|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

Факультет: «Специальное машиностроение»

Кафедра: СМ11 «Подводные аппараты и роботы»

**Лабораторная работа №2**

по курсу «Электронные устройства мехатронных и робототехнических систем»

по теме «Рассмотрение принципов работы микроконтроллеров при помощи стенда EASYAVR 5»

Выполнили: Бочкарев М. А.

Группа: СМ11-61Б

Проверил:

Москва 2024 г

**Изучение структуры стенда EasyAVR 5 и САПР AVRStudio**

**1 Цель работы**

Изучить функциональные возможности учебно-отладочного стенда, внутреннюю структуру и систему команд МК ATmega16. Получить первичные навыки программирования МК ATmega16.

**2 Краткие теоретические сведения**

2.1 Память

В соответствии с гарвардской архитектурой память AVR-микроконтроллера разделена на две области: память данных и память программ. Кроме того, ATmega16 содержит память на EEPROM(ЭСППЗУ) для энергонезависимого хранения данных. Все три области памяти являются линейными и равномерными.

Внутрисистемно программируемая флэш-память программ ATmega16 содержит 16 кбайт внутренней внутрисистемно перепрограммируемой флэш-памяти для хранения программы. Поскольку все AVR-инструкции являются 16 или 32-разрядными, то флэш-память организована как 8 кбайт × 16.

2.2 Порты ввода/вывода

В составе данного МК имеет четыре 8-разрядных двунаправленных порта.

Для обслуживания каждого порта отведено три регистра: регистр данных PORTx, регистр направления данных - DDRx и регистр выводов порта PINx (вместо х подставляется соответственно буквы A, B, C, D). Регистр выводов порта предназначен только для чтения, в то время как регистр данных и регистр направления данных - для чтения/записи.

Все выводы портов имеют отдельно подключаемые подтягивающие резисторы. Выходы порта могут поглощать ток до 20 мА и непосредственно управлять светодиодными индикаторами. Выводы используются как входы и замыкаются на землю, если включены внутренние подтягивающие резисторы, при этом выводы являются источниками тока (IIL).

Все 8 бит каждого из портов при использовании для ввода/вывода одинаковы.

Бит DDxn регистра DDRx выбирает направление передачи данных. Если бит установлен (1), вывод сконфигурирован как выход. Если бит сброшен (0) - вывод сконфигурирован как вход. Если PORTxn установлен и вывод сконфигурирован как вход, включается КМОП подтягивающий резистор. Для отключения резистора, PORTxn должен быть сброшен (0) или вывод должен быть сконфигурирован как выход (см. таблицу 1).

Таблица 1 – Влияние DDxn на выводы порта

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| DDxn | PORTxn | Вх/Вых | Подт.резист | Комментарий |
| 0 | 0 | Вход | Нет | Третье состояние (Hi-Z) |
| 0 | 1 | Вход | Да | Pxn источник тока IIL, если извне соединен с землей |
| 1 | 0 | Выход | Нет | Выход установлен в 0 |
| 1 | 1 | Выход | Нет | Выход установлен в 1 |

n = 7,6...0 - номер вывода.

Стенд Easyavr5 представляет собой лабораторный макет с микроконтроллером. К каждому выводу микроконтроллера подключена кнопка и светодиод. На макете установлено четыре семисегментных индикатора. Также имеется возможность подключения к макету внешних устройств через специальные устройства. Питание макета осуществляется либо от внешнего источника напряжения 8–16В, либо через USB-порт. Кроме этого на макете установлен внутрисхемный программатор.

В лабораторных работах будет использоваться микроконтроллер фирмы ATMEL – ATmega16.

Условные обозначения:

* Flash ROM - объем энергонезависимой памяти программ (в килобайтах);
* EEPROM - объем энергонезависимой памяти данных (в байтах);
* RAM - объем статической памяти данных (в байтах);
* External RAM - возможность подключения к микроконтроллеру дополнительной микросхемы внешней статической памяти данных (в килобайтах);
* ISP - возможность программирования микроконтроллера в системе (на целевой плате) при основном напряжении питания;
* SPM - функция самопрограммирования Flash ROM памяти микроконтроллера в системе без участия внешнего программатора;
* JTAG - встроенный JTAG - интерфейс;
* I/O (pins) - максимальное количество доступных линий ввода / вывода;
* Timer(s) 8/16 bit - количество и разрядность таймеров/счетчиков;
* USI - универсальный коммуникационный интерфейс;
* AC - аналоговый компаратор;
* ADC (channels) - количество каналов аналого-цифрового преобразования;
* Internal RC - наличие внутренней RC-цепочки для автономной работы микроконтроллера (без внешнего источника опорной частоты);
* WDT - сторожевой таймер;
* BDC - аппаратный программируемый блок защиты от сбоев при внезапном (в том числе и кратковременном) пропадании напряжения питания микроконтроллера;
* UART - асинхронный последовательный приемопередатчик;
* SPI - синхронный трехпроводной последовательный интерфейс;
* I2C - двухпроводной последовательный интерфейс;
* RTC - система реального времени;
* PWM (channels) - количество независимых каналов широтно - импульсной модуляции;
* Command Set - количество различных инструкций в системе команд микроконтроллера;
* Vcc - диапазон рабочих напряжений питания (в Вольтах);
* Clock - диапазон рабочих частот (в мегагерцах);

