МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего образования «Казанский национальный исследовательский технический университет им. А.Н. Туполева-КАИ»
(КНИТУ-КАИ)
Чистопольский филиал «Восток»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ЛАБОРАТОРНЫМ РАБОТАМ

по дисциплине

ПЕРИФЕРИЙНЫЕ УСТРОЙСТВА

Индекс по учебному плану: **Б1.В.05.02**
Направление подготовки: **09.03.01 Информатика и вычислительная техника**
Квалификация: **Бакалавр**
Профиль подготовки: **Автоматизированные системы обработки информации и Управления**
Типы задач профессиональной деятельности: **проектная, производственно-технологическая**

Рекомендовано УМК ЧФ КНИТУ-КАИ

Чистополь
2023 г.

Введение

Данные методические указания предназначены для проведения лабораторных работ по дисциплине «Периферийные устройства»

Лабораторные работы ориентированы на изучение различных внешних устройств, характерных для современных компьютерных систем.

№ п/п	№ раздела	Наименование лабораторных работ	Трудоемкость (час.)
1	1	Работа с внешними устройствами по готовности	4
2	1	Работа с внешними устройствами по прерыванию	4
3	2	Управление шаговым двигателем	4
4	2	Управление светодиодной матрицей	4

1 Лабораторная работа №1

«Работа с внешними устройствами по готовности»

1.1 Цель работы

Целью настоящей лабораторной работы является изучение процесса организации взаимодействия процессора и внешних устройств (ВУ) в составе ЭВМ

1.2 Общие положения

В лабораторной работе №1 применяется программное обращение процессора к регистру состояния внешнего устройства с последующим анализом значения разряда готовности слова состояния ВУ.

Такое обращение следует предусмотреть в программе с некоторой периодичностью, независимо от фактического наступления контролируемого события (например, нажатие клавиши).

1.3 Задание на лабораторную работу №1

1. Ознакомиться с построением и работой программной модели ЭВМ (Приложение 4).
2. Написать программу с использованием режима работы с ВУ по готовности и произвести ассемблирование программы, пример программы приведен в табл. 1 (Приложение 1).
3. Зафиксировать результаты выполнения команд в виде копии экрана компьютера.
4. Оформить отчет по лабораторной работе в соответствии с правилами оформления текстовых документов (Приложение 5).
5. Титульный лист отчета оформить в соответствии с Приложением 3.
6. Отчет по лабораторной работе разместить на прилагаемом к отчетам лазерном диске типа CD-RW.

1.4 Ход работы

В процессе работы выполнить следующие действия:

1) включить компьютер;

2) загрузить программную модель ЭВМ;

3) загрузить выполняемую программу;

4) зафиксировать результаты выполнения программы в виде копий экрана компьютера.

1.5 Выполнение работы с ВУ по готовности

В программной модели учебной ЭВМ использован стандартный интерфейс Windows, реализованный в нескольких окнах.

1.5.1Окно «Текст программы»

Ввести последовательность команд в рабочее поле и произвести компиляцию (рис. 1).

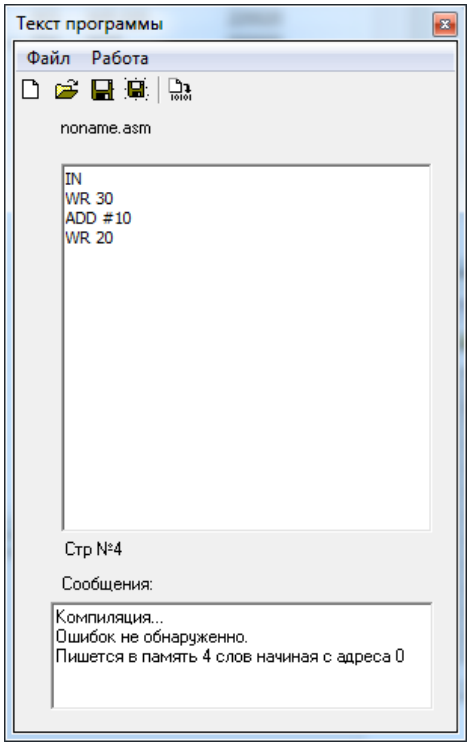


Рисунок 1 - Окно «Текст программы»

1.5.2 Окно «Подключаемые устройства»

В окне «Подключаемые устройства» подключить контроллеры клавиатуры и дисплея (рис. 2).

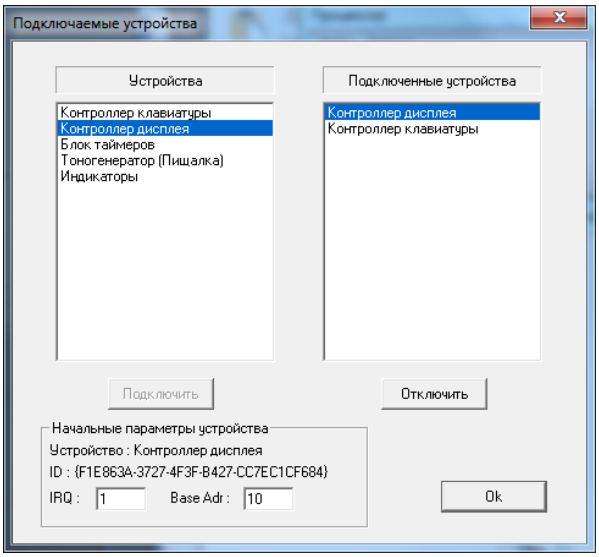


Рисунок 2 – Подключение дисплея и клавиатуры

1.5.3 Выполнение программы работы с ВУ по готовности

Открыть окно «Контроллер клавиатуры» (рис. 3), после чего вводить русские буквы и цифры.

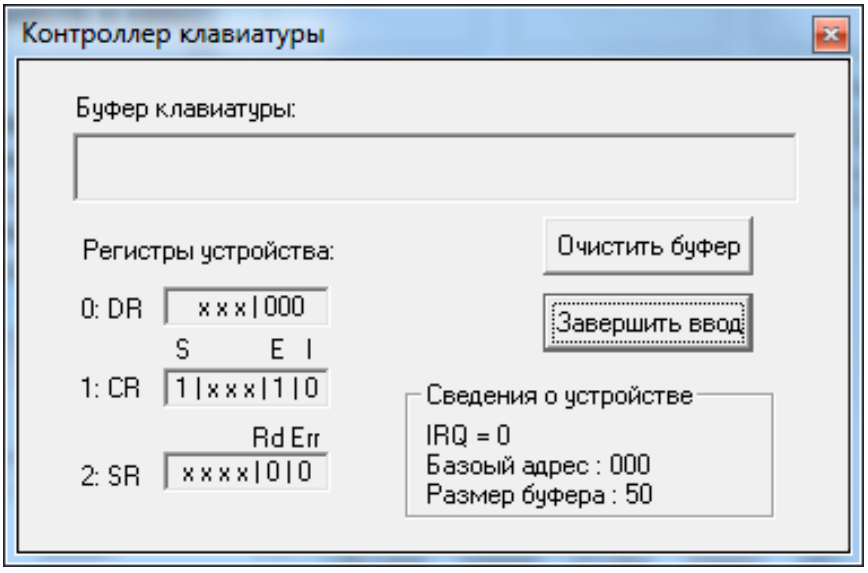


Рисунок 3 – Окно «Контроллер клавиатуры»

Зафиксировать происходящее в окне «Дисплей» (рис. 4).

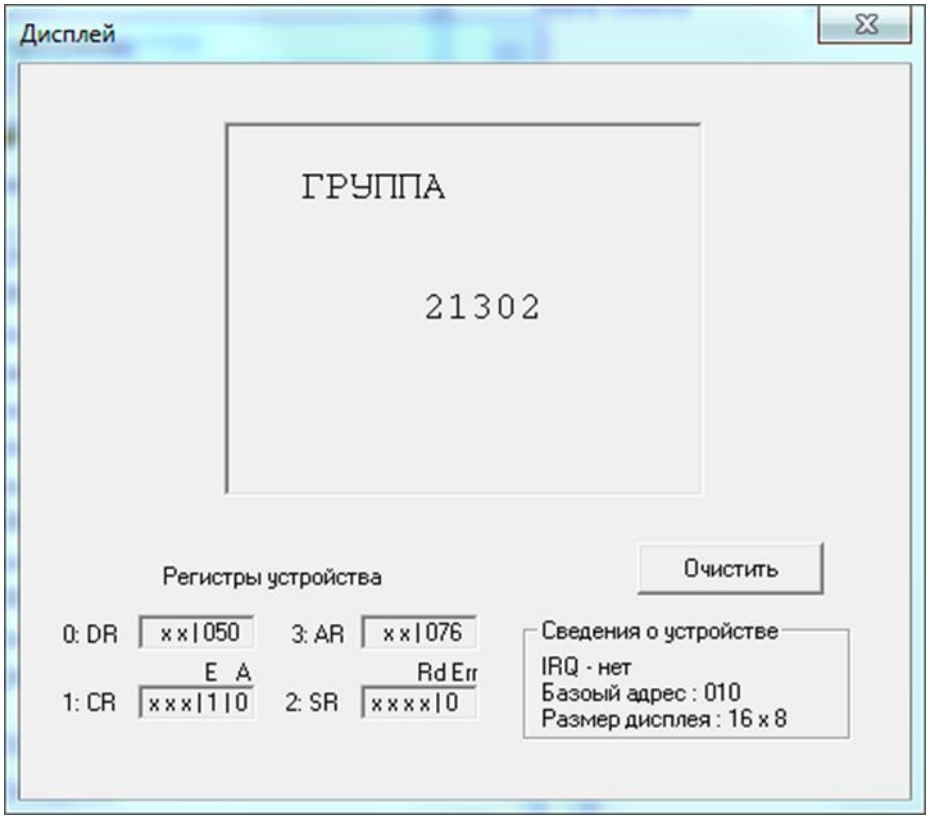


Рисунок 4 – Окно «Дисплей»

1.6 Содержание отчета

Отчет по лабораторной работе должен содержать:

1) оглавление;
2) описание цели работы;
3) описание хода работы в соответствии с данными методическими указаниями;
4) выводы по результатам выполнения работы;
5) список литературы;

Выводы

В результате выполнения данной лабораторной работы приобретены навыки выполнения работы с внешними устройствами в режиме ожидания готовности ВУ.

Список литературы

1. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации : учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. — М. : КУРС, 2018. — 208 с. - ISBN 978-5-906923-55-4. - Текст: электронный. - URL: https://znanium.com/catalog/product/1017280 (дата обращения: 06.11.2020)

2. Жмакин А.П. Архитектура ЭВМ.- СПб.: БХВ-Петербург, 2006. – 320 с.

4. Интерфейсы периферийных устройств [Электронный ресурс] : учеб. пособие / А.О. Ключев [и др.]. — Электрон. дан. — Санкт-Петербург : НИУ ИТМО, 2010. — 290 с. — Режим доступа: https://e.lanbook.com/book/43548. — Загл. с экрана.

5. Самарин, А.В. Жидкокристаллические дисплеи. Схемотехника, конструкция и применение [Электронный ресурс] : справ. — Электрон. дан. — Москва : СОЛОН-Пресс, 2007. — 304 с. — Режим доступа: https://e.lanbook.com/book/13651. — Загл. с экрана.

2 Лабораторная работа №2

«Работа с внешними устройствами по прерыванию»

2.1 Цель работы

Целью настоящей лабораторной работы является изучение процесса организации взаимодействия процессора и внешних устройств (ВУ) в составе ЭВМ

2.2 Общие положения

В лабораторной работе №2 при возникновении контролируемого события внешнее устройство выставляет процессору запрос на прерывание программы.

После обработки прерывания процессор осуществляет связь с внешним устройством.

2.3 Задание на лабораторную работу №2

1. Ознакомиться с построением и работой программной модели ЭВМ (Приложение 4).
2. Написать программу с использованием режима работы с ВУ по готовности и произвести ассемблирование программы, пример программы приведен в табл. 1 (Приложение 1).
3. Зафиксировать результаты выполнения команд в виде копии экрана компьютера.
4. Оформить отчет по лабораторной работе в соответствии с правилами оформления текстовых документов (Приложение 5).
5. Титульный лист отчета оформить в соответствии с Приложением 3.
6. Отчет по лабораторной работе разместить на прилагаемом к отчетам лазерном диске типа CD-RW.

2.4 Ход работы

В процессе работы выполнить следующие действия:

1) включить компьютер;

2) загрузить программную модель ЭВМ;

3) загрузить выполняемую программу;

4) зафиксировать результаты выполнения программы в виде копий экрана компьютера.

2.5 Выполнение работы с ВУ по готовности

В программной модели учебной ЭВМ использован стандартный интерфейс Windows, реализованный в нескольких окнах.

2.5.1Окно «Текст программы»

Ввести последовательность команд в рабочее поле и произвести компиляцию (рис. 1).

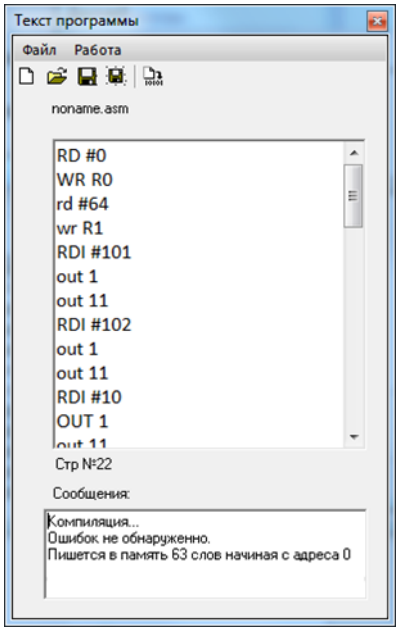


Рисунок 1 - Окно «Текст программы»

2.5.2 Окно «Подключаемые устройства»

В окне «Подключаемые устройства» подключить контроллеры клавиатуры и дисплея (рис. 2).

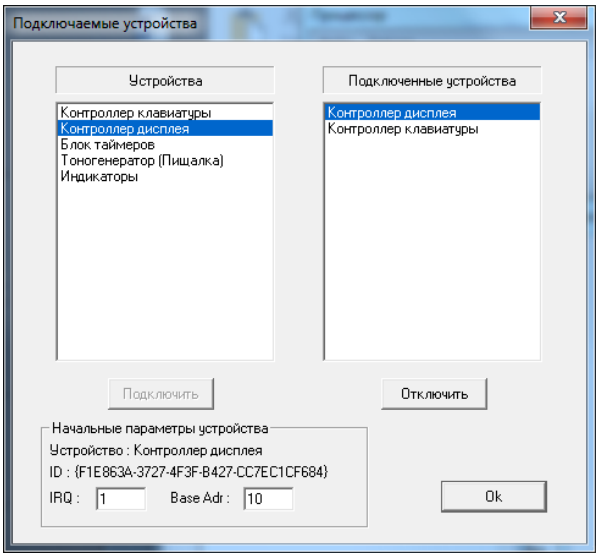


Рисунок 2 – Подключение дисплея и клавиатуры

2.5.3 Выполнение программы работы с ВУ по прерыванию

Открыть окно «Контроллер клавиатуры» (рис. 3), после чего вводить русские буквы и цифры.

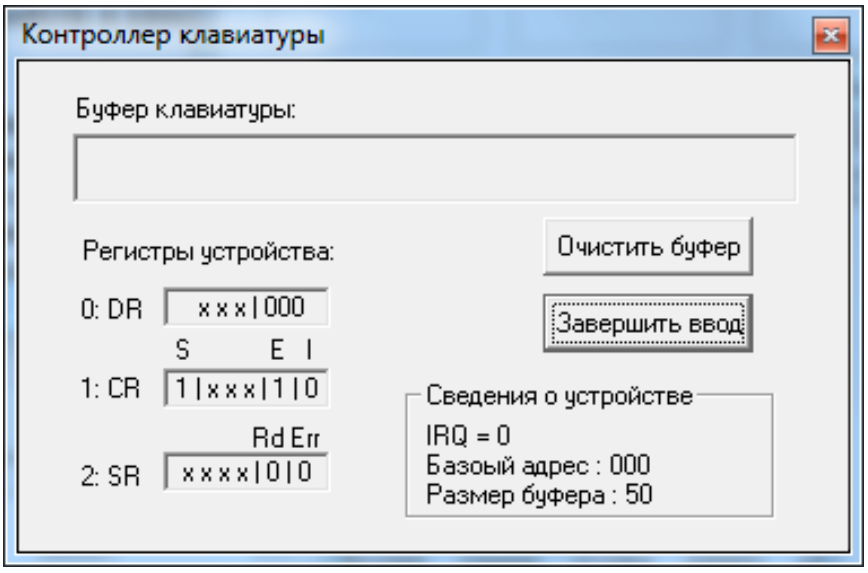


Рисунок 3 – Окно «Контроллер клавиатуры»

Зафиксировать происходящее в окне «Дисплей» (рис. 4).

