МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Тульский государственный университет»

Институт *высокоточных систем им. В.П. Грязева*
Кафедра «Приборы управления»

Утверждено на заседании кафедры
«Приборы управления»
«22» января 2024 г., протокол №1

Заведующий кафедрой

_____ В.В. Матвеев

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
по проведению лабораторных занятий
по дисциплине (модулю)
«Микропроцессоры в оптотехнике»

основной профессиональной образовательной программы
высшего образования – программы бакалавриата

по направлению подготовки
12.03.03 Фотоника и оптоинформатика

с направленностью (профилем)
Интеллектуальные фотонные системы
Форма обучения: очная

Идентификационный номер образовательной программы: 120303-01-24

Тула 2024 год

Разработчик методических указаний:

Алалуев В.В., доц. каф., к.т.н.,_____ 

Содержание

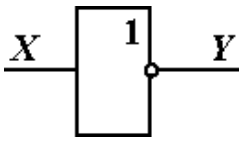
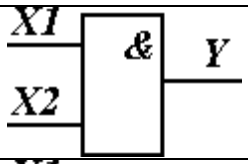
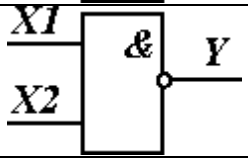
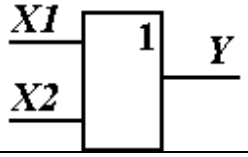
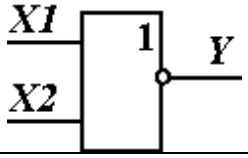
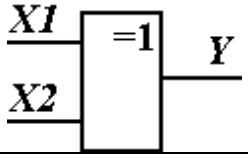
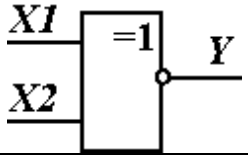
Лабораторная работа №1. Исследование базовых логических элементов

Цель работы: Изучение базовых логических элементов и простейших схем на их основе

Основные теоретические сведения

В основе всех цифровых схем лежат базовые логические элементы. В таблице 1 приведены их основные параметры

Таблица 1. Базовые логические элементы

Логическая функция	Условное графическое обозначение	Булево выражение	Таблица истинности		
			Входы		Выход
			X		*Y*
НЕ		$Y = \overline{X}$	0		1
			1		0
			X1	*X2*	*Y*
И		$Y = X1 \bullet X2$	0	0	0
			0	1	0
			1	0	0
			1	1	1
И-НЕ		$Y = \overline{X1 \bullet X2}$	0	0	1
			0	1	1
			1	0	1
			1	1	0
ИЛИ		$Y = X1 + X2$	0	0	0
			0	1	1
			1	0	1
			1	1	1
ИЛИ-НЕ		$Y = \overline{X1 + X2}$	0	0	1
			0	1	0
			1	0	0
			1	1	0
Исключающее ИЛИ		$Y = X1 \oplus X2$	0	0	0
			0	1	1
			1	0	1
			1	1	0
Исключающее ИЛИ-НЕ		$Y = \overline{X1 \oplus X2}$	0	0	1
			0	1	0
			1	0	0
			1	1	1

Порядок выполнения работы:

1. Зарисовать схемы на карточках I-1, I-9
2. Составить таблицу истинности по каждой карточке включив в нее все задействованные в карточке входы и выходы схемы. Заполнить ее при помощи стенда ОАВТ

(При этом положение переключателя SA1-4 вверх соответствует подаче логической единцы, а вниз – логическому нулю, если светодиод горит это означает логическую единицу на выходе схемы если светодиод не горит, то на выходе схемы логический ноль)

3. По таблице истинности определить тип каждого логического элемента изображенного на карточке λ (например, на карточке I-2 таких элементов два).

Содержание отчета:
1. Название и цель работы.
2. Исследуемые схемы
3. Таблицы истинности
4. Выводы по работе

Контрольные вопросы:

1. Описать базовые логические элементы привести их таблицы истинности.
2. По приведенному Булевскому выражению составить электрическую принципиальную схему.

У=(X1&X2)⊕(X1&X3)	Y=X1+X2+X3&X1
Y=X1&X2&X3+(X1⊕X2)	Y=X1+X2+(X3&X1)
Y=((X1+X2)⊕X3)&X1	Y=(X1+X2)&(X1⊕X2)

3. По заданной таблице истинности составить принципиальную схему

X1	X2	X3	Y1	Y2	Y3	Y4	Y5	Y6
0	0	0	1	0	1	0	1	0
0	0	1	1	0	1	0	1	0
0	1	0	1	0	1	0	0	0
0	1	1	1	1	0	0	1	1
1	0	0	0	0	1	1	1	0
1	0	1	1	0	1	0	1	0
1	1	0	0	0	1	1	1	0
1	1	1	1	1	0	0	0	1

5. Дайте определение мультиплексора. Область применения мультиплексоров.
6. Дайте определение демультиплексора. Область применения демультиплексоров.
7. Приведите электрическую принципиальную схему мультиплексора на 2 информационных канала. Объясните принцип работы.
8. Приведите электрическую принципиальную схему демультиплексора на 2 информационных канала. Объясните принцип работы.
9. Приведите электрическую принципиальную схему мультиплексора на 4 информационных канала. Объясните принцип работы.
10. Приведите электрическую принципиальную схему демультиплексора на 4 информационных канала. Объясните принцип работы.

Лабораторная работа №2. Исследование регистров и триггеров

Цель работы: Изучить работу триггеров и простейших схем на их основе

Основные теоретические сведения

***RS* - триггер**

Условное графическое изображение ***RS*** - триггера показано на рис. 1. ***RS***-триггер имеет два входа S установки 1 (SET) и R установки 0 (RESET) и два выхода Q и $\overline{Q}$ (прямой и инверсный). В триггерах выходы всегда находятся в противоположных (комплементарных) состояниях. Другими словами, если на выходе $\overline{Q}$ мы имеем уровень логической 1 ($\overline{Q} = 1$), то на выходе Q будет уровень логического 0 ($Q = 0$), и наоборот.

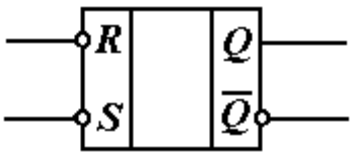


Рис. 1. Условное графическое обозначение RS-триггера.

Рассмотрим работу триггера на основе логических элементов И-Не

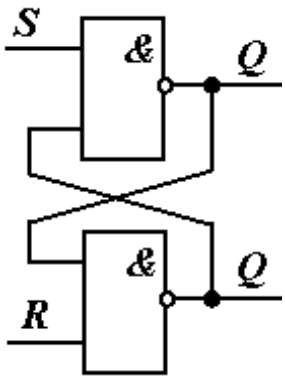


Рис. 2. Схема RS-триггера на основе элементов И-НЕ.

Принцип работы RS-триггера иллюстрирует его таблица истинности (табл.1).

Таблица 1. Таблица истинности RS-триггера.

Режим работы	Входы		Выходы		
	S	R	Q	Q	Влияние на выход Q
Запрещенное состояние	0	0	1	1	Запрещено - не используется
Установка 1	0	1	1	0	Для установки Q в 1
Установка 0	1	0	0	1	Для установки Q в 0
Хранение	1	1	Б/И	Б/И	Зависит от предыдущего состояния

При подаче на оба входа триггера уровня логического 0 ($S = R = 0$) на обоих выходах устанавливается уровень логической 1 ($Q = \overline{Q} = 1$). Это запрещенное состояние триггера; оно не используется. Согласно второй строке таблицы истинности, при $S = 0$ и $R = 1$ на выходе Q триггера устанавливается уровень логической 1. В этом случае говорят, что триггер установлен в состоянии 1. Согласно третьей строке, при $R = 0$ и $S = 1$ происходит сброс сигнала на выходе Q (очистка выхода Q) к уровню логического 0. Это значит, что триггер установлен в

состояние 0. Четвертая строка таблицы истинности соответствует $S = R = 1$. В этом случае триггер находится в состоянии покоя: на выходах Q и Q сохраняются прежние комплементарные уровни сигнала. Это режим хранения.

Из таблицы 1 видно, что установку триггера в состояние 1 (установку 1 на выходе Q) инициирует логический 0 на входе S. Точно также установку триггера в состояние 0 (установку 0 на выходе Q) инициирует логический 0 на входе R. Обратите внимание на инвертирующие кружки у входов S и R. Они показывают, что активным уровнем сигнала для установки триггера в состояние 1 и 0 является уровень логического 0 на одном из входов.

***D*-триггер**

Условное графическое обозначение ***D***-триггера показано на рис. 3. У этого триггера имеется только один информационный вход D, а также синхронизирующий вход C. Выходы по-прежнему обозначаются Q и $\overline{Q}$. ***D***-триггер часто называют триггером с задержкой. Слово "задержка" здесь характеризует то, что происходит с данными (информацией), поступающими на вход D. Информационный сигнал (0 или 1), поступающий на этот вход, задерживается в триггере ровно на один такт, прежде чем появляется на выходе Q. Таблица истинности для ***D***-триггера приведена в таблице 2.

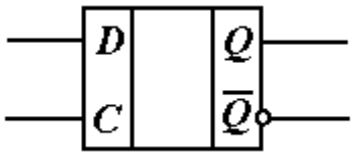


Рис. 3. Условно графическое обозначение *D*-триггера

Таблица 2. Таблица истинности *D* -триггера

№ тактового импульса	D	C	Q	$\overline{Q}$	Комментарий
0	0	---	0	1	Подаем 0 на вход данных
0	1	---	0	1	Подаем 1 на вход данных, при этом данные на выходе Q не изменяются
1	1	⎍	1	0	Подаем 1 тактовый импульс, при этом данные на выходе изменяются (запись данных)
1	0	---	1	0	убираем 1 со входа (на выходе Q она остается)
2	0	⎍	0	1	Подаем 2 тактовый импульс, при этом данные на выходе Q изменяются (запись данных)
2	0	---	0	1	

Обратите внимание, что сигнал на выходе Q в после подачи тактового импульса повторяет сигнал, который был на входе D до подачи тактового импульса.

Чаще всего придется использовать ***D***-триггеры, выполненные в виде монолитных ИС. На рис. 3 показано условное графическое обозначение типичного серийно выпускаемого интегрального ***D***-триггера.

D триггер может иметь два дополнительных входа - предварительной установки (***PS***) и очистки (***CLR***). Логический 0 на входе ***PS*** инициирует установку логической 1 на выходе Q. Логический 0 на входе ***CLR*** инициирует очистку выхода Q (установку логического 0 на выходе Q). В активных состояниях входы ***PS*** и ***CLR*** блокируют действие входов D и C, при разблокировании входы D и C действуют точно так же, как в обычном ***D***-триггере.

***JK*-триггер.**

JK-триггер - это наиболее широко используемый универсальный триггер, обладающий характеристиками всех других типов триггеров. Условное графическое обозначение ***JK***-триггера показано на рис. 4. ***JK***-триггер имеет два информационных входа J и K, синхронизирующий вход C и, как и все триггеры, два комплементарных входа Q и $\overline{Q}$. В таблице 3 приведена таблица истинности для ***JK***-триггера. Когда на оба входа J и K подается уровень логического 0, триггер блокируется, и состояния его выходов не изменяются. В этом случае триггер находится в режиме хранения.

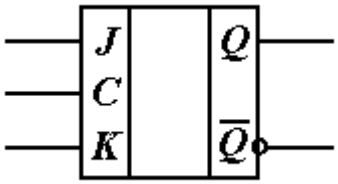


Рис.4. Условно-графическое обозначение *JK*-триггера

Таблица 3. Таблица истинности *JK*-триггера

Режим работы	Входы			Выходы		
	C	J	K	Q	Q	Влияние на выход Q
Хранение	⎍	0	0	Б/И	Б/И	Без изменений - блокировка
Установка 0	⎍	0	1	0	1	Сброс или очистка в состоянии 0
Установка 1	⎍	1	0	1	0	Установка в состояние 1
Переключение	⎍	1	1	Переклю-чаются		Изменение состояния на противоположное

Строки 2 и 3 таблицы истинности описывают режимы, соответствующие установке триггера в состояние 0 и 1. Строка 4 иллюстрирует очень важный переключательный режим работы ***JK***-триггера. Если на обоих входах J и K установлен уровень логической 1, то следующие друг за другом тактовые импульсы будут вызывать перебросы уровней сигналов на выходах триггера от 1 к 0, от 0 к 1,снова от 1 к 0 и т. д. Такая работа триггера напоминает последовательно производимые переключения тумблера, откуда и происходит название режима.

Порядок выполнения работы:

1. Зарисовать схемы на карточках II-1, II-7
2. Продумать и составить таблицу истинности по каждой карточке и заполнить ее при помощи стенда ОАВТ
3. По таблице истинности определить режимы работы триггеров и вписать их в таблицу истинности (они могут не совпадать с режимами работы, приведенными в данных методических указаниях).

Содержание отчета:
1. Название и цель работы.
2. Исследуемые схемы
3. Таблицы истинности
4. Выводы по работе.

Контрольные вопросы:
1. Область применения триггеров. Виды триггеров.
2. ***RS*** триггер – принцип работы, УГО, таблица истинности
3. ***D*** триггер – принцип работы, УГО, таблица истинности
4. ***JK*** триггер – принцип работы, УГО, таблица истинности

5. Принцип работы, назначение и область применения счетчиков
6. Трехразрядный асинхронный счетчик со сквозным переносом. Принципиальная схема на триггерах. Таблица истинности. Принцип работы.
7. Трехразрядный синхронный счетчик со сквозным переносом. Принципиальная схема на триггерах. Таблица истинности. Принцип работы.
8. Трехразрядный асинхронный счетчик со переносом по модулю 9. Принципиальная схема на триггерах. Таблица истинности. Принцип работы.

Лабораторная работа №3. Исследование арифметических устройств

Цель работы: Изучение регистров и простейших арифметических схем.

Основные теоретические сведения

Регистры - цифровые устройства, предназначенные для хранения двоичной информации. Различают последовательные и параллельные регистры.

Последовательный регистр.

Рассмотрим последовательный 4-разрядный регистр сдвига на основе ***D*** триггеров Рис.1.

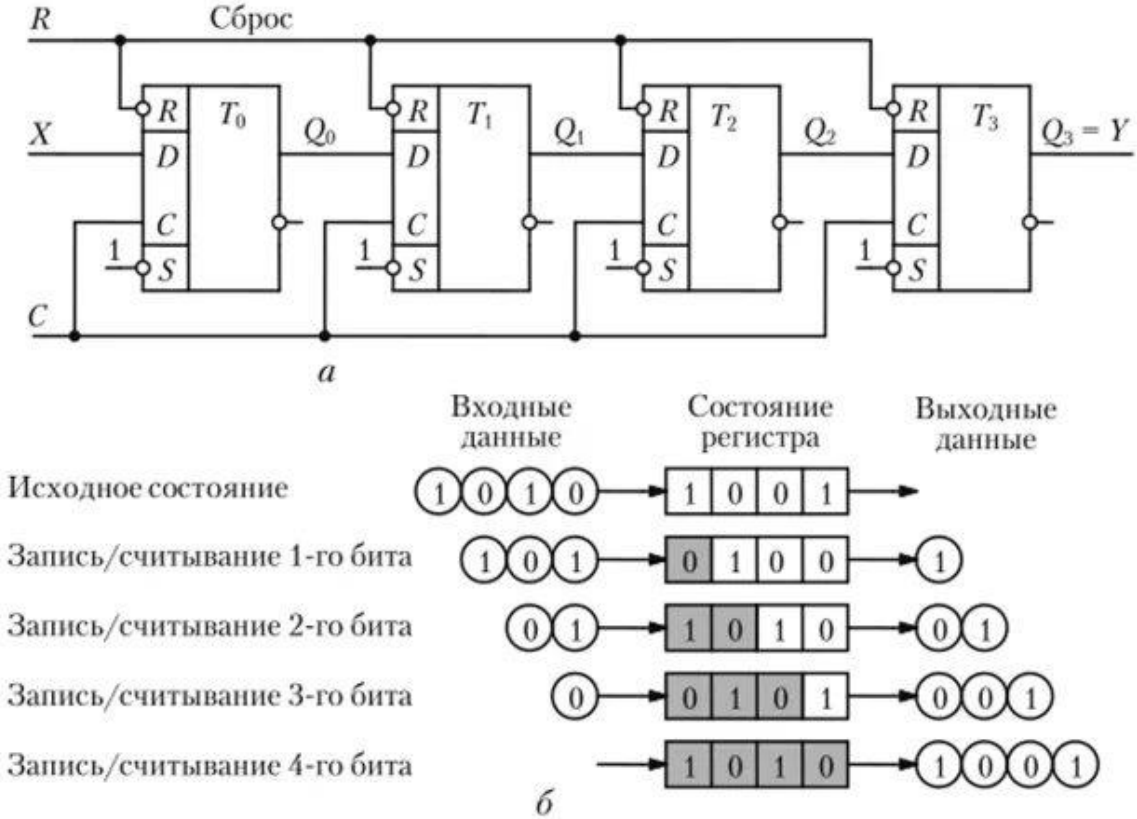


Рис1 . 4 –разрядный последовательный регистр сдвига.

Позволяет хранить 4 бита информации (по одному в каждом триггере). Загрузка регистра осуществляется последовательной подачей битов информации на вход **данных**. Каждый бит информации загружается при помощи тактового импульса **такт.** Вход **сброс** предназначен для установки всех триггеров регистра в 0.

При подаче такта данные с входа данных передаются в первый регистр (на выход d). Данные из первого регистра передаются во второй (d>>c), из второго в третий (c>>b) и из третьего в четвертый (b>>a). Передачу данных иллюстрирует таблица 1.

Таблица .1. Загрузка числа 0011 в последовательный регистр сдвига

Данные	Такт	сброс	d	c	b	a
0	0	1	0	0	0	0
1	0	0	0	0	0	0
1	⎍	0	1	0	0	0
1	⎍	0	1	1	0	0
0	⎍	0	0	1	1	0
0	⎍	0	0	0	1	1

Параллельный регистр.

В отличие от последовательного регистра в параллельном загрузка данных в регистр осуществляется параллельно (одновременно во все триггеры).

Рассмотрим 4-разрядный кольцевой регистр сдвига с параллельной загрузкой данных на основе ***JK*** триггеров Рис.2.

Данный регистр имеет 4 входа (предварительной установки) данных d_i, c_i, b_i, a_i, и 4 выхода d_0, c_0, b_0, a_0, вход сброса и вход тактовых импульсов. Данные в регистр можно записать параллельно при помощи подачи сигналов на входы d_i, c_i, b_i, a_i Это иллюстрирует строка 2 таблицы 2, а в дальнейшем при помощи подачи синхроимпульсов на вход **такт** происходит сдвиг регистра.

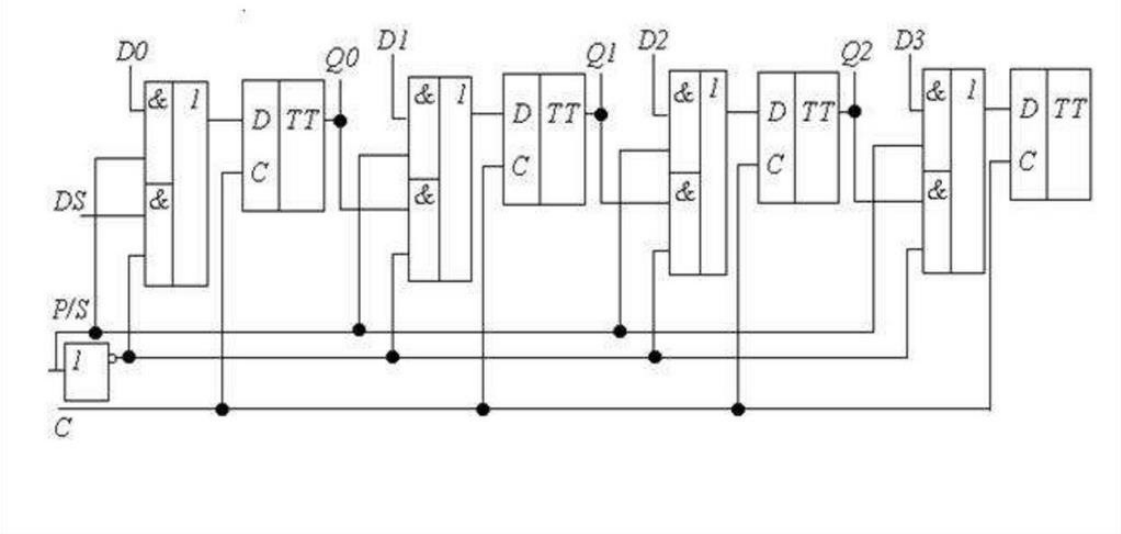


Рис.2. 4-Разрядный параллельный регистр сдвига

Работу регистра иллюстрирует таблица истинности.

