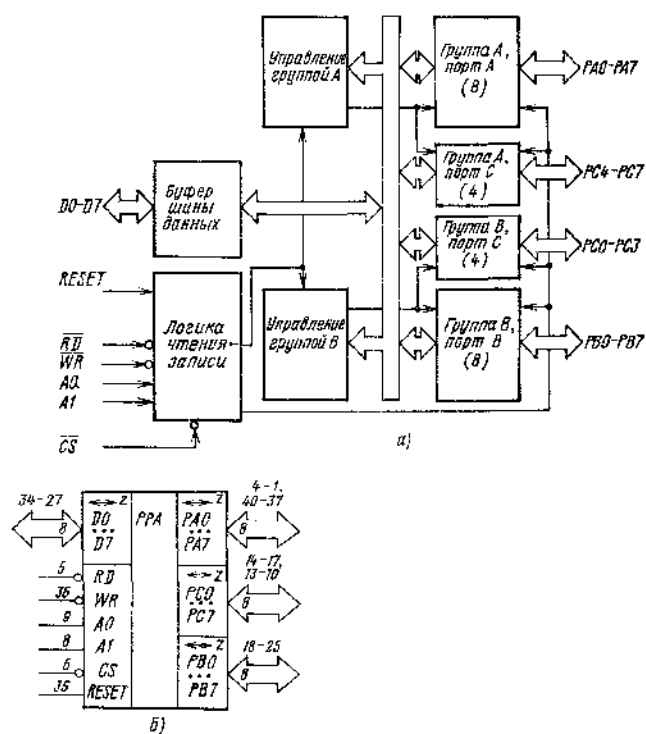


Рисунок 8.1 – Программируемый периферийный адаптер *i8255*:
а – структурная схема; б – условное графическое обозначение

Таблица 8.1 – Управляющие сигналы и режимы работы *i8255*

$A1$	$A0$	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	Операция
0	0	0	1	0	$D \leftarrow \text{порт } A$
0	1	0	1	0	$D \leftarrow \text{порт } B$
1	0	0	1	0	$D \leftarrow \text{порт } C$
1	1	0	1	0	Недопустимо
0	0	1	0	0	$\text{порт } A \leftarrow D$
0	1	1	0	0	$\text{порт } B \leftarrow D$
1	0	1	0	0	$\text{порт } C \leftarrow D$
1	1	1	0	0	$\text{Управление} \leftarrow D$
x	x	1	1	0	Нет операции
x	x	x	x	1	Нет операции

Дальнейшая настройка ППА выполняется программно с помощью специального управляющего слова *MS* (*Mode Selection*), которое назначает режим работы каждому каналу. Эти режимы могут быть изменены в любое время. Для хранения *MS* используется регистр *CW*. Формат *MS* приведен на рисунке 8.2, а. Управляющее слово определяется по признаку $D7=1$.

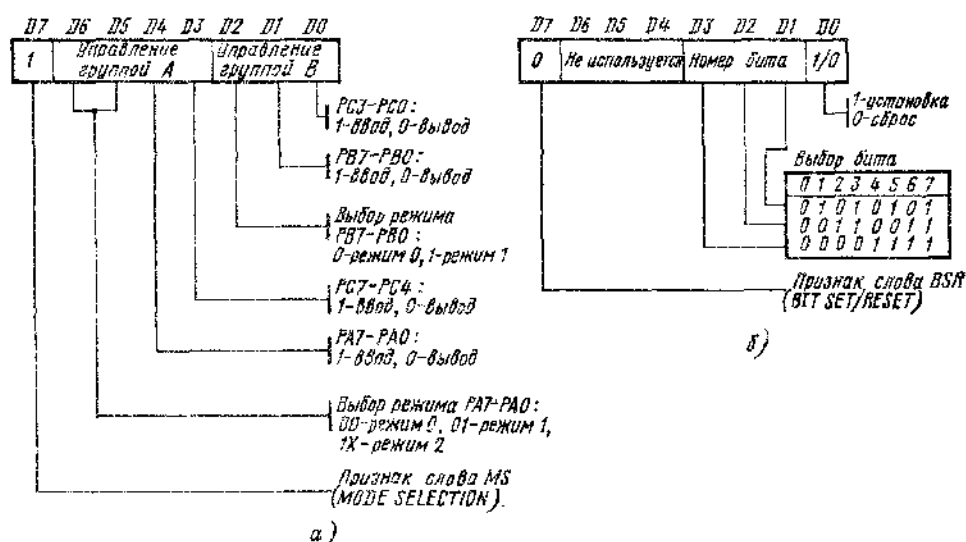


Рисунок 8.2 – Форматы управляющих слов *MS* (а) и *BSR* (б) ППА

При $D7=0$ управляющее слово применяется для независимой установки (сброса) разрядов выходного порта *C* и носит название *BSR* (*Bit Set/Reset*). Выборочная манипуляция содержимым выходного буферного регистра порта *C* важна при его использовании в качестве шины управления ПУ. Формат слова *BSR* представлен на рисунке 8.2, б.

Адаптер поддерживает три режима работы портов:

- «режим 0» – однонаправленный ВВ без квитирования (применим к любому из трех портов);
- «режим 1» – однонаправленный ВВ с квитированием (применим к портам *A* и *B*);
- «режим 2» – двунаправленный ВВ (допускается только для порта *A*).

При работе портов *A* и *B* в «режимах 1 и 2» часть линий порта *C* из соответствующей группы используется для управления обменом с внешними

ПУ. Часть порта *C*, занятая под управление обменом, реализует функцию слова состояния в соответствии с форматом, представленным на рисунке 8.3.

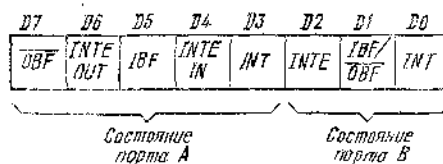


Рисунок 8.3 – Формат слова состояния *SW* ППА

Более подробно ознакомиться со структурой, организацией портов, режимами работы и способами программирования исследуемой БИС можно в основной [1] и дополнительной [1 - 5] литературе и по материалам курса лекций дисциплины «Основы микропроцессорной техники».