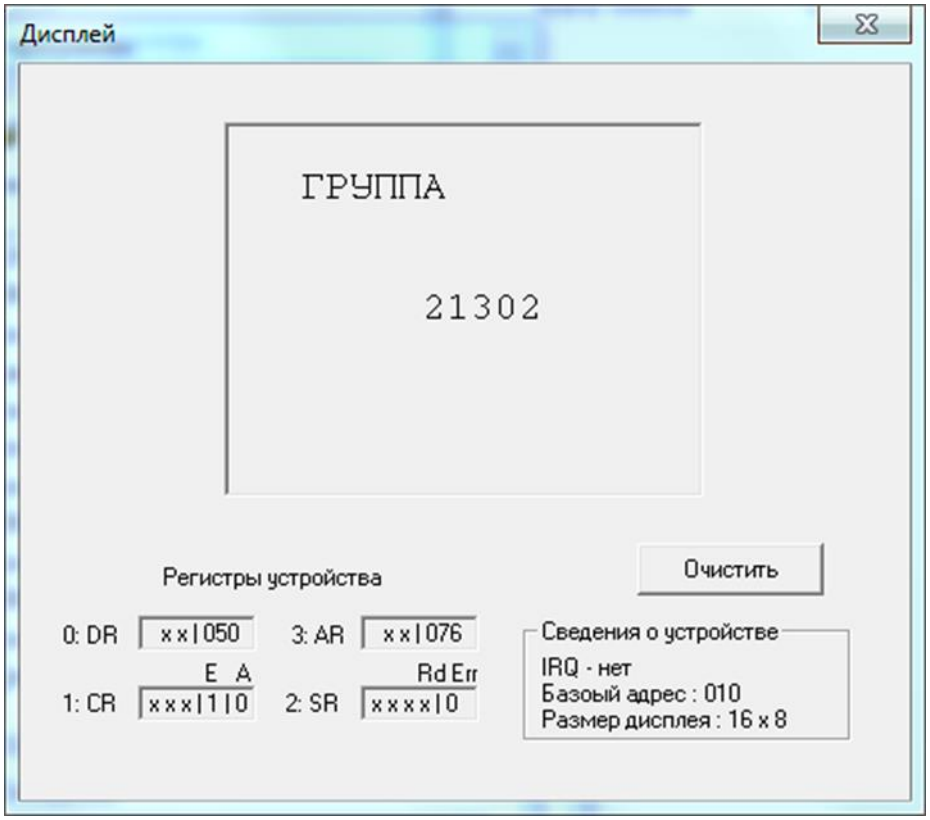


Рисунок 4 – Окно «Дисплей»

2.6 Содержание отчета

Отчет по лабораторной работе должен содержать:

1) оглавление;
2) описание цели работы;
3) описание хода работы в соответствии с данными методическими указаниями;
4) выводы по результатам выполнения работы;
5) список литературы;

Выводы

В результате выполнения данной лабораторной работы приобретены навыки выполнения работы с внешними устройствами в режиме прерывания от внешнего устройства.

Список литературы

1. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации : учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. — М. : КУРС, 2018. — 208 с. - ISBN 978-5-906923-55-4. - Текст: электронный. - URL: https://znanium.com/catalog/product/1017280 (дата обращения: 06.11.2020)

2. Жмакин А.П. Архитектура ЭВМ.- СПб.: БХВ-Петербург, 2006. – 320 с.

4. Интерфейсы периферийных устройств [Электронный ресурс] : учеб. пособие / А.О. Ключев [и др.]. — Электрон. дан. — Санкт-Петербург : НИУ ИТМО, 2010. — 290 с. — Режим доступа: https://e.lanbook.com/book/43548. — Загл. с экрана.

5. Самарин, А.В. Жидкокристаллические дисплеи. Схемотехника, конструкция и применение [Электронный ресурс] : справ. — Электрон. дан. — Москва : СОЛОН-Пресс, 2007. — 304 с. — Режим доступа: https://e.lanbook.com/book/13651. — Загл. с экрана.

3 Лабораторная работа №3 «Управление шаговым двигателем»

3.1 Цель работы

Целью настоящей лабораторной работы является управление шаговым двигателем в прямом и обратном направлении

3.2 Общие положения

В лабораторной работе №3 необходимо написать программу для управления шаговым двигателем с помощью микроконтроллера Arduino Uno в прямом и обратном направлении.

3.3 Задание на лабораторную работу №3

1. Ознакомиться с построением и работой программной модели ЭВМ (Приложение 4).
2. Написать программу с использованием режима работы с ВУ по готовности и произвести ассемблирование программы, пример программы приведен в табл. 1 (Приложение 1).
3. Зафиксировать результаты выполнения команд в виде копии экрана компьютера.
4. Оформить отчет по лабораторной работе в соответствии с правилами оформления текстовых документов (Приложение 5).
5. Титульный лист отчета оформить в соответствии с Приложением 3.
6. Отчет по лабораторной работе разместить на прилагаемом к отчетам лазерном диске типа CD-RW.

3.4 Ход работы

В процессе работы выполнить следующие действия:

1) подключить микроконтроллер Arduino Uno к компьютеру в соответствии со схемой, приведенной на Рис. 1 и на Рис. 2;

2) включить компьютер;

3) загрузить программу управления шаговым двигателем;

4) зафиксировать результаты управления шаговым двигателем.

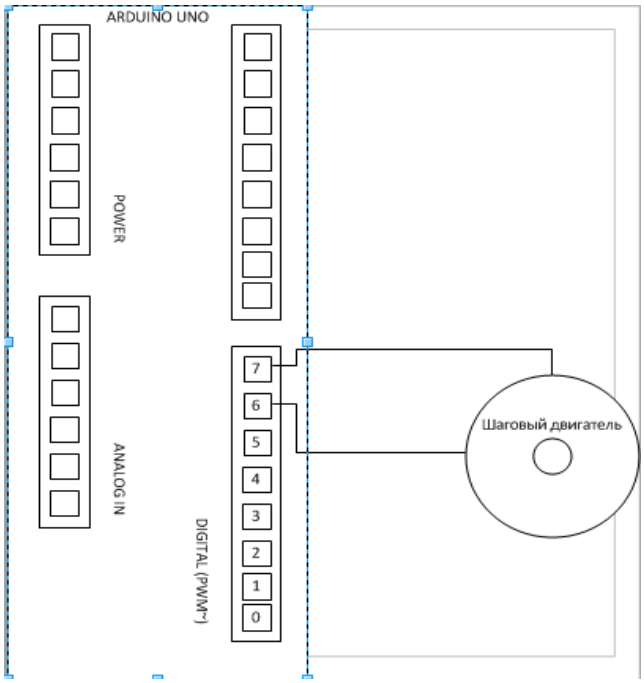


Рисунок 1 - Схема подключения шагового двигателя

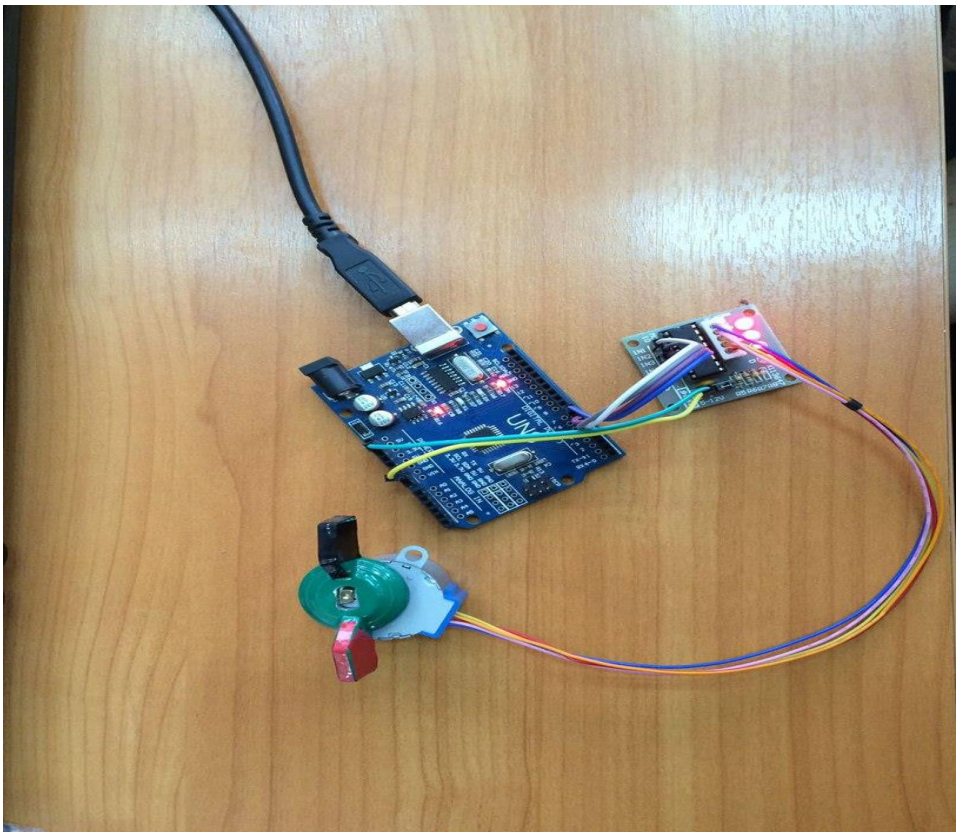


Рисунок 2 - Подключение шагового двигателя

3.5 Выполнение управления шаговым двигателем

После загрузки кода программы управления шаговым двигателем в микроконтроллер Arduino Uno проверить результаты работы программы.

Для этого открыть монитор порта (Рис. 3) и вводя букву “R” убедиться, что шаговый двигатель меняет направление вращения.

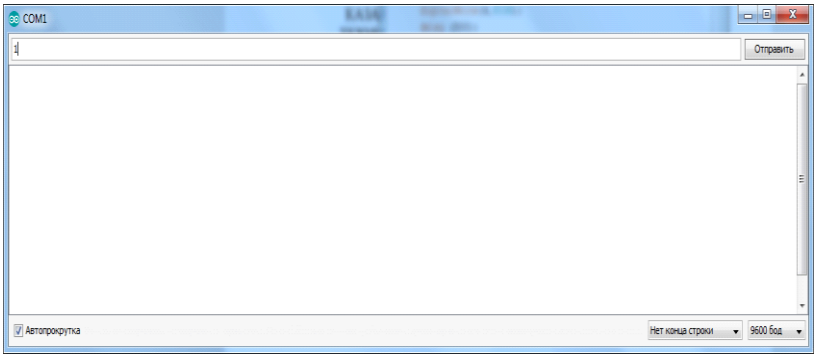


Рисунок 3 - Монитор порта

3.6 Содержание отчета

Отчет по лабораторной работе должен содержать:

1) оглавление;
2) описание цели работы;
3) описание хода работы в соответствии с данными методическими указаниями;
4) выводы по результатам выполнения работы;
5) список литературы;

Выводы

В результате выполнения данной лабораторной работы приобретены навыки управления направлением движения шагового двигателя.

Список литературы

1. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации : учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. — М. : КУРС, 2018. — 208 с. - ISBN 978-5-906923-55-4. - Текст: электронный. - URL: https://znanium.com/catalog/product/1017280 (дата обращения: 06.11.2020)

2. Жмакин А.П. Архитектура ЭВМ.- СПб.: БХВ-Петербург, 2006. – 320 с.

4. Интерфейсы периферийных устройств [Электронный ресурс] : учеб. пособие / А.О. Ключев [и др.]. — Электрон. дан. — Санкт-Петербург : НИУ ИТМО, 2010. — 290 с. — Режим доступа: https://e.lanbook.com/book/43548. — Загл. с экрана.

5. Самарин, А.В. Жидкокристаллические дисплеи. Схемотехника, конструкция и применение [Электронный ресурс] : справ. — Электрон. дан. — Москва : СОЛОН-Пресс, 2007. — 304 с. — Режим доступа: https://e.lanbook.com/book/13651. — Загл. с экрана.

4 Лабораторная работа №4 «Управление светодиодной матрицей»

4.1 Цель работы

Целью настоящей лабораторной работы является управление светодиодной матрицей

4.2 Общие положения

В лабораторной работе №4 необходимо написать программу для управления светодиодной матрицей с помощью микроконтроллера Arduino Uno

4.3 Задание на лабораторную работу №4

1. Ознакомиться с построением и работой программной модели ЭВМ (Приложение 4).
2. Написать программу с использованием режима работы с ВУ по готовности и произвести ассемблирование программы, пример программы приведен в табл. 1 (Приложение 1).
3. Зафиксировать результаты выполнения команд в виде копии экрана компьютера.
4. Оформить отчет по лабораторной работе в соответствии с правилами оформления текстовых документов (Приложение 5).
5. Титульный лист отчета оформить в соответствии с Приложением 3.
6. Отчет по лабораторной работе разместить на прилагаемом к отчетам лазерном диске типа CD-RW.

4.4 Ход работы

В процессе работы выполнить следующие действия:

1) подключить микроконтроллер Arduino Uno к компьютеру в соответствии со схемой, приведенной на Рис. 1 и на Рис. 2;

2) включить компьютер;

3) загрузить программу управления светодиодной матрицей;

4) зафиксировать результаты управления светодиодной матрицей.

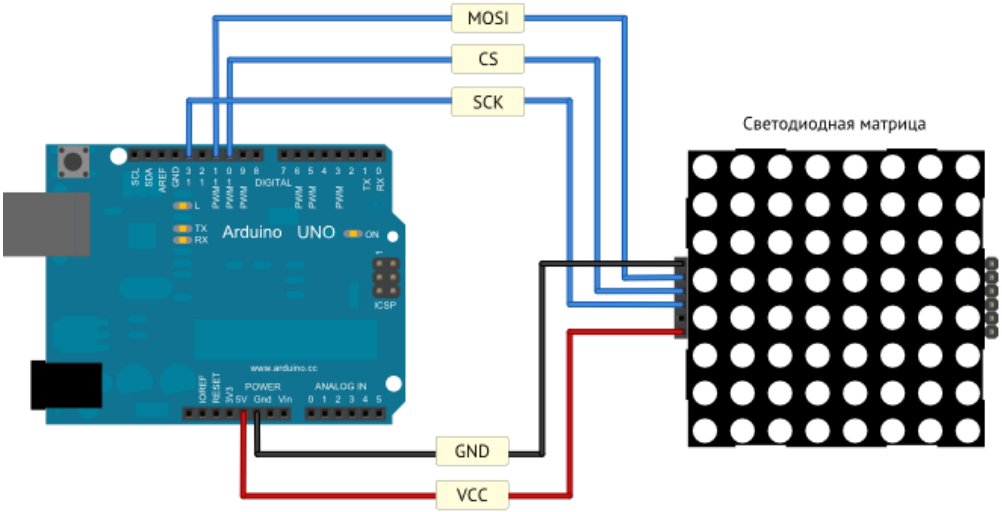


Рисунок 1 - Схема подключения светодиодной матрицы

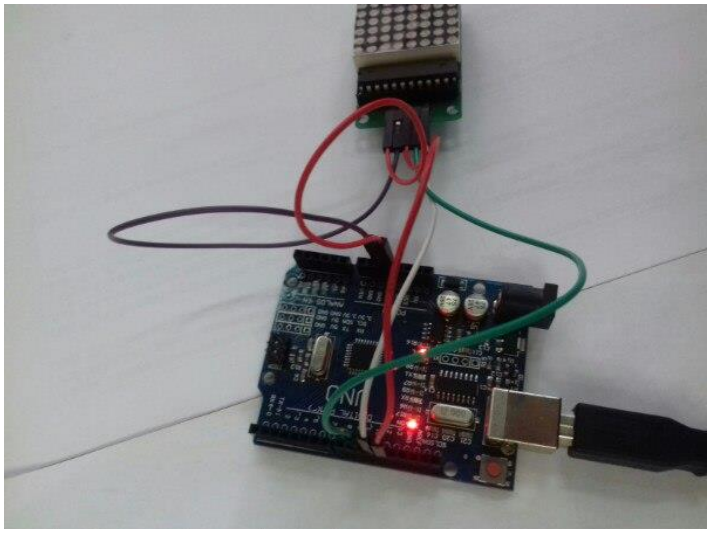


Рисунок 2 - Подключение светодиодной матрицы

4.5 Выполнение управления светодиодной матрицей

После загрузки кода программы светодиодной матрицей в микроконтроллер Arduino Uno проверить результаты работы программы при изображении числа (Рис. 3) и при изображении фигуры (Рис. 4).