Таблица 2. Работа параллельного регистра сдвига

Такт	сброс	d_i	c_i	b_i	a_i	d_0	c_0	b_0	a_0
0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	1	1	0
0	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1
2	0	0	0	0	0	1	0	0	1
3	0	0	0	0	0	1	1	0	0

Мы рассмотрели типичные примеры последовательного и параллельного регистров. В настоящее время промышленностью выпускается большое количество разнообразных регистров сдвига. В лабораторной работе Вам необходимо изучить работу регистров и составить их таблицы истинности. (Примечание: Работа регистров на карточках не соответствует приведенной в методичке)

Арифметические устройства выполняющие операцию сложения
Полусумматор

Полусумматор реализует сложение 2х бит информации $A+B=\sum+C_0$

Обозначим: A - 1 слагаемое, B - 2 слагаемое, $\sum$ - сумма, C_0 - перенос в старший разряд. Рассмотрим правила сложения двух одноразрыдных чисел A и B Рис.3..

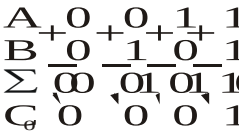


Рис.3. Правила сложения одноразрядных чисел

Электрическая принципиальная схема полусумматора реализующего эти правила приведена на рис.4.

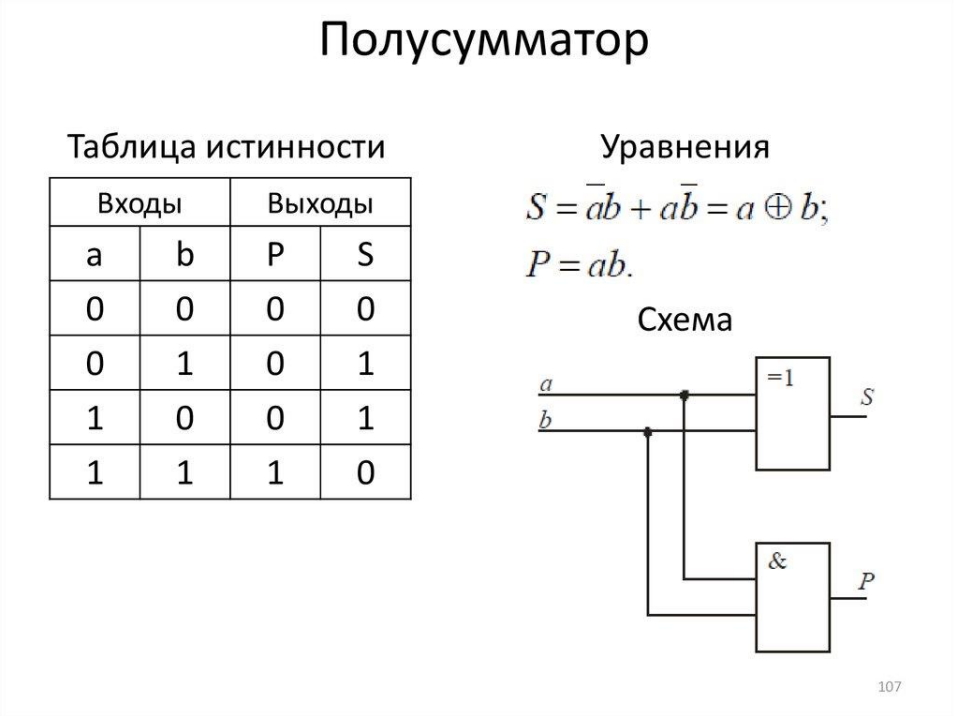


Рис.4. Полусумматор

Полный сумматор

Полный Сумматор – устройство для сложения двух бит информации с учетом возможного переноса из младшего разряда, то есть $A + B + C_{in} = \sum + C_0$ где C_{in} - перенос из младшего разряда

Электрическая принципиальная схема полного сумматора приведена на рис.5. и таблица 3 иллюстрирует принцип работы

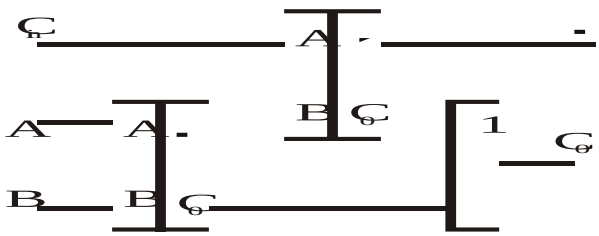


Рис. 5. Полный Сумматор

Таблица 3.Таблица истинности полного сумматора

C_{in}	A	B	∑	C_0
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Многоразрядный сумматор

Многоразрядный сумматор предназначен для сложения многоразрядных двоичных чисел. Рассмотрим трехразрядный сумматор. Он предназначен для сложения 3х разрядных двоичных чисел и состоит из 1 полусумматора и 2 полных сумматоров.
Пример сложения трехразрядных чисел

A 0 1 1
B + 1 0 1
∑ 1 0 0 0

Схема трехразрядного сумматора реализующего данный пример представлена на рис.3.

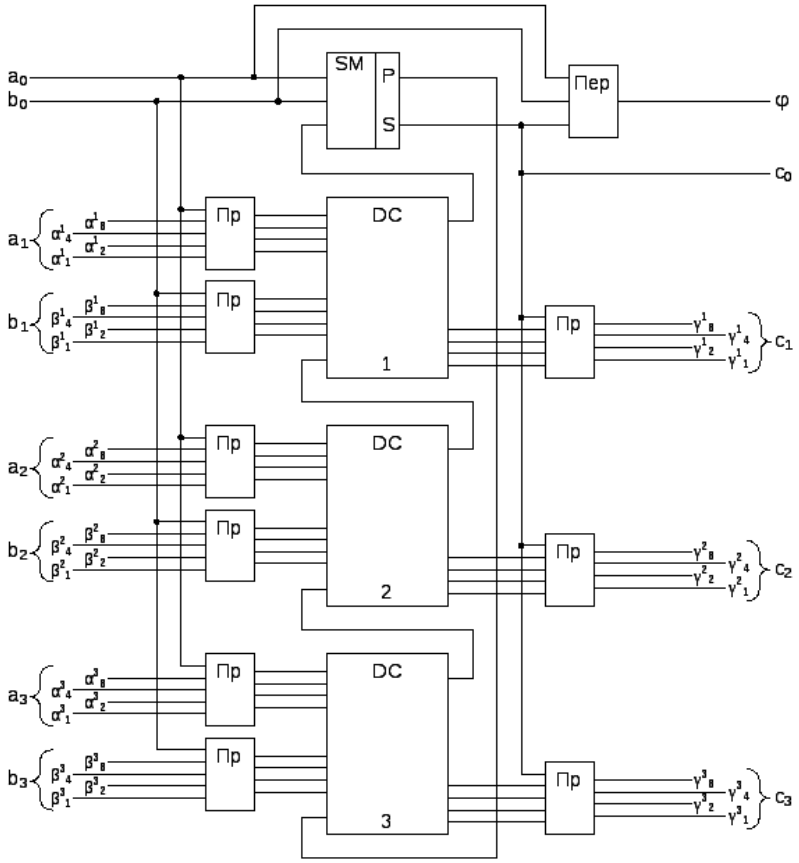


Рис. 6. 3-разрядный сумматор

Порядок выполнения работы:
1. Зарисовать схемы на карточках III-1, III-3
2. Продумать и составить таблицу истинности по каждой карточке и заполнить ее при помощи стенда ОАВТ
3. Для карточки III-3 составить таблицу с примерами сложения двух чисел согласно варианта

№ варианта	1 слагаемое (16-ричный код)	2 слагаемое (16-ричный код)	Состояние кнопки SB3 (при нажатой кнопке считается, что есть перенос из младшего разряда)
1	3	5	Без нажатия кнопки SB3
1	7	6	Без нажатия кнопки SB3
1	A	8	Без нажатия кнопки SB3
2	7	2	Без нажатия кнопки SB3
2	A	4	Без нажатия кнопки SB3
2	B	9	Без нажатия кнопки SB3
3	2	3	Без нажатия кнопки SB3
3	B	2	Без нажатия кнопки SB3
3	C	A	Без нажатия кнопки SB3
4	4	3	Без нажатия кнопки SB3

4	5	6	Без нажатия кнопки SB3
4	D	8	Без нажатия кнопки SB3
1	3	5	С нажатием кнопки SB3
1	7	6	С нажатием кнопки SB3
1	A	8	С нажатием кнопки SB3
2	7	2	С нажатием кнопки SB3
2	A	4	С нажатием кнопки SB3
2	B	9	С нажатием кнопки SB3
3	2	3	С нажатием кнопки SB3
3	B	2	С нажатием кнопки SB3
3	C	A	С нажатием кнопки SB3
4	4	3	С нажатием кнопки SB3
4	5	6	С нажатием кнопки SB3
4	D	8	С нажатием кнопки SB3

Содержание отчета:

1. Название и цель работы.
2. Исследуемые схемы
3. Таблицы истинности
4. Выводы по работе.

Контрольные вопросы:

1. Параллельный регистр. Назначение, Область применения, электрическая принципиальная схема, УГО параллельного регистра.
2. Последовательный регистр. Назначение, Область применения, электрическая принципиальная схема, УГО последовательного регистра..
1. Операции двоичного сложения, вычитания и умножения.
2. Полусумматор. Электрическая принципиальная схема, принцип работы, таблица истинности.
3. Полный сумматор. Электрическая принципиальная схема, принцип работы, таблица истинности.
4. Полувычитатель. Электрическая принципиальная схема, принцип работы, таблица истинности.
5. Полный вычитатель. Электрическая принципиальная схема, принцип работы, таблица истинности.
6. Многоразрядные сумматоры. Назначение. Электрическая принципиальная схема, принцип работы, таблица истинности.
7. Многоразрядные вычитатели. Назначение. Электрическая принципиальная схема, принцип работы, таблица истинности.
8. Многотактный умножитель. Привести структурную схему умножителя разрядности(5*3). Объяснить принцип и порядок работы.
9. Матричный умножитель. Привести структурную схему умножителя (4*4). Объяснить принцип и порядок работы.
10. Устройство и назначение семисегментного индикатора.
11. Принцип работы дешифратора.
12. Приведите часть электрической принципиальной схемы дешифратора для какого либо одного сегмента.

Лабораторная работа №4 Исследование счетчиков

Цель работы: Изучение ***T***-триггеров и счетчиков.

Основные теоретические сведения

T-триггер представляет собой счетный триггер. Схема ***T***-триггера с использованием ***D***- триггера приведена на рис.1.а. Принцип работы ***T***- триггера поясняет временная диаграмма работы рис.1.б. На каждые два импульса входного тактового сигнала ***T*** на выходе ***Q*** появляется только один импульс. ***T***-триггер может быть снабжен входом сброса ***R.***

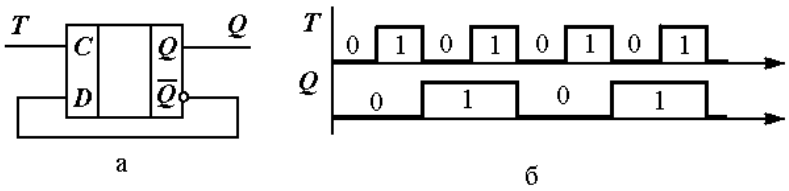


Рис. 1. Схема ***T***-триггера а) принципиальная схема, б) временная диаграмма

Счетчики

Счетчик – схема, собранная с использованием триггеров, осуществляющая подсчет количества импульсов поданных на тактовый вход.

Классификация счетчиков

Рис. 2. Классификация счетчиков

Замечания:

Суммирующий счетчик считает импульсы путем прибавления 1 к сумме.

Вычитающий счетчик считает импульсы путем вычитания 1 из суммы.

Счетчик со сквозным переносом осуществляет автоматический переход к началу счета после достижения модуля счета.

Счетчик с остановкой счета останавливается при достижении модуля счета.

Счетчик с фиксированным модулем 2^n осуществляет подсчет количества импульсов от нуля до переполнения мантиссы.

Счетчик с произвольным модулем предназначен для подсчета любого произвольно заданного количества импульсов.

В синхронном счетчике триггеры переключаются параллельно (одновременно, синхронно).

В асинхронном счетчике триггеры переключаются последовательно («неодновременно», по цепочке).

Трехразрядный асинхронный счетчик со сквозным переносом на JK триггерах

В таком счетчике все триггеры работают в режиме переключения. Тактовые импульсы подаются на первый триггер в схеме. При подаче тактовых импульсов каждый триггер делит частоту следования тактовых импульсов на два. Работу схемы поясняет таблица истинности 1 и временная диаграмма рис.4.

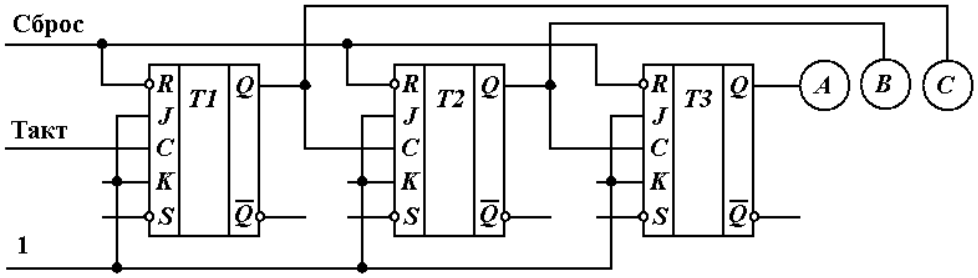


Рис. 3. Схема трехразрядного асинхронного счетчика со сквозным переносом на *JK* – триггерах

Таблица.1. Работа трехразрядного асинхронного счетчика

№ п/п	Такт	Сброс	A	B	C	Комментарий
1	---	0	0	0	0	Начальный сброс счетчика
2	---	1	0	0	0	Перевод схемы в режим счета импульсов
3	⎍	1	0	0	1	При первом тактовом импульсе переключается только триггер ***T1***
4	⎍	1	0	1	0	триггер ***T1*** переключается в 0, тем самым, формируется 1 тактовый импульс для триггера ***T2***, следовательно, триггер ***T2*** переключается 1 раз при двух переключениях триггера ***T1***
5	⎍	1	0	1	1	При дальнейшей подаче тактовых импульсов на вход схемы на выходах формируется двоичная последовательность, соответствующая числу тактовых импульсов поданных на счетчик
6	⎍	1	1	0	0	
7	⎍	1	1	0	1	
8	⎍	1	1	1	0	
9	⎍	1	1	1	1	
10	⎍	1	0	0	0	При переполнении мантиссы все триггеры сбрасываются

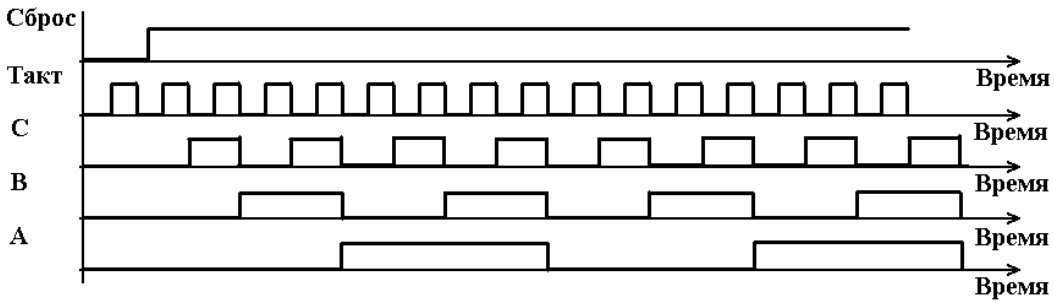


Рис.4. Временная диаграмма работы счетчика

Трехразрядный синхронный счетчик с переносом на JK триггерах

В отличие от асинхронного счетчика в синхронном счетчике тактовые импульсы подаются на все триггеры одновременно (параллельно). При этом реакция триггера зависит о того, в каком режиме работы он находиться. Если на входах J и K присутствует логическая единица, то триггер находиться в режиме переключения. Если на входах J и K присутствует логический ноль, то триггер находиться в режиме хранения и не реагирует на тактовые импульсы.

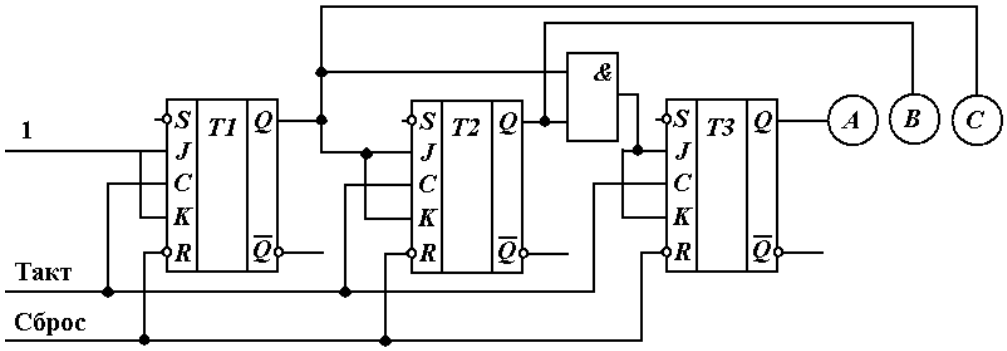


Рис. 5. Схема трехразрядного синхронного счетчика со сквозным переносом на *JK* - триггерах

Работу схемы поясняет таблица истинности 2.

Таблица.2. Работа трехразрядного синхронного счетчика

№ п/п	Сброс	Такт	Состояние выходов триггеров			Режимы работы триггеров после подачи такта (П-переключение, Х-хранение)			Комментарий
			A	***B***	***C***	Т3	Т2	Т1	
1	0	---	0	0	0	Х	Х	П	Начальный сброс схемы
2	1	---	0	0	0	Х	Х	П	Перевод схемы в режим счета импульсов
3	1	⎍	0	0	1	Х	П	П	По первому такту переключается только триггер ***T1.*** Поскольку на выходе триггера ***T1*** появилась 1 триггер ***T2*** перешел в режим переключения
4	1	⎍	0	1	0	Х	Х	П	По второму такту переключаются триггеры ***T1*** и ***T2.***
5	1	⎍	0	1	1	П	П	П	На входах *J* и *K* всех триггеров логические единицы, следовательно они все находятся в режиме переключения
6	1	⎍	1	0	0	Х	Х	П	При дальнейшей подаче тактовых импульсов на вход схемы на выходах формируется двоичная последовательность, соответствующая числу тактовых импульсов поданных на счетчик
7	1	⎍	1	0	1	Х	П	П	
8	1	⎍	1	1	0	Х	Х	П	
9	1	⎍	1	1	1	П	П	П	
10	1	⎍	0	0	0	Х	Х	П	Сброс счетчика при переполнении мантиссы

Порядок выполнения работы:
1. Зарисовать схемы на карточках V-1, V-3
2. Продумать и составить таблицу истинности по каждой карточке и заполнить ее при помощи стенда ОАВТ
3. Для карточки V-1 составить таблицу истинности с примерами счета, до модуля счета. Варианты заданий представлены в таблице. Модуль счета устанавливается при помощи соединительных проводов на картридже.

№ п/п	**Модуль счета**
1	7
2	8
3	9
4	A
5	B
6	C

3. Для карточки V-3 составить таблицу истинности с примерами счета, согласно варианта. (Назначение кнопок SB1-для подачи тактовых импульсов, SB2 – для подачи тактовых импульсов загрузки начального значения счетчика, SB3 –для сброса счетчика. Назначение переключателей: SA1-SA4 для выбора начального значения счетчика SA5 –для выбора типа счетчика (суммирующий/вычитающий))

№ п/п	**Тип счетчика**	**Начальное значение счетчика**	**Конечное значение счетчика**	
1	суммирующий	3	A	
2	суммирующий	5	C	
3	суммирующий	7	D	
4	вычитающий	A	3	
5	вычитающий	C	5	
6	вычитающий	D	7	

Содержание отчета:
1. Название и цель работы.
2. Исследуемые схемы.
3. Таблицы истинности.
4. Выводы по работе.