Оборудование и материалы

В ходе выполнения лабораторной работы будет использоваться эмулятор «Виртуальная МикроЭВМ *Intel® 8080*» версии 1.0 (далее ВМ). Эмулятор находится в папке *PyREStd/VDM8080*.

Эмулятор представляет собой набор модулей объединённых общим интерфейсом и позволяющих моделировать поведение следующих устройств:

- микропроцессор *Intel® 8080*;
- синхронную статическую системную память размером 64К x 8;
- контроллер прямого доступа к памяти *Intel® 8257*;
- программируемый контроллер прерываний *Intel® 8259*;
- программируемый интервальный таймер *Intel® 8253*;
- программируемый связной адаптер *Intel® 8251*;
- программируемый параллельный интерфейс *Intel® 8255*.

Кроме этого в пакете используется модуль компилятора *Easy 8080 Assembler* и редактора программ на языке *Assembler Intel® 8080*.

Подробное описание каждого модуля находится во вкладке «Помощь» верхнего меню главного окна программы. Во вкладке «Как пользоваться программой» необходимо ознакомиться со следующими пунктами:

- состав и структура комплектации;
- главное окно;
- рабочие модули комплекса.

Поскольку синтаксис написания программ в данном эмуляторе имеет некоторые особенности, то важно ознакомиться с ними во вкладке «Как пользоваться транслятором». Невыполнение указанных требований синтаксиса приведет к выводу сообщения об ошибке при попытке компиляции и, соответственно, невозможности компиляции написанной программы в память эмулятора.

Во вкладке «Структура “Виртуальной МикроЭВМ”» необходимо ознакомиться с содержанием следующих пунктов:

- карта адресов устройств;
- таблица включения модулей *i8080*, *i8228*, *i8255*.

Незнание содержания указанных пунктов приведет к невозможности работы с эмулятором и ППА в частности.

БИС *i8228* (отечественный аналог – КР580ВК28/ВК38) представляет собой системный контроллер и формирователь шины данных для МП ВМ80. Схема формирует базовый набор управляющих стробов магистрали типа И41 [2, 4] и обеспечивает двунанравленную буферизацию шины данных МП от основной памяти и устройств ВВ. Более подробно ознакомиться со структурой, организацией и режимами работы этой БИС можно в основной [1] и дополнительной [1 - 5] литературе.

Необходимо помнить, что работа эмулятора осуществляется только в ручном режиме. После компиляции программы в память эмулятора рекомендуется выполнить аппаратный сброс микропроцессора и ППА,

установив ручную «галочки» для сигналов сброса, и выполнив «Шаг команды».

Указания по технике безопасности

Перед проведением лабораторной работы следует изучить инструкцию по технике безопасности при работе на компьютере. Также следует соблюдать правила и требования, установленные для данного компьютерного класса.

Задания

1 Задание на подготовку к выполнению экспериментальной части работы.

1.1 Изучите структурную схему программируемого периферийного адаптера *i8255*.

1.2 Изучите особенности организации портов ППА.

1.3 Изучите режимы работы ППА и сигналы управления. Ознакомьтесь с временными диаграммами режимов работы.

1.4 Изучите форматы управляющих слов *MS* и *BSR*, способы программирования ППА.

2 Задание на выполнение экспериментальной части работы.

2.1 Исследовать работу ППА в «режиме 0», для чего выполнить следующее:

- настроить порт «X» на ввод в «режиме 0»;
- считать через настроенный порт из внешнего устройства *n* байт данных и записать в память ВМ, начиная с адреса *XXXX*;
- выполнить заданную операцию над введенными данными. Результат разместить в памяти ВМ через два адреса после данных, а также в регистрах

микропроцессора;

– настроить порт «X» на вывод в «режиме 0»;

– вывести через настроенный порт во внешнее устройство результат выполнения. Если результат получился двухбайтный, то для вывода старшего байта необходимо использовать любой другой порт ППА, настроенный на вывод в «режиме 0».

Исходные данные приведены в таблице 8.2.

Таблица 8.2 – Исходные данные для п. 2.1

Номер варианта	Кол-во байт	Порт ввода	Смещение	Операция	Порт вывода
1	2	3	4	5	6
1	50	A	20	ИЛИ	A
2	48	A	22	И	B
3	46	A	24	ИЛИ-НЕ	C
4	44	B	26	И-НЕ	B
5	42	B	28	Искл. ИЛИ	C
6	40	B	30	Искл. ИЛИ-НЕ	A
7	16	C	32	Арифм. сложение	C
8	12	C	34	Арифм. вычитание	A
9	52	C	36	ИЛИ	B
10	10	A	38	Арифм. вычитание	A
11	14	A	40	Арифм. сложение	B
12	38	A	42	И	C
13	41	B	44	ИЛИ-НЕ	B
14	47	B	46	И-НЕ	C
15	43	B	48	Искл. ИЛИ	A

Продолжение таблицы 8.2

1	2	3	4	5	6
16	45	<i>C</i>	50	Искл. ИЛИ-НЕ	<i>C</i>
17	49	<i>C</i>	52	Искл. ИЛИ-НЕ	<i>A</i>
18	11	<i>C</i>	54	Арифм. сложение	<i>B</i>
19	51	<i>A</i>	56	ИЛИ-НЕ	<i>A</i>
20	9	<i>A</i>	58	Арифм. вычитание	<i>B</i>

Данные для ввода студент выбирает самостоятельно. В разных вариантах данные не должны совпадать.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется для нахождения начального адреса расположения данных:

$$AddData = AddEndOf Prog + SS, \quad (8.1)$$

где *AddData* – начальный адрес для размещения данных;

AddEndOf Prog – адрес окончания программы;

SS – смещение.

При написании программы необходимо учитывать следующее:

- доступная пользователю память ВМ начинается с нулевого адреса;
 - при компиляции программы в память и файл необходимо задать начальный адрес загрузки, который зависит от смещения и составляет удвоенное его значение;
 - для организации стека в ВМ выделен определенный диапазон памяти.
- Размещать программу и данные необходимо так, чтобы не допустить пересечения этих областей.