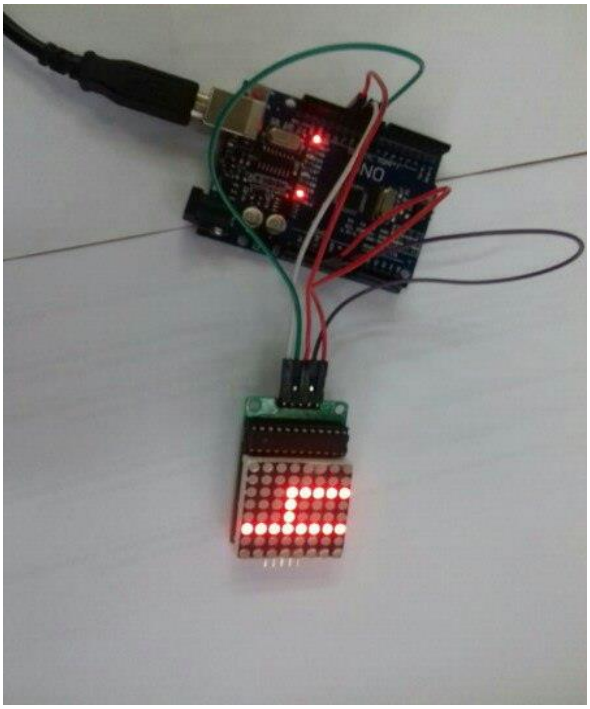


Рисунок 3 - Изображение числа на светодиодной матрице

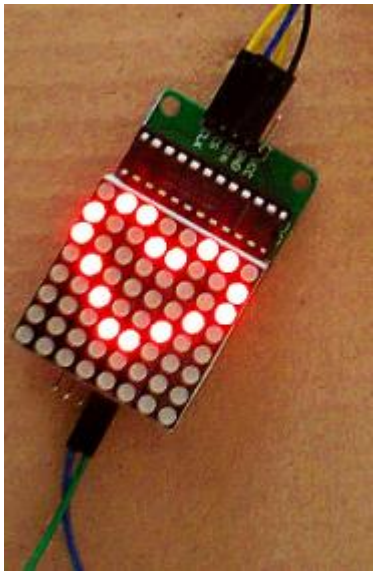


Рисунок 4 - Изображение фигуры на светодиодной матрице

4.6 Содержание отчета

Отчет по лабораторной работе должен содержать:

1) оглавление;
2) описание цели работы;
3) описание хода работы в соответствии с данными методическими указаниями;
4) выводы по результатам выполнения работы;
5) список литературы;

Выводы

В результате выполнения данной лабораторной работы приобретены навыки управления светодиодной матрицей для формирования изображений разного вида.

Список литературы

1. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации : учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. — М. : КУРС, 2018. — 208 с. - ISBN 978-5-906923-55-4. - Текст: электронный. - URL: https://znanium.com/catalog/product/1017280 (дата обращения: 06.11.2020)

2. Жмакин А.П. Архитектура ЭВМ.- СПб.: БХВ-Петербург, 2006. – 320 с.

4. Интерфейсы периферийных устройств [Электронный ресурс] : учеб. пособие / А.О. Ключев [и др.]. — Электрон. дан. — Санкт-Петербург : НИУ ИТМО, 2010. — 290 с. — Режим доступа: https://e.lanbook.com/book/43548. — Загл. с экрана.

5. Самарин, А.В. Жидкокристаллические дисплеи. Схемотехника, конструкция и применение [Электронный ресурс] : справ. — Электрон. дан. — Москва : СОЛОН-Пресс, 2007. — 304 с. — Режим доступа: https://e.lanbook.com/book/13651. — Загл. с экрана.

Приложение 1

Таблица 1 - Текст программы с анализом флагов и готовности ВУ

Команда 1	RD #0
Команда 2	WR R0
Команда 3	RD #64
Команда 4	WR R1
Команда 5	RDI #101
Команда 6	OUT 1
Команда 7	OUT 11
Команда 8	RDI #102
Команда 9	OUT 1
Команда 10	OUT 11
Команда 11	RDI #10
Команда 12	OUT 1
Команда 13	OUT 11
Команда 14	RDI #103
Команда 15	OUT 1
Команда 16	LOOP:IN 2
Команда 17	DIV #10
Команда 18	WR R3
Команда 19	DIV #2
Команда 20	MUL #2
Команда 21	SUB R3
Команда 22	JZ LOOP
Команда 23	IN 0
Команда 24	WR R2
Команда 25	SBI #48

Команда 26	JS RESET
Команда 27	RD R2
Команда 28	SBI #58
Команда 29	JNS CMPRUS
Команда 30	RD R1
Команда 31	OUT 13
Команда 32	ADI #1
Команда 33	WR R1
Команда 34	RD R2
Команда 35	OUT 10
Команда 36	JMP RESET
Команда 37	CMPRUS:RD R2
Команда 38	SBI #128
Команда 39	JS RESET
Команда 40	RD 80
Команда 41	OUT 13
Команда 42	ADI #1
Команда 43	WR R0
Команда 44	RD R2
Команда 45	OUT 10
Команда 46	RESET:RDI #101
Команда 47	OUT 1
Команда 48	RDI #103
Команда 49	OUT 1
Команда 50	JMP LOOP

Таблица 2 - Текст программы с обработкой прерываний

Команда 1	RD #27
Команда 2	WR 100
Команда 3	RD #0
Команда 4	WR R0
Команда 5	RD #64
Команда 6	WR R1
Команда 7	RDI #101
Команда 8	OUT 1
Команда 9	OUT 11
Команда 10	RDI #102
Команда 11	OUT 1
Команда 12	OUT 11
Команда 13	RDI #11
Команда 14	OUT 1
Команда 15	RDI #10
Команда 16	OUT 11
Команда 17	RDI #103
Команда 18	OUT 1
Команда 19	EI
Команда 20	LOOP:
Команда 21	NOP
Команда 22	NOP
Команда 23	JMP LOOP
Команда 24	INT0:IN 0

Команда 25	WR R2
Команда 26	SBI #48
Команда 27	JS RESET
Команда 28	RD R2
Команда 29	SBI #58
Команда 30	JNS CMPRUS
Команда 31	RD R1
Команда 32	OUT 13
Команда 33	ADI #1
Команда 34	WR R1
Команда 35	RD R2
Команда 36	OUT 10
Команда 37	JMP RESET
Команда 38	CMPRUS:RD R2
Команда 39	SBI #128
Команда 40	JS RESET
Команда 41	RD R0
Команда 42	OUT 13

1 Микроконтроллер Arduino Uno

Микроконтроллер Arduino Uno (Рис. 1) - это устройство на основе микропроцессора ATmega328. В его состав входит все необходимое для удобной работы с микроконтроллером: 14 цифровых входов/выходов (из них 6 могут использоваться в качестве ШИМ-выходов), 6 аналоговых входов, кварцевый резонатор на 16 МГц, разъем USB, разъем питания, разъем для внутрисхемного программирования (ICSP) и кнопка сброса.

Для начала работы с устройством достаточно просто подать питание от AC/DC-адаптера или батарейки, либо подключить его к компьютеру посредством USB-кабеля [1].

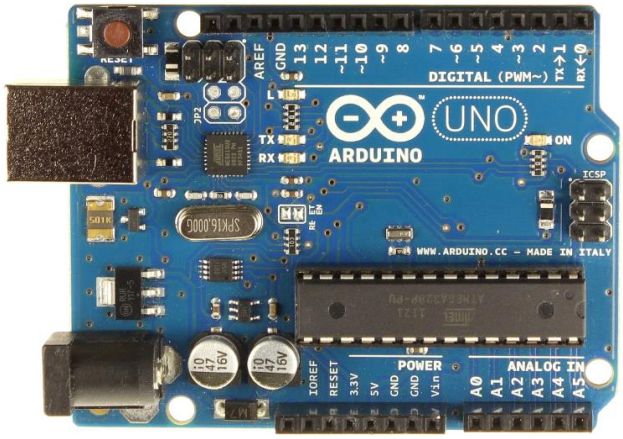


Рисунок 1- Внешний вид микроконтроллера Arduino Uno

1.1 Характеристики микроконтроллера

Микроконтроллер	ATmega328
Рабочее напряжение	5В
Напряжение питания (рекомендуемое)	7-12В
Напряжение питания (предельное)	6-20В
Цифровые входы/выходы	14 (из них 6 могут использоваться в качестве ШИМ-выходов)

Аналоговые входы	6
Максимальный ток одного вывода	40 мА
Максимальный выходной ток вывода 3.3V	50 мА
Flash-память	32 КБ (ATmega328) из которых 0.5 КБ используются загрузчиком
SRAM	2 КБ (ATmega328)
EEPROM	1 КБ (ATmega328)
Тактовая частота	16 МГц

1.2 Схема микроконтроллера

Схема микроконтроллера Arduino Uno представлена на Рис. 2.

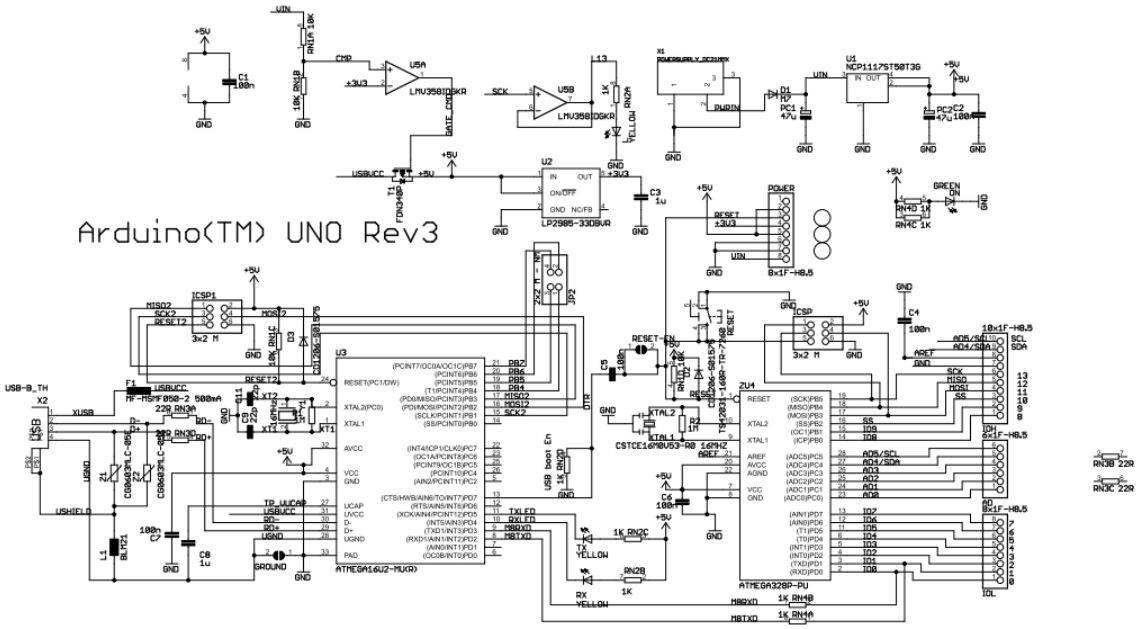


Рисунок 2 - Схема микроконтроллера Arduino Uno

1.3 Питание микроконтроллера

Микроконтроллер Arduino Uno может быть запитан от USB либо от внешнего источника питания - тип источника выбирается автоматически.

В качестве внешнего источника питания (не USB) может использоваться сетевой AC/DC-адаптер или аккумулятор/батарея. Штекер адаптера (диаметр - 2.1мм, центральный контакт - положительный) необходимо вставить в соответствующий разъем питания на плате. В случае питания от аккумулятора/батареи, ее провода необходимо подсоединить к выводам Gnd и Vin разъема POWER.

Напряжение внешнего источника питания может быть в пределах от 6 до 20 В. Однако, уменьшение напряжения питания ниже 7В приводит к уменьшению напряжения на выводе 5V, что может стать причиной нестабильной работы устройства. Использование напряжения больше 12В может приводить к перегреву стабилизатора напряжения и выходу платы из строя. С учетом этого, рекомендуется использовать источник питания с напряжением в диапазоне от 7 до 12В.

Ниже перечислены выводы питания, расположенные на плате:

1) VIN. Напряжение, поступающее в Arduino непосредственно от внешнего источника питания (не связано с 5В от USB или другим стабилизированным напряжением). Через этот вывод можно как подавать внешнее питание, так и потреблять ток, когда устройство запитано от внешнего адаптера;

2) 5V. На вывод поступает напряжение 5В от стабилизатора напряжения на плате, вне независимости от того, как запитано устройство: от адаптера (7 - 12В), от USB (5В) или через вывод VIN (7 - 12В). Запитывать устройство через выводы 5V или 3V3 не рекомендуется, поскольку в этом случае не используется стабилизатор напряжения, что может привести к выходу платы из строя;

3) 3V3. 3.3В, поступающие от стабилизатора напряжения на плате. Максимальный ток, потребляемый от этого вывода, составляет 50 мА;

4) GND. Выводы земли;

5) IOREF. Этот вывод предоставляет платам расширения информацию о рабочем напряжении микроконтроллера Arduino. В зависимости от напряжения, считанного с вывода IOREF, плата расширения может переключиться на соответствующий источник питания либо задействовать преобразователи уровней, что позволит ей работать как с 5В, так и с 3.3В-устройствами [3].

1.4 Память микроконтроллера

Объем флеш-памяти ATmega328 составляет 32 КБ (из которых 0.5 КБ используются загрузчиком). Микроконтроллер также имеет 2 КБ памяти SRAM и 1 КБ EEPROM (из которой можно считывать или записывать информацию с помощью библиотеки EEPROM).

1.5 Входы и выходы микроконтроллера

Использованием функций pinMode(), digitalWrite() и digitalRead() каждый из 14 цифровых выводов может работать в качестве входа или выхода. Уровень напряжения на выводах ограничен 5В. Максимальный ток, который может отдавать или потреблять один вывод, составляет 40 мА. Все выводы сопряжены с внутренними подтягивающими резисторами (по умолчанию отключенными) номиналом 20-50 кОм.

Последовательный интерфейс: выводы 0 (RX) и 1 (TX). Используются для получения (RX) и передачи (TX) данных по последовательному интерфейсу. Эти выводы соединены с соответствующими выводами микросхемы ATmega8U2, выполняющей роль преобразователя USB-UART.

Внешние прерывания: выводы 2 и 3. Могут служить источниками прерываний, возникающих при фронте, спаде или при низком уровне сигнала на этих выводах.

ШИМ: выводы 5, 6, 9, 10 и 11. С помощью функции analogWrite() могут выводить 8-битные аналоговые значения в виде ШИМ-сигнала.

Интерфейс SPI: выводы 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). С применением библиотеки SPI данные выводы могут осуществлять связь по интерфейсу SPI.

В Arduino Uno есть 6 аналоговых входов (A0 - A5), каждый из которых может представить аналоговое напряжение в виде 10-битного числа (1024 различных значения). По умолчанию, измерение напряжения осуществляется относительно диапазона от 0 до 5 В. Тем не менее, верхнюю границу этого диапазона можно изменить, используя вывод AREF и функцию analogReference().

TWI: вывод A4 или SDA и вывод A5 или SCL. С использованием библиотеки Wire данные выводы могут осуществлять связь по интерфейсу TWI.

AREF. Опорное напряжение для аналоговых входов. Может задействоваться функцией analogReference().

Reset. Формирование низкого уровня (LOW) на этом выводе приведет к перезагрузке микроконтроллера. Обычно этот вывод служит для функционирования кнопки сброса на платах расширения [2].