Контрольные вопросы:

1. Принцип работы, назначение и область применения счетчиков
2. Трехразрядный асинхронный счетчик со сквозным переносом. Принципиальная схема на триггерах. Таблица истинности. Принцип работы.
3. Трехразрядный синхронный счетчик со сквозным переносом. Принципиальная схема на триггерах. Таблица истинности. Принцип работы.
4. Трехразрядный асинхронный счетчик со переносом по модулю 9. Принципиальная схема на триггерах. Таблица истинности. Принцип работы.
5. Т триггер - Принципиальная схема и принцип работы.

Лабораторная работа № 5 Начальные сведения по модулю УМПК - 80/ВМ

Цель работы:

1. Изучение технических характеристик, функциональной схемы и элементарной базы **УМПК - 80/ВМ.**

2. Процедура включения **УМПК - 80/ВМ.**

3. Изучение работы клавиатуры **УМПК - 80/ВМ.**

1. Технические характеристики, функциональная схема и элементарная **база** УМПК - 80/ВМ.

1.1. Технические характеристики:

Напряжение питания	-5В ± 5% +5В ± 5% 12В±5%
Разрядность магистрали адреса	16 бит
Разрядность магистрали данных	8 бит
Тактовая частота	2 МГц
Опорная частота генерации	18МГц
Объем ПЗУ	2 Кбайт
Объем ОЗУ	2 Кбайт

Модуль УМПК - 80/ВМ производит индикацию светодиодами состояния:

- магистрали адреса;
- магистрали данных;
- магистрали управления;
- входного регистра;
- выходного регистра.

Модуль УМПК - 80/ВМ обеспечивает возможность записи и чтения информации при помощи бытового магнитофона.

В состав программного обеспечения входит операционная система и ряд демонстрационных программ. Операционная система позволяет выводить на светодиодный индикатор содержимое **ПЗУ, ОЗУ** и регистров процессора.

Модуль УМПК - 80/ВМ содержит имитатор входного устройства.

1.2. Функциональная схема модуля У МПК - 80/ВМ приведена на рис 1.

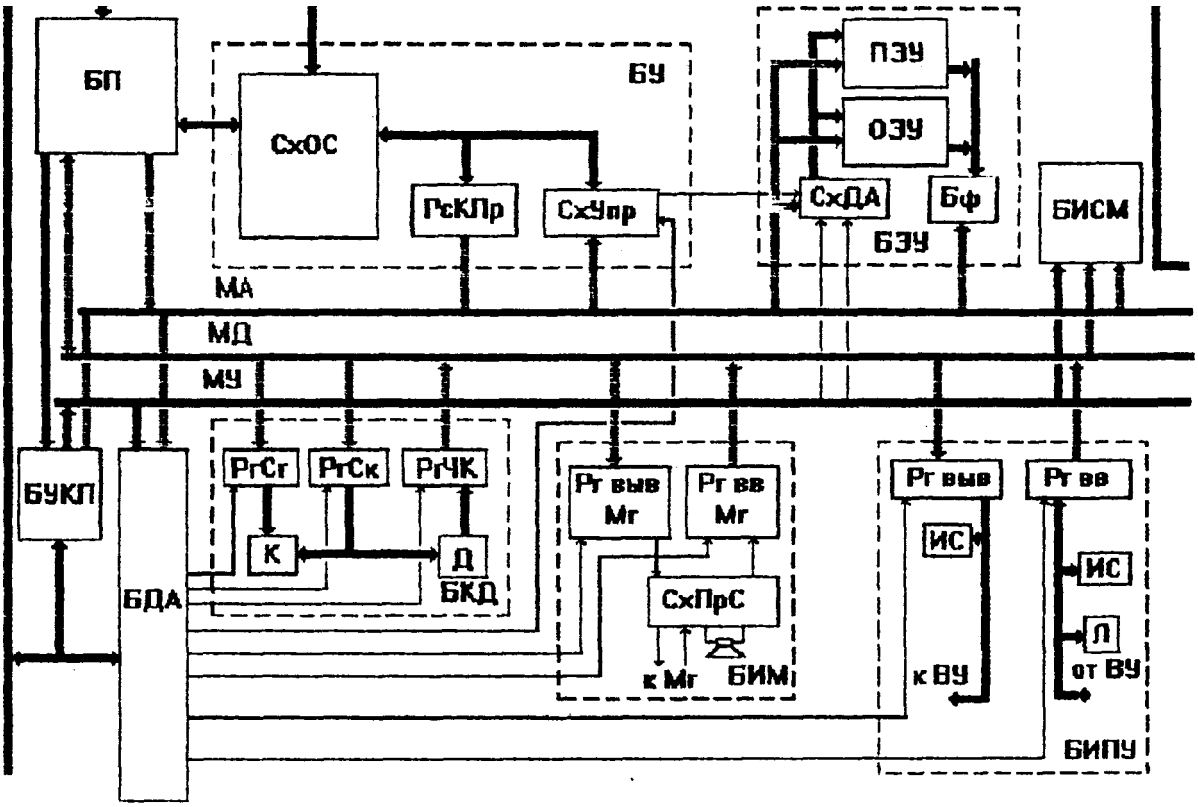


Рис.1. Функциональная схема модуля УМПК - 80/ВМ.

Блок процессора **(БП)** состоит из:
- БИС процессора КР580ВМ80А;
- БИС тактового генератора КР580ГФ24;
- буфер магистрали адреса КР580ВА86;
- буфер магистрали данных и системного контроллера КР580ВК28;
- буфер внешней магистрали К 155ЛП 10.

Блок управления картой памяти **(БУКП)** состоит из:
- мультиплексора сигналов магистрали управления К531 КП 11 П;
- логических элементов К555ЛИ 1, К555ЛЛ 1, К555ЛА2.

Блок запоминающих устройств **(БЗУ)** состоит из:
-ОЗУ БИС КР537РУ8Б;
-ПЗУ БИСКР556РТ7;
- дешифратора адреса К555ИД7. Объем ОЗУ и ПЗУ по 2 Кбайта.

Дешифратор адреса определяет адреса:
ПЗУ от ОООООН до 07FFH;
ОЗУ от 0800Н до OFFFH.

Переключателем SA6 БЗУ может быть отключен (верхнее положение), а переключателем SA2.4 может быть включена защита ОЗУ от записи (верхнее положение).

Блок дешифрации адресов (БДА) устройств ввода-вывода построен на микросхеме Kl 55РЕ3.

Устройствами ввода-вывода модуля УМПК - 80/ВМ являются:
- клавиатура;
- светодиодный знаковый дисплей;
- входной регистр РгВв;
- выходной регистр РгВыв;
- схема звуковой сигнализации;
- регистр ввода-вывода информации на магнитофон.

Блок клавиатуры и дисплея **(БКД)** состоит из:
- регистра сканирования КР580ИР83;
- регистра сегментов КР580ЛП83;
- регистра чтения клавиатуры Kl 55ЛП 11;

- светодиодных индикаторов **АЛС333Б;**
- кнопок клавиатуры.

Клавиатура модуля УМПК - 80/ВМ содержит 26 кнопок. Дисплей предназначен для индикации в 16-ричном коде содержимого ячеек текущего адреса, данных и регистров процессора. Блок имитации периферийных устройств (БИПУ). Блок управления (БУ) режимами работы процессора. Этот блок обеспечивает отладку программы в 2-х пошаговых режимах:

- выполнение программы по шагам;
- выполнение программы по шагам машинных циклов.

Расположение элементов на плате показано на рис.2.

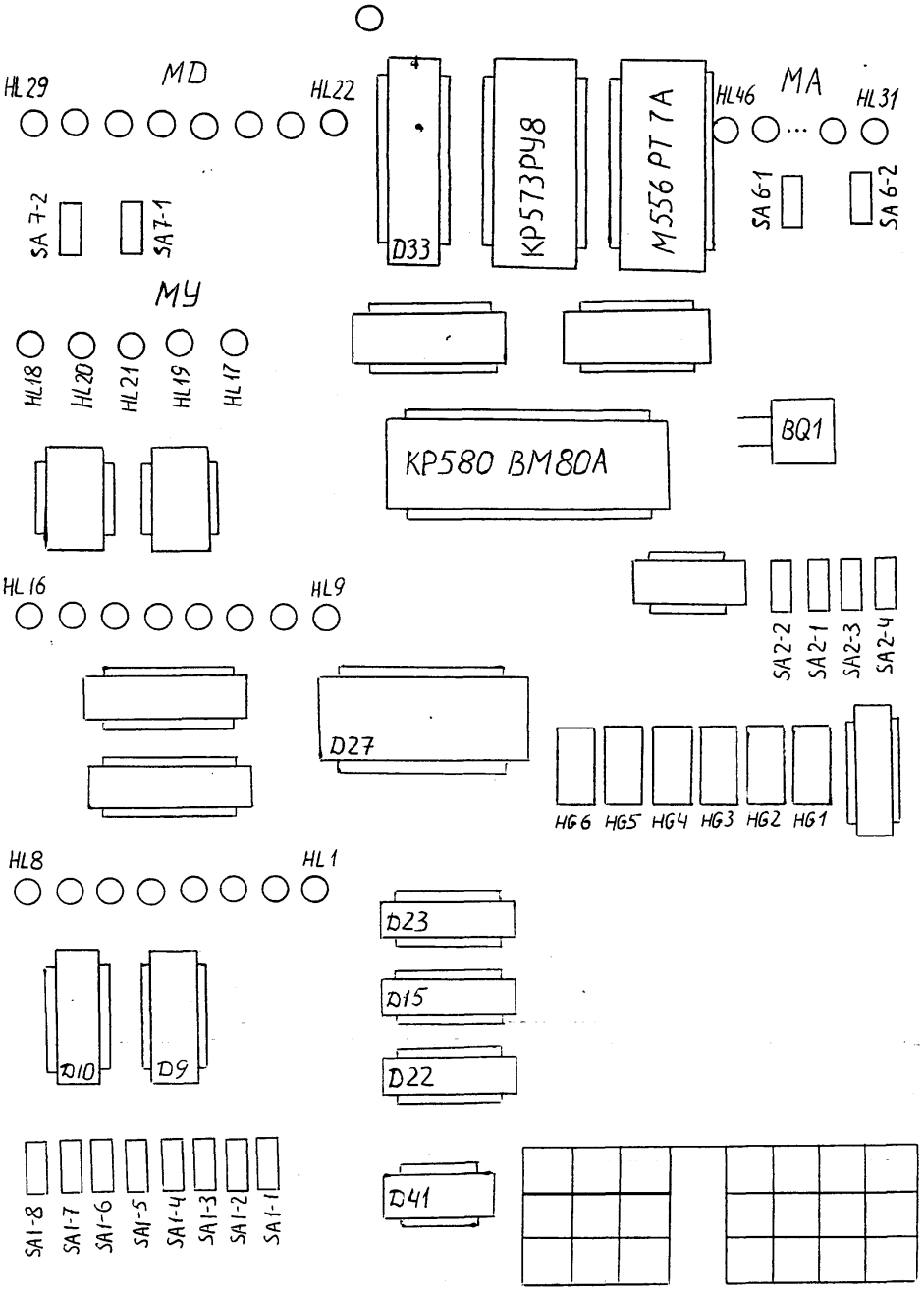


Рис.2. Расположение элементов на плате.

2. Процедура включения модуля УМПК - 80/ВМ.

1) Проверьте, что сетевая кнопка находится в отжатом положении.

2) Вставьте сетевую вилку в розетку с напряжением 220 вольт.

3) Нажмите сетевую кнопку. При этом выполняется тест, включающий в себя:

- начальную установку регистров процессора;

- тест ячеек ОЗУ.

ПРИ НЕИСПРАВНОСТИ ОЗУ НА ДИСПЛЕЕ ВЫСВЕЧИВАЕТСЯ НАДПИСЬ "ОЗУ", СОПРОВОЖДАЕМАЯ ЗВУКОВЫМ СИГНАЛОМ!

- проверку светодиодного дисплея, во время которой должны загореться все сегменты светодиодных индикаторов;

- проверку выходного регистра путем записи в него логических единиц во все разряды, при этом должны загореться все светодиоды выходного регистра.

4) При успешном завершении теста подается звуковой сигнал и на дисплее появляется надпись "НАЧАЛО". Данная надпись свидетельствует о готовности модуля к работе. ПРИ НЕОБХОДИМОСТИ ТЕСТ МОЖНО ПОВТОРИТЬ, НАЖАВ КЛАВИШУ [R].

3. Изучение работы клавиатуры модуля УМПК - 80/ВМ.

После успешного завершения теста модуль УМПК - 80/ВМ воспринимает информацию от следующих клавиш: [от А],[от Рг], [Пр Сч].

3.1. При нажатии клавиши [Пр Сч] на дисплее высвечивается содержимое программного счетчика (после окончания теста 0800_00 -начальный адрес ОЗУ).

3.2. При нажатии клавиши [от Рт] модуль УМПК - 80/ВМ переводится в режим индикации на дисплее состояния регистров процессора. Всего процессор содержит 10 регистров:

A 00 - аккумулятор;

FL 00 - флаговый регисгр;

B 00 - регистр B;

C 00 - регистр C;

D 00 - регистр D;

E 00 - регистр E;

H 00 - регистр H;

L 00 - регистр L;

SPH OF - старший байт стекового регистра;

SPL BO - младший байт стекового регистра;

PCH 08 - старший байт программного счетчика;

PCL 00 - младший байт программного счетчика.

Используя клавиши [Зп Ув] и [Ум], можно просмотреть содержимое этих регистров. Клавиша [Зп Ув] обеспечивает просмотр в том порядке, как они были перечислены, а [Ум] в обратном порядке. В приведенном списке регистров справа расположены их названия, а слева - обозначения соответствующего регистра на светодиодном дисплее и его содержимое после проведения теста.

3.3. Запись данных в регистр процессора.

Запись производится следующим образом:

- модуль переводится в режим индикации регистров процессора;

- при помощи кнопок [Зп Ув] и [Ум] выбирается нужный регистр;

- вводится информация при помощи цифровых клавиш [O...F]. При вводе данных на индикаторе младшего разряда загорается запятая - это означает, что данные записаны в регистр дисплея, но не записаны в регистр процессора. Окончательная запись происходит при нажатии кнопки [Зп Ув]. При этом на дисплее высвечивается информация о следующем регистре.

3.4 Ввод адреса.

При нажатии на кнопку [От А] на дисплее загораются нижние горизонтальные сегменты индикаторов адреса. Это означает, что управляющая программа готова к вводу адреса, который вводится путем нажатия кнопок [0...F]. При этом на дисплее высвечивается содержимое ячеек памяти, соответствующих введенному адресу.

В этом режиме возможен просмотр и модификация ячеек **ОЗУ,** что производится аналогично такой же процедуре с регистрами процессора (используются клавиши [Зп Ув], [Ум] и цифровые клавиши [0...F]).

Следует отметить, что при попытке модификации ячеек ПЗУ (что невозможно) выдается предупреждение в виде звукового сигнала, а информация на дисплее не меняется.

3.5 Запуск программы.

После занесения программы в ячейку ОЗУ ее можно запустить при помощи клавиши [П] из режима просмотра и модификации памяти. При этом программа будет запущена с адреса, который показан на дисплее.

При необходимости программу можно остановить нажатием клавиши [Ст], при этом на дисплее будет показан адрес и содержание ячейки памяти, на которой произведена остановка программы.

3.6 Пошаговая трассировка программы.

Пошаговая трассировка может быть осуществлена двумя способами:

- покомандная трассировка может производиться путем нажатия клавиши [Шк];
- трассировка по машинным циклам осуществляется нажатием клавиши [Шц].

В обоих случаях в паузах трассировки возможен просмотр и модификация ячеек ОЗУ и регистров процессора, как это было описано выше.

Выход из режима трассировки происходит по нажатию клавиши [Cm].

4. Задание для самоконтроля.

4.1. Найти на модуле **УМПК** - 80/ВМ **БИС** процессора, тактового генератора, **ПЗУ, ОЗУ.**

4.2. Найти на клавиатуре все перечисленные в тексте клавиши.

4.3. Включить модуль УМПК - 80/ВМ.

4.4. Запустить тест с клавиатуры.

4.5. Просмотреть содержимое регистров процессора и записать в аккумулятор сначала все единицы, а потом все нули.

4.6. Просмотреть несколько ячеек **ОЗУ,** начиная с адреса 0900. Записать в них сначала все единицы, а потом все нули.

4.7. Просмотреть содержимое нескольких произвольных ячеек ПЗУ. Убедиться в невозможности их модификации.

Лабораторная работа №6. Изучение основ программирования микропроцессора КР580ВМ80А на языке ассемблера. Команды передачи данных

Цель работы изучение архитектуры микропроцессора КР580ВМ80А и команд передачи данных.

1. Общие сведения.

Архитектура микропроцессора **КР580ВМ80А.** Микропроцессор КР580ВМ80А представляет собой однокристальный 8-разрядный параллельный микропроцессор с фиксированной системой команд.

Микропроцессор содержит 16-разрядную шину адреса, двунаправленную 8-разрядную шину данных, 4 выходных и 6 входных линий управления.

Для пользователя, работающего на языке ассемблера, микроЭВМ представляется в виде следующих элементов:

- память;
- порты ввода-вывода;
- регистры общего назначения микропроцессора **(МП);**
- счетчик команд;
- слово состояния **МП;**
- указатель стека;
- система прерываний;
- система адресации;
- система команд.

Память, используемая, для хранения программ, исходных данных и результатов вычислений, для программиста представляет собой одномерный массив байт, имеющих адреса от **ОН** до **FFFFH,** т.е. адресное пространство составляет 64 Кбайт. Для связи с внешними устройствами в составе **микроЭВМ** допускается использовать 256 портов ввода и 256 портов вывода. Количество портов ввода-вывода, относящихся к одному периферийному устройству, определяется структурой самого устройства.

В состав микропроцессора входят семь 8-битных регистров общего назначения **(РОН).**

Регистр-аккумулятор "А" используется как источник одного из операндов при выполнении операции и как регистр хранения результата операции **(т.е.** результат операции всегда заносится в аккумулятор). Остальные регистры **(B, C, D, E, H, L)** могут использоваться в качестве источников операндов, а также для хранения адресов операндов (при использовании регистровых пар).

Все РОН имеют номера, используемые для адресации регистров в машинных инструкциях. В программе на языке ассемблера к регистрам можно обращаться по именам. Соответствие имен и номеров РОН приведено ниже:

Имена РОН	A	B	C	D	E	H	L
Номера РОН	7	0	1	2	3	4	5

Некоторые команды МП используют в качестве операнда не регистры, а регистровые пары, которые обозначаются именем старшего из регистров и имеют следующие номера:

Регистровая пара	Регистры	Номер пары
B	B, C	0
D	D,E	1
H	H.L	2

Разряды в регистрах и ячейках памяти нумеруются справа налево, таким образом, младшим значащим битом является бит "О", а старшим - бит "7".

Счетчик команд **(PC)** представляет собой 16-разрядный программно-доступный счетчик и содержит адрес первого байта следующей выполняемой команды.

Слово состояния процессора или регистровая пара PSW состоит из двух регистров: аккумулятора "А" и признакового регистра "F".

Регистр "F" содержит признаки выполнения операций, которые используются при выполнении команд условного перехода и в команде десятичной коррекции. Регистр "F" имеет следующий формат:

7	6	5	4	3	2	1	0
S	Z	0	AC	0	P	1	CY

Назначение битов регистра "F": CY - бит переноса. Устанавливается в единицу при возникновении переноса или заёма в старшем разряде; P - бит четности. Устанавливается в единицу, если число единичных бит аккумулятора четно; AC - бит вспомогательного переноса. Указывает на наличие переноса из третьего разряда аккумулятора. Используется только в команде десятичной коррекции; Z - бит нулевого результата операции. Устанавливается в единицу при нулевом значении бит аккумулятора;

S - бит знака. Принимает единичное значение, если результат операции отрицательный, т.е. значение бита совпадает с 7 разрядом аккумулятора.

Указатель стека (SP) предназначен для работы со стеком. Стек представляет собой область памяти, используемую для временного хранения данных. Занесение данных в стек и извлечение их из стека производится по правилу "ПОСЛЕДНИЙ ПРИШЕЛ - ПЕРВЫЙ УШЕЛ" (LIFO), т.е. последний записанный в стек элемент извлекается из него первым.

Существуют только две операции со стеком: запись данных в стек (PUSH) и чтение данных из стека (POP).