После выполнения задания данного пункта необходимо сделать выводы. Результаты выполнения представить в виде снимков экрана.

2.2 Исследовать работу ППА в смешанном режиме работы портов, для

чего выполнить следующее:

- настроить порт «*X*» на ввод в «режиме 1», порт «*Y*» – на вывод в «режиме 0»;
- считать через настроенный порт «*X*» из внешнего устройства *n* байт данных и записать в память ВМ, начиная с адреса *XXXX*. Каждое вводимое с порта «*X*» число должно сразу выводиться в порт «*Y*»;
- выполнить заданную операцию над введенными данными. Результат разместить в памяти ВМ через два адреса после данных, а также в регистрах микропроцессора;
- вывести через настроенный порт «*Y*» во внешнее устройство результат выполнения.

Исходные данные приведены в таблице 8.3.

Таблица 8.3 – Исходные данные для п. 2.2

Номер варианта	Кол-во байт	Порт ввода	Смещение	Операция	Порт вывода
1	2	3	4	5	6
1	30	<i>A</i>	20	Искл. ИЛИ	<i>B</i>
2	28	<i>A</i>	22	Искл. ИЛИ-НЕ	<i>C</i>
3	26	<i>B</i>	24	И-НЕ	<i>A</i>
4	24	<i>B</i>	26	ИЛИ-НЕ	<i>C</i>
5	22	<i>A</i>	28	И	<i>B</i>
6	20	<i>A</i>	30	ИЛИ	<i>C</i>
7	18	<i>B</i>	32	Искл. ИЛИ	<i>A</i>
8	16	<i>B</i>	34	Искл. ИЛИ-НЕ	<i>C</i>
9	14	<i>A</i>	36	И-НЕ	<i>B</i>
10	15	<i>A</i>	38	ИЛИ-НЕ	<i>C</i>
11	17	<i>B</i>	40	И	<i>A</i>

Продолжение таблицы 8.3

1	2	3	4	5	6
12	19	<i>B</i>	42	ИЛИ	<i>C</i>
13	21	<i>A</i>	44	ИЛИ	<i>B</i>
14	23	<i>A</i>	46	И	<i>C</i>
15	25	<i>B</i>	48	Искл. ИЛИ-НЕ	<i>A</i>
16	27	<i>B</i>	50	Искл. ИЛИ	<i>C</i>
17	29	<i>A</i>	52	Искл. ИЛИ-НЕ	<i>B</i>
18	31	<i>A</i>	54	И-НЕ	<i>C</i>
19	13	<i>B</i>	56	ИЛИ	<i>A</i>
20	32	<i>B</i>	58	И-НЕ	<i>C</i>

Данные для ввода студент выбирает самостоятельно. В разных вариантах данные не должны совпадать.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде, начальный адрес расположения данных рассчитывается по формуле (8.1).

При написании программы необходимо учитывать те же условия, что и в п. 2.1.

После выполнения задания данного пункта необходимо сделать выводы. Результаты выполнения представить в виде снимков экрана.

2.3 Исследовать работу ППА в «режиме 1», для чего выполнить следующее:

- настроить порт «*X*» на ввод в «режиме 1», порт «*Y*» – на вывод в «режиме 1»;
- установить байт данных на входе порта «*X*»;
- выполнить условия квитирования для ввода байта данных и считать байт;
- выполнить условия квитирования для вывода байта данных в порт

«Y» и осуществить вывод;

- повторить операции ввода-вывода с квитированием для n байт.

Количество байт выбирается согласно варианту по таблице 8.3.

Выбор портов осуществляется следующим образом:

- для четных вариантов в качестве порта «X» использовать порт A , а в качестве порта «Y» – порт B ;
- для нечетных вариантов – наоборот.

Данные для ввода студент выбирает самостоятельно. В разных вариантах данные не должны совпадать.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя.

При написании программы необходимо учитывать те же условия, что и в п. 2.1.

После выполнения задания данного пункта необходимо сделать выводы. Результаты выполнения представить в виде снимков экрана.

2.4 Исследовать работу ППА в «режиме 2», для чего выполнить следующее:

- настроить ППА в «режиме 2»;
- установить байт данных на входе порта A ;
- выполнить условия квитирования для ввода байта данных и считать байт;
- записать в память ВМ, начиная с адреса $XXXX$;
- выполнить заданную операцию над введенным байтом. Результат разместить в памяти ВМ сразу за введенным байтом;
- выполнить условия квитирования для вывода байта данных в порт A и осуществить вывод;
- повторить операции ввода-вывода с квитированием для n байт.

Исходные данные приведены в таблице 8.4.

Таблица 8.4 – Исходные данные для п. 2.4

Номер варианта	Кол-во байт	Смещение	Операция
1	8	20	И с числом 01h
2	10	22	ИЛИ с числом 88 h
3	12	24	Искл. ИЛИ с числом 18h
4	14	26	И с числом 55h
5	16	28	ИЛИ с числом A2h
6	18	30	Искл. ИЛИ с числом 63h
7	20	32	И с числом 87h
8	19	34	ИЛИ с числом F0h
9	17	36	Искл. ИЛИ с числом 0Fh
10	15	38	И с числом BBh
11	13	40	ИЛИ с числом 41h
12	11	42	Искл. ИЛИ с числом 14h
13	9	44	И с числом 3Ch
14	8	46	ИЛИ с числом 02h
15	15	48	Искл. ИЛИ с числом 20h
16	12	50	И с числом 04h
17	14	52	ИЛИ с числом 4Eh
18	13	54	Искл. ИЛИ с числом FFh
19	16	56	И с числом 60h
20	11	58	ИЛИ с числом 29h

Данные для ввода студент выбирает самостоятельно. В разных вариантах данные не должны совпадать.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется для

нахождения начального адреса расположения данных по формуле (8.1).

При написании программы необходимо учитывать те же условия, что и в п. 2.1.

После выполнения задания данного пункта необходимо сделать выводы. Результаты выполнения представить в виде снимков экрана.

3 Задание на обработку экспериментальных данных.

3.1 Привести текст программы на языке Ассемблер для задания п. 2.1.