1.6 Программирование микроконтроллера

Arduino Uno программируется с помощью программного обеспечения Ардуино (Рисунок 3). Для этого из меню Tools > Board необходимо выбрать "Arduino Uno" с микроконтроллером, соответствующим вашей плате.

ATmega328 в Arduino Uno выпускается с прошитым загрузчиком, позволяющим загружать в микроконтроллер новые программы без необходимости использования внешнего программатора. Взаимодействие с ним осуществляется по оригинальному протоколу STK500.

Тем не менее, микроконтроллер можно прошить и через разъем для внутрисхемного программирования ICSP (In-Circuit Serial Programming), не обращая внимания на загрузчик.

Исходный код прошивки микроконтроллера ATmega16U2 находится в свободном доступе. Прошивка ATmega16U2/8U2 включает в себя DFU-загрузчик (Device Firmware Update), позволяющий обновлять прошивку микроконтроллера [1].

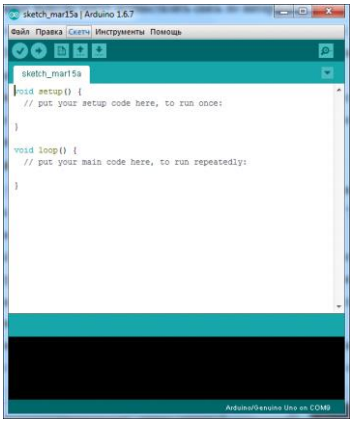


Рисунок 3 - Программное обеспечение Arduino

1.7 Программа управления шаговым двигателем

Текст программы управления шаговым двигателем в прямом и обратном направлении:

```
long val = '0';
void setup() {
      Serial.begin(9600);
      pinMode(7, OUTPUT);
      pinMode(6, OUTPUT);
}
void loop() {
  if (Serial.available() > 0) {
    val = Serial.read();}
    if (val=='0') {
      digitalWrite(7,LOW);
      digitalWrite(6,HIGH);
      delay (100);
      digitalWrite(7,HIGH);
      digitalWrite(6,HIGH);
      delay (150);
    }
    if (val=='1') {
```

```
    digitalWrite(7,HIGH);
    digitalWrite(6,LOW);
    delay (70);
    digitalWrite(7,HIGH);
    digitalWrite(6,HIGH);
    delay (70);
   }
   if (val=='R') {
   digitalWrite(7,HIGH);
    digitalWrite(6,HIGH);
   }
}
```

1.8 Программа управления светодиодной матрицей

1.8.1 Программа для изображения числа на светодиодной матрице.

```
LedControl lc=LedControl(12,11,10,1);
unsigned long delaytime=250;
void setup() {
  Serial.begin(9600);
  lc.shutdown(0,false);
  lc.setIntensity(0,8);
  lc.clearDisplay(0);
}
void draw_0()
{
  lc.setRow(0,0,24);
  lc.setRow(0,7,24);
  lc.setColumn(0,2,126);
  lc.setColumn(0,5,126);
}
void draw_1()
{
  lc.setLed(0,2,2,1);
  lc.setLed(0,1,3,1);
  lc.setColumn(0,4,255);
}
void draw_2()
{
  lc.setColumn(0,2,99);
  lc.setColumn(0,3,133);
  lc.setColumn(0,4,137);
  lc.setColumn(0,5,113);
}
void draw_3()
{
  lc.setColumn(0,2,130);
  lc.setColumn(0,3,145);
  lc.setColumn(0,4,177);
  lc.setColumn(0,5,206);
```

```
}
void draw_4()
{
  lc.setColumn(0,2,248);
  lc.setRow(0,4,56);
  lc.setColumn(0,5,255);
}
void draw_5()
{
  lc.setColumn(0,2,242);
  lc.setColumn(0,3,161);
  lc.setColumn(0,4,161);
  lc.setColumn(0,5,158);
}
void draw_6()
{
  lc.setColumn(0,2,126);
  lc.setColumn(0,3,145);
  lc.setColumn(0,4,145);
  lc.setColumn(0,5,78);
}
void draw_7()
{
  lc.setRow(0,0,60);
  lc.setLed(0,1,5,1);
  lc.setColumn(0,4,191);
}
void draw_8()
{
  lc.setColumn(0,2,110);
  lc.setColumn(0,3,145);
  lc.setColumn(0,4,145);
  lc.setColumn(0,5,110);
}
void draw_9()
{
  lc.setColumn(0,2,114);
  lc.setColumn(0,3,137);
  lc.setColumn(0,4,137);
  lc.setColumn(0,5,126);
}
void Print(int a)
{
  switch(a)
  {
    case 48:draw_0();break;
    case 49:draw_1();break;
    case 50:draw_2();break;
    case 51:draw_3();break;
    case 52:draw_4();break;
    case 53:draw_5();break;
    case 54:draw_6();break;
```

```
    case 55:draw_7();break;
    case 56:draw_8();break;
    case 57:draw_9();break;
    default:Serial.println("Digit only");break;
   }
  }

  /*void scrollDigits()
   {
    lc.setRow(0,5,126);
    for(int i=2;i<6;i++)
    lc.setLed(0,0,i,true);
      lc.setLed(0,1,1,true);
        lc.setLed(0,1,6,true);
        for(int i=2;i<6;i++)
        {
           lc.setLed(0,i,0,true);
           lc.setLed(0,i,7,true);
        }
    lc.setLed(0,6,1,true);
        lc.setLed(0,6,6,true);
        for(int i=2;i<6;i++)
    lc.setLed(0,7,i,true)
   }*/

  void loop() {
   while(Serial.available()<=0)
   {}
   while(Serial.available()!=0)
   {
    int a=Serial.read();
    Print(a);
    delay(2000);
    lc.clearDisplay(0);
   }
  }
```

1.8.2 Программа для изображения фигуры на светодиодной матрице

```
unsigned char i;
unsigned char j;
int Max7219_pinCLK = 10;
int Max7219_pinCS = 9;
int Max7219_pinDIN = 8;

unsigned char disp1[19][8]={
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x40, 0x00, 0x00, 0x00,
```

```
  0x00, 0x00, 0x00, 0x40, 0x40, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x80, 0x40, 0x40, 0x00, 0x00, 0x00,
  0x00, 0x80, 0x80, 0x40, 0x40, 0x00, 0x00, 0x00,
  0x40, 0x80, 0x80, 0x40, 0x40, 0x00, 0x00, 0x00,
  0x60, 0x80, 0x80, 0x40, 0x40, 0x00, 0x00, 0x00,
  0x60, 0x90, 0x80, 0x40, 0x40, 0x00, 0x00, 0x00,
  0x60, 0x90, 0x88, 0x40, 0x40, 0x00, 0x00, 0x00,
  0x60, 0x90, 0x88, 0x44, 0x40, 0x00, 0x00, 0x00,
  0x60, 0x90, 0x88, 0x44, 0x44, 0x00, 0x00, 0x00,
  0x60, 0x90, 0x88, 0x44, 0x44, 0x08, 0x00, 0x00,
  0x60, 0x90, 0x88, 0x44, 0x44, 0x08, 0x10, 0x00,
  0x60, 0x90, 0x88, 0x44, 0x44, 0x08, 0x10, 0x20,
  0x60, 0x90, 0x88, 0x44, 0x44, 0x08, 0x10, 0x60,
  0x60, 0x90, 0x88, 0x44, 0x44, 0x08, 0x90, 0x60,
  0x60, 0x90, 0x88, 0x44, 0x44, 0x88, 0x90, 0x60
};

void Write_Max7219_byte(unsigned char DATA)
{
  unsigned char i;
  digitalWrite(Max7219_pinCS,LOW);
  for(i=8;i>=1;i--)
  {
    digitalWrite(Max7219_pinCLK,LOW);
    digitalWrite(Max7219_pinDIN,DATA&0x80);
    DATA = DATA<<1;
    digitalWrite(Max7219_pinCLK,HIGH);
  }
}

void Write_Max7219(unsigned char address,unsigned char dat)
{
  digitalWrite(Max7219_pinCS,LOW);
  Write_Max7219_byte(address);
  Write_Max7219_byte(dat);
  digitalWrite(Max7219_pinCS,HIGH);
}

void Init_MAX7219(void)
{
  Write_Max7219(0x09, 0x00);
  Write_Max7219(0x0a, 0x03);
  Write_Max7219(0x0b, 0x07);
  Write_Max7219(0x0c, 0x01);
  Write_Max7219(0x0f, 0x00);
}

void setup()
{
   pinMode(Max7219_pinCLK,OUTPUT);
  pinMode(Max7219_pinCS,OUTPUT);
  pinMode(Max7219_pinDIN,OUTPUT);
```

```
  delay(50);
  Init_MAX7219();
}
void loop()
{
  for(j=0;j<19;j++)
  {
    for(i=1;i<9;i++)
      Write_Max7219(i,disp1[j][i-1]);
    delay(500);
  }
}
```

Приложение 3

Министерство науки и высшего образования Российской Федерации
федеральное государственное бюджетное образовательное учреждение высшего образования «Казанский национальный исследовательский технический университет им. А.Н. Туполева-КАИ»
(КНИТУ-КАИ)
Чистопольский филиал «Восток»

Кафедра компьютерных и телекоммуникационных систем

Отчет

по лабораторной работе № 1

по дисциплине «Периферийные устройства»

Работа с внешними устройствами по готовности

Выполнил ст. группы 21302 Петров И.И.

Проверил к. т. н., доцент Белош В.В.

г. Чистополь

2023 г.

Отчет

по лабораторной работе № 2

по дисциплине «Периферийные устройства»

Работа с внешними устройствами по прерыванию

Отчет

по лабораторной работе № 3

по дисциплине «Периферийные устройства»

Управление шаговым двигателем

Отчет

по лабораторной работе № 4

по дисциплине «Периферийные устройства»

Управление светодиодной матрицей

Приложение 4

1 Описание архитектуры учебной ЭВМ

Современные процессоры и операционные системы — не слишком благоприятная среда для начального этапа изучения архитектуры ЭВМ.

Одним из решений этой проблемы может быть создание программных моделей учебных ЭВМ, которые, с одной стороны, достаточно просты, чтобы обучаемый мог освоить базовые понятия архитектуры (система команд, командный цикл, способы адресации, уровни памяти, способы взаимодействия процессора с памятью и внешними устройствами), с другой стороны — архитектурные особенности модели должны соответствовать тенденциям развития современных ЭВМ.

Программная модель позволяет реализовать доступ к различным элементам ЭВМ, обеспечивая удобство и наглядность. С другой стороны, модель позволяет игнорировать те особенности работы реальной ЭВМ, которые на данном уровне рассмотрения не являются существенными.

Далее приводится описание программной модели учебной ЭВМ, предназначенной для начальных этапов изучения архитектуры (в т. ч. на младших курсах вуза и даже в школе). Именно этим объясняется использование в модели десятичной системы счисления для кодирования команд и представления данных.

1.1 Структура ЭВМ

Моделируемая ЭВМ включает процессор, оперативную (ОЗУ) и сверхоперативную память, устройство ввода (УВв) и устройство вывода (УВыв). Процессор, в свою очередь, состоит из центрального устройства управления (УУ), арифметического устройства (АУ) и системных регистров (CR, PC, SP и др.). Структурная схема ЭВМ показана на рис. 8.1.

В ячейках ОЗУ хранятся команды и данные. Емкость ОЗУ составляет 1000 ячеек. По сигналу MWr выполняется запись содержимого регистра данных (MDR) в ячейку памяти с адресом, указанным в регистре адреса (MAR). По сигналу MRd происходит считывание — содержимое ячейки памяти с адресом, содержащимся в MAR, передается в MDR.

Сверхоперативная память с прямой адресацией содержит десять регистров общего назначения R0—R9. Доступ к ним осуществляется (аналогично доступу к ОЗУ) через регистры RAR и RDR.

АУ осуществляет выполнение одной из арифметических операций, определяемой кодом операции (COP), над содержимым аккумулятора (Acc) и регистра операнда (DR). Результат операции всегда помещается в Acc. При завершении выполнения операции АУ вырабатывает сигналы признаков результата: Z (равен 1, если результат равен нулю); S (равен 1, если результат отрицателен); OV (равен 1, если при выполнении операции произошло переполнение

разрядной сетки). В случаях, когда эти условия не выполняются, соответствующие сигналы имеют нулевое значение.

В модели ЭВМ предусмотрены внешние устройства двух типов. Во-первых, это регистры IR и OR, которые могут обмениваться с аккумулятором с помощью безадресных команд IN (Асе := IR) и OUT (OR := Acc). Во-вторых, это набор моделей внешних устройств, которые могут подключаться к системе и взаимодействовать с ней в соответствии с заложенными в моделях алгоритмами. Каждое внешнее устройство имеет ряд программно-доступных регистров, может иметь собственный *обозреватель* (окно видимых элементов). Подробнее эти внешние устройства описаны в разд. 8.6.

УУ осуществляет выборку команд из ОЗУ в последовательности, определяемой естественным порядком выполнения команд (т. е. в порядке возрастания адресов команд в ОЗУ) или командами передачи управления; выборку из ОЗУ операндов, задаваемых адресами команды; инициирование выполнения операции, предписанной командой; останов или переход к выполнению следующей команды.

В качестве сверхоперативной памяти в модель включены регистры общего назначения (РОН), и может подключаться модель кэш-памяти.

В состав УУ ЭВМ входят:

PC — счетчик адреса команды, содержащий адрес текущей команды;

CR — регистр команды, содержащий код команды;

RB — регистр базового адреса, содержащий базовый адрес;

SP — указатель стека, содержащий адрес верхушки стека;

RA— регистр адреса, содержащий исполнительный адрес при косвенной адресации.

Регистры Acc, DR, IR, OR, CR и все ячейки ОЗУ и РОН имеют длину 6 десятичных разрядов, регистры PC, SP, RA и RB — 3 разряда.

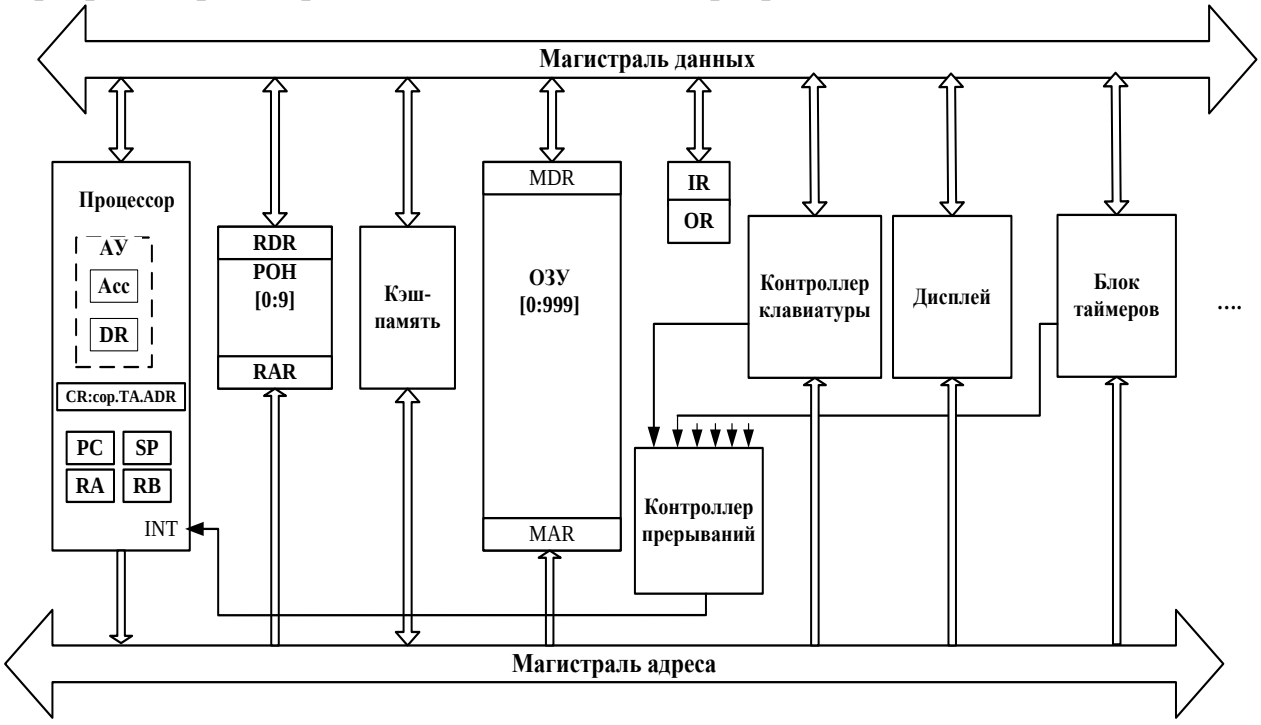


Рисунок 1 - Общая структура учебной ЭВМ

1.2 Представление данных в модели

Данные в ЭВМ представляются в формате, показанном на рис.2. Это целые десятичные числа, изменяющиеся в диапазоне "- 99 999...+ 99 999", содержащие знак и 5 десятичных цифр.