Операция PUSH используется для записи в стек данных из регистровой пары или счетчика команд. Запись данных производится с использованием указателя стека SP следующим образом:

- старший байт счетчика команд или первый регистр регистровой пары запоминается в памяти по адресу, на единицу меньше, чем значение указателя стека;
- младший байт счетчика команд или второй регистр регистровой пары запоминается в памяти по адресу, на два меньше, чем значение указателя стека;
- содержимое указателя стека уменьшается на два. Операция POP используется для извлечения из стека данных в регистровую пару или счетчик команд. Чтение данных производится с использованием указателя стека SP следующим образом:
- второй регистр регистровой пары или старший байт счетчика команд считывается из памяти по адресу, определяемому указателем стека;
- первый регистр регистровой пары или старший байт счетчика команд считывается из памяти по адресу, на единицу больше, чем значение указателя стека;
- содержимое указателя стека увеличивается на 2. Перед выполнением операций со стеком должно быть установлено значение указателя стека, определяющее адрес вершины стека. Вершиной стека называется байт в стеке, адрес которого определяется текущим значением указателя стека.

Система прерываний. В МП имеется триггер разрешения прерывания (РПР), который может быть установлен в "О" или "1" при помощи специальных команд. Если триггер РПР сброшен в "О", то система прерываний блокирована, и запросы на прерывание не обрабатываются МП. Если же триггер РПР установлен в "I", то запрос на прерывание обрабатывается МП следующим образом:

- заканчивается выполнение текущей команды;

- сбрасывается в "О" триггер разрешения прерывания ;

- периферийное устройство аппаратно формирует команду, обеспечивающую переход на обработку прерывания, и передает ее в МП для выполнения.

Система адресации. Под системой адресации понимается совокупность способов задания адреса операнда в команде (режим адресации) и механизма доступа к ячейкам памяти.

Каждая из команд МП может работать только с определенными видами адресации, которые рассматриваются ниже.

Регистровая адресация предусматривает указание в коде команды одного или двух регистров A, B, C, D, E, H, L, являющихся источниками и приемниками операндов. В случае использования регистровой пары имя регистра определяет соответствующую регистровую пару.

При непосредственной адресации операнд является частью команды и следует непосредственно за кодом операции. В зависимости от длины операнд располагается во втором (байт) или во втором и третьем (слово) байтах команды.

В случае прямой адресации во втором (втором и третьем) байте команды содержится адрес данных, используемых в операции. Таким адресом может быть либо номер порта ввода-вывода, либо 16-разрядный адрес памяти.

При косвенной адресации в коде команды задается номер регистровой пары (B, D, H), содержимое которой определяет адрес операнда, участвующего в операции.

Форматы данных. Данные, используемые в МП, по способу их интерпретации аппаратными и программными средствами МП можно разделить на следующие группы:

- числа без знака;
- числа со знаком;
- адреса;
- символьные коды.

Числа без знака представляются с использованием всех разрядов байта или слова и имеют следующий диапазон представления:

единица представления - байт

0...377(8) или 0...255(10) или О...FFH

единица представления - слово

0...177777(8) или 0...65535(10) или О...FFFFH

Числа со знаком. При представлении в МП этих чисел старший бит байта или слова определяет знак числа, а остальные биты используются для определения значения числа. Числа представляются в дополнительном коде и имеют следующий диапазон:

единица представления - байт

-128(10)...+127(10)

единица представления - слово

-32768(10)...+32767(10)

Адрес задается 16-разрядным словом, определяющим номер байта или слова памяти, содержащего операнд. Величина адреса может лежать в пределах О...65535(10) или О...FFFFH.

Символьный код используется для представления в программе текстовой информации в коде КОИ-7. Каждый символ занимает один байт, причем старший бит всегда равен 0.

Система команд. Команды МП КР580ВМ80А состоят из одного, двух или трех последовательно расположенных в памяти байт в зависимости от типа команды.

Мнемоника команд МП приведена в приложении.

В однобайтных командах байт используется для задания кода выполняемой операции и операндов.

Двухбайтные команды содержат в первом байте код операции, а во втором - номер порта ввода-вывода или данные.

Трехбайтные команды в первом байте содержат код операции, а во втором и третьем байтах - либо адрес памяти, либо данные. В командах этого вида во втором байте содержится младший байт, а в третьем - старший байт адреса или данных.

Команды передачи данных обеспечивают пересылку данных между регистрами или между памятью и регистрами.

Формат команд:

КОП DST.SRC

где **КОП** - код операции;

DST - приемник информации **(DESTINATION);**

SRC - источник информации **(SOURCE).**

Команды передачи данных не изменяют состояние битов условий.

Команды передачи данных можно разделить на следующие группы:

- засылки константы;
- пересылки;
- чтения-записи.

В результате выполнения команды засылки константы в регистр или регистровую пару загружается константа, содержащаяся во втором или по втором и третьем байтах команды. Формат команд засылки констант:

MVI DST,D8

где DST-любой из РОН (A, B, C, D, E, H, L) или ячейка памяти (M);

D8 - 8-разрядная величина константы;

или

LXI RP,D16 где RP - регистровая пара B, D, H или указатель стека SP;

D 16-16-разрядная величина константы.

При использовании команды

MVI M,D8

т.е. засылки константы в память, адрес соответствующего байта определяется содержимым регистровой пары (H, L).

В любом случае, когда в мнемонике команды присутствует символ M, это означает, что одним из операндов является ячейка памяти, адресуемая через регистровую пару (H, L).

При выполнении команд пересылки содержимое источника SRC пересылается в приемник DST, при этом содержимое источника не изменяется. В качестве источника и Приемника операндов может быть использован любой из регистров общего назначения или ячейка памяти, адресуемая через регистровую пару (H, L).

Примечание. Пересылка данных типа "ПАМЯТЬ-ПАМЯТЬ" запрещена.

Формат команд пересылки MOV DST, SRC

где DST, SRC-любой из РОН (A, B, C, D, E, H, L) или ячейка памяти (M).

К командам пересылки можно также отнести команду XCHG, в результате выполнения которой регистровые пары (H, L) и (D, E) обмениваются содержимым следующим образом:

H с D

L с E

Команды чтения-записи:

- LDAX RP-запись в аккумулятор содержимого ячейки памяти, адресуемой через регистровую пару (B, C) или (D, E);
- STAX RP-запись аккумулятора в ячейку памяти адресуемую через регистровую пару (B, C) или (D, E);
- LDA Adr-запись в аккумулятор содержимого ячейки, адрес которой определяется 16-разрядным адресом Adr;
- LHLD Adr-запись в регистровую пару (H, L) содержимого двух последовательных ячеек памяти с адресами Adr и Adr+1, причем в регистр H загружается содержимое ячейки по адресу Adr+1, а в регистр L - по адресу Adr;

- SHLD Adr-запись содержимого регистровой пары (H, L) в две последовательные ячейки памяти с адресами Adr и Adr+1, причем содержимое регистра H записывается в ячейку с адресом Adr+1, а регистра L - в ячейку с адресом Adr.

2. Примеры использования команд передачи данных.

Одна из наиболее часто встречающихся задач при программировании - засылка некоторого числа в аккумулятор. В таблице приведены различные варианты загрузки аккумулятора.

Команда	Выполняемое действие
MOV A,B	Загрузить число из регистра B
MVI A,23H	Загрузить непосредственно число 23H
LDA 4098H	Загрузить число из ячейки памяти 4098H
LDAX B	Загрузить число из ячейки памяти, адресуемой через пару (B, C)
LDAX D	Загрузить число из ячейки памяти, адресуемой через пару (D, E)
LHLD 4098H MOV A,M	Загрузить число из ячейки, адрес которой находится в ячейках 4098H-4099H (косвенная адресация)

3. Порядок выполнения работ.

А. Составить программу на языке ассемблера **(см.** приложение), обеспечивающую формирование массива из 5 байт, содержащих заданную информацию, согласно номеру варианта.

Вариант	Адрес массива	Набор констант	Адресация массива	Адресация констант	Регистр-источник
1	0860H	1.2,3,4,5	H.L	Непосред-ственная	B
2	0870H	F,E,D,C,B	B. C	То же	D
3	0880H	1,3,5,7,9	D.E	Тоже	H
4	0890H	2,4,6,8,A	Прямая	Тоже	A
5	08A0H	1,F,3,E,5	H.L	Тоже	C
6	08B0H	2,A,4,B,6	B, C	Тоже	E
7	08C0H	F,D.B,9.7	D.E	Тоже	L
8	08E0H	E,C,A,8,6	Прямая	Прямая	A

Б. Произвести ассемблирование программы вручную, получив тем самым машинные коды программы.

Пример. Пусть задана программа, осуществляющая занесение константы 55H во все регистры процессора. Предположим, что адрес начала программы равен 2100H.

```
START:
MVI   A, 55   ;Выборка константы
MOV   B, A    ;Запись константы в регистр B
MOV   C, A    ;Запись константы в регистр C
MOV   D, A    ;Запись константы в регистр D
MOV   E, A    ;Запись константы в регистр E
MOV   H, A    Запись константы в регистр H
MOV   L, A    ;Запись константы в регистр L
```

Для ассемблирования программы вручную используем приложение. Для каждой команды программы находим в приложении ее машинный код и выписываем его, помечая соответствующим адресом ячейку, в которой этот код будет записан. Например, для команды MVI A, 55H машинный код будет состоять из 2 байт, в первом байте будет записан код команды 3EH, а во втором - константа 55H. Эти байты будут записываться в память по адресам 0800H и 0801 H соответственно.

Если в программе есть команда, использующая 16-разрядную константу, например, LXI H, 2345H то машинный код этой команды будет состоять из трех байт. В первом байте будет записан код команды 21H, а во втором - младший байт константы 45H, а в третьем - старший байт константы 23H.

Выполнив соответствующие действия для каждой команды, получим следующие машинные коды программы:

0800 3E 55 START: MVI A,55H
0802 47 MOV B,A
0803 4F MOV C,A
0804 57 MOV D,A
0805 5F MOV E.A
0806 67 MOV H.A
0807 6F MOV L.A

которые необходимо записать в память контроллера.

Примечание. Для корректного завершения программы в контроллере необходимо, чтобы программа заканчивалась кодами C3H, 00H и 00H. В этом случае после окончания программы в первой позиции экрана будет инициироваться цифра "8".

В. Записать программу в память контроллера и выполнить ее, исследуя при этом содержимое рабочих регистров и ячеек памяти.

4. Содержание отчета.

Отчет по работе должен содержать:

- краткие записи о командах;
- текст программы с указанием адресов ячеек и кодов программы;
- протокол выполнения программы, т.е. содержимое ячеек массива и рабочих регистров после выполнения каждой команды программы.

Лабораторная работа № 7. Изучение основ программирования микропроцессора КР580ВМ80А на языке ассемблера. Арифметические и логические команды

Цель работы: изучение арифметико-логических команд и команд сдвигов.

1. Общие сведения.

Арифметические команды обеспечивают выполнение операций сложения и вычитания, изменение операнда на единицу.

Арифметические операции можно разделить на следующие группы:

- операции с одним операндом (регистром или регистровой парой);

- операции с двумя операндами .причем в качестве первого используется аккумулятор, а в качестве второго - регистр, ячейка памяти или константа. Результат операции всегда записывается в аккумулятор.

Арифметические команды изменяют состояние битов условий. При выполнении команды сложения в аккумулятор заносится результат сложения аккумулятора и операнда-источника. Операндом-источником может быть регистр, ячейка памяти или константа, которая в этом случае записывается в следующем за кодом команды байте.

К этим командам относятся:

ADD SRC - сложение содержимого аккумулятора с регистром или ячейкой памяти;

ADC SRC - сложение содержимого аккумулятора со значением бита CY и содержимым регистра или ячейки памяти;

ADI DB - сложение содержимого аккумулятора с константой;

ACI DB - сложение содержимого аккумулятора со значением бита CY и константой.

Эти команды изменяют состояние всех битов условий. **DAD RP** - сложение содержимого регистровой пары RP с содержимым регистровой пары **(H, L).** Результат сложения записывается в пару **(H, L). При** сложении содержимое регистровой пары рассматривается как 16-разрядное число.

Эта команда изменяет состояние только бита **CY. При** выполнении команд вычитания в аккумулятор заносится результат вычитания операнда-источника из аккумулятора. Операндом-источником может быть регистр, ячейка памяти или константа, которая в этом случае записывается в следующем за кодом команды байте.

К этим командам относятся:

SUB SRC-вычитание из содержимого аккумулятора содержимого регистра или ячейки памяти;

SBB SRC-вычитание из содержимого аккумулятора значения бита **CY** и содержимого регистра или ячейки памяти;

SUI DS-вычитание константы из содержимого аккумулятора;

SBI DS-вычитание из содержимого аккумулятора значения бита **CY** и константы.

Эти команды изменяют все биты условий.

Команды изменения операнда на единицу предназначены для увеличения или уменьшения операнда на единицу. Операндом может являться содержимое регистра, ячейки памяти или регистровой пары.

Эти команды часто применяются для организации счетчиков или изменения адресов, используемых при косвенной адресации (обработка массивов, матриц и т.п.).

INR SRC - увеличение на единицу содержимого регистра или ячейки памяти;

DCR SRC - уменьшение на единицу содержимого регистра или ячейки памяти.

Команды **INR** и **DCR** изменяют состояние всех битов условий за исключением бита **CY.**

INX RP-увеличение на единицу содержимого регистровой пары;

DCX RP-уменьшение на единицу содержимого регистровой пары.

Команды **INX и DCX** не изменяют состояние битов условий.

При выполнении команды десятичной коррекции (формат **DAA**) 8-битное число в аккумуляторе рассматривается как две 4-битные десятичные двоично-кодированные цифры. Коррекция содержимого аккумулятора производится по следующим правилам:

- если значение младшей тетрады аккумулятора больше 9 или если признак вспомогательного переноса **AC** равен 1, то к содержимому аккумулятора добавляется число 6;

- если значение старшей тетрады аккумулятора больше 9 или если признак переноса **CY** равен 1, то к содержимому аккумулятора добавляется число **96.**

Команда **DAA** изменяет состояние всех битов условий.

При выполнении логических команд в аккумулятор заносится результат логической операции над операндом-источником и аккумулятором. Операндом-источником может быть регистр, ячейка памяти или константа, которая в этом случае записывается в следующем за кодом команды байте.

К логическим командам относятся:

ANA SRC - "Логическое И" содержимого аккумулятора и операнда-источника;

ORA SRC- "Логическое ИЛИ" содержимого аккумулятора и операнда-источника;

XRA SRC- "Исключающее ИЛИ" (сложение по модулю 2) содержимого аккумулятора и операнда-источника;

CMP SRC- сравнение содержимого аккумулятора и операнда-источника. При выполнении этой операции операнд вычитается из содержимого аккумулятора без изменения участвующих в операции операндов. Состояние битов условий устанавливается по результату вычитания:

ANI D8-"Логическое **И**" содержимого аккумулятора и константы;

ORI D8-"Логическое **ИЛИ"** содержимого аккумулятора и константы;

XRI D8-" Исключающее **ИЛИ"** (сложение по модулю 2) содержимого аккумулятора и константы;

CPI D8-сравнение содержимого аккумулятора с константой. При выполнении этой операции константа вычитается из содержимого аккумулятора без изменения участвующих в операции операндов. Состояние битов условий устанавливается по результату вычитания.

Команды изменяют состояние всех битов условий. Все команды, за исключением команд **CMP** и **CPI**, устанавливают значение бита **CY**, равное 0.

Команды сдвигов выполняются над расширенным 9-битным операндом, состоящим из аккумулятора и бита **CY** признакового регистра **F**. Операции сдвигов часто используются для умножения на число, представляющее собой степень 2, т.е. на числа 2; 4; 8;

16... или 0,5; 0,25; 0,125...

Сдвиг на один разряд влево равносилен удвоению операнда, а сдвиг вправо - делению операнда на 2. При этом необходимо помнить, что младший разряд при сдвиге влево (или старший разряд при сдвиге вправо) должен заполняться 0 для положительных чисел и 1 - для отрицательных.

К командам сдвига относятся:

RRC - циклический сдвиг вправо. Значение младшего бита заносится одновременно в бит **CY** признакового регистра **F** и старший разряд аккумулятора, остальные биты сдвигаются на один разряд вправо;

RLC - циклический сдвиг влево. Значение старшего бита заносится одновременно в бит CY и младший разряд аккумулятора, остальные биты сдвигаются на один разряд влево;

RAR - арифметический сдвиг вправо. Значение младшего бита заносится в бит **CY**. Значение бита **CY** заносится в старший разряд аккумулятора, остальные биты сдвигаются на один разряд вправо;

RAL - арифметический сдвиг влево. Значение старшего бита заносятся в бит **CY**. Значение бита **CY** заносится в младший разряд аккумулятора, остальные биты сдвигаются на один разряд влево.

Команды сдвигов изменяют состояние только бита **CY.**

2. Примеры программирования.

Используя арифметические .логические команды и команды сдвига, можно составлять программы, реализующие различные вычислительные операции, такие, как операции с повышенной точностью, преобразование кодов, формирование контрольных разрядов и т.п.

Операции с повышенной точностью.

При выполнении операций с повышенной точностью каждый операнд, участвующий в операции, имеет длину 2 и более байт.

В этом случае алгоритм операции сложения или вычитания данных с повышенной точностью будет следующим:

1) выполнить операцию над младшими байтами операндов;

2) выполнить требуемую операцию над следующей парой байт операндов, учитывая при этом возникший перенос (для сложения) или заем (для вычитания);

3) повторить п. 2 алгоритма для всех пар байт операндов. В качестве примера приведена программа, реализующая сложение двух 3-байтовых чисел, первое из которых расположено в регистрах **B, C, D**, а второе - в регистрах **E, H, L**.

```
MADD:MVI B, 023H ;
MVI C, OFFH ; Подготовить
MVI D, OADH; данные
MVI E.075H ;для
MVI H, 002H ; сложения
MVI L, 03AH ;
MOV A, D ; Передать 1 байт первого слагаемого
ADD L ; Получить сумму
MOV D, A ; Запомнить результат
MOV A, C ; Передать 2 байт первого слагаемого
ADC H ; Получить сумму с учетом переноса
MOV C, A ; Запомнить результат
MOV A, B ; Передать 3 байт первого слагаемого
ADC E ; Получить сумму с учетом переноса
MOV B, A ; Запомнить результат
```

В этой программе:

- первые шесть команд осуществляют запись в регистры байт слагаемых, первое из которых равно **023FFAD**H, а второе **-075023A**H;

- следующие три команды выполняют сложение младших байт слагаемых, в результате чего будет получен младший байт результата, равный 0E7H, и бит **CY**=0;

- следующие три команды выполняют сложение вторых байт , в результате чего будет получен второй байт результата, равный 01H, и бит **CY**=1;

- последние три команды выполняют сложение старших байт слагаемых, в результате чего будет получен старший байт результата, равный **097**H, и бит **CY**=0;

- после выполнения программы регистры **B, C, D** будут содержать результат сложения двух трехбайтовых чисел, равный **09701E7**H.

Если в приведенной программе команды сложения заменить соответствующими командами вычитания, то получится программа многобайтового вычитания.

3. Порядок выполнения работы.

А. Составить программу, выполняющую действия в соответствии с вариантом задания:

1. Сложение шестиразрядных двоично-десятичных чисел.

2. Подсчет количества единичных битов в 16-разрядном слове.

3. Подсчет количества нулевых битов в шестнадцатиразрядном слове.

Б. Произвести ассемблирование полученной программы вручную и загрузить ее в ОЗУ контроллера.

В. Выполнить программу, исследуя при этом содержимое рабочих регистров и ячеек памяти.

4. Содержание отчета.

Отчет по работе должен содержать:

- краткие сведения о командах;
- текст программы с указанием адресов ячеек и кодов программы;
- протокол выполнения программы, т. е. содержимое рабочих ячеек памяти и регистров после выполнения каждой команды программы.

Лабораторная работа № 8. Изучение основ программирования микропроцессора КР580ВМ80А на языке ассемблера. Команды передачи управления и специальных команд

Цель работы: изучение команд передачи управления и специальных команд.

1. Общие сведения.

Команды передачи управления. К ним относятся команды безусловного и условного переходов, команды безусловного и условного вызова и выхода из подпрограммы, а также команды перезапуска.

Команды этой группы не изменяют биты условий, а используют их при своем выполнении, осуществляя переход по указанному адресу, если один из четырех битов условий находится в состоянии «1» или «0».

Условия перехода, которые могут быть использованы этими командами, приведены в таблице.

Условие	Условия перехода	
NZ	Результат операции не равен 0	(Z=0)
Z	Результат операции равен 0	(Z=1)
NC	Переноса не было	(CY=0)
C	Был перенос	(CY=1)
PE	Число единиц в аккумуляторе четно	(P=1)
PO	Число единиц в аккумуляторе нечетно (P=0)	
P	Результат операции положителен	(S=0)
M	Результат операции отрицателен	(S=1)

Команды условного перехода имеют следующий формат:

Jcond Adr

где **cond** - одно из условий перехода перечисленных в таблице;

Adr - адрес перехода, расположенный в следующих двух байтах команды.