Указать фрагменты, отвечающие за:

- инициализацию ППА в «режиме 0»;
- ввод и вывод через заданные порты;
- выполнение операции над введенными данными;
- вывод результата размером более 1 байта.

Управляющие слова должны быть выписаны отдельно и сопровождаться комментариями по каждому биту или группе.

3.2 Привести текст программы на языке Ассемблер для задания п. 2.2.

Указать фрагменты, отвечающие за:

- инициализацию портов ППА в требуемых режимах;
- ввод и вывод через заданные порты;
- выполнение операции над введенными данными.

Управляющие слова должны быть выписаны отдельно и сопровождаться комментариями по каждому биту или группе.

3.3 Привести текст программы на языке Ассемблер для задания п. 2.3.

Указать фрагменты, отвечающие за:

- инициализацию ППА в «режиме 1»;
- ввод с квитированием через заданный порт;
- вывод с квитированием через заданный порт.

Управляющие слова должны быть выписаны отдельно и сопровождаться комментариями по каждому биту или группе.

3.4 Привести текст программы на языке Ассемблер для задания п. 2.4.

Указать фрагменты, отвечающие за:

- инициализацию ППА в «режиме 2»;
- ввод с квитированием через порт *A*;
- выполнение операции над введенными данными;
- вывод с квитированием через порт *A*.

Управляющие слова должны быть выписаны отдельно и сопровождаться комментариями по каждому биту или группе.

Содержание отчета

Отчет по лабораторной работе выполняется на листах формат А4 согласно требований ГОСТ и ЕСКД и должен содержать следующие пункты:

- а) титульный лист с названием работы, дисциплины, кода группы, ФИО исполнителя и преподавателя;
- б) цель и содержание работы;
- в) условно-графическое обозначение и структурную схему ППА;
- г) тексты программ, выполненных по пунктам 2.1 – 2.4;
- д) результаты выполнения заданий по пунктам 2.1 – 2.4;
- е) выводы по результатам работы.

Контрольные вопросы

- 1 Каково назначение ППА *i8255*?
- 2 Какова структура ППА *i8255*? Рассказать о назначении каждого блока.
- 3 С помощью каких сигналов осуществляется управление ППА со стороны микропроцессора? Описать назначение каждого.
- 4 Какие режимы управления ППА использует микропроцессор? Указать необходимые сигналы управления для каждого случая.

5 Какого назначение управляющего слова *MS*? Рассмотреть формат слова на примерах.

6 Какого назначение управляющего слова *BSR*? Рассмотреть формат слова на примерах.

7 Какие режимы работы портов поддерживает ППА? Охарактеризовать каждый режим.

8 Что представляет собой обмен с квитированием? Привести примеры.

9 Какие сигналы используются в режиме квитирования? Какое устройство их формирует и в какой последовательности?

10 Привести примеры использования линий порта *C* в режиме квитирования.

11 Описать формат слова состояния *SW* ППА.

Список литературы, рекомендованный к использованию по данной теме

Основная учебная литература

1 Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд., испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

Дополнительная учебная литература

1 Шевкопляс Б.С. Микропроцессорные структуры. Инженерные решения: Справочник. - М.: Радио и связь, 1990 – 512 с.

2 Щелкунов, Н. Н. Микропроцессорные средства и системы / Н. Н. Щелкунов, А. П. Дианов. – М. : Радио и связь, 1989. – 288 с.

3 Коффон, Дж. Технические средства микропроцессорных систем: практический курс / Дж. Коффон. Пер. с англ. – М. : Мир, 1983. – 344 с.

4 Шахнов, В. А. Микропроцессоры и микропроцессорные комплекты

интегральных микросхем : справочник / В.-Б. Б. Абрайтис, Н. Н. Аверьянов, А. И. Белоус [и др.]: под ред. В. А. Шахнова. – Том 1. – М. : Радио и связь, 1988. – 368 с.

5 Соппа, И. В. Микропроцессоры и микроконтроллеры / И. В. Соппа, А. С. Самардак, В. А. Иванов. – Владивосток : Издательство Дальневосточного университета, 2009. – 211 с.

6 Алексенко, А. Г. Основы микросхемотехники / А. Г. Алексенко. 3-е изд. перераб. и доп. – М. : БИНОМ. Лаборатория знаний, 2009. – 448 с.

ЛАБОРАТОРНАЯ РАБОТА 9

Тема: Исследование работы средств последовательного ввода-вывода.

Цель работы: исследовать возможности настройки программируемого связного адаптера; исследовать работу устройства в асинхронном режиме ввода и вывода; исследовать работу устройства в синхронном режиме ввода и вывода.

Формируемые компетенции: ОК-6, ОК-7, ОПК-6, ОПК-8, ОПК-9, ПК-1, ПК-3.

Теоретическая часть

Параллельные каналы обычно используются при расстоянии до 10 или 15 м. При больших расстояниях стоимость многопроводного кабеля достаточно высока. В этих случаях применяются каналы с последовательной (поразрядной) передачей данных, что позволяет снизить стоимость линий, однако уменьшается пропускная способность канала. Функция стыка МС с последовательными каналами возлагается на связные адаптеры (СА).

К числу простейших интерфейсов последовательной передачи принадлежит интерфейс радиальный последовательный (ИРПС) или асинхронный старт-стоповый [2, 4]. Временные диаграммы протокола ИРПС представлены на рисунке 9.1. Приведенный вид посылки называется кадром.

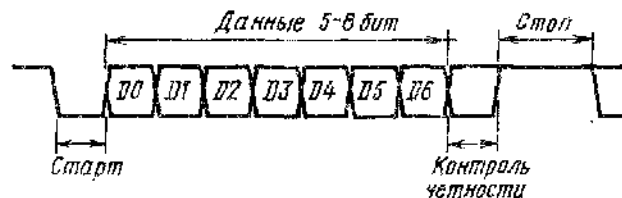


Рисунок 9.1 – Временные диаграммы протокола ИРПС

В простейшем случае функцию стыка последовательного канала с МС выполняют программируемые связные адаптеры (ПСА), в более сложных

случаях – программируемые связные контроллеры и сопроцессоры. К числу однокристальных ПСА относится БИС фирмы *Intel* – *i8251* (отечественный аналог КР580ВВ51).