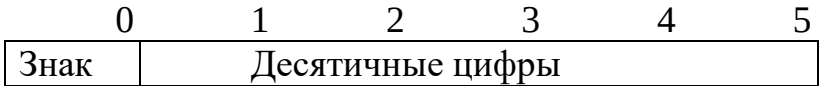


Рисунок 2 - Формат десятичных данных учебной ЭВМ

Старший разряд слова данных используется для кодирования знака: плюс (+) изображается как 0, минус (-) – как 1. Если результат арифметической операции выходит за пределы указанного диапазона, то говорят, что произошло переполнение разрядной сетки. АЛУ в этом случае вырабатывает сигнал переполнения OV = 1. Результатом операции деления является целая часть частного. Деление на ноль вызывает переполнение.

1.3 Система команд

При рассмотрении системы команд ЭВМ обычно анализируют три аспекта: форматы, способы адресации и систему операций.

1.3.1 Форматы команд

Большинство команд учебной ЭВМ являются одноадресными или безадресными, длиной в одно машинное слово (6 разрядов). Исключение составляют двухсловные команды с непосредственной адресацией и команда MOV, являющаяся двухадресной.

В форматах команд выделяется три поля:

- два старших разряда [0: 1] определяют код операции COP;
- разряд 2 может определять тип адресации (в одном случае (формат 5а) он определяет номер регистра);
- разряды [3:5] могут определять прямой или косвенный адрес памяти, номер регистра (в команде MOV номера двух регистров), адрес перехода или короткий непосредственный операнд. В двухсловных командах непосредственный операнд занимает поле [6:11].

Полный список форматов команд показан на рис.3, где приняты следующие обозначения:

COP — код операции;
ADR — адрес операнда в памяти;
ADC — адрес перехода;
I — непосредственный операнд;
R, Rl, R2 — номер регистра;
TA — тип адресации;
X — разряд не используется.

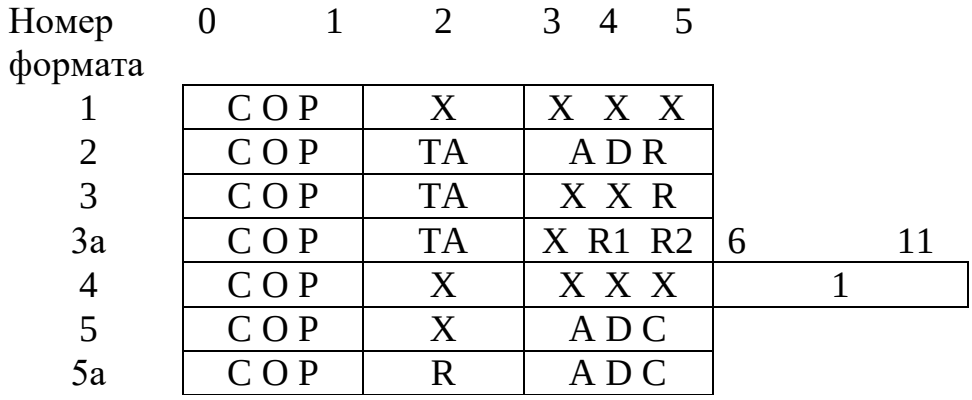


Рисунок 3 - Форматы команд учебной ЭВМ

1.3.2 Способы адресации

В ЭВМ принято различать пять основных способов адресации: *прямая, косвенная, непосредственная, относительная, безадресная.*

Каждый способ имеет разновидности. В модели учебной ЭВМ реализованы семь способов адресации, приведенные в табл. 1.

Таблица 1 -Адресация в командах учебной ЭВМ

Код ТА	**Тип адресации**	**Исполнительный адрес**
0	Прямая (регистровая)	ADR (R)
1	Непосредственная	-
2	Косвенная	ОЗУ (ADR) [3:5]
3	Относительная	ADR + RB
4	Косвенно-регистровая	РОН (R)[3:5]
5	Индексная с постинкрементом	РОН (R)[3:5], R:=R+1
6	Индексная с преддекрементом	R:=R-1, РОН(R)[3:5]

1.3.3 Система операций

Система команд учебной ЭВМ включает команды следующих классов:

- *арифметико-логические* и *специальные*: сложение, вычитание, умножение, деление;

- *пересылки* и *загрузки:* чтение, запись, пересылка (из регистра в регистр), помещение в стек, извлечение из стека, загрузка указателя стека, загрузка базового регистра;

- *ввода/вывода:* ввод, вывод;

- *передачи управления:* безусловный и шесть условных переходов, вызов подпрограммы, возврат из подпрограммы, цикл, программное прерывание, возврат из прерывания;

- *системные:* пустая операция, разрешить прерывание, запретить прерывание, стон.

Список команд учебной ЭВМ приведен в табл. 4 и в табл. 6.

1.4 Состояния и режимы работы ЭВМ

Ядром УУ ЭВМ является управляющий автомат (УА), вырабатывающий сигналы управления, которые инициируют работу АЛУ, РОН, ОЗУ и УВВ, передачу информации между регистрами устройств ЭВМ и действия над содержимым регистров УУ.

ЭВМ может находиться в одном из двух состояний: **Останов** и **Работа.**

В состояние **Работа** ЭВМ переходит по действию команд **Пуск** или **Шаг.** Команда **Пуск** запускает выполнение программы, представляющую собой последовательность команд, записанных в ОЗУ, в автоматическом режиме до команды HLT или точки останова. Программа выполняется по командам, начиная с ячейки ОЗУ, на которую указывает PC, причем изменение состояний объектов модели отображается в окнах обозревателей.

В состояние **Останов** ЭВМ переходит по действию команды **Стоп** или автоматически в зависимости от установленного режима работы.

Команда **Шаг,** в зависимости от установленного режима работы, запускает выполнение одной команды или одной микрокоманды (если установлен **Режим микрокоманд),** после чего переходит в состояние **Останов.**

В состоянии **Останов** допускается просмотр и модификация объектов модели: регистров процессора и РОН, ячеек ОЗУ, устройств ввода/вывода. В процессе модификации ячеек ОЗУ и РОН можно вводить данные для программы, в ячейки ОЗУ — программу в кодах. Кроме того, в режиме **Останов** можно менять параметры модели и режимы ее работы, вводить и/или редактировать программу в мнемокодах, ассемблировать мнемокоды, выполнять стандартные операции с файлами.

1.5 Интерфейс пользователя

В программной модели учебной ЭВМ использован стандартный интерфейс Windows, реализованный в нескольких окнах.

Основное окно модели **Модель учебной ЭВМ** содержит основное меню и кнопки на панели управления. В рабочее поле окна выводятся сообщения о функционировании системы в целом. Эти сообщения группируются в файле logfile.txt (по умолчанию), сохраняются на диске и могут быть проанализированы после завершения сеанса работы с моделью.

Меню содержит следующие пункты и команды:

Файл:

- неактивные команды;
- **Выход.**

Вид:

- **Показать все;**
- **Скрыть все;**
- **Процессор;**
- **Микрокомандный уровень;**
- **Память;**
- **Кэш-память;**
- **Программа;**
- **Текст программы.**

Внешние устройства:

- **Менеджер ВУ;**
- окна подключенных ВУ;

Работа:

- **Пуск;**
- **Стоп;**
- **Шаг;**
- **Режим микрокоманд;**
- **Кэш-память;**
- **Настройки.**

Команды меню **Вид** открывают окна соответствующих обозревателей, описанные далее. Менеджер внешних устройств позволяет подключать/отключать внешние устройства, предусмотренные в системе. Команда вызова менеджера внешних устройств выполняется при нажатии кнопки на панели инструментов. Подробнее о внешних устройствах и их обозревателях смотрите в разд. 1.6.

Команды меню **Работа** позволяют запустить программу в автоматическом (команда **Пуск)** или шаговом (команда **Шаг)** режиме, остановить выполнение программы в модели процессора (команда **Стоп).** Эти команды могут выполнять-

ся при нажатии соответствующих одноименных кнопок на панели инструментов основного окна.

Команда **Режим микрокоманд** включает/выключает микрокомандный режим работы процессора, а команда **Кэш-память** подключает/отключает в системе модель этого устройства.

Команда **Настройки** открывает диалоговое окно **Параметры системы,** позволяющее установить задержку реализации командного цикла (при выполнении программы в автоматическом режиме), а так же установить параметры файла logfile.txt, формируемого системой и записываемого на диск.

1.5.1 Окна основных обозревателей системы

Окно *Процессор*

Окно **Процессор** (рис. 4) обеспечивает доступ ко всем регистрам и флагам процессора.

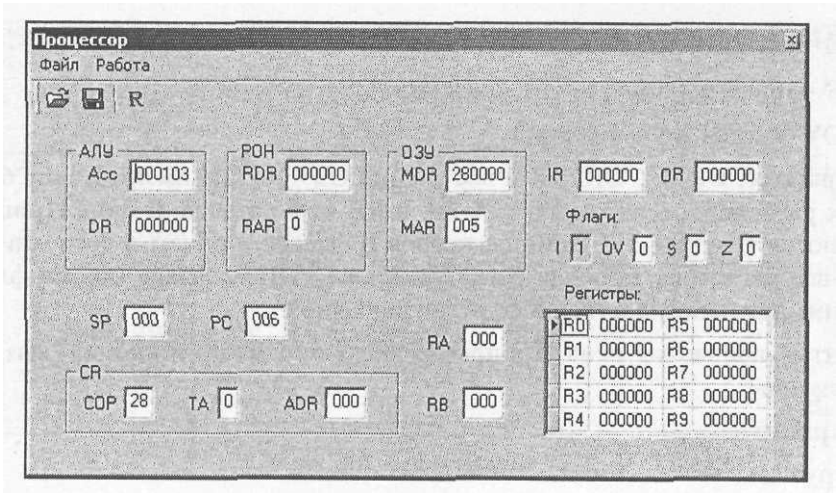


Рисунок 4 - Окно Процессор

Программно-доступные регистры и флаги:

- Acc — аккумулятор;
- PC — счетчик адреса команды, содержащий адрес текущей команды;
- SP — указатель стека, содержащий адрес верхушки стека;
- RB — регистр базового адреса, содержащий базовый адрес;
- RA — регистр адреса, содержащий исполнительный адрес при косвенной адресации;
- IR — входной регистр;
- OR — выходной регистр;
- I — флаг разрешения прерываний.

Системные регистры и флаги:

- DR — регистр данных АЛУ, содержащий второй операнд;
- MDR — регистр данных ОЗУ;

- MAR — регистр адреса ОЗУ;
- RDR — регистр данных блока РОН;
- RAR — регистр адреса блока РОН;
- CR — регистр команд, содержащий поля:
  COP — код операции;
  TA — тип адресации;
  ADR — адрес или непосредственный операнд;
- Z — флаг нулевого значения Acc;
- S — флаг отрицательного значения Acc;
- OV — флаг переполнения.

Регистры Acc, DR, IR, OR, CR и все ячейки ОЗУ и РОН имеют длину 6 десятичных разрядов, регистры PC, SP, RA и RB — 3 разряда. В окне **Процессор** отражаются текущие значения регистров и флагов, причем в состоянии **Останов** все регистры, включая регистры блока РОН, и флаги (кроме флага I) доступны для непосредственного редактирования.

Элементы управления окна **Процессор** включают меню и кнопки, вызывающие команды:

- **Сохранить;**
- **Загрузить;**
- **Reset;**
- **Reset R0-R9** (только команда меню **Работа).**

Команды **Сохранить, Загрузить** позволяют сохранить текущее значение регистров и флагов процессора в файле и восстановить состояние процессора из файла. Команда Reset и кнопка R устанавливают все регистры (в т. ч. блок РОН) в начальное (нулевое) значение. Содержимое ячеек памяти при этом не меняется. Выполняемая лишь из меню **Работа** команда **Reset R0-R9** очищает только регистры блока РОН.

Окно *Память*

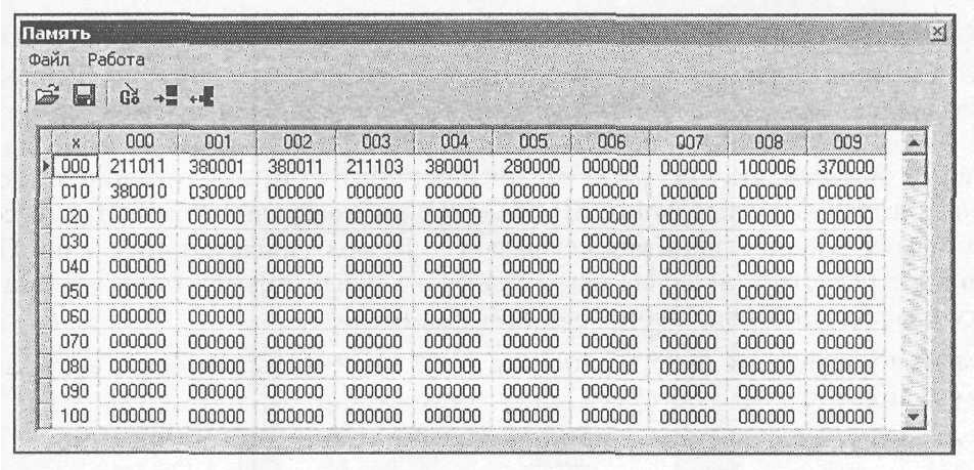
Окно **Память** (рис. 5) отражает текущее состояние ячеек ОЗУ. В этом окне допускается редактирование содержимого ячеек, кроме того, предусмотрена возможность выполнения (через меню или с помощью кнопок панели инструментов) пяти команд: **Сохранить, Загрузить, Перейти к, Вставить, Убрать.**

Команды **Сохранить, Загрузить** во всех окнах, где они предусмотрены, работают одинаково— сохраняют в файле текущее состояние объекта (в данном случае памяти) и восстанавливают это состояние из выбранного файла, причем файл в каждом окне записывается по умолчанию с характерным для этого окна расширением.

Команда **Перейти к** открывает диалоговое окно, позволяющее перейти на заданную ячейку ОЗУ.

Команда **Убрать** открывает диалог, в котором указывается диапазон ячеек с m по n. Содержимое ячеек в этом диапазоне теряется, а содержимое ячеек

[(n+1): 999] перемещается в соседние ячейки с меньшими адресами. Освободившиеся ячейки с адресами 999, 998, ... заполняются нулями.



x	000	001	002	003	004	005	006	007	008	009
000	211011	380001	380011	211103	380001	280000	000000	000000	100006	370000
010	380010	030000	000000	000000	000000	000000	000000	000000	000000	000000
020	000000	000000	000000	000000	000000	000000	000000	000000	000000	000000
030	000000	000000	000000	000000	000000	000000	000000	000000	000000	000000
040	000000	000000	000000	000000	000000	000000	000000	000000	000000	000000
050	000000	000000	000000	000000	000000	000000	000000	000000	000000	000000
060	000000	000000	000000	000000	000000	000000	000000	000000	000000	000000
070	000000	000000	000000	000000	000000	000000	000000	000000	000000	000000
080	000000	000000	000000	000000	000000	000000	000000	000000	000000	000000
090	000000	000000	000000	000000	000000	000000	000000	000000	000000	000000
100	000000	000000	000000	000000	000000	000000	000000	000000	000000	000000

Рисунок 5 - Окно **Память**

Команда **Вставить,** позволяющая задать номера ячеек, перемещает содержимое всех ячеек, начиная от m-й на m-n позиций в направлении больших адресов, ячейки заданного диапазона [m:n] заполняются нулями, а содержимое последних ячеек памяти теряется.