Выполнение команд условных переходов происходит следующим образом: если условие перехода истинно, то управление передается по указанному в команде адресу перехода, в противоположном случае выполняется следующая команда. В соответствии с приведенными в таблице условиями перехода в систему команд МП входят следующие команды условного перехода:

JNZ JZ JNC JC JPE JPO JP JM

Команда безусловного перехода, имеющая формат **JMP Adr** осуществляет безусловный переход по указанному адресу.

Команды условного вызова подпрограмм имеют следующий формат:

Ccond Adr

где **cond** - одно из условий перехода, перечисленных в таблице;

Adr - адрес перехода, расположенный в последующих двух байтах команды.

Выполнение команд условных вызовов подпрограмм происходит следующим образом: если условие истинно, то управление передается по указанному в команде адресу перехода, а содержимое **PC** загружается в стек, в противоположном случае выполняется следующая команда.

В соответствии с приведенными в таблице условиями перехода в систему команд МП входят следующие команды условного вызова подпрограмм:

CNZ CZ CNC CC CPE CPO CP CM

Команда безусловного вызова, подпрограммы, имеющая формат **CALL Adr** осуществляет безусловный вызов подпрограммы по указанному адресу.

Команды условного возврата из подпрограммы имеют следующий формат:

Rcond

где **cond** - одно из условий перехода, перечисленных в таблице.

Команда возврата из под программы выполняется следующим образом: если условие возврата истинно, то в **PC** заносится содержимое вершины стека - адрес возврата, в противоположном случае выполняется следующая команда подпрограммы.

В соответствии с приведенными в таблице условиями перехода в систему команд МП входят следующие команды условного возврата из под программы:

RNZ RZ RNC RC RPE RPO RP RM

По команде **RET** осуществляется безусловный возврат из под программы.

Команда перезапуска имеет формат **RST n**

При выполнении этой команды происходит прерывание работы процессора и текущее состояние счетчика команд записывается в стек.

Управление передается команде, адрес которой определяется как произведения числа n на 8.

Команды управления работой МП предназначены для управления работой МП. К ним относятся:

HLT - останов процессоров. Выполнение текущей программы приостанавливается до появления запроса прерывания от устройства ввода-вывода. Состояние РОН и счетчика команд не изменяется;

DI - выключение системы прерывания;

El - включение системы прерываний;

NOP - отсутствие операции. Никаких действий по команде не производится и управление передается следующей команде.

Команды этой группы не изменяют значения битов условий.

Команды работы со стеком предназначены для выполнения следующих операций со стеком:

PUSH RP-запись содержимого регистровой пары в стек (регистровая пара **SP** не может быть использована в данной команде);

POP RP-восстановление содержимого регистровой пары из стека (регистровая пара **SP** не может быть использована в данной команде);

XTHL - обмен содержимого регистровой пары **(H, L)** с вершиной стека;

SPHL - запись содержимого регистровой пары **(H, L)** в указатель стека.

Состояние битов условия не изменяется командами этой группы, за исключением команды **POP PSW,** которая изменяет все биты.

Специальные команды выполняют операции установки, инверсии бита **CY** и инверсию содержимого аккумулятора. К ним относятся следующие команды:

CMA - инверсия содержимого аккумулятора, т. е. разряды, имеющие значение 1, принимают значение 0, и наоборот. Эта команда не изменяет значение битов условий;

CMC - инверсия бита **CY;**

STC - установка бита **CY** в единичное состояние.

Команды ввода-вывода используются для обмена информацией с периферийными устройствами. К ним относятся следующие команды:

IN port - прием данных из порта, определяемого адресом **port;**

OUT port - выдача содержимого аккумулятора в порт, указанный адресом **port.**

Команды ввода-вывода не изменяют состояние битов условий.

Организация подпрограмм. Подпрограмма записывается как последовательность операторов на ассемблере и вызывается по метке первой команды подпрограммы, определяющей адрес начала подпрограммы.

Основной задачей при организации подпрограмм является передача в них параметров.

В простейшем случае передача данных в подпрограмму осуществляется с использованием **РОН.** В других случаях целесообразно передавать данные в подпрограмму, используя для этого список параметров.

При использовании списка параметров все передаваемые подпрограмме параметры записываются в определенную область памяти, адрес которой загружается в пару **(H, L)** и впоследствии используется в подпрограмме.

В качестве примера рассмотрим программу, в которой используется подпрограмма, вычисляющая сумму первых двух параметров и возвращая результат в третьем параметре.

1. Передача параметров через регистры:

START:
LDA SECOND ;прием второго
MOV B; A ;параметра
LDA FIRST ;прием первого параметра
CALL ADDPAR ;определение результата
STA REZULT ;запись результата
..... ;продолжение программы

ADDPAR: ADD B ;вычисление суммы
RET ;возврат из подпрограммы

FIRST: DB 23 ;первый параметр - слагаемое
SECOND: DB 15 ;второй параметр - слагаемое
REZULT: DB 0 ;третий параметр - результат

2. Передача параметров списком параметров:

START: LXI H, FIRST ;занесение адреса списка параметров
CALL ADDPAR ;определение результата
..... ;продолжение программы

ADDPAR:
MOV A.M ;выбор первого параметра
INX H ;переход к след. параметру
ADD M ;вычисление суммы
INX H ;переход к результату
MOV M.A ;запись результата
RET ;возврат из подпрограммы
..... ;продолжение программы

FIRST: DB 23 ;первый параметр-слагаемое
SECOND: DB 15 ;второй параметр-слагаемое
REZULT: DB 0 ;третий параметр-результат

2. Порядок выполнения работы.

А. Составить вызывающую программу и подпрограмму, выполняющие действие согласно варианту задания:

1. Умножение восьмиразрядных двоичных чисел, выполняемое по следующему алгоритму:

1) установить частичную сумму равной 0;

2) установить счетчик циклов равным числу разрядов множимого;

3) если младший разряд множителя равен 1, то прибавить множимое к частичной сумме;

4) сдвинуть множимое арифметически на один разряд влево, а множитель - на один разряд вправо;

5) вычесть 1 из счетчика циклов;

6) если счетчик циклов не равен 0, перейти к п.3; иначе -закончить выполнение программы.

2. Умножение 8-разрядных чисел со знаком.

3. Деление 8-разрядного двоичного числа на 4-разрядное двоичное число, выполняемое по следующему алгоритму:

1) установить счетчик циклов равным числу разрядов делителя
n;

2) установить частное равным 0;

3) сдвинуть делитель арифметически на N разрядов влева;

4) сравнить делимое и модефицированный делитель: если делимое больше, то деление невозможно (частное имеет длину больше N разрядов); иначе

5) сдвинуть делитель арифметически на один разряд вправо, а частное - на один разряд влево;

6) вычесть делитель из делимого; если результат вычитания положителен, то установить младший разряд частного в 1 и перейти к п.8; иначе

7) восстановить делимое, т. е. прибавить к полученному в п. 6 отрицательному остатку делитель; установить младший разряд частного в 0;

8) вычесть 1 из счетчика циклов. Если счетчик циклов не равен О, перейти к п. 5; иначе закончить программу.

Б. Произвести ассемблирование программы вручную и загрузить ее в ОЗУ контроллера.

В. Произвести отладку программы.

3. Содержание отчета.

Отчет по работе должен содержать:

- краткие сведения о командах;
- структурную схему алгоритма программы;
- текст программы с указанием адресов ячеек и кодов программы;
- результаты выполнения программы.

Лабораторная работа № 9. Программируемый адаптер параллельного интерфейса КР580ВВ55

Цель работы: изучение структуры, режимов работы и управления программируемого адаптера параллельного интерфейса ***КР580ВВ55.***

1. Общие сведения.

Для реализации ЭВМ необходимы средства обмена данными в параллельном коде с различными периферийными устройствами. Общность функций ввода-вывода стимулировала разработку БИС периферийных адаптеров, представляющих собой гибкие программные приборы, ориентированные на ввод-вывод. Типичным примером адаптера является БИС программируемого адаптера параллельного интерфейса КР580ВВ55 (в дальнейшем - адаптер).

Подключение периферийного оборудования к адаптеру производится через три двунаправленные 8-битных порта (канала) А, В и С. Порты А и В состоят из 8-разрядных регистров ввода-вывода и могут использоваться для двунаправленной передачи байтов. Порт С может быть разделен на два 4-разрядных регистра, которые могут использоваться совместно с портами А и В или независимо от них для передачи информации или для выдачи и приема сигналов управления.

Порт А и старшие 4 разряда порта С образуют группу А, порт В и младшие 4 разряда порта С - группу В.

Структурная схема адаптера на рис. 1. Чтобы запрограммировать адаптер, достаточно загрузить управляющее слово в регистр управляющего слова **(РУС).** Управляющее слово задает один из трех режимов работы портов:

режим 0 - нестробируемый однонаправленный ввод-вывод (основной режим);

режим 1 - стробируемый однонаправленный ввод-вывод;

режим 2 - стробируемый двунаправленный ввод-вывод. Формат управляющего слова, определяющего режим работы адаптера (слово режима), приведен на рис. 2.

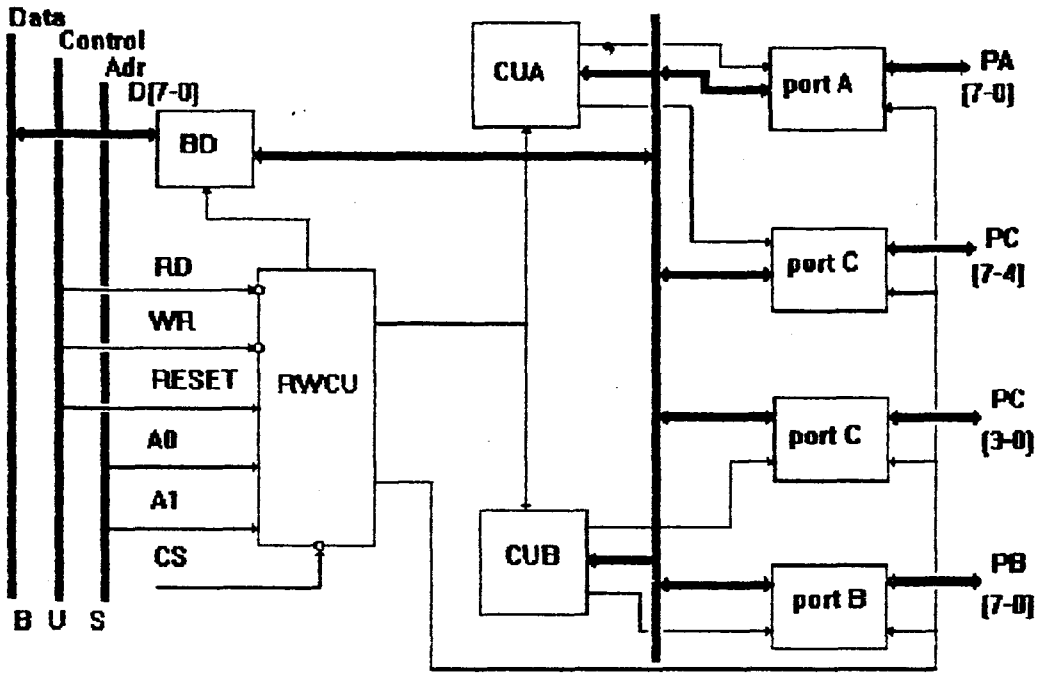

Data
Control
Adr
D[7-0]
BD
RD
WR
RESET
A0
A1
CS
RWCU
B U S
CUA
CUB
port A
port C
port C
port B
PA
[7-0]
PC
[7-4]
PC
[3-0]
PB
[7-0]

рис. 1.

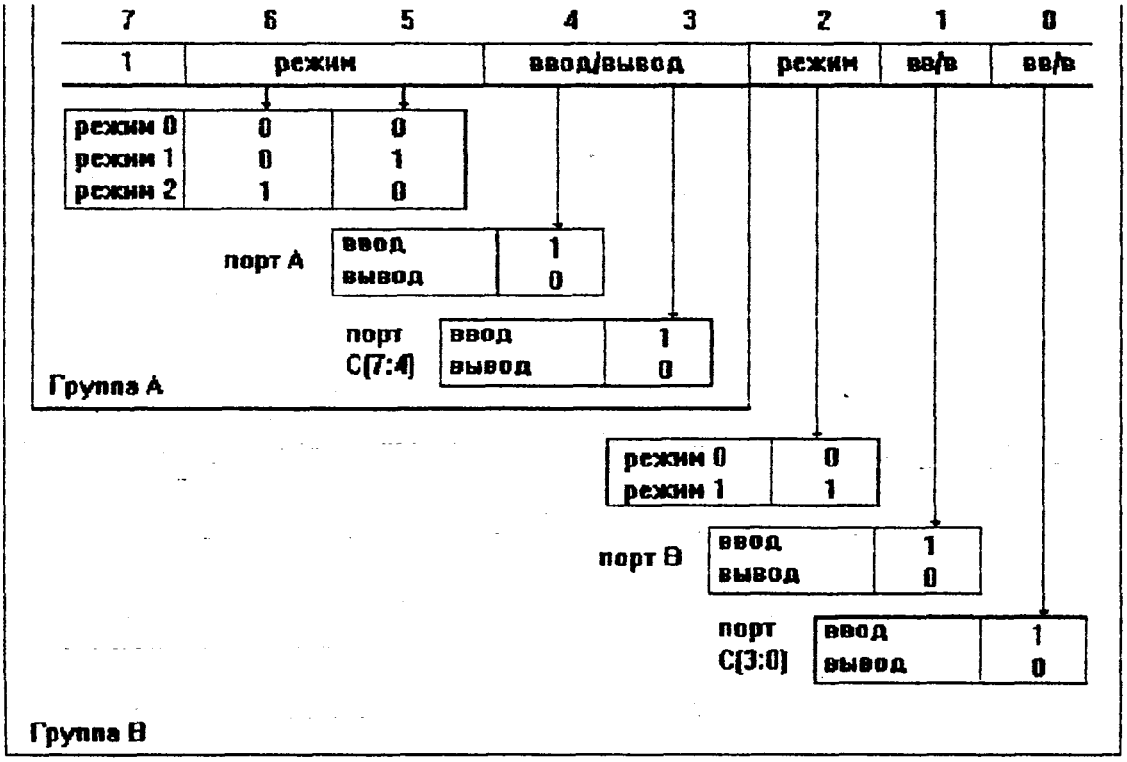

7 6 5 4 3 2 1 0
1
режим
ввод/вывод
режим
вв/в
вв/в
режим 0 0 0
режим 1 0 1
режим 2 1 0
порт А
ввод 1
вывод 0
порт С[7:4]
ввод 1
вывод 0
Группа А
режим 0 0
режим 1 1
порт В
ввод 1
вывод 0
порт С[3:0]
ввод 1
вывод 0
Группа В

рис. 2.

Если в управляющем слове старший бит равен 0, то оно используется для установки или сброса битов порта С. В этом случае разряды 1-3 управляющего слова определяют номер бита порта С, над которым производится операция установки, а бит 0 указывает на тип операции:

0 - очистка бита, указанного разрядами 1-3;

1 - установка бита, указанного разрядами 1-3;

Если нужно изменить значение бита в портах А и В, то эта операция производится в следующей последовательности:

- содержание порта вводится в аккумулятор;

- модифицируется нужный бит;

- слово с модифицированным битом выводится в тот же порт. Рассмотрим подробнее режимы работы адаптера. Основной режим (режим 0). В этом режиме могут работать все 3 порта, причем порт С разделяется на 2 независимых 4-битных порта. В итоге получается четыре порта параллельного ввода или вывода, два из которых 8-битные, а два - 4-битные. Таким образом, адаптер в режиме 0 может иметь одну из 16 возможных конфигураций, определяемых управляющим словом таб. 1.

Таблица 1

Состояние разрядов РУС				Направление передачи данных в портах			
4	2	1	0	А	С(7:4)	в	С(3:0)
0	0	0	0	вывод	вывод	вывод	вывод
0	0	0	1	вывод	вывод	вывод	ввод
0	0	1	0	вывод	вывод	ввод	вывод
0	0	1	1	вывод	вывод	ввод	ввод
0	1	0	0	вывод	Ввод	вывод	вывод
0	1	0	1	вывод	Ввод	вывод	ввод
0	1	1	0	вывод	Ввод	ввод	вывод
0	1	1	1	вывод	Ввод	ввод	ввод
1	0	0	0	ввод	вывод	вывод	вывод
1	0	0	1	ввод	вывод	вывод	ввод
1	0	1	0	ввод	вывод	ввод	вывод
1	0	1	1	ввод	вывод	ввод	ввод
1	1	0	0	ввод	Ввод	вывод	вывод
1	I	0	1	ввод	Ввод	вывод	ввод
1	1	1	0	ввод	Ввод	ввод	вывод
1	1	I	1	ввод	Ввод	ввод	ввод

Режим стробируемого однонаправленного ввода-вывода (режим1). Этот режим предназначен для однонаправленных передач данных, инициируемых прерываниями. Передача данных осуществляется через поты А и В, а линии порта С используются для управления обменом, причем в этом режиме порты А и В могут работать как на ввод, так и на вывод.

Режим стробируемого двунаправленного ввода-вывода (режим 2). В этом режиме порт А используется как 8-разрядная двунаправленная шина данных. Порт В может программироваться для работы в режимах 0 и 1. Разряды порта С используются для управления обменом.

Порядок работы с модулем зависит от конкретных задач, реализующихся на основе функционирования составных частей модуля, описанных в соответствующем разделе. Ниже приводятся возможные направления работы с модулем.

Ввод-вывод информации в режиме 0. Возможен простой (нестробируемый) побитный ввод данных с помощью переключателей устройства ввода канала В или канала А и отображение информации, выводимой в канале А или С, с помощью светодиодов.

Примечание. При выводе информации в канал А необходимо переключить буферы D9, D10 на вывод удерживанием кнопки SA2 или записью " 1 " в разряд D0 канала В.

В качестве примера предлагается программа 1, выполняющая следующие действия:

- программирование канала А на простой вывод, канала В на ввод информации;
- циклический опрос состояния переключателей устройства ввода канала В и вывод считанной из канала В информации в канал А.

Переключатель SA2 в положении **"РАБ",** SA21 - в положении **"АВТ".**

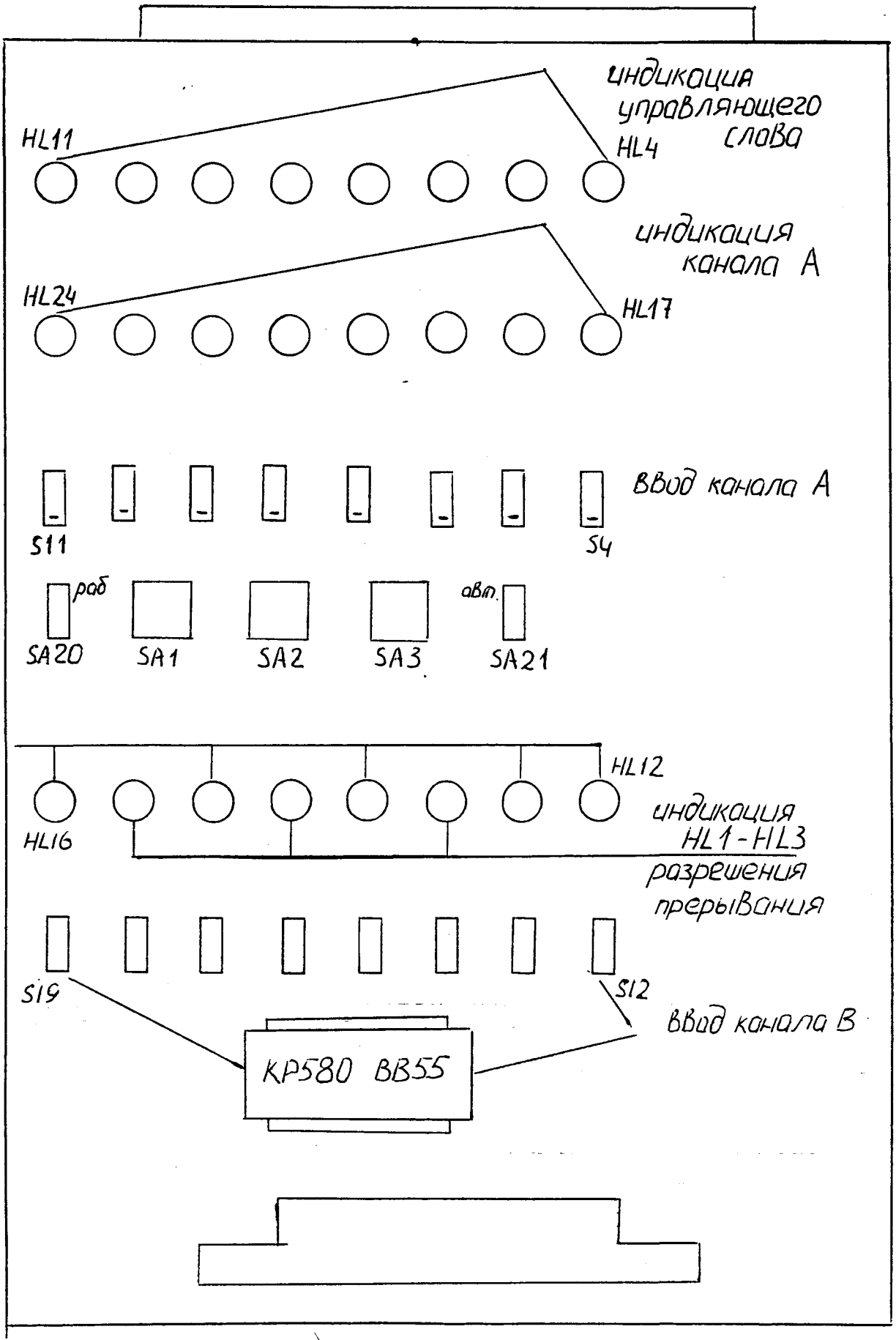

индикация управляющего слова
HL11
HL4
индикация канала А
HL24
HL17
ввод канала А
S11
S4
раб
авт.
SA20
SA1
SA2
SA3
SA21
HL12
HL16
индикация HL1-HL3 разрешения прерывания
S19
S12
ввод канала В
КР580 ВВ55

Рис. 3.