Программируемый связной адаптер *i8251* или универсальный синхронно-асинхронный приемопередатчик (УСАПП, *USART* – *Universal Synchronous/Asynchronous Reciver/Transmitter*) представляет собой однокристальное программируемое устройство, реализующее интерфейс МС с синхронно-асинхронными каналами последовательной связи.

В состав ПСА (рисунок 9.2) входят передатчик, приемник, буфер шины данных и ряд схем управляющего типа. Основу передатчика составляет 13-разрядный сдвиговый регистр, хранящий очередное выходное слово. Приемник содержит два девятиразрядных сдвиговых регистра.

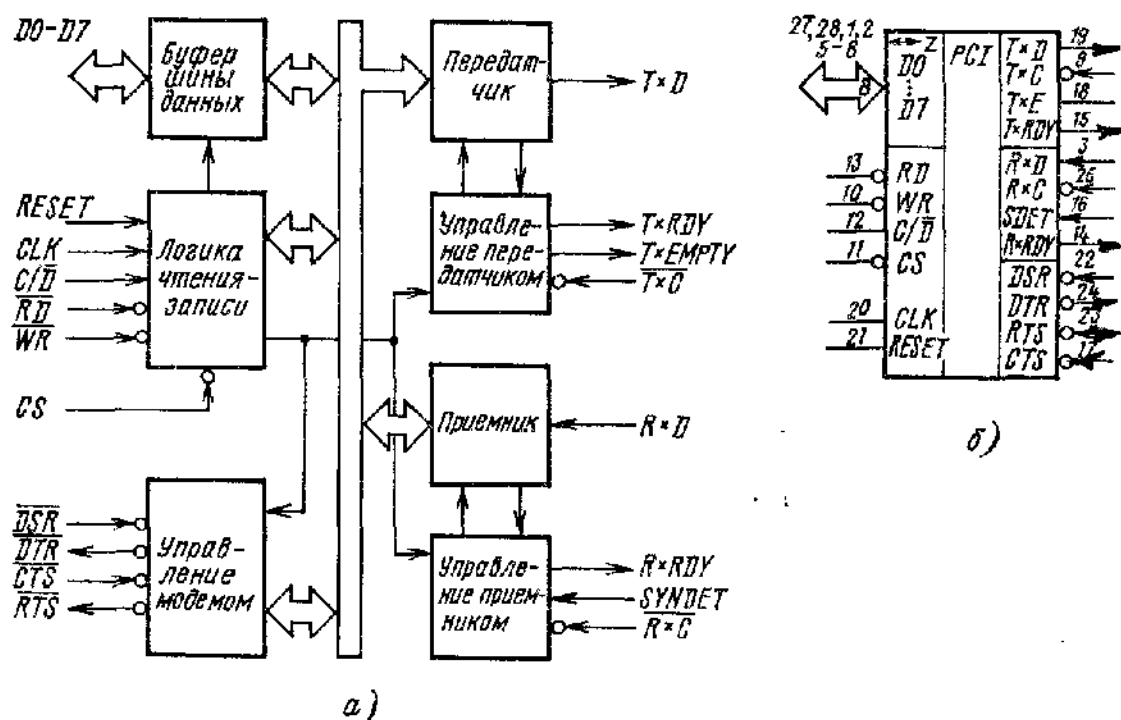


Рисунок 9.2 – Программируемый связной адаптер *i8251*:

а – структурная схема; б – условное графическое обозначение

Все операции по обмену информацией приведены в таблице 8.1. Они возможны только при низком уровне напряжения на линии выбора кристалла

$(\overline{CS} = 0)$.

Таблица 9.1 – Управляющие сигналы и операции *i8255*

$C / \overline{D}$	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	Операция
0	0	1	0	$D \leftarrow$ Приемник
0	1	0	0	Передатчик $\leftarrow D$
1	0	1	0	$D \leftarrow SW$
1	1	0	0	$CW \leftarrow D$
x	1	1	0	Нет операции
x	x	x	1	Нет операции

В таблице 9.1 приняты следующие обозначения:

- x – любое значение (лог. «1» или лог. «0»);
- D – шина данных.

Дальнейшее управление работой ПСА осуществляется двумя управляющими словами: MI (*Mode Instruction*) и CI (*Command Instruction*) в соответствии с таблицей 9.2 Слово MI определяет основной режим работы ПСА, оно должно быть передано после операции сброса первым.

Слово CI используется для оперативного управления работой ПСА: разрешения (запрещения) приема (передачи), перехода в режим ожидания синхронизации, а также для программной установки ПСА в исходное состояние с целью его переинициализации ($CIIR=1$). Внимание: для надежного сброса ПСА длительность сигнала $RESET$ должна быть больше шести периодов тактовой частоты CLK .

Формат слова MI приведен на рисунке 9.3, а слова CI – на рисунке 9.4.

Для организации программно-управляемого обмена по условию в составе ПСА предусмотрено слово состояния SW , формат которого представлен на рисунке 9.5.

Таблица 9.2 – Управление работой ПСА

Данные $D7, \dots, D0$	Управляющие сигналы				
	$C / \overline{D}$	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	$RESET$
Произвольно	x	x	x	0	1
<i>Mode Instruction</i>	1	0	1	0	0
Синхросимвол 1	1	0	1	0	0
Синхросимвол 2	1	0	1	0	0
<i>Command Instruction</i>	1	0	1	0	0
Данные, <i>Status Word</i>	0	x	x	0	0
<i>Command Instruction</i>	1	0	1	0	0
Данные, <i>Status Word</i>	0	x	x	0	0