Окно *Текст программы*

Окно **Текст программы** (рис. 6) содержит стандартное поле текстового редактора, в котором можно редактировать тексты, загружать в него текстовые файлы и сохранять подготовленный текст в виде файла.

Команды меню **Файл:**

Новая — открывает новый сеанс редактирования;

Загрузить — открывает стандартный диалог загрузки файла в окно редактора;

Сохранить — сохраняет файл под текущим именем;

Сохранить как — открывает стандартный диалог сохранения файла;

Вставить — позволяет вставить выбранный файл в позицию курсора.

Все перечисленные команды, кроме последней, дублированы кнопками на панели инструментов окна. На той же панели присутствует еще одна кнопка — **Компилировать,** которая запускает процедуру ассемблирования текста поле редактора.

Ту же процедуру можно запустить из меню Работа. Команда **Адрес вставки** позволяет задать адрес ячейки ОЗУ, начиная с которой программа будет размещаться в памяти. По умолчанию этот адрес принят равным 0.

Ниже области редактирования в строку состояния выводится позиция текущей строки редактора — номер строки, в которой находится курсор.

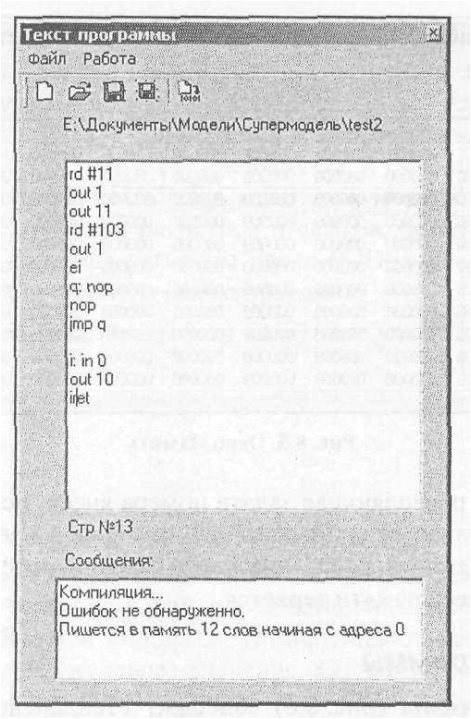


Рисунок 6 - Окно **Текст программы**

В случае обнаружения синтаксических ошибок в тексте программы диагностические сообщения процесса компиляции выводятся в окно сообщений и запись в память кодов (даже безошибочного начального фрагмента программы) не производится.

После исправления ошибок и повторной компиляции выдается сообщение об отсутствии ошибок, о расположении и размере области памяти, занятой под ассемблированную программу.

Набор текста программы производится по стандартным правилам языка ассемблера. В каждой строке может содержаться метка, одна команда и комментарий. Метка отделяется от команды двоеточием, символы после знака "точка с запятой" до конца строки игнорируются компилятором и могут рассматриваться как комментарии. Строка может начинаться с ; и, следовательно, содержать только комментарии.

Окно *Программа*

Окно **Программа** (рис. 7) отображает таблицу, имеющую 300 строк и 4 столбца. Каждая строка таблицы соответствует дизассемблированной ячейке ОЗУ. Второй столбец содержит адрес ячейки ОЗУ, третий — дизассемблированный мнемокод, четвертый - машинный код команды. В первом столбце может помещаться указатель → на текущую команду (текущее значение PC) и точка останова — красная заливка ячейки.

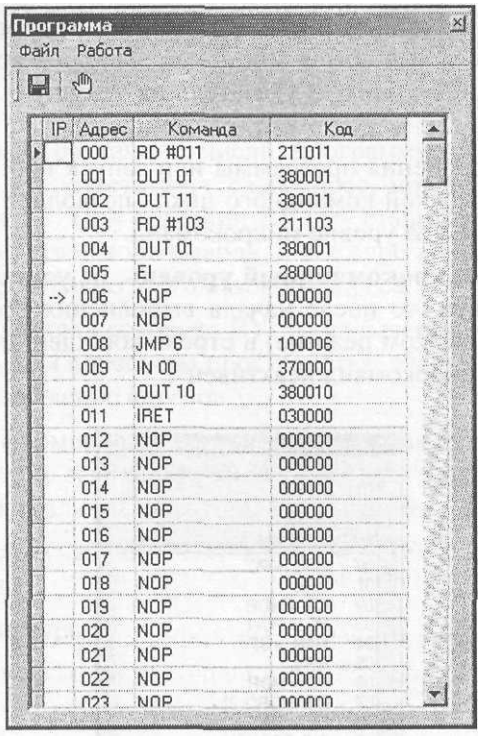


Рисунок 7 - Окно **Программа**

Окно **Программа** позволяет наблюдать процесс прохождения программы. В этом окне ничего нельзя редактировать. Органы управления окна позволяют сохранить содержимое окна в виде текстового файла, выбрать начальный адрес области ОЗУ, которая будет дизассемблироваться (размер области постоянный — 300 ячеек), а также установить/снять точку останова. Последнее можно проделать тремя способами: командой **Точка останова** из меню Работа, кнопкой на панели инструментов или двойным щелчком мыши в первой ячейке соответствующей строки. Характерно, что прочитать в это окно ничего нельзя. Сохраненный текстовый asm-файл можно загрузить в окно **Текст программы,** ассемблировать его и тогда дизассемблированное значение заданной области памяти автоматически появится в окне **Программа.** Такую процедуру удобно использовать, если программа изначально пишется или редактируется непосредственно в памяти в машинных кодах.

Начальный адрес области дизассемблирования задается в диалоге командой **Начальный адрес** меню **Работа.**

Окно *Микрокомандный уровень*

Окно **Микрокомандный уровень** (рис. 8) используется только в режиме микрокоманд, который устанавливается командой **Режим микрокоманд** меню **Работа. В** это окно выводится мнемокод выполняемой команды, список микрокоманд, ее реализующих, и указатель на текущую выполняемую микрокоманду.

Шаговый режим выполнения программы или запуск программы в автоматическом режиме с задержкой командного цикла позволяет наблюдать процесс выполнения программы на уровне микрокоманд.

Если открыть окно **Микрокомандный уровень,** не установив режим микрокоманд в меню **Работа,** то после начала выполнения программы в режиме **Шаг** (или в автоматическом режиме) в строке сообщений окна будет выдано сообщение "Режим микрокоманд неактивен".

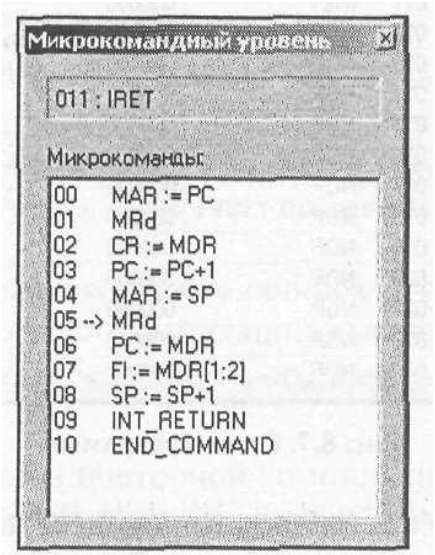


Рисунок 8 - Окно **Микрокомандный уровень**

Окно *Кэш-память*

Окно **Кэш-память** используется в режиме с подключенной кэш-памятью. Подробнее смотрите об этом режиме в разд. 1.8.

1.6. Внешние устройства

Модели внешних устройств (ВУ), используемые в описываемой системе, реализованы по единому принципу. С точки зрения процессора они представляют собой ряд программно-доступных регистров, лежащих в адресном пространстве ввода/вывода. Размер регистров ВУ совпадает с размером ячеек памяти и регистров данных процессора — шесть десятичных разрядов.

Доступ к регистрам ВУ осуществляется по командам IN aa, OUT aa, где aa — двухразрядный десятичный адрес регистра ВУ. Таким образом, общий объем адресного пространства ввода/вывода составляет 100 адресов. Следует помнить, что адресные пространства памяти и ввода/вывода в этой модели разделены.

Разные ВУ содержат различное число программно-доступных регистров, каждому из которых соответствует свой адрес, причем нумерация адресов всех ВУ начинается с 0. При создании ВУ ему ставится в соответствие *базовый адрес* в пространстве ввода/вывода, и все адреса его регистров становятся *смещениями* относительно этого базового адреса.

Если в системе создаются несколько ВУ, то их базовые адреса следует выбирать с учетом величины адресного пространства, занимаемого этими устройствами, исключая наложение адресов.

Если ВУ способно формировать запрос на прерывание, то при создании ему ставится в соответствие *вектор прерывания* — десятичное число. Разным ВУ должны назначаться различные векторы прерываний.

Программная модель учебной ЭВМ комплектуется набором внешних устройств, включающим:

- контроллер клавиатуры;
- дисплей;
- блок таймеров;
- тоногенератор,

которым по умолчанию присвоены параметры, перечисленные в табл. 2.

Таблица 2 - Параметры внешних устройств

Внешнее устройство	**Базовый адрес**	**Адрес регистров**	**Вектор прерывания**
Контроллер клавиатуры	0	0, 1, 2	0
Диспллей	10	0, 1, 2, 3	Нет
Блок таймеров	20	0, 1, 2, 3, 4, 5, 6	2
Тоногенератор	30	0, 1	Нет

При создании устройств пользователь может изменить назначенные по умолчанию базовый адрес и вектор прерывания.

В описываемой версии системы не предусмотрена возможность подключения в систему нескольких одинаковых устройств.

Большинство внешних устройств содержит регистры *управления* CR и *состояния* SR, причем обычно регистры CR доступны только по записи, а SR — по чтению.

Регистр CR содержит флаги и поля, определяющие режимы работы ВУ, а SR — флаги, отражающие текущее состояние ВУ. Флаги SR устанавливаются аппаратно, но сбрасываются программно (или по внешнему сигналу). Поля и флаги CR устанавливаются и сбрасываются программно при записи кода данных в регистр CR или специальными командами.

Контроллер ВУ интерпретирует код, записываемый по адресу CR как команду, если третий разряд этого кода равен 1, или как записываемые в CR данные, если третий разряд равен 0. В случае получения командного слова запись в регистр CR не производится, а пятый разряд слова рассматривается как код операции.

1.6.1 Контроллер клавиатуры

Контроллер клавиатуры (рис. 9) представляет собой модель внешнего устройства, принимающего ASCII-коды от клавиатуры ПЭВМ.

Символы помещаются последовательно в *буфер символов,* размер которого установлен равным 50 символам, и отображаются в окне обозревателя (рис.8. 10).

В состав контроллера клавиатуры входят три программно-доступных регистра:

DR (адрес 0) — регистр данных;

CR (адрес 1) — регистр управления, определяет режимы работы контроллера и содержит следующие флаги:

- E — флаг разрешения приема кодов в буфер;
- I — флаг разрешения прерывания;
- S — флаг режима посимвольного ввода.

SR (адрес 2) — регистр состояния, содержит два флага:

- Err — флаг ошибки;
- Rd — флаг готовности.

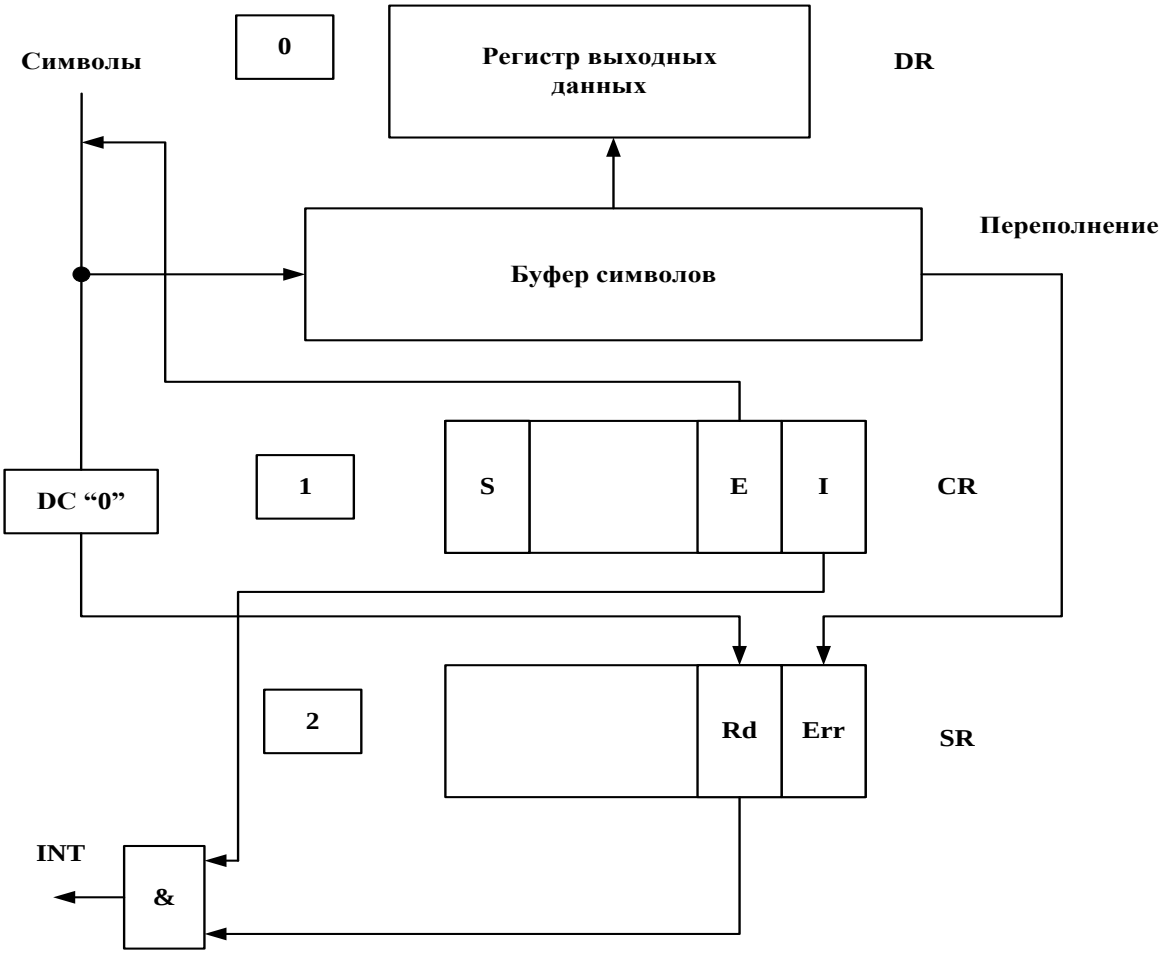


Рисунок 9 - Контроллер клавиатуры

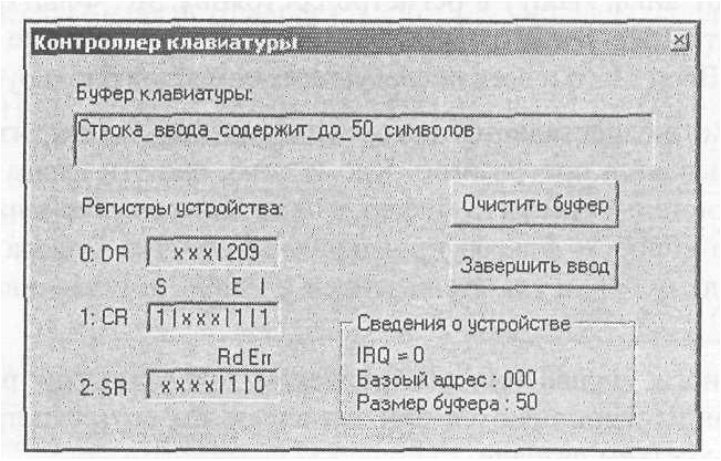


Рисунок 10 - Окно обозревателя контроллера клавиатуры

Регистр данных DR доступен только для чтения, через него считываются ASCII-коды из буфера, причем порядок чтения кодов из буфера соответствует порядку их записи в буфер — каждое чтение по адресу 0 автоматически перемещает указатель чтения буфера. В каждый момент времени DR содержит код символа по адресу указателя чтения буфера.

Флаги *регистра управления CR* устанавливаются и сбрасываются программно.

Флаг E, будучи установленным, разрешает прием кодов в буфер. При E = 0 контроллер игнорирует нажатие на клавиатуре, прием кодов в буфер не производится. На считывание кодов из буфера флаг E влияния не оказывает.