Программа 1.
Работа в режиме 2.

```
0C00 3EOB      LOAD MYI A, CW0 ;запись управляющего
0C02 D3B3      OUT RGCW ;слова в регистр
0C04 D3B1      LOOP IN KB ;чтение данных из KB
0C06 D3B0      OUT КА ;запись данных в КА
OC08 C3000C IMP LOOP ;зацикливание
```

В результате работы программы 1 данные, задаваемые с помощью переключателей устройства ввода канала В, отображаются на светодиодах HL7...HL24.

Ввод-вывод информации в режиме 1. Ввод информации осуществляется через канал В, входные данные задаются с помощью переключателей SA12...SA19 устройства ввода канала В. Вывод информации осуществляется через канал А и отображается на светодиодах HL17.. .HL24.

В качестве примера работы в режиме 1 предлагается программа 2, выполняющая следующие действия:

- программирование канала В на вывод данных в режиме 1, канала А на вывод данных в режиме 1;
- установка триггеров разрешения прерывания каналов А и В**;**
- опрос сигнала IRQ.B и ввод данных по каналу В при наличии логической " 1 " этого сигнала;
- вывод считанной информации в канал А и ожидание установки сигнала IRQ.A;
- зацикливание.

Переключатель SA20 в положении "РАБ", переключатель SA21 - в положение "РУЧ".

Программа 2
Работа в режиме 1.

```
080B 3EAF LOAD: MVI A.CW1 ;запись управляющего
080D D3B3 OUT RGCW ;слова в регистр
080F 3E05 MVI A.SEN0 ;остановка триггера
0811 D3B3 OUT RGCW ;INTEO
0813 3E0D MVI A,CW2 ;остановка триггера
0815 D3B3 OUT RGCW ;INTE2
0817 DBB2 WAIT1:IN КС ;ожидание установки
0819 E601 ANI 1 ;сигнала IRQ.B
081B CA1708 JZ WAIT1 ;
081E DBB1 IN KB ;чтение данных из KB
0820 D3BO OUT КА ;запись данных в КА
0822 DBB2 WAIT2:IN КС ;ожидание установки
0824    E608        ANIB            ;сигнала IRQ.A
0826    CA2208      JZ WAIT2        ;
0829    C31708      JMP WAIT1  ;зацикливание
```

После запуска программы необходимо установить какую-либо кодовую комбинацию с помощью переключателей устройства ввода канала В, далее, в соответствии с временной диаграммой обмена в режиме 1, нажать кнопку SA1 (сигнал STB RC/.B), после чего нажать кнопку SA3 (сигнал АСК WR/.A). На светодиодах HL17...HL24 должна отобразиться установленная кодовая комбинация. Программу полезно выполнять по шагам машинных циклов.

Ввод-вывод информации в режиме 2. В этом режиме в качестве входной информации используется кодовая комбинация, задаваемая с помощью переключателей устройства ввода канала А или В. Информация выводится через устройство индикации канала А.

В качестве примера работы в режиме 2 предлагается программа 3, выполняющая следующие действия:

- программирование канала А на работу в режиме 2 и канала В для ввода данных в режиме 1;
- установка триггеров разрешение прерываний;
- анализ сигналов запроса прерывания **IRQ.A и IRQ.B.** Ввод данных из канала А по сигналу **IRQ.A** или из канала В по сигналу **IRQ.B;**
- запись полученных данных в канал А и ожидание снятия сигнала **IRQ.A;**
- зацикливание.

Переключатель SA20 в положении "РАБ" .переключатель SA21 в положении **"РУЧ".**

Программа 3.
Работа в режиме 2.

```
085E 3EC6 LOAD: MVI A,CW ;запись управляющего
085C D3B3 OUT RGCW ;слова в регистр
085E 3E05 MVI A, SET0 ;установка триггера
0860 D3B3       OUT RGCW ;INTEO
0862 3E09 MVI A,SET1 ;установка триггера
0864 D3B3 OUT RGCW ;INTE1
0866 3EOD MVI A.SET2 ;установка триггера
0868 D3B3 OUT RGCW ;INTE2
086A DBB2 TESTB: IN КС ;ожидание установки
086C E601 ANI 1 ;сигнала IRQ.B
086E CA7608 JC TESTA ;
0871 DBB1 IN KB ;чтение данных из KB
0873 C37F08 JMP WRITE ;переход к записи в КА
0876 DBB2 TESTA: IN КС    ;ожидание установки
0878 E620 ANI 20H     ;сигнала IRQ.A
087A CA6A08, JZ TESTB     ;
087D DBB0 IN KA       ;чтение данных из КА
087F D3B0 WRITE: OUT KA          ;запись данных в КА
0881 DBB2 WAIT2: IN RC     ;ожидание снятия
0883 E608 ANI 8        ; сигнала IRQ.A
0885 CA8108 JZ WAIT          ;
0888 C36A08 JMP TESTB     зацикливание
```

После запуска программы необходимо установить 2 различных кодовых комбинации с помощью переключателей устройства ввода каналов А и В. Нажатием кнопки SA1 (сигнал STB RC/.B) считывается комбинация УВ канала В, после чего считанную информацию можно ввести в канал А с помощью кнопки SA3 (сигнал АСК WR/.A). Нажатием кнопки SA2 (сигнал STB RC/.A) считывается комбинация УВ канала А, которая также может быть выведена в канал А с помощью кнопки SA3.

2. Порядок выполнения работы:

1. Составить схемы алгоритмов приведенных программ.
2. Набрать программы на УПМК-80.
3. Продемонстрировать работу интерфейса преподавателю и объяснить алгоритм функционирования.

3. Содержание отчета:

1. Краткие сведения о параллельном интерфейсе.
2. Программы, схемы алгоритмов.
3. Выводы и пояснения.

Лабораторная работа № 10 Программируемый последовательный интерфейс КР580ВВ51

Цель** работы: изучение структуры, режимов работы и принципов программирования универсального синхронно-асинхронного приемопередатчика **КР580ВВ51.

1. Общие сведения.

В некоторых случаях обмен данными между мини- и микро ЭВМ осуществляется в последовательном формате. Для реализации интерфейса паралльного процессора мини- или микро ЭВМ с периферийным оборудованием, имеющим последовательный интерфейс, используются программируемые БИС, которые называются программируемым связным интерфейсом, универсальным приемопередатчиком или адаптером последовательной связи.

На практике для последовательного обмена информацией может быть использован один из двух режимов: асинхронный или синхронный.

В асинхронном или старт-стопном режиме каждый символ передается автономно в любой произвольный момент времени. Передача начинается со стартового бита, за которым следует от 5 до 8 бит самого символа, которые оканчиваются необязательным битом контроля на четность (нечетность).

Передача заканчивается одним или двумя битами. Скорость передачи измеряется либо числом символов в секунду, либо числом битовых посылок в секунду.

Синхронная передача символа начинается с одного или двух символов синхронизации, после которых последовательно без всяких разделителей передаются 5 - 8-битные коды символов с необязательными контрольными битами.

В обоих режимах передачи необходимо контролировать правильность передачи символа по битам паритета, если они есть, выдерживать необходимые временные соотношения, а для асинхронного режима, кроме этого необходимо выдерживать установленный формат символа.

Наиболее широкое распростропспис для реализации последовательного интерфейса получили универсальные синхронно-асинхронные приемопередатчики **(УСАПП),** примером которых может служить **БИС КР580ВВ51**, называемая - для краткости адаптером последовательнного интерфейса.

Адаптер предназначен для автоматического параллельно-последовательного при передаче и последовательно-параллельного при приеме преобразования форматов символов. Адаптер обеспечивает одновременную одностороннюю связь процессора с периферийными устройствами за счет возможности работы как в полудуплексном, так и дуплексном режимах; кроме этого, адаптер формирует и воспринимает сигналы управления модемом.

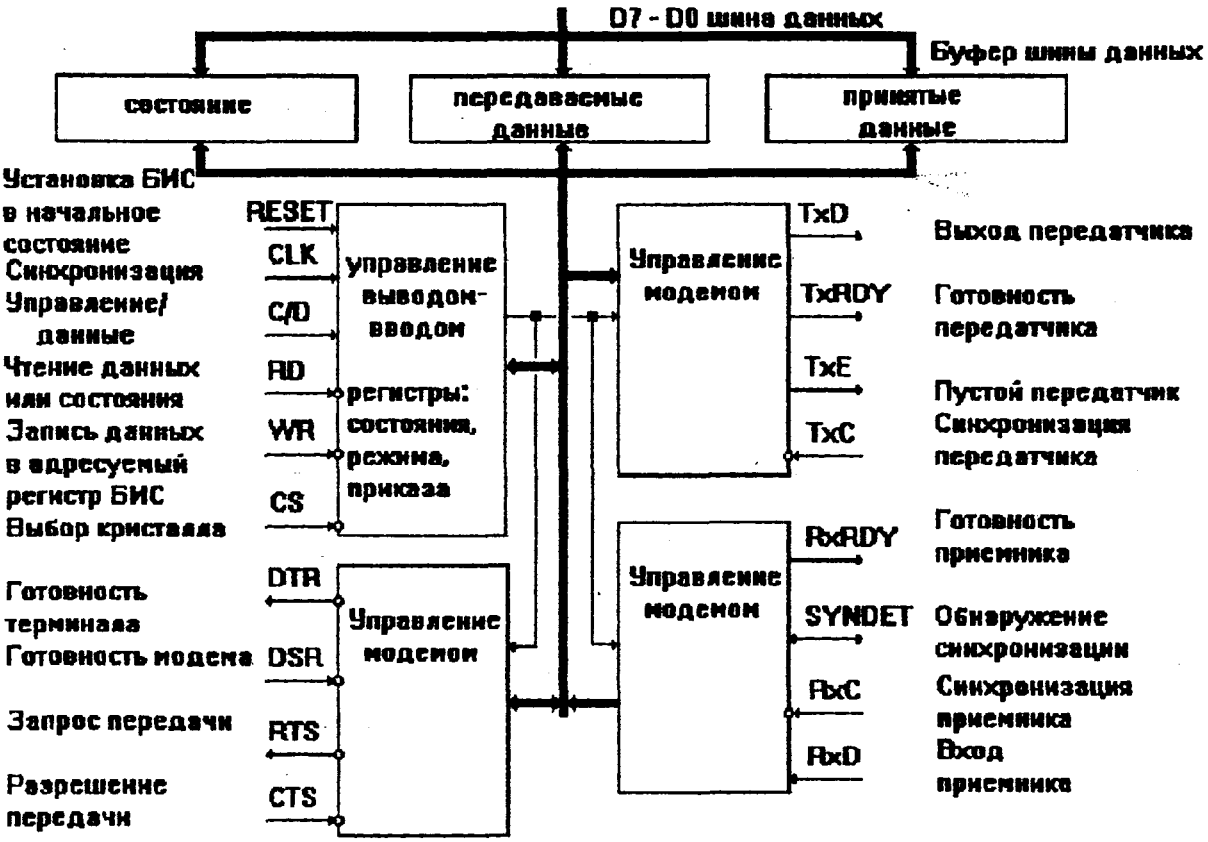


Рис. 1. Функциональная схема последовательного интерфейса.

Рассмотрим подробно отдельные узлы и сигналы адаптера КР580ВВ51.

Параллельный 8-битный двунаправленный буфер шины данных.

- предназначен для передачи собственно данных, управляющих слов и информации состояния. Обмен с буфером, а следовательно, прием или передачаинформации происходит по командам ввода IN или вывода OUT.

Схема управления.

- воспринимает сигналы с шины управления и вырабатывает внутренние управляющие сигналы. В состав схемы управления входят регистры слова команды, в которых хранятся управляющие слова, определяющие режим работы ада1 пера.

Узел передатчика.

- включает собственную схему управления и предназначен для выполнения функций, определяющих передачу данных в последовательном коде. Передатчик принимает от процессора данные в параллельном коде, автоматически формирует служебные биты и символы синхронизации и выдает информацию в последовательном коде на выход.

Узел приемника.

- включает собственную схему управления и предназначен для приема данных в последовательном коде, преобразования их в параллельный код с одновременным исключением служебных бит и передачи принятого символа в процессор.

Программирование адаптера сводится к загрузке в него нескольких управляющих слов, которые определяют:

- скорость передачи;
- длину символа;
- число стоповых бит;
- режим работы;
- наличие контроля и его вид.

Кроме этого, при синхронном режиме адаптера необходимо задать тип синхронизации (внутренняя или внешняя) и один или два символа синхронизации. После программирования адаптер готов к работе.

Управляющие слова, определяющие режим работы адаптера, должны загружаться в него сразу после операции сброса.

Два формата управляющего слова: формат слова режима и формат слова команды.

Слово режима задает общие характеристики адаптера и загружается первым. После слова режима для случая синхронной работы загружается один или два символа синхронизации. Последним в адаптер загружается слово команды, определяющее конкретные действия адаптера. Слово режима имеет четыре поля и загружается во внутренний регистр слова режима.

Следует отметить, что биты 6 и 7 слова в зависимости от режима работы имеют различный смысл. В асинхронном режиме они определяют число стоповых бит, а в синхронном режиме управляют процессом синхронизации.

Слово команды задает операцию, выполняемую адаптером и содержит информацию о разрешении передачи или приема символа, сброс ошибок, управление модемом и т.д.

При работе адаптера в некоторых случаях возникает необходимость проанализировать его состояние. Состояние адаптера фиксируется в слове состояния и может быть считано в любой момент времени с помощью команда IN.

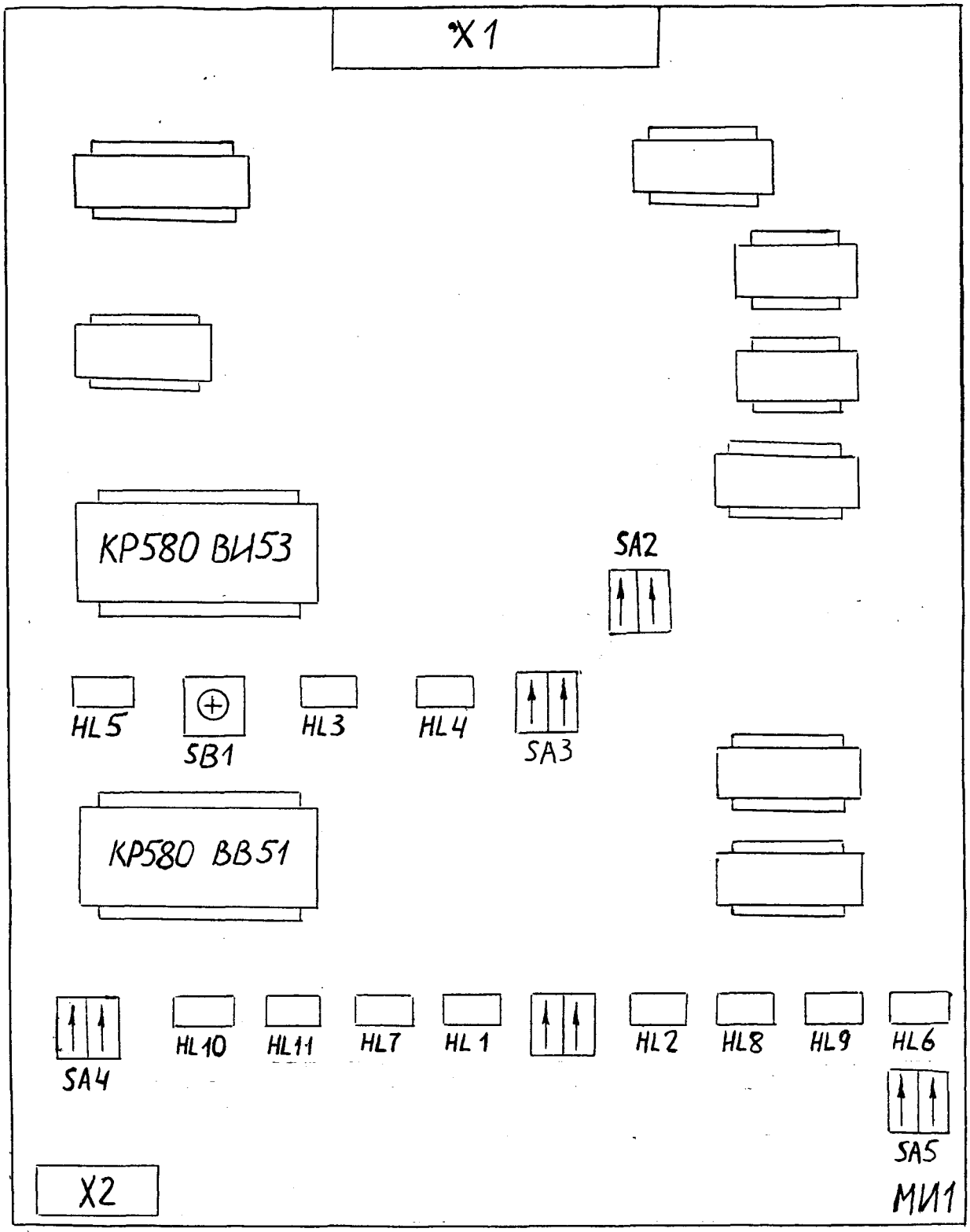


Рис. 2. Расположение элементов.

Бит	Функция	Название и содержание бита
0	TxEN - разрешение передачи	TxRDY - готовность передатчика
1	DIR- готовность	RxRDY - готовность приемника
2	RxE- разрешение приема	TxE- передатчик пуст
3	SBRK - разрыв	PE- ошибка паритета
4	ER - сброс ошибок	OE- ошибка переполнения
5	RTS - запрос передачи	FE- ошибка кадра
6	IR - внутренний сброс	SYNDEN
7	EX - поиск символа синхронизации	DSR

Бит OE устанавливается в единицу, если процессор вовремя не считал символ из адаптера.

Бит FE устанавливается в единицу, если в асинхронном режиме в конце символа не обнаружен стоп-бит.

Все биты ошибок сбрасываются, если бит ER слова команды установлен в единицу. Возникновение любой ошибки не вызывает останова адаптера.

2. Порядок выполнения работы.

Для того, чтобы визуально наблюдать передачу данных на индикаторах (HL6 и HL1), требуется низкая частота (желательно ниже 10 Гц). Поэтому нужно запрограммировать интервальный таймер так, чтобы он делил входную тактовую частоту - 2,5 МГц. Интервальный таймер имеет 6 режимов. Для делителя частоты прямоугольных импульсов используем 3-й режим - генераторный. Программа, осуществляющая деление частоты будет выглядеть следующим образом:

0800 MVI A, 96 3E; Ввод управляющего
0801 96; слова в аккумулятор
0802 OUT 9B D3; Пересылка управляю-
0803 9B; щего слова в регистр
0804 MVI A, 00 3E; Ввод числа в
0805 00; аккумулятор
0806 OUT 9A D3**;** Пересылка числа в
0807 9A; счетчик
0808 END CF; Конец программы

Эта программа позволяет снизить частоту до 17 Гц. Таймер программируется на передатчике.