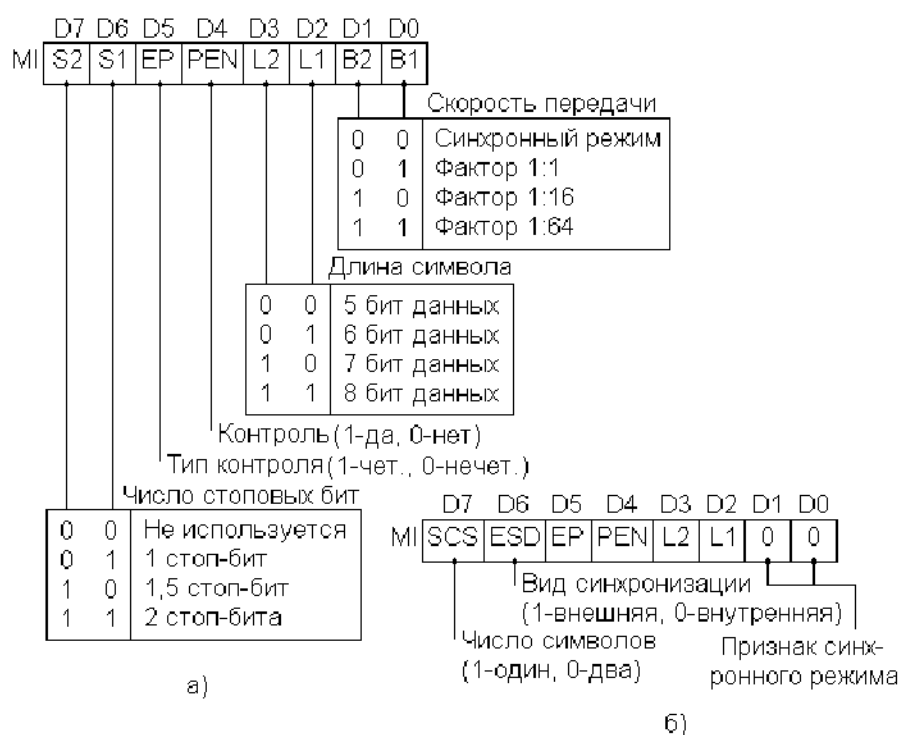


Рисунок 9.3 – Формат слова выбора режима ПСА:

а – асинхронный режим; б – синхронный режим

Режим работы ПСА определяется программными средствами. После записи *CI* адаптер может быть переведен в один из основных режимов

работы или в их допустимую комбинацию:

- а) асинхронная передача;
- б) асинхронный прием;
- в) синхронная передача;
- г) синхронный прием:
 - 1) с внутренней синхронизацией;
 - 2) с внешней синхронизацией.

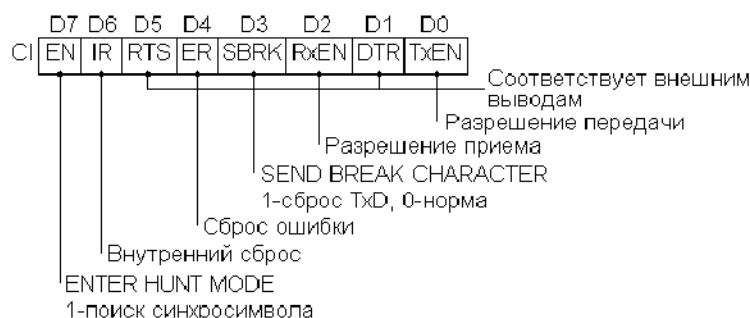


Рисунок 9.4 – Формат команды ПСА



Рисунок 9.5 – Формат слова состояния ПСА

Более подробно ознакомиться со структурой, режимами работы и способами программирования исследуемой БИС можно в основной [1] и дополнительной [1 - 5] литературе и по материалам курса лекций дисциплины «Основы микропроцессорной техники».

Оборудование и материалы

В ходе выполнения лабораторной работы будет использоваться эмулятор «Виртуальная МикроЭВМ *Intel® 8080*» версии 1.0 (далее ВМ). Эмулятор находится в папке *PyREStd/ VDM8080*. более подробное описание

приведено в лабораторной работе 8.

Во вкладке «Структура “Виртуальной МикроЭВМ”» необходимо ознакомиться с содержанием следующих пунктов:

- карта адресов устройств;
- таблица включения модулей *i8080*, *i8228*, *i8251*.

Указания по технике безопасности

Перед проведением лабораторной работы следует изучить инструкцию по технике безопасности при работе на компьютере. Также следует соблюдать правила и требования, установленные для данного компьютерного класса.

Задания

1 Задание на подготовку к выполнению экспериментальной части работы.

1.1 Изучите структурную схему программируемого связного адаптера *i8251*.

1.2 Изучите особенности организации передатчика и приемника ПСА.

1.3 Изучите временные диаграммы протокола последовательной передачи и формат посылки.

1.4 Изучите режимы работы ПСА и сигналы управления. Ознакомьтесь с временными диаграммами режимов работы.

1.5 Изучите форматы управляющих слов *MI* и *CI*, способы программирования ПСА.

1.6 Изучите формат слова состояния *SW* и назначение каждого бита.

2 Задание на выполнение экспериментальной части работы.

2.1 Исследовать работу ПСА в режиме асинхронного приема, для чего выполнить следующее:

- настроить адаптер на ввод в асинхронном режиме с фактором «1:1»;
- считать из внешнего устройства n слов данных и записать в память ВМ, начиная с адреса XXXX (понятие «слова» в данном контексте необходимо понимать как единицу информации заданного размера);
- выполнить заданную операцию над введенными данными. Результат разместить в памяти ВМ через два адреса после данных, а также в регистрах микропроцессора.

Исходные данные приведены в таблице 9.3.