Флаг I, будучи установленным, разрешает при определенных условиях формирование контроллером запроса на прерывание. При I = 0 запрос на прерывание не формируется.

Флаг S = 1 устанавливает т. н. *режим посимвольного ввода,* иначе контроллер работает в обычном режиме. Флаг S устанавливается и сбрасывается программно, кроме того, S сбрасывается при нажатии кнопки **Очистить буфер** в окне **Контроллер клавиатуры.**

Условия формирования запроса на прерывание определяются, с одной стороны, значением флага разрешения прерывания I, с другой — режимом работы контроллера. В режиме посимвольного ввода запрос на прерывание формируется после ввода каждого символа (разумеется, при I= 1), в обычном режиме запрос будет сформирован по окончании набора строки.

Завершить набор строки можно, щелкнув по кнопке **Завершить ввод** в окне **Контроллер клавиатуры** (см. рис. 8.10). При этом устанавливается флаг готовности Rd (от англ. *ready)* в регистре состояния SR. Флаг ошибки Err (от англ. *error)* в том же регистре устанавливается при попытке ввода в буфер 51 -го символа. Ввод 51 -го и всех последующих символов блокируется.

Сброс флага Rd осуществляется автоматически при чтении из регистра DR, флаг Err сбрасывается программно. Кроме того, оба эти флага сбрасываются при нажатии кнопки **Очистить буфер** в окне **Контроллер клавиатуры;** одно-

временно со сбросом флагов производится очистка буфера— весь буфер заполняется кодами 00h, и указатели записи и чтения устанавливаются на начало буфера.

Для программного управления контроллером предусмотрен ряд командных слов. Все команды выполняются при записи по адресу регистра управления CR кодов с 1 в третьем разряде.

Контроллер клавиатуры интерпретирует следующие командные слова:

xxx10l — очистить буфер (действие команды эквивалентно нажатию кнопки **Очистить буфер);**

xxx102 — сбросить флаг Err в регистре SR;

xxx103 — установить флаг S в регистре CR;

xxx104 — сбросить флаг S в регистре CR.

Если по адресу 1 произвести запись числа *xxx0nn,* то произойдет изменение 4-го и 5-го разрядов регистра CR по следующему правилу:

$$n = \begin{cases} 0 - \text{записать } 0; \\ 1 - \text{записать } 1; \\ 2,\ldots, 9 - \text{сохранить разряд без изменения.} \end{cases} \qquad (8.1)$$

8.6.2 Дисплей

Дисплей (рис. 11) представляет собой модель внешнего устройства, реализующую функции символьного дисплея. Дисплей может отображать символы, задаваемые ASCII-кодами, поступающими на его регистр данных. Дисплей включает:

видеопамять объемом 128 слов (ОЗУ дисплея);

символьный экран размером 8 строк по 16 символов в строке;

четыре программно-доступных регистра:

- DR (адрес 0) — регистр данных;
- CR (адрес 1) — регистр управления;
- SR (адрес 2) — регистр состояния;
- AR (адрес 3) — регистр адреса.

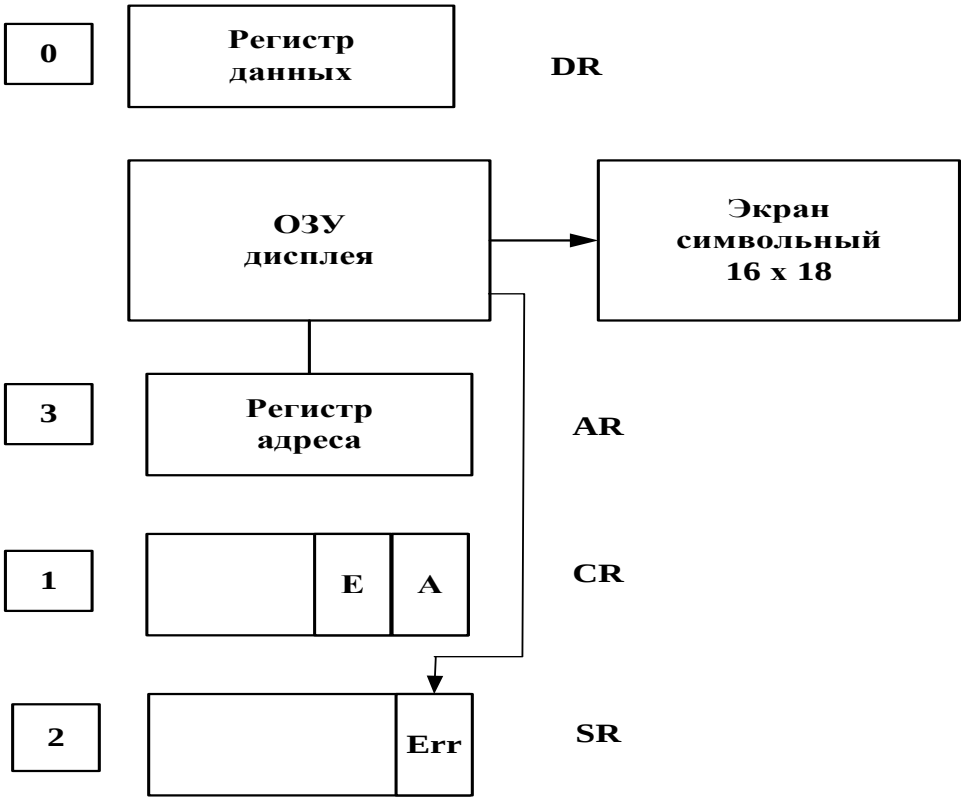


Рисунок 11 - Контроллер дисплея

Через *регистры адреса* AR и *данных* DR по записи и чтению осуществляется доступ к ячейкам видеопамяти. При обращении к регистру DR по записи содержимое аккумулятора записывается в DR и в ячейку видеопамяти, адрес которой установлен в регистре AR.

Регистр управления CR доступен только по записи и содержит в 4-м и 5-м разрядах соответственно два флага:

E — флаг разрешения работы дисплея; при E = 0 запись в регистры AR и DR блокируется;

A — флаг автоинкремента адреса; при A = 1 содержимое AR автоматически увеличивается на 1 после любого обращения к регистру DR — по записи или чтению.

Изменить значения этих флагов можно, если записать по адресу CR (по умолчанию — 11) код *xxx0nn,* при этом изменение 4-го и 5-го разрядов регистра CR произойдет согласно выражению (8.1).

Для программного управления дисплеем предусмотрены две команды, коды которых должны записываться по адресу регистра CR, причем в третьем разряде командных слов обязательно должна быть 1:

xxxl0l — очистить дисплей (действие команды эквивалентно нажатию кнопки **Очистить** в окне **Дисплей),** при этом очищается видеопамять (в каждую ячейку записывается код пробела— 032), устанавливается в 000 регистр адреса AR и сбрасываются флаги ошибки Err и автоинкремента A;

xxx102 — сбросить флаг ошибки Err.

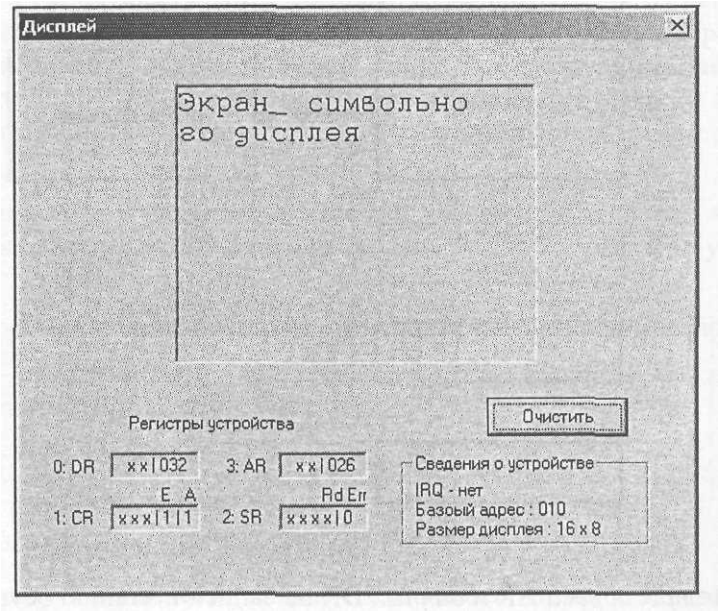


Рисунок 12 - Окно обозревателя контроллера дисплея

Регистр состояния SR доступен только по чтению и содержит единственный флаг (в пятом разряде) ошибки Err. Этот флаг устанавливается аппаратно при попытке записать в регистр адреса число, большее 127, причем как в режиме прямой записи в AR, так и в режиме автоинкремента после обращения по адресу 127. Сбрасывается флаг Err программно или при нажатии кнопки Очистить в окне **Дисплей** (рис. 12).

1.6.3 Блок таймеров

Блок таймеров (рис. 13) включает в себя три однотипных канала, каждый из которых содержит:

пятиразрядный десятичный реверсивный счетчик Т, на вход которого поступают метки времени (таймер);

программируемый предделитель D;

регистр управления таймером CTR;

флаг переполнения таймера FT.

Регистры таймеров Т доступны по записи и чтению (адреса 1, 3, 5 соответственно для Tl, Т2, Т3). Программа в любой момент может считать текущее содержимое таймера или записать в него новое значение.

На входы предделителей поступает общие для всех каналов метки времени CLK с периодом 1 мс. Предделители в каждом канале программируются независимо, поэтому таймеры могут работать с различной частотой.

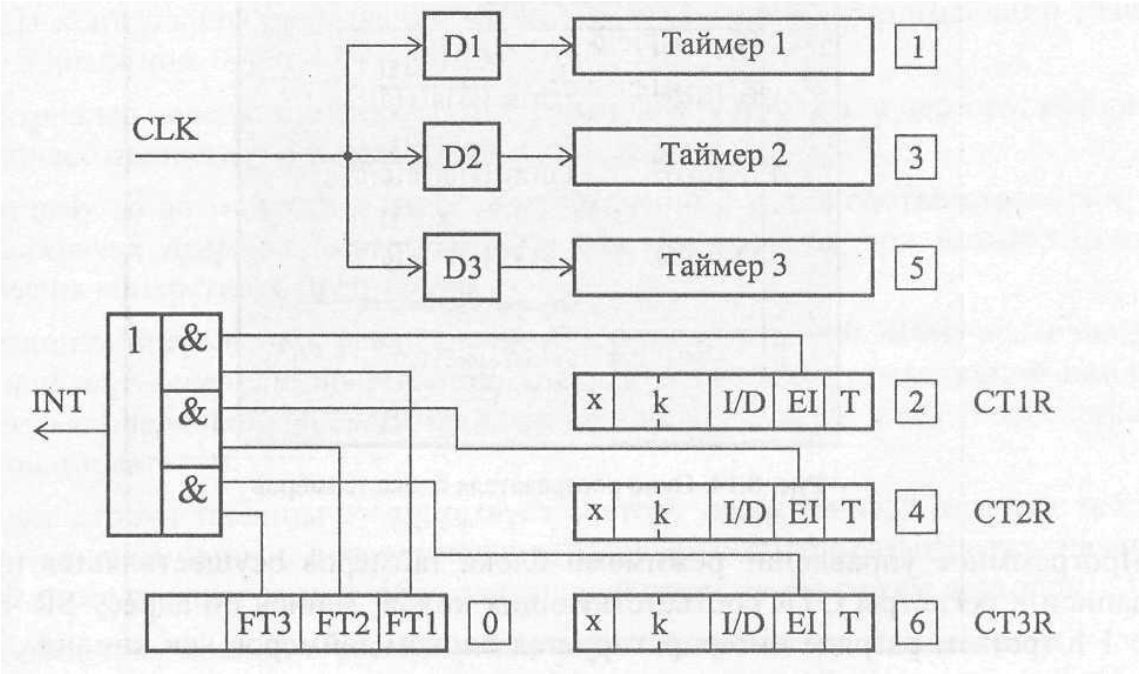


Рисунок 13 - Блок таймеров

Регистры управления CTR доступны по записи и чтению (адреса 2, 4, 6) и содержат следующие поля:

T (разряд 5) — флаг включения таймера;

EI (разряд 4) — флаг разрешения формирования запроса на прерывание при переполнении таймера;

I/D (разряд 3) — направление счета (инкремент/декремент), при I/D =0 таймер работает на сложение, при I/D =1 — на вычитание;

k (разряды [1:2]) — коэффициент деления предделителя (от 1 до 99).

Флаги переполнения таймеров собраны в один регистр — доступный только по чтению регистр состояния SR, имеющий адрес 0. Разряды регистра (5, 4 и 3 для Tl, T2, T3 соответственно) устанавливаются в 1 при переполнении соответствующего таймера. Для таймера, работающего на сложение, переполнение наступает при переходе его состояния из 99 999 в 0, для вычитающего таймера — переход из 0 в 99 999.

В окне обозревателя (рис. 8.14) предусмотрена кнопка **Сброс,** нажатие которой сбрасывает в 0 все регистры блока таймеров, кроме CTR, которые устанавливаются в состояние 001000. Таким образом, все три таймера обнуляются, переключаются в режим инкремента, прекращается счет, запрещаются прерывания, сбрасываются флаги переполнения и устанавливаются коэффициенты деления предделителей равными 01.

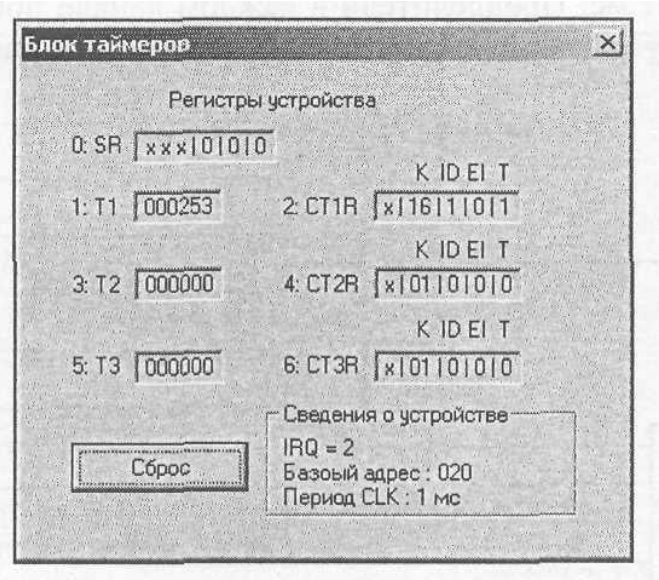


Рисунок 14 - Окно обозревателя блока таймеров

Программное управление режимами блока таймеров осуществляется путем записи в регистры CTR соответствующих кодов. Запись по адресу SR числа с 1 в третьем разряде интерпретируется блоком таймеров как команда, причем младшие разряды этого числа определяют код команды:

xxxl00 — общий сброс (эквивалентна нажатию кнопки Сброс в окне обозревателя);

xxxl0l — сброс флага переполнения таймера FT1;

xxx102 — сброс флага переполнения таймера FT2;

xxx103 — сброс флага переполнения таймера FT3.

1.6.4 Тоногенератор

Модель этого простого внешнего устройства не имеет собственного обозревателя, содержит всего два регистра, доступных только для записи:

FR (адрес 0) — регистр частоты звучания (Гц):

LR (адрес 1) — регистр длительности звучания (мс).

По умолчанию базовый адрес тоногенератора — 30. Сначала следует записать в FR требуемую частоту тона в герцах, затем в LR — длительность звучания в миллисекундах. Запись числа по адресу регистра LR одновременно является командой на начало звучания.

1.7 Подсистема прерываний

В модели учебной ЭВМ предусмотрен механизм векторных внешних прерываний. Внешние устройства формируют запросы на прерывания, которые поступают на входы *контроллера прерываний.* При подключении ВУ, способного формировать запрос на прерывание, ему ставится в соответствие номер

входа контроллера прерываний — вектор прерывания, принимающий значение в диапазоне 0—9.

Контроллер передает вектор, соответствующий запросу, процессору, который начинает процедуру обслуживания прерывания.

Каждому из возможных в системе прерываний должен соответствовать т. н. *обработчик прерывания*— подпрограмма, вызываемая при возникновении события конкретного прерывания.

Механизм прерываний, реализованный в модели учебной ЭВМ, поддерживает *таблицу векторов прерывании,* которая создается в оперативной памяти *моделью операционной системы* (если она используется) или непосредственно пользователем.