Для того чтобы данные передать с одной машины на другую, одну из двух машин нужно запрограммировать на передачу. Программируется та машина, на которой до этого был запрограммирован таймер. Программа, осуществляющая передачу данных (в данном случае передается число $AA_{16}=10101010$ будет выглядеть следующим образом:

```
0809    MVI A, 40    3E;    Сброс микросхемы в
080A                 40;    исходное состояние
080B    OUT 9D       D3;
080C                 9D;
080D    MVI A, 4E    3E;    Ввод слова режима в
080E                 4E;    аккумулятор
080F    OUT9D        D3;    Пересылка слова
0810                 9D;    режима в регистр
0811    MVIA.01      3E;    Ввод слова команды в
```

```
0812                  01;   аккумулятор
0813   OUT9D          D3;   Пересылка слова
0814                  9D;   команды в аккумулятор
0815   MVIA,AA        3E;   Ввод передаваемого
0816                  AA;   байта в аккумулятор
0817 OUT9C   D3;     Пересылка передава-
0818                  9C;   емого байта в регистр
0819 END      CF;    Конец программы
```

После программирования передатчика нужно включить следующие ключи:

- SA1.1 и SA1.2 в положение вкл. (вверх);
- SA2.2 в положение вкл.;
- SA4.1 и SA4.2 в положение выкл.

Затем запускаем таймер. Индикатор HL2 должен моргать с частотой 17Гц. Потом запускаем передатчик. Включаем ключ SA4.1. Индикатор HL6 должен сигнализировать о работе передатчика.

Включаем вторую машину. Соединяем обе машины шнуром через разъемы X2.

На второй машине должна моргать лампочка HL2, сигнализирующая о том, что обе машины синхронизированы частотой 17 Гц.

Далее программируем **УСАПП** второй машины на прием. При этом ключи должны находиться в следующем положении:

- SA 2.1 и SA 2.2 в положении выкл.
- SA4.1 и SA4.2 в положение выкл**.**
- SA1.1 и SA 1.2 в положение вкл.:

Программа, осуществляющая прием:

```
0800 MVI A, 40 3E; Сброс микросхемы в
0801           40; исходное состояние
0802 OUT 9D    D3;
0803           9D;
0804 MVI A, 4E 3E; Ввод слова режима в
0805 4E; аккумулятор
0806   OUT9D          D3;   Пересылка слова
0807                  9D;   режима в регистр
0808   MVI.A,04       3E;   Ввод слова команды в
0809                  04;   аккумулятор
080A OUT9C   D3;     Пересылка слова
080B           9C;   команды в регистр
080C END       CF;   Конец программы
```

Далее УСАПП программируется на считывание информации. Программа выглядит следующим образом:

```
080D IN 9C DB; Считывание переданного
080E 9C; байта в аккумулятор
080C END CF; Конец программы
```

После того как УСАПП будет запрограммирован на прием и на считывание, на первой машине запускается программа на передачу. Индикаторы HL1 и HL6 синхронно моргают, сигнализируя о том, что информация передается и принимается. HL6 - на передатчике, HL1 -на приемнике.

Затем на приемнике просматриваем аккумулятор, там должен находиться переданный байт.

3. Содержание отчета.

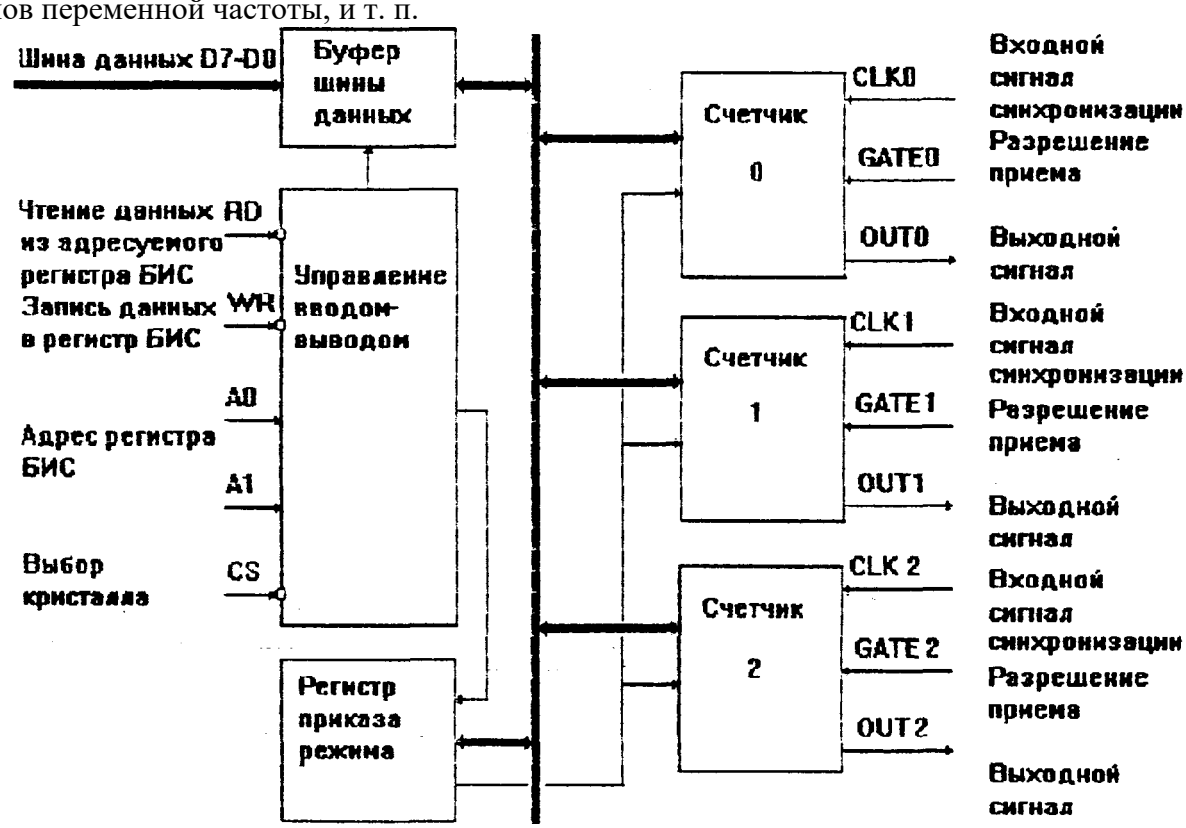
1. Краткие сведения о последовательном интерфейсе.
2. Программы, схемы алгоритмов.
3. Выводы и пояснения.

Лабораторная работа № 11. Программируемый интервальный таймер КР580ВИ53

Цель работы: изучение структуры и режимов работы программируемого интервального таймера ***КР580ВИ53.***

1. Общие сведения.

Программируемый интервальный таймер КР580ВИ53 предназначен для реализации таких распространенных управляющих функций микроЭВМ, как формирование временных интервалов, подсчет числа внешних событий, генерации сигналов переменной частоты, и т. п.



В состав БИС КР580ВИ53 входят три независимых 16-разрядных счетчика, работающих на вычитание в двоичном или двоично-десятичном коде. Сигналы тактового генератора или подсчитываемые сигналы из ВУ подаются на вход счетчиков CLK0, CLK1, CLK2. Прием сигналов разрешается внешними управляющими сигналами GATE0, GATE1, GATE2 соответственно. Как только содержимое какого-либо счетчика становится равным нулю, вырабатывается один из выходных сигналов OUT0, OUT1 или OUT3, которые используются как запросы на прерывание.

Инициализация каждого счетчика производится записью управляющего слова в регистр приказа. При этом устанавливаются начальное значение счетчика и один из ниже следующих шести режимов его работы - условий формирования входного сигнала OUT.

Режим 0 - сигнал на выходе OUT формируется по окончании счета (программируемая задержка),

Режим 1 - программируемый одновибратор, т.е. формирование на выходе **OUT** сигнала длительностью N периодов сигнала **CLK.**

Режим 2 - генератор программируемой частоты, т. е. на выходе OUT формируется периодический сигнал с высоким уровнем в течении **N-1** периодов сигнала CLK и с низким - в один период сигнала CLK.

Режим 3 - генератор прямоугольных импульсов со скважностью 2, т. е. на выходе OUT формируется сигнал высокого уровня в

течении — N+1 (N - четное) или (N+1)/2 (N - нечетное) периодов сигнала **CLK.**

Режим 4 - формирование одиночного строба с программным запуском, т. е. на выходе **OUT** формируется одиночный сигнал длительностью в один период сигнала **CLK.** Начало счета инициируется загрузкой в счетчик N.

Режим 5 - формирование одиночного строба с программным запуском аналогично режиму 4, по начало счета инициируется фронтом сигнала **GATE.**

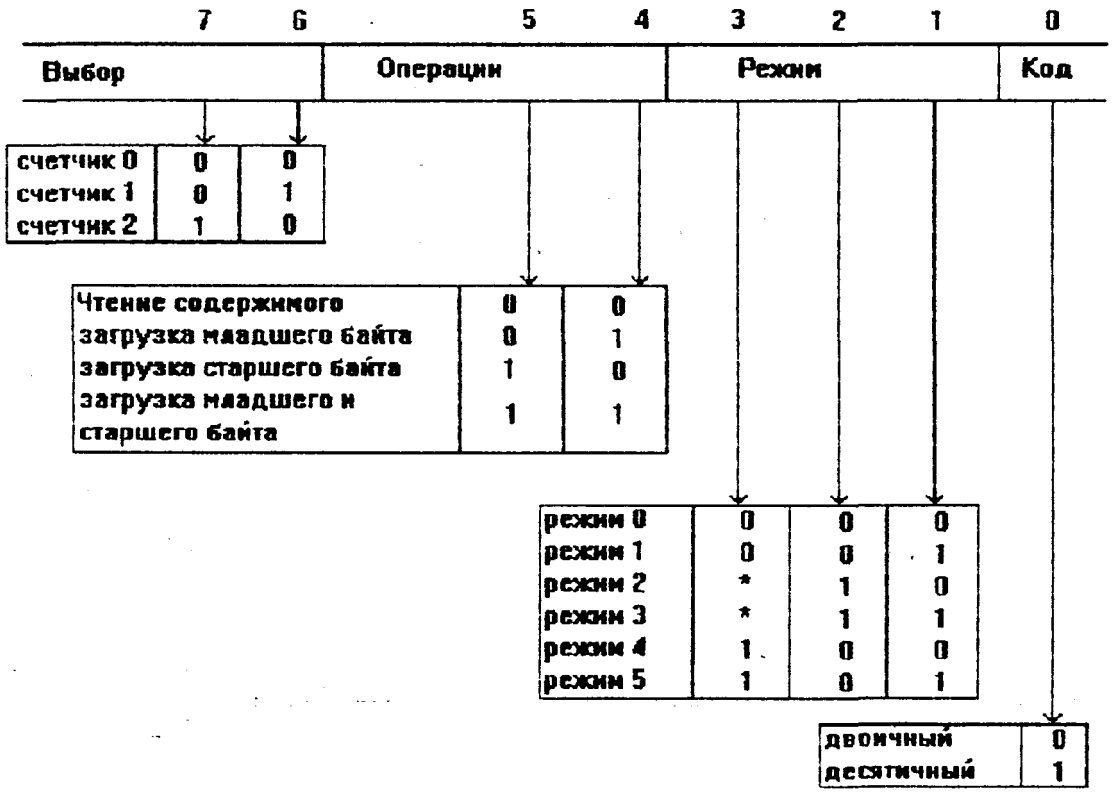
Обмен информацией между системным интерфейсом микроЭВМ и регистрами БИС осуществляется по шине данных D7-DO с помощью управляющих сигналов RW, WR, CS. При этом внутренние регистры БИС адресуются по линиям АО и А1.

Считывание содержимого любого счетчика для его последующего анализа осуществляется двумя способами.

Первый - реализуется обычными командами IN, в которых указывается порт, соответствующего счетчика. Считывание содержимого счетчика осуществляется в соответствии с операцией, указанной в разрядах 4-5 управляющего слова. Недостаток этого способа заключается в том, что на время считывания необходимо прекращать работу счетчика.

Второй - "считывание на ленту", не нарушает работы счетчика, но требует для своей реализации предварительной загрузки в регистр режима соответствующего управляющего слова.

Ре-жим	Состояние входа GATE		
	Низкий уровень от-рицательный фронт	Положительный фронт	Высокий уровень
0	Запрещает счет		Разрешает счет
1		Инициирует счет сбрасывает OUT в следующем такте	
2	Запрещает счет ус-танавливает OUT=1	Инициирует счет	Разрешает счет
3	Запрещает счет ус-танавливает OUT= 1	Инициирует счет	Разрешает счет
4	Запрещает счет		Разрешает счет
5		Инициирует счет	



2. Порядок выполнения работы.

Для запуска таймера необходимо загрузить в регистр режима управляющее слово и дописать в него начальное значение n. При работе с **КР580ВИ53** используются следующие адреса:

- таймер 0 - 98;
- таймер 1 - 99;
- таймер 2 - 9A;
- рус - 9B.

Для того, чтобы запрограммировать 0-ой счетчик на работу в режиме 0 с n=4, необходимо записать следующую программу:

0800 MVI A, 10; Запись управляющего слова в РУС

0802 OUT, 9B;

0804 MVI A, 04; Загрузить в таймер 0 начальное
0806 OUT 98; значение n
0808 KST1;

Для выполнения работы использовать карту параллельного интерфейса **МИ - 1,** (см. лабораторную работу № 6) на которой:

SB1 - кнопка подачи входных импульсов;
HL5 - индикатор OUT;
HL3 - индикатор входных импульсов;
HL4 - индикатор сигнала GATE;
SA31 - включение сигнала GATE.

3. Задание на выполнение работы.

1. Режим 1, n=7;
2. Режим 2, n=ll;
3. Режим 3, n=5;
4. Режим 4, n=3;

4. Содержание отчета.

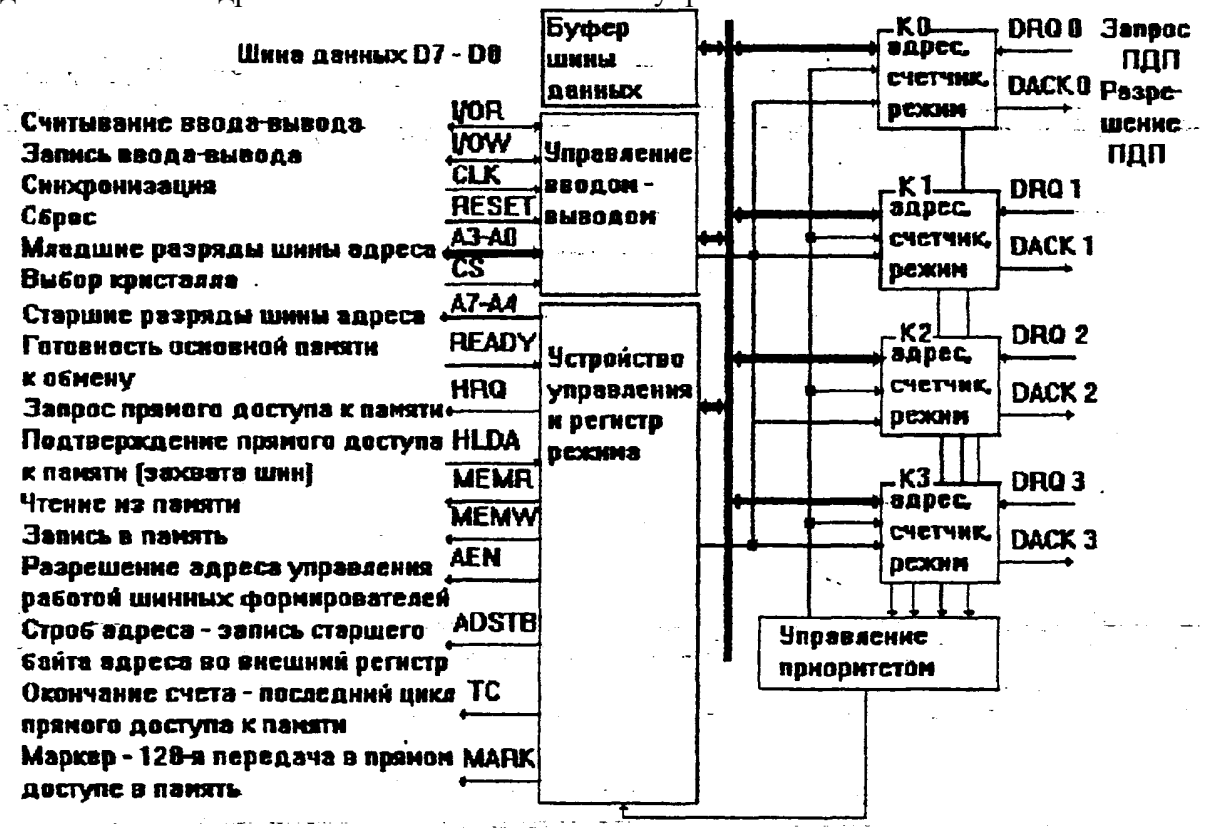
Отчет должен содержать:

1. Краткие характеристики, назначение и устройство программируемого интервального таймера.
2. Программа задания режима.
3. Графики работы таймера по программам.

Лабораторная работа № 12. Программируемый контроллер прямого доступа к памяти КР580ВТ57

Цель работы: изучение структуры и режимов работы программируемого контроллера прямого доступа к памяти ***КР580ВТ57.***

Программируемый контроллер прямого доступа к памяти КР580ВТ57 предназначен для управления обменом данными между ВУ и основной памятью **микроЭВМ** в режиме прямого доступа к памяти (ПДП). Контроллер управляет процессом предоставления прямого доступа к памяти, формирует в процессе обмена последовательность адресов ячеек памяти и сигналы управления обменом.



В БИС КР580ВТ57 реализованы четыре независимых канала ПДП К0, ..., К3. Управление работой каждого канала ПДП осуществляется с помощью двух 16-разрядных регистров: регистра начального адреса и регистра управления. В регистр начального адреса при программировании БИС заносится начальный адрес передаваемого массива данных. В 14 младших разрядах регистра управления размещается счетчик, в который заносится число на единицу меньше длины передаваемого массива данных, т. е. размер массива не может превышать 16Кбайт. 15-й и 14-й разряды регистра управления определяют тип операции обмена:

00 - контроль;

01 - запись в память;

10 - чтение из памяти;

11 - запрещенное состояние.

Связь контроллера с системным интерфейсом микроЭВМ осуществляется по шине данных **D7...DO** через двунаправленный буфер шины данных. Запись информации в регистры начального адреса и управления каналов контроллера производится программным путем. Адресуются внутренние регистры контроллера по линиям А3...АО.

Помимо указанных регистров в состав контроллера входят: регистр режима, определяющий общие функции контроллера, в т. ч. и приоритеты каналов **ПДП;**

регистр состояния, отображающий состояние окончания счета (сигнал **ТС),** т. е. окончание передачи массива данных по соответствующим каналам **ПДП.**

Приоритеты входных запросов на предоставление прямого доступа устанавливаются программно записью информации в регистр приказов: либо фиксированные **(DRQ0** - высший, **DRQ3** -низший), либо циклические, т. е. последнему обслуженному запросу присваивается низший приоритет, а приоритеты остальных запросов изменяются в круговой последовательности.

Для формирования 16-ти разрядного адреса ячейки памяти на шине адреса системного интерфейса контроллеру прямого доступа требуется внешний 8-разрядный регистр. В этот регистр в начале цикла ПДП контроллер с использованием строба ADSTB по шине данных D7...DO записывает восемь старших разрядов адреса A15...A8. Выходы внешнего регистра подключаются к линиям A15...A8 шины адреса системного интерфейса микроЭВМ, а младшие разряды адреса формируются на выходах A7...AO контроллера.

Порядок выполнения работы.

Назначение переключателей.

Клавиша SA9.

При нажатии клавиши SA9 формируется бездребезговый сигнал записи информации с набранного поля SA1...SA8 в буферный регистр. По спаду этого сигнала, то есть при отпускания клавиши SA9 формируется сигнал запроса прямого доступа памяти по каналу DRQ2, который поступает на КПДП. Если КПДП запрограммирован для работы с каналом ПДП2, то запрос будет воспринят и контроллер перейдет к циклу обмена.

Клавиша SA10.