Таблица 9.3 – Исходные данные для п. 2.1

Номер варианта	Кол- во слов	Смещение	Операция	Длина слова, бит	Контр. четности/ тип	Кол-во стоп- битов
1	2	3	4	5	6	7
1	10	20	ИЛИ	8	+/0	2
2	12	22	И	8	+/1	1
3	14	24	ИЛИ-НЕ	8	+/0	0
4	16	26	И-НЕ	7	+/1	2
5	18	28	Искл. ИЛИ	7	+/0	1
6	20	30	Искл. ИЛИ-НЕ	7	+/1	0
7	22	32	Арифм. сложение	6	+/0	2
8	24	34	Арифм. вычитание	6	+/1	1
9	26	36	ИЛИ	6	нет	0
10	28	38	Арифм. вычитание	5	нет	2
11	27	40	Арифм. сложение	5	нет	1
12	25	42	И	5	нет	0

Продолжение таблицы 9.3

1	2	3	4	5	6	7
13	29	44	ИЛИ-НЕ	5	нет	2
14	23	46	И-НЕ	6	нет	1
15	21	48	Искл. ИЛИ	7	нет	0
16	19	50	Искл. ИЛИ-НЕ	8	+/0	2
17	17	52	Искл. ИЛИ-НЕ	7	+/1	1
18	15	54	Арифм. сложение	8	+/0	0
19	13	56	ИЛИ-НЕ	7	+/1	2
20	11	58	Арифм. вычитание	8	+/0	1

В таблице 9.3 приняты следующие обозначения:

- «+» – выполнять контроль четности;
- «0» – контроль нечетности;
- «1» – контроль четности;
- «нет» – не выполнять контроль четности.

Данные для ввода студент выбирает самостоятельно. В разных вариантах данные не должны совпадать.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется для нахождения начального адреса расположения данных:

$$AddData = AddEndOf Prog + SS, \quad (9.1)$$

где $AddData$ – начальный адрес для размещения данных;

$AddEndOf Prog$ – адрес окончания программы;

SS – смещение.

При написании программы необходимо учитывать следующее:

- доступная пользователю память ВМ начинается с нулевого адреса;
- при компиляции программы в память и файл необходимо задать

начальный адрес загрузки, который зависит от смещения и составляет удвоенное его значение;

- для организации стека в ВМ выделен определенный диапазон памяти. Размещать программу и данные необходимо так, чтобы не допустить пересечения этих областей.

После выполнения задания данного пункта необходимо сделать выводы. Результаты выполнения представить в виде снимков экрана.

2.2 Исследовать работу ПСА в режиме асинхронной передачи, для чего выполнить следующее:

- настроить адаптер на вывод в асинхронном режиме с фактором «1:1»;
- передать во внешнее устройство n слов данных, записанных в п. 2.1 в память ВМ;
- передать во внешнее устройство результат выполнения операции п. 2.1 из памяти ВМ.

При написании программы необходимо учитывать те же условия, что и в п. 2.1.

После выполнения задания данного пункта необходимо сделать выводы. Результаты выполнения представить в виде снимков экрана.

2.3 Исследовать работу ПСА в режиме синхронного приема, для чего выполнить следующее:

- настроить адаптер на ввод в синхронном режиме с заданным типом синхронизации;
- считать из внешнего устройства n слов данных и записать в память ВМ, начиная с адреса XXXX (понятие «слова» в данном контексте необходимо понимать как единицу информации заданного размера);
- выполнить действия, необходимые для моделирования заданного типа синхронизации.

Исходные данные приведены в таблице 9.4.

Таблица 9.4 – Исходные данные для п. 2.3

Номер варианта	Кол- во слов	Смещение	Длина слова, бит	Контр. четности/ тип	Тип синхро- низации	Кол-во с/символов
1	2	3	5	6	7	8
1	22	20	5	нет	Внешн.	1
2	20	22	5	нет	Внутр.	2
3	18	24	6	нет	Внешн.	1
4	16	26	6	нет	Внутр.	2
5	14	28	7	нет	Внешн.	1
6	12	30	7	нет	Внутр.	2
7	10	32	8	нет	Внешн.	1
8	8	34	8	+/0	Внутр.	2
9	9	36	8	+/1	Внешн.	1
10	11	38	8	+/0	Внутр.	2
11	13	40	7	+/1	Внешн.	1
12	15	42	7	+/0	Внутр.	2
13	17	44	6	+/1	Внешн.	1
14	19	46	6	+/0	Внутр.	2
15	21	48	5	+/1	Внешн.	1
16	23	50	6	нет	Внутр.	2
17	22	52	7	нет	Внешн.	1
18	24	54	7	нет	Внутр.	2
19	25	56	6	нет	Внешн.	1
20	26	58	5	нет	Внутр.	2

Обозначения в таблице 9.4 соответствуют таблице 9.3.

Данные для ввода и формат синхросимвола (синхросимволов) студент

выбирает самостоятельно. В разных вариантах они не должны совпадать.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде и используется для нахождения начального адреса расположения данных по формуле (9.1).

При написании программы необходимо учитывать те же условия, что и в п. 2.1.

После выполнения задания данного пункта необходимо сделать выводы. Результаты выполнения представить в виде снимков экрана.

2.4 Исследовать работу ПСА в режиме синхронной передачи, для чего выполнить следующее:

- настроить адаптер на вывод в синхронном режиме с заданным количеством синхроимпульсов;
- передать во внешнее устройство n слов данных, записанных в п. 2.1 в память ВМ;
- выполнить действия, необходимые для моделирования синхронизации, причем в таком количестве, чтобы отобразить заданное количество синхросимволов и формат каждого.

Формат синхросимвола (синхросимволов) студент выбирает самостоятельно. В разных вариантах они не должны совпадать.

При написании программы необходимо учитывать те же условия, что и в п. 2.1.

После выполнения задания данного пункта необходимо сделать выводы. Результаты выполнения представить в виде снимков экрана.

2.5 Исследовать работу ПСА в полудуплексном режиме, для чего выполнить следующее:

- настроить адаптер для работы в полудуплексном режиме;
- считать из внешнего устройства два слова данных и записать в память ВМ, начиная с адреса XXXX;
- передать во внешнее устройство два слова данных, записанные ранее

в память ВМ;

- повторить операции ввода-вывода три раза.

В данном пункте необходимо написать программы процедуры полудуплексного обмена для различных режимов работы ПСА, указанных в теоретическом обосновании данной лабораторной работы (для которых это возможно). Исследовать в эмуляторе только одну из процедур.

Данные для ввода необходимо использовать те же, что и в пунктах предыдущих заданий. В разных вариантах данные не должны совпадать.

Номер варианта соответствует порядковому номеру студента в журнале преподавателя. Смещение задано в десятичном виде (таблица 9.3) и используется для нахождения начального адреса расположения данных по формуле (9.1).