Номер строки таблицы соответствует вектору прерывания, а элемент таблицы — ячейка памяти, в трех младших разрядах которой размещается начальный адрес подпрограммы, обслуживающей прерывание с этим вектором.

Таблица прерываний в рассматриваемой модели жестко фиксирована— она занимает ячейки памяти с адресами 100—109. Таким образом, адрес обработчика с вектором 0 должен располагаться в ячейке 100, с вектором 2 — в ячейке 102. При работе с прерываниями не рекомендуется использовать ячейки 100—109 для других целей.

Процессор начинает обработку прерывания (если они разрешены), завершив текущую команду. При этом он:

- получает от контроллера вектор прерывания;
- формирует и помещает в верхушку стека слово, три младших разряда([3:5]) которого — текущее значение PC (адрес возврата из прерывания), а разряды [1:2] сохраняют десятичный эквивалент шестнадцатеричной цифры, определяющей значение вектора флагов (I, OV, S, Z). Например, если I=1, OV = 0, S = 1, Z = 1,то в разряды [1:2] запишется число $11_{10} = 1011_{2;}$
- сбрасывает в 0 флаг разрешения прерывания I;
- извлекает из таблицы векторов прерываний адрес обработчика, соответствующий обслуживаемому вектору, и помещает его в PC, осуществляя тем самым переход на подпрограмму обработчика прерывания.

Таким образом, вызов обработчика прерывания, в отличие от вызова подпрограммы, связан с помещением в стек не только адреса возврата, но и текущего значения вектора флагов. Поэтому последней командой подпрограммы обработчика должна быть команда IRET, которая не только возвращает в PC три младшие разряда ячейки — верхушки стека (как RET), НО И восстанавливает те значения флагов, которые были в момент перехода на обработку прерывания.

Не всякое событие, которое может вызвать прерывание, приводит к прерыванию текущей программы. В состав процессора входит программно-доступный флаг I разрешения прерывания. При I = 0 процессор не реагирует на запросы прерываний. После сброса процессора флаг I так же сброшен и все прерывания запрещены. Для того чтобы разрешить прерывания, следуете программе выполнить команду EI (от англ. *enable interrupt).*

Выше отмечалось, что при переходе на обработку прерывания флаг I автоматически сбрасывается, в этом случае прервать обслуживание одного прерыва-

ния другим прерыванием нельзя. По команде IRET значение флагов восстанавливается, в т. ч. вновь устанавливается I = 1, следовательно, в основной программе прерывания опять разрешены.

Если требуется разрешить другие прерывания в обработчике прерывания, достаточно в нем выполнить команду EI. Контроллер прерываний и процессор на аппаратном уровне блокируют попытки запустить прерывание, если его обработчик начал, но не завершил работу.

Таким образом, флаг I разрешает или запрещает все прерывания системы. Если требуется выборочно разрешить некоторое подмножество прерываний, используются программно-доступные флаги разрешения прерываний непосредственно на внешних устройствах.

Как правило, каждое внешнее устройство, которое может вызвать прерывание, содержит в составе своих регистров разряд флага разрешения прерывания (см. формат регистров CR и CTR на рис. 9, 13), по умолчанию установленный в 0. Если оставить этот флаг в нуле, то внешнему устройству запрещается формировать запрос контроллеру прерываний.

Иногда бывает удобно (например, в режиме отладки) иметь возможность вызвать обработчик прерывания непосредственно из программы. Если использовать для этих целей команду CALL, которая помещает в стек только адрес возврата, то команда IRET, размещенная последней в обработчике, может исказить значения флагов (все они будут сброшены в 0, т. к. команда CALL формирует только три младшие разряда ячейки верхушки стека, оставляя остальные разряды в 000).

Поэтому в системах команд многих ЭВМ, в т. ч. и нашей модели, имеются команды вызова прерываний— INT n (в нашей модели $n \in \{0, 1,\ldots,9\}$), где n— вектор прерывания. Процессор, выполняя команду INT n, производит те же действия, что и при обработке прерывания с вектором n.

Характерно, что с помощью команды INT n можно вызвать обработчик прерывания даже в том случае, когда флаг разрешения прерывания I сброшен.

Приложение 5

Правила оформления текстовых документов

Рефераты, выпускная квалификационная работа и другие текстовые документы выполняются на отдельных листах бумаги формата А4 (210х297мм) с помощью текстового редактора MicrosoftWord.

На страницах оставляются поля: слева – 25 мм, справа – 15 мм, сверху и снизу – 20 мм. Использовать шрифт *TimesNewRoman;* размер шрифта – 14; межстрочное расстояние – 1,5, выравнивание по ширине. Абзацный отступ – 1,25 см.

В начале реферата приводится его оглавление, которое должно включать все разделы и подразделы работы с указанием страниц начала каждого раздела и подраздела (прилож. 2).

Все разделы и подразделы реферата должны иметь заголовки и обязательно нумеруются.

Заголовки разделов и подразделов следует записывать с красной строки с прописной буквы, не подчеркивая, например:

1 Анализ существующих решений по заданной предметной области

Переносы слов в заголовке не допускаются. В конце заголовка точка не ставится. Если заголовок состоит из двух предложений, их разделяют точкой.

Отступ между заголовком и текстом должно быть 15 пт, а между заголовками раздела и подраздела – 8 пт. (*правая кнопка мыши →меню Абзац→ Интервал После – 15 пт)*

Каждый раздел начинается с новой страницы.

Разделы должны иметь порядковые номера, обозначенные арабскими цифрами, в пределах всей работы, после цифры **НЕ** ставится точка, а текст начинается с заглавной буквы.

Подразделы должны иметь нумерацию в пределах каждого раздела. Номер подраздела состоит из номеров раздела и подраздела, разделенных точкой,

например: 1.2; 1.3 и т.д. Нумерация пунктов должна быть в пределах подраздела. Номер пункта должен состоять из номеров раздела, подраздела и пункта, разделенных точками, например: 1.2.1 и т.д., например:

1 Анализ существующих решений по заданной предметной области

Некоторый вводный текст (2 или 2 абзаца)

1.1 Принцип действия и структурная схема существующей системы

Некоторый вводный текст

1.1.1 Принцип действия существующей системы

Текст о принципе действия существующей системы

1.1.2 Структурная схема существующей системы

Текст о структурной схеме существующей системы. После номера пункта до конца страницы должно быть не менее 3-х строк. В противном случае пункт надо переносить на следующую страницу.

Страницы работы нумеруются арабскими цифрами, начиная со второй.

Текст должен быть кратким, четким и не допускать различных толкований. В тексте не допускаются сокращения слов, кроме общепринятых.

Перечисление некоторой информации оформляется следующим образом:

К параметрам непрерывного преобразования следует отнести:

1) выбор значений масштабного коэффициента a, по которому производится разложение;
2) шаг изменения масштабного коэффициента;
3) выбор коэффициента обратного преобразования.

Если после номера ставится точка, тогда нумерованный список оформляется следующим образом:

Согласно [24] в основе диагностики оборудования по параметрам механических колебаний лежат два утверждения:

1. Все работающее оборудование вибрирует, что связано с неточностью изготовлению, сборки, монтажа;

2. Вибрационные процессы вращающегося оборудования несут в себе полную информацию о характере дефекта, его локализации и степени развития.

В тексте реферата ДОЛЖНЫ ПРИСУТСТВОВАТЬ ссылки на источники, приведенные в списке литературы. После упоминания источника, в квадратных скобках проставляют номер, под которым этот источник значится в списке[25].

Таблицы, используемые в работе (за исключением таблиц приложения), помещаются в соответствии с логикой изложения и нумеруются арабскими цифрами в пределах каждой главы.

По центру строки без отступа абзаца пишется:

Таблица 1 – Название таблицы

Заголовки граф и строк таблицы начинаются с прописных букв, заголовки подграф – со строчных. Высота строк в таблице должна быть не менее 8 мм (см. *Образец оформления таблицы*).

Иллюстрации могут быть расположены как по тексту, так и в приложении. Их следует нумеровать арабскими цифрами сквозной нумерацией, за исключением иллюстраций приложений. Можно использовать сквозную нумерацию рисунков по всему тексту реферата (Рисунок 1, Рисунок 2 и т.д.). Допускается нумеровать иллюстрации в пределах раздела, например: «Рисунок 1.1». Иллюстрации должны иметь наименование, которое должно располагаться под ним (см. *Образец оформления рисунков*). Рисунок должен располагаться ниже текста документа, где первый раз упоминается о нем. НЕ ДОПУСКАЕТСЯ, чтобы иллюстрация и подпись к ней располагались на разных страницах!

На каждую таблицу и рисунок должна быть ссылка в тексте с анализом приводимых данных.

Формулы, содержащиеся в тексте, выполняются с помощью **редактора формул!**!! Формулы располагаются на отдельных строках в начале строки с отступом и имеют нумерацию в пределах раздела. Номер формулы состоит из

номеров раздела и номера формулы, заключенных в круглые скобки. Номер формулы помещается в конце строки. Под формулой приводится расшифровка символов и числовых коэффициентов, если они не были пояснены ранее в тексте. Первая строка расшифровки начинается словом «где» без двоеточия после него. Выше и ниже каждой формулы должен быть интервал не менее 6 пт.

Пример: Зная коэффициент температурной нестабильности, можно найти величину приращения коллекторного тока $\Delta I_к$ при изменении температуры в заданном интервалеΔT по формуле:

$$\Delta I_к = S \cdot \left[\Delta I_{к0} + \frac{\varepsilon \cdot \Delta T}{R_э + R_б} + \left(I_б + I_{к0}\right) \frac{\Delta h_{21э}}{h_{21э}} \right], \qquad (1.2)$$

где $R_б = \dfrac{R_1 \cdot R_2}{R_1 + R_2}$, $\varepsilon = -2{,}5$ мВ/град.

ГДЕ *пишется с начала строки без отступа!!!*

Ссылки на разделы, подразделы, пункты, формулы, таблицы, рисунки следует указывать их порядковым номером, например: «в разделе 1», «в подразделе 1.2», «по формуле (1.2)», «по данным таблицы 1.2», «на рисунке 1.1».

Текст, таблицы, иллюстрации вспомогательного материала рекомендуется оформлять в приложениях. Приложение оформляют как продолжение пояснительной записки, располагают на отдельных страницах и помещают после списка литературы. Каждое приложение следует начинать с новой страницы с указанием наверху справа слова «Приложение», после которого следует номер (арабскими или римскими цифрами). Если в работе используется только одно приложение, оно обозначается без номера.

Приложение должно иметь заголовок, который записывают симметрично тексту с прописной буквы отдельной строкой. Все приложения должны быть перечислены в содержании документа с указанием их номеров и заголовков. В тексте пояснительной записки на все приложения должны быть даны ссылки, например: «в приложении 1», «(приложении 4)».

Нумерация страниц приложений продолжает общую нумерацию работы.

В целом эти материалы должны наглядно отражать объекта особенности исследования, цели и задачи ВКР, результаты проведенного научного исследования.

Оформление списка литературы

Образец

Список литературы

Книга одного автора

1. Витязев, В.В. Вейвлет-анализ временных рядов: Учебное пособие. – СПб.: Изд-во С.-Петерб. ун-та, 2001. – 58 с.

Книга двух и более авторов

2. Баркова, Н.А. Неразрушающий контроль технического состояния горных машин и оборудования: учебное пособие. / Н.А. Баркова, Ю.С. Дорошев. – Владивосток: Изд-во ДВГТУ, 2009. – 157 с.

Статья из журнала одного автора

3. Астафьева, Н.М. Вейвлет-анализ: основы теории и примеры применения// Успехи физических наук. – 1996. –Т. 166, №11. – С. 1145 – 1170.

Статья из журнала двух и более авторов

4. Баданин, Е.Ю., Дрозденко В.А. Диагностика и анализ вибрационного состояния ГЦН энергоблока БН-600 / Е.Ю. Баданин, В.А. Дрозденко // Известия высших учебных заведений. Ядерная энергетика. - 2009. - N 2. - С. 30-34

ГОСТ

5. ГОСТ 16504-81 Система государственных испытаний продукции. Испытания и контроль качества продукции. Основные термины и определения. М.: Стандартинформ, 2011. – 23 с.

Патент

6. Патент РФ 2007113529/28, 11.04.2007. Костюков В.Н., Науменко А.П., Бойченко С.Н. Способ вибродиагностики технического состояния поршневых машин по спектральным инвариантам // Патент России № 2337341. 2008. Бюл. №30.

Электронный ресурс

7. Скворцов В. Разделение школьных предметов на образовательные и воспитательные – ошибка: челябинский эксперт [Электронный ресурс] // http://regnum.ru. [2011]. URL: http://regnum.ru/news/cultura/1374311.html (дата обращения: 03.03.2011).

Оформление оглавления реферата

Образец

Оглавление

Образец оформления материала в форме таблицы

Таблица 1.1 – Рабочие параметры роторных машин

Наименование оборудования	Рабочие параметры
Паровая турбина	Температура, давление, выходная мощность, частота вращения, расход масла, давление масла, производительность, крутящий момент
Газовая турбина	Температура, давление, расход топлива, давление масла, расход масла, частота вращения, производительность
Насос	Температура, давление, частота вращения, производительность, входная мощность, давление масла, расход масла
Компрессор	Температура, давление, отношение давлений, входная мощность, крутящий момент, частота вращения, производительность
Электрогенератор	Температура, давление, электрический ток, напряжение, сопротивление, входная мощность, выходная мощность, крутящий момент, частота вращения

При увеличении входного напряжения $\Delta U_{вх}$ увеличение напряжения стабилитрона (нагрузки) получается небольшим. Это обусловлено тем, что при небольшом увеличении напряжения стабилитрона происходит значительное увеличение тока стабилитрона (см. $\Delta U_{н}$ и $\Delta I_{ст}$ на рис. 1), в результате чего происходит значительное увеличение напряжения на балластном сопротивлении $U_{Rб}$ (примерно равное увеличению $\Delta U_{вх}$). Аналогично происходит и при уменьшении входного напряжения.

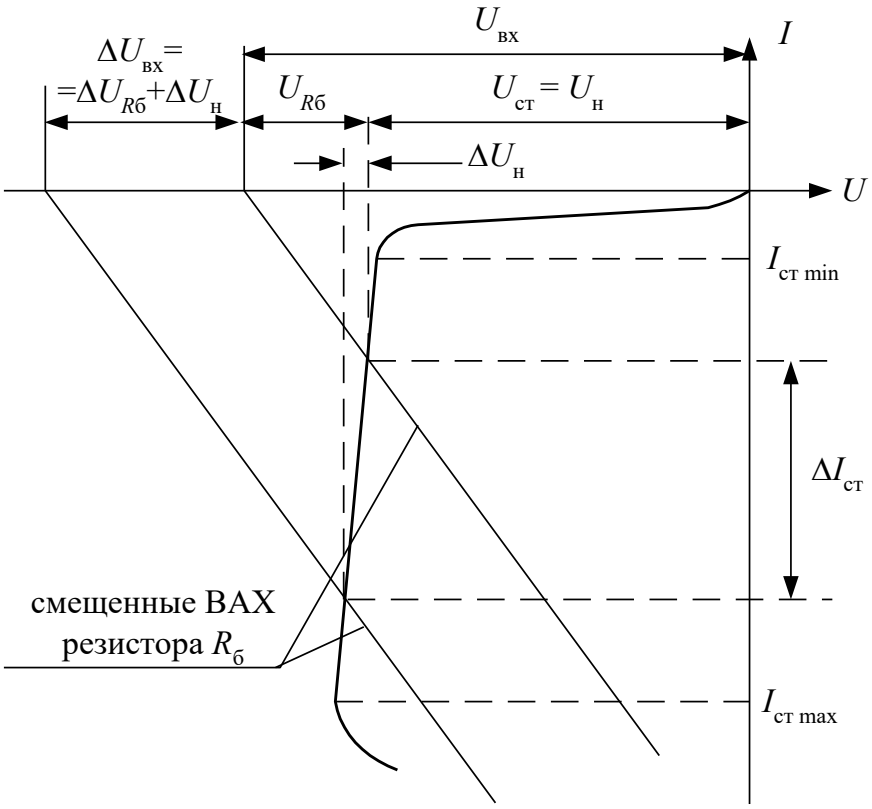


Рисунок 1 – Вольтамперная характеристика стабилитрона