При нажатии клавиши SA10 формируется бездребезговый сигнал запроса ПДП по каналу DRQ3. Если КПДП запрограммирован для работы по каналу ПДПЗ, то запрос будет воспринят и контроллер перейдет к выполнению цикла обмена по каналу ПДПЗ.

Переключатель SA12.

Если переключатель SA12 разомкнут (в левом положении), то контроллер, после того как воспримет сигнал запроса ПДП и на светодиодах HL9...HL12 отразится сигнал подтверждения ПДП по соответствующему каналу, он перейдет в режим ожидания (сигнал готовности на входе КПДП будет сброшен).

Для завершения цикла обмена необходимо подать сигнал готовности на вход КПДП путем нажатия клавиши SA11.

При замкнутом переключателе SA12 сигнал готовности на входе КПДП будет всегда высокого уровня и БИС КПДП не будет переходить в состояние ожидания, а цикл обмена произойдет "мгновенно".

Переключатель на центральной плате SA2.

При прогаммировании КПДП на центральной плате микропроцессора КР580ВМ80А, НЕОБХОДИМО переключить переключатель SA2 в следующие положения:

SA2-2 - нижнее;

SA2-1 -верхнее;

SA2-3 - верхнее;

SA2-4 - нижнее.

Программирование **КПДП** для приема данных по каналу **DRQ2.**

При этом данные с набранного поля SA1...SA8 передаются в ячейку памяти с адресом (в данном случае) **0900H.**

0800 3E 00 MVI A.00; запись младшего байта адреса в PA2

0802 D3 D4 OUT D4;

0804 3E 09 MVI A,09; запись старшего байта адреса в PA2

0806 D3 D4 OUT D4;

0808 3E 01 MVI A,01; запись младшего байта в регистр

080A D3 D5 OUT D5; СК2
080C 3E 40 MVI A,40; запись старшего полубайта в регистр
080E D3 D5 OUT D5; СК2 и загрузка РЖК
0810 3E 44 MVI A.44; запись управляющего слова в РУС
0812 D3 D8 OUT D8;
0814 CF RST 1 ;

Если в адресе 0810 записано УС 44, то передается один блок данных.

Если в адрес 0810 записать УС 04, то возможно передать последовательно несколько блоков данных без перепрограммирования КПДП. Для этого достаточно ввести новые данные и подать сигнал запроса ПДП. При каждом следующем цикле обмена адрес ячейки памяти увеличивается на единицу.

Программирование **КПДП** для передачи данных по каналу **DRQ3.**

При этом данные из ячейки памяти (в данном случае) с адресом 09 ЮН передаются на светодиоды **(HL13...HL20)** индикации данных выходного устройства.

0800 3E 10 MVI A, 10;запись младшего байта в РА3
0802 D3 D6 OUT D6;
0804 3E 09 MVI A,09;запись старшего байта в РА3
0806 D3 D6 OUT D6;
0808 3E 01 MVI A,01;запись младшего байта в СК2
080A D3 D7 OUT D7;
080C 3E 80 MVI A,80;запись старшего байта в **СК2**
080E DE D7 OUT D7;
0810 3E 48 MVI A,48;запись управляющего слова
0812 D3 D8 OUT D8;
0814 CF RST 1 ;

Если в адресе 0810 записано **УС** 48 , то передаётся один блок данных.

Если в адрес 0810 записать УС 08 , то возможно передавать последовательно несколько блоков данных без перепрограммирования **КПДП.**

Программирование КПДП для приема и передачи данных по каналам **DRQ2 и DRQ3.**

После завершения цикла обмена по одному каналу, контроллер может сразу приступить к обмену по другому каналу без перепрограммирования.

0800 3E 00 MVI A,00;запись младшего байта в РА2
0802 D3 D4 OUT D4;
0804 3E 09 MVI A,09;запись старшего байта в РА2
0806 D3 D4 OUT D4;
0808 3E 01 MVI A,01;запись младшего байта в СК2
080A D3 D5 OUT D5;
080C 3E 40 MVI A,40;запись старшего байта в СК2
080E D3 D5 OUT D5;
0810 3E 10 MVI A, 10;запись младшего байта в РА3
0812 D3 D6 OUT D6;
0814 3E 09 MVI A,09;запись старшего байта в РА3
0816 D3 D6 OUT D6;
0818 3E 01 MVI A,01;запись младшего байта в СК3
082A D3 D7 OUT D7;
082C 3E 80 MVI A,80;запись старшего байта в СК3
082E D3 D7 OUT D7;
0830 3E 5C MVI A,5C;запись управляющего слова
0832 D3 D8 OUT D8;
0834 CF RST 1 ;

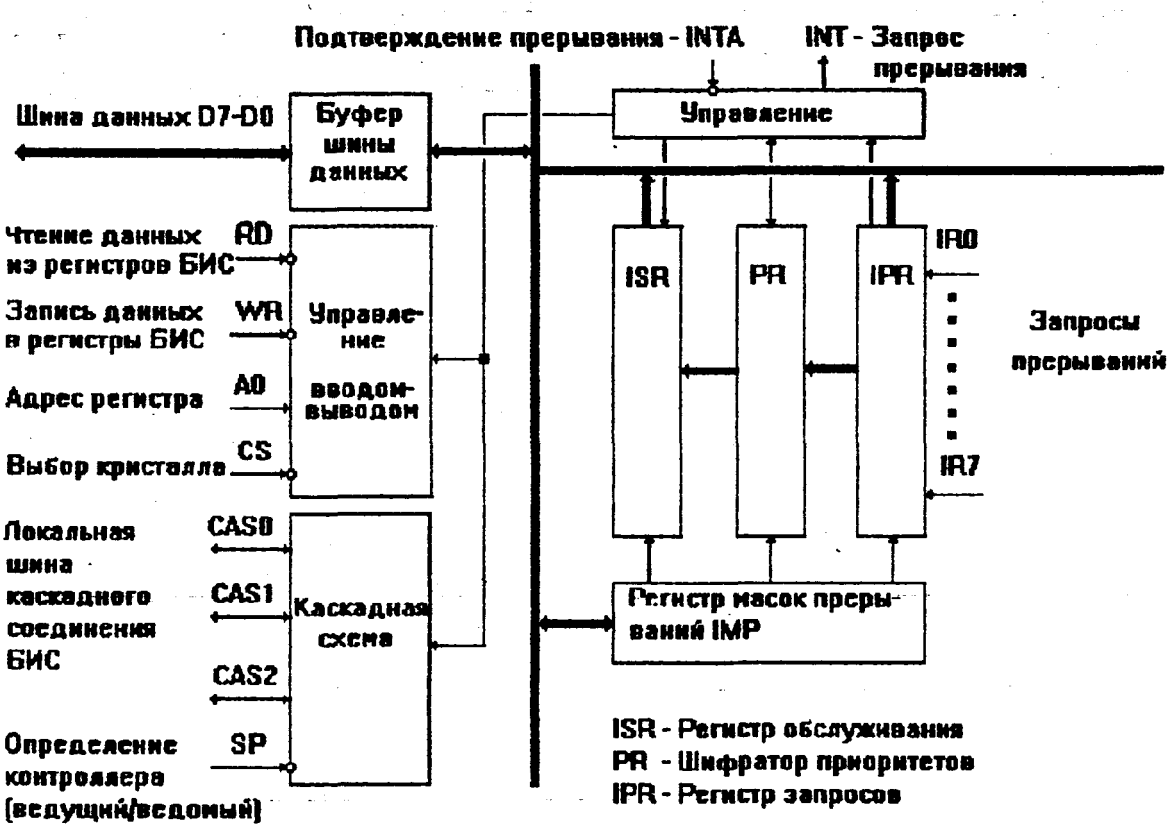
Все прогаммы выполняются при замкнутом и при разомкнутом переключателе SA12.

Содержание отчета.
1. Краткие сведения о контроллере прямого доступа к памяти.
2. Программы и схемы алгоритмов.
3. Выводы и пояснения.

Лабораторная работа № 13. Программруемый контроллер прерываний КР580ВН59

Цель работы: изучение структуры и режимов работы программируемого контроллера прерываний КР580ВН59.

Программируемый контроллер прерываний КР580ВН59 используется для организации в микроЭВМ многоуровневой векторной системы прерываний. Одна БИС КР580ВН59 обеспечивает прием и обработку восьми сигналов прерываний. Возможно совместное использование до восьми БИС, т. е. увеличение числа сигналов прерываний до 64. Любой сигнал или группа сигналов прерываний могут быть программно запрещены (замаскированы).



Связь БИС с системным интерфейсом микроЭВМ осуществляется через двунаправленный буфер шины данных по линиям D7...DO с использованием управляющих сигналов RD, ER, AO и CS. Через буфер данных в процессор передаются содержимое регистра состояния БИС и команда CALL, вызывающая переключение процессора на выполнение подпрограммы обработки сигнала прерывания с наивысшим приоритетом. Процессор передает в БИС управляющие слова - приказы, которые настраивают БИС на один из режимов работы.

Вложенные прерывания. Каждому входу запроса прерывания IR7...IRO присваивается фиксированный приоритет в порядке изменения номеров: IRO - наивысший приоритет, IR7 - низший приоритет. Запрос прерывания с большим приоритетом прерывает обслуживание прерываний с меньшим приоритетом.

Циклический приоритет. Каждому входу запроса прерывания IR7...IRO присваивается приоритет, как и в предыдущем случае, но после обслуживания какого-либо запроса прерывания приоритеты всех входов изменяются в циклическом порядке, так что последнему обслуженному входу присваивается низший приоритет.

Адресуемые приоритеты. Этот режим аналогичен режиму циклических приоритетов с той разницей, что вход, которому присваивается низший приоритет, задается программно.

Режим опроса. В этом режиме прерывания процессора запрещены, а идентификация запросов прерывания производится после считывания в процессор состояния БИС.

Для всех режимов прерывания идентификация сигналов прерывания, поступающих в процессор через БИС КР580ВН59, производится с помощью вектора прерывания - адреса начала подпрограммы обработки соответствующего сигнала прерывания. Старшие разряды (A15...A5) вектора прерывания, общие для всех входов IR7...IRO контроллера, передаются в него приказом инициализации, а младшие разряды, индивидуальные для каждого входа, формируются непосредственно в контроллере. В ответ на сигнал запроса прерывания INT процессор выдает сигналы подтверждения INTA. Сигнал INTA заставляет контроллер выдавать на шину данных код операции команды CALL и затем в ответ на два дополнительных сигнала процессора INTA выдать два байта вектора прерывания. Процессор, получив от контроллера вектор прерывания, переключается на подпрограмму обработки-прерывания.

Совместное использование нескольких БИС программируемого контроллера прерывания обеспечивается схемой каскадирования, с помощью которой можно объединить до 8 БИС. Выбор БИС, требующий обслуживание, осуществляется по линиям CAS2...CAS0, объединяющим все контроллеры, а сигнал SP используется для определения контроллера как ведущего или ведомого.

Порядок выполнения работы.

Назначение переключателей.

Клавиша SA1.

При нажатии клавиши SA1 формируется бездребезговый сигнал запроса прерывания **IRQ1.** Если запрос по этому входу не маскирован (0 в соответствующем разряде **СКО1),** то запрос будет обслужен.

Клавиша SA2.

При нажатии клавиши SA2 формируется бездребезговый сигнал запроса прерывания IRQ3. Обслуживание этого запроса аналогично IRQ1. Кроме этого запрос IRQ3 индицируется с помощью светодиода HL1.

Клавиша **SA3.**

При нажатии клавиши SA3 формируется бездребезговый сигнал запроса прерывания IRQ4. Обслуживание этого запроса аналогично IRQ1. Наличие запроса IRQ4 индицируется светодиодом HL2.

Клавиша SA4.

При нажатии клавиши SA4 формируется бездребезговый сигнал запроса прерывания IRQ5. Обслуживание этого запроса аналогично IRQ I. Наличие запроса индицируется светодиодом HL3.

Переключатель SA5.

Переключатель SA5 производит подключение выхода **INT** ведомого **ПКП** либо к входу **IRQO,** либо к входу **IRQ7** ведущего пкп.

Сброс триггеров формирования запросов **IRQ3...IRQ5** происходит путем записи во внешние устройства с адресами **B1...B3** соответственно.

Переключатель на центральной плате SA2. При программировании ПКП на центральной плате микропроцессора КР580ВМ80А необходимо переключить переключатель SA2 в следующие положения:

SA2.2 - верхнее;

SA2.1 - верхнее;

SA2.3 - нижнее;

SA2.4 - верхнее.

1) Запись в **УВВ** с адресами **B1H...B3H** - строб записи в триггеры запросов прерываний.

2) Запись или чтение в УВВ с адресами **98H...9BH -** выборка ведущего ПКП.

3) Запись в УВВ с адресом 99Н - строб записи последнего управляющего слова (УС) в регистр индикации и отображения УС с помощью светодиодов HL7...HL14 в инверсном режиме.

4) Запись или чтение в УВВ с адресами **9СН...9ЕН** - выборка ведомого ПКП.

При программировании ПКП для начальной установки и ввода слова команды операции 1 (СК01) необходимо задаться следующими условиями: в схеме один ПКП, начальный адрес подпрограммы обслуживания прерываний - 0820, далее адреса идут через четыре байта, режим вложенных прерываний.

```
0800 D3 Bl OUT B 1; сброс триггера запроса IRQ3
0802 D3 B2 OUT B2; сброс триггера запроса IRQ4
0804 D3 B3 OUT B3; сброс триггера запроса IRQ5
0806 3E 36 MVI A,36; запись слова команды инициализации СКИ1
0808 D3 98 OUT 98; запись СКИ1 в ПКП
080A D3 99 OUT 99;
080C 3E 08 MVI A,08; запись слова команды инициализации СКИ2
080E D3 9B OUT 9B; запись СКИ2 в ПКП
0810 D3 99 OUT 99;
0812 3E ** MVI A,**; запись маски (СК01)
0814 D3 9B OUT 9B; запись СК01 в ПКП
0816 D3 99 OUT 99;
0818 76 HLT ; останов.
```

Содержание отчета.

1. Краткие сведения о назначении, характеристиках и установке контроллера прерывания.

2. Программы, схемы алгоритма загрузки контроллера.

КОМАНДЫ МИКРОПРОЦЕССОРА КР580ВМ80А

ЗАСЫЛКА КОНСТАНТЫ		ПЕРЕСЫЛКА					
06	MVI B,D8	40	MOV B,B	55	MOV D,L	6B	MOV L,E
0E	MVI C,D8	41	MOV B,A	56	MOV D,M	6C	MOV L,H
16	MVI D.D8	42	MOV B,D	57	MOV D,A	6D	MOV L,L
1e	MVI E,D8	43	MOV B,E	58	MOV E,B	6E	MOV L,M
26	MVI H,D8	44	MOV B,H	59	MOV E,C	6F	MOV L,A
2E	MVI L,D8	45	MOV B,L	5A	MOV E,D	70	MOV M,B
36	MVI M,D8	46	MOV B,M	5B	MOV E,E	71	MOV M,C
3E	MVI A,D8	47	MOV B,C	5C	MOV E,H	72	MOV M,D
01	LXI B,D16	48	MOV C,P	5D	MOV E,L	73	MOV M,E
11	LXI D,D16	49	MOV C,C	5E	MOV E,M	74	MOV M,H
21	LXI H,D16	4A	MOV C,D	5F	MOV E,A	75	MOV M,L
31	LXI P,D16	4B	MOV C,E	60	MOV H,B	77	MOV M,A
		4C	MOV C,H	61	MOV H,C	78	MOV A,B
		4D	MOV C,L	62	MOV H,D	79	MOV A,C
		4E	MOV C,M	63	MOV H,E	7A	MOV A,D
		4F	MOV C,A	65	MOV H,L	7B	MOV A,E
		50	MOV D,B	66	MOV H,M	7C	MOV A,H
		51	MOV D,C	67	MOV H,A	7D	MOV A,L
		52	MOV D,D	68	MOV L,B	7E	MOV A,M
		53	MOV D,E	69	MOV L,C	7F	MOV A,A
		54	MOV D,H	6A	MOV L,D		

ЧТЕНИЕ ЗАПИСЬ		ОПЕРАЦИИ СО СТЕКОМ		ВВОД-ВЫВОД		УПРАВЛЕНИЕ	
0A	LDAX B	C5	PUSH B	D3	OUT D8	00	NOP
1A	LDAX D	D5	PUSH D	DB	IN D8	76	HLT
2A	LHLD ADR	E5	PUSH H			F3	DI
3A	LDA ADR	F5	PUSH PSW			FB	EI
02	STAX B	C1	POP B				
12	STAX D	D1	POP D				
22	SHLD ADR	E1	POP H				
32	STA ADR	F1	POP PSW*				
		E3	XTHL				
		F9	SPHL				

АККУМУЛЯТОР С РЕГИСТРОМ*							
80	ADD B	90	SUB B	A0	ANA B	B0	ORA B
81	ADD C	91	SUB C	A1	ANA C	B1	ORA C
82	ADD D	92	SUB D	A2	ANA D	B2	ORA D
83	ADD E	93	SUB E	A3	ANA E	B3	ORA E
84	ADD H	94	SUB H	A4	ANA H	B4	ORA H
85	ADD L	95	SUB L	A5	ANA L	B5	ORA L
86	ADD M	96	SUB M	A6	ANA M	B6	ORA M
87	ADD A	97	SUB A	A7	ANA A	B7	ORA A
88	ADC B	98	SBB B	A8	XRA B	B8	CMP B
89	ADC C	99	SBB C	A9	XRA C	B9	CMP C
8A	ADC D	9A	SBB D	AA	XRA D	BA	CMP D
8B	ADC E	9B	SBB E	AB	XRA E	BB	CMP E
8C	ADC H	9C	SBB H	AC	XRA H	BC	CMP H
8D	ADC L	9D	SBB L	AD	XRA L	BD	CMP L
8E	ADC M	9E	SBB M	AE	XRA M	BE	CMP M
8F	ADC A	9F	SBB A	AF	XRA A	BF	CMP A

ШЕСТНАДЦАТИРАЗРЯДНОЕ СЛОЖЕНИЕ +		ЦИКЛИЧЕСКИЙ СДВИГ +		АККУМУЛЯТОР С КОНСТАНТОЙ*		ИНКРЕМЕНТ**		ДЕКРЕМЕНТ**	
09	DAD B	07	RLC	C6	ADI D8	04	INR B	05	DCR B
19	DAD D	0F	RRC	CE	ACI D8	0C	INR C	0D	DCR C
29	DAD H	17	RAL	D6	SUI D8	14	INR D	15	DCR D
39	DAD SP	1F	RAR	DE	SBI D8	1C	INR E	1D	DCR E
				E6	ANI D8	24	INR H	25	DCR H
				EE	XRI D8	2C	INR L	2D	DCR L
				F6	ORI D8	34	INR M	35	DCR M
				FE	CPI D8	3C	INR A	3D	DCR A
						03	INX B	0B	DCX B
						13	INX D	1B	DCX D
						23	INX H	2B	DCX H
						33	INX SP	3B	DCX SP

ПЕРЕХОД	ВОЗВРАТ	ВЫЗОВ	СПЕЦИАЛЬНЫЙ ВЫЗОВ
C3 JMP ADR	C9 RET	CD CALL ADR	C7 RCT 0
C2 JNZ ADR	C0 RNC	C4 CNZ ADR	CF RST 1
CA JZ ADR	C8 RZ	CC CZ ADR	D7 RST 2
D2 JNZ ADR	D0 RNC	D4 CNC ADR	DF RST 3
DA JC ADR	D8 RC	DC CC ADR	E7 RST 4
E2 JPO ADR	E0 RPO	E4 CPO ADR	EF RST 5
EA JPE ADR	E8 RPE	EC CPE ADR	F7 RST 6
F2 JP ADR	F0 RP	F4 CP ADR	FF RST 7
FA JM ADR	F8 RM	FC CM ADR	
E9 PCHL			

ПРОЧИЕ ОПЕРАЦИИ	ОПЕРАТОРЫ
EB XCHB	( )
27 DAA *	* / MOD SHL SHR
2F CMA	+ -
37 STC +	NOT
3F CMC +	AND
	OR XOR

D8 - константа или арифметико-логическое выражение, имеющее своим значением 8-разрядную величину;

D 16-константа или арифметико-логическое выражение, имеющее своим значением 16-разрядную величину;

ADR - 16-разрядный адрес;

* - устанавливаются все флаги **CY, Z, S, P, Ac;**

+ - устанавливается только флаг **CY;**

** - устанавливаются флаги **Z, S, P, Ac.**