При написании программы необходимо учитывать те же условия, что и в п. 2.1.

После выполнения задания данного пункта необходимо сделать выводы. Результаты выполнения представить в виде снимков экрана.

3 Задание на обработку экспериментальных данных.

3.1 Привести текст программы на языке Ассемблер для задания п. 2.1.

Указать фрагменты, отвечающие за:

- инициализацию ПСА в режиме асинхронного приема;
- ввод данных;
- выполнение операции над введенными данными.

Использованные управляющие слова (слово состояния) должны быть выписаны отдельно и сопровождаться комментариями по каждому биту или группе.

3.2 Привести текст программы на языке Ассемблер для задания п. 2.2.

Указать фрагменты, отвечающие за:

- инициализацию ПСА в режиме асинхронной передачи;
- вывод данных.

Использованные управляющие слова (слово состояния) должны быть выписаны отдельно и сопровождаться комментариями по каждому биту или группе.

3.3 Привести текст программы на языке Ассемблер для задания п. 2.3.

Указать фрагменты, отвечающие за:

- инициализацию ПСА в режиме синхронного приема;
- ввод данных;
- условия выполнения заданного типа синхронизации.

Использованные управляющие слова (слово состояния) должны быть выписаны отдельно и сопровождаться комментариями по каждому биту или группе.

3.4 Привести текст программы на языке Ассемблер для задания п. 2.4.

Указать фрагменты, отвечающие за:

- инициализацию ПСА в режиме синхронной передачи;
- вывод данных;
- условия выполнения заданного типа синхронизации.

Использованные управляющие слова (слово состояния) должны быть выписаны отдельно и сопровождаться комментариями по каждому биту или группе.

3.5 Привести текст программы на языке Ассемблер для задания п. 2.5.

Указать фрагменты, отвечающие за:

- инициализацию ПСА в полудуплексном режиме;
- ввод данных;
- вывод данных.

Использованные управляющие слова (слово состояния) должны быть выписаны отдельно и сопровождаться комментариями по каждому биту или группе.

Содержание отчета

Отчет по лабораторной работе выполняется на листах формат А4 согласно требований ГОСТ и ЕСКД и должен содержать следующие пункты:

- а) титульный лист с названием работы, дисциплины, кода группы, ФИО исполнителя и преподавателя;
- б) цель и содержание работы;
- в) условно-графическое обозначение и структурную схему ПСА;
- г) тексты программ, выполненных по пунктам 2.1 – 2.5;
- д) результаты выполнения заданий по пунктам 2.1 – 2.5;
- е) выводы по результатам работы.

Контрольные вопросы

- 1 Каково назначение ПСА *i8251*?
- 2 Какова структура ППА *i8251*? Рассказать о назначении каждого блока.
- 3 С помощью каких сигналов осуществляется управление ПСА со стороны микропроцессора? Описать назначение каждого.
- 4 Какие режимы управления ПСА использует микропроцессор? Указать необходимые сигналы управления для каждого случая.
- 5 Какого назначения управляющего слова *MI*? Рассмотреть формат слова на примерах.
- 6 Какого назначения управляющего слова *CI*? Рассмотреть формат слова на примерах.
- 7 Какие режимы работы поддерживает ППА? Охарактеризовать каждый режим.
- 8 Что представляет собой асинхронный режим? Какие существуют разновидности? Привести примеры.

9 Что представляет собой синхронный режим? Какие существуют разновидности? Привести примеры.

10 Что представляет собой полудуплексный режим?

11 Возможен ли полудуплексный режим при асинхронном приеме-передаче? Если да, то описать протокол обмена.

12 Возможен ли полудуплексный режим при синхронном приеме-передаче? Если да, то описать протокол обмена.

13 Описать формат слова состояния *SW* ПСА.

Список литературы, рекомендованный к использованию по данной теме

Основная учебная литература

1 Новиков, Ю. В. (канд. техн. наук). Основы микропроцессорной техники : учебное пособие / Ю. В. Новиков, П. К. Скоробогатов. - 4-е изд., испр. - Москва : Интернет-университет информационных технологий, 2009. - 357 с. : ил., табл. ; 22.

Дополнительная учебная литература

1 Шевкопляс Б.С. Микропроцессорные структуры. Инженерные решения: Справочник. - М.: Радио и связь, 1990 – 512 с.

2 Щелкунов, Н. Н. Микропроцессорные средства и системы / Н. Н. Щелкунов, А. П. Дианов. – М. : Радио и связь, 1989. – 288 с.

3 Коффрон, Дж. Технические средства микропроцессорных систем: практический курс / Дж. Коффрон. Пер. с англ. – М. : Мир, 1983. – 344 с.

4 Шахнов, В. А. Микропроцессоры и микропроцессорные комплекты интегральных микросхем : справочник / В.-Б. Б. Абрайтис, Н. Н. Аверьянов, А. И. Белоус [и др.]: под ред. В. А. Шахнова. – Том 1. – М. : Радио и связь, 1988. – 368 с.

5 Соппа, И. В. Микропроцессоры и микроконтроллеры / И. В. Соппа,

А. С. Самардак, В. А. Иванов. – Владивосток : Издательство
Дальневосточного университета, 2009. – 211 с.

6 Алексенко, А. Г. Основы микросхемотехники / А. Г. Алексенко. 3-е
изд. перераб. и доп. – М. : БИНОМ. Лаборатория знаний, 2009. – 448 с.

ОСНОВЫ МИКРОПРОЦЕССОРНОЙ ТЕХНИКИ

Методические указания по выполнению лабораторных работ

по направлению подготовки

11.03.04 Электроника и нанoeлектроника

Квалификация выпускника

Бакалавр

Составитель: Лапин А.А.

Редактор:

Подписано в печать _____			
Формат _____	Усл. п. л. – _____	Уч.-изд. л. – _____.	
Бумага _____.	Печать _____.	Заказ _____	Тираж _____ экз.
ФГАОУ ВО «Северо-Кавказский федеральный университет»			
355029, г. Ставрополь, пр. Кулакова, 2			

Издательство Северо-Кавказского федерального университета
Отпечатано в типографии СКФУ