**Блок-схема ATmega16**

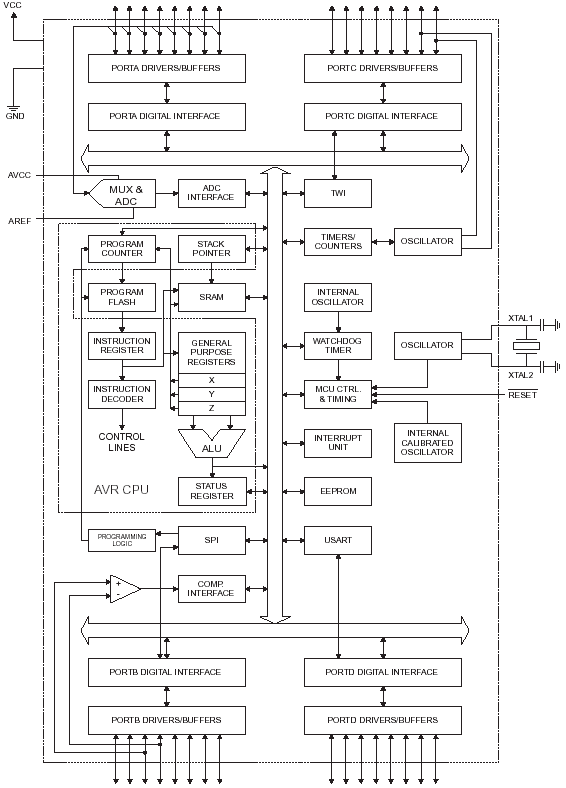
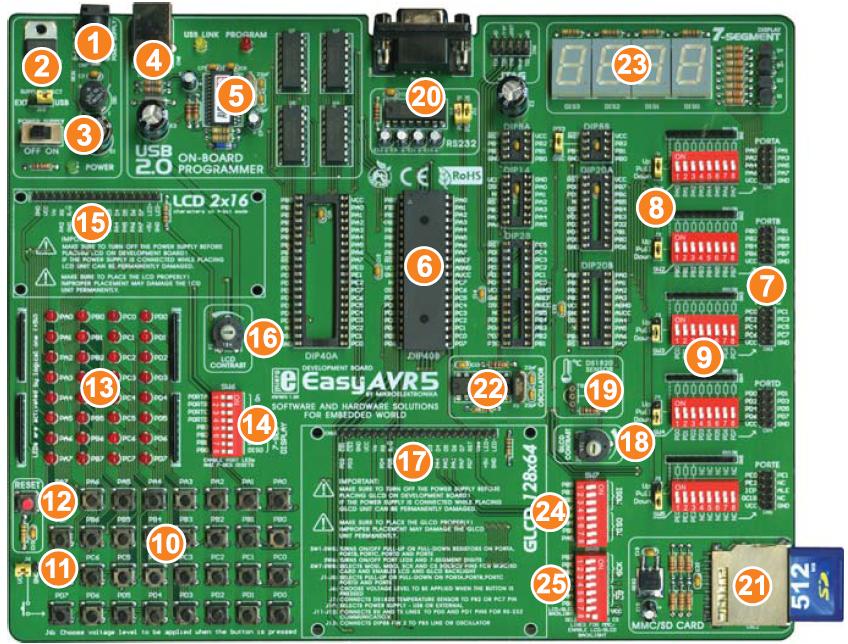


Рисунок 1 – Блок схема ATmega16

**Работа со стендом EasyAVR5**

****

1 – разъем для подключения внешнего питания 8–16В;

2 – выбор между внешним питанием и питанием от USB;

3 – выключатель;

4 – разъем USB;

5 – встроенный внутрисхемный программатор;

6 – панельки для установки различных типов МК;

7 – разъемы для доступа к портам МК;

8 – джамперы для выбора типа подтягивающих резисторов (к «1» или к «0»);

9 – группы переключателей для подключения подтягивающих резисторов к выводам МК;

10 – 32 кнопки, подключенные к выводам МК;

11 – джампер выбора высокого или низкого состояния при нажатии кнопки:

12 – кнопка сброса;

13 – светодиоды, каждый подключенный к выводу МК;

14 – группа переключателей, подключающая или отключающая светодиоды к МК;

15 – разъем для подключения символьной ЖК-матрицы;

16 – потенциометр для регулировки яркости символьной ЖК-матрицы;

17 – разъем для подключения графической ЖК-матрицы;

18 – потенциометр для регулировки яркости графической ЖК-матрицы;

19 – разъем для подключения датчика температуры DS1820;

20 – разъем порта RS232;

21 – слот для подключения MMC/SD карт памяти;

22 – встроенный генератор частоты;

23 – 4 семисегментных индикатора;

24 – группа переключателей для разрешения внутрисхемного программирования;

25 – группа переключателей для разрешения работы с MMC/SD картами памяти.

Рисунок 2 – Внешний вид стенда EasyAVR5

Внешний вид стенда представлен на рисунке 2. Данный стенд позволяет подключить к каждому выводу микроконтроллера светодиод, кнопку или подать на него любой внешний сигнал. На стенде предустановленны 4 семисегментных индикатора.

Схема подключения светодиодов к выводам МК представлена на рисунке 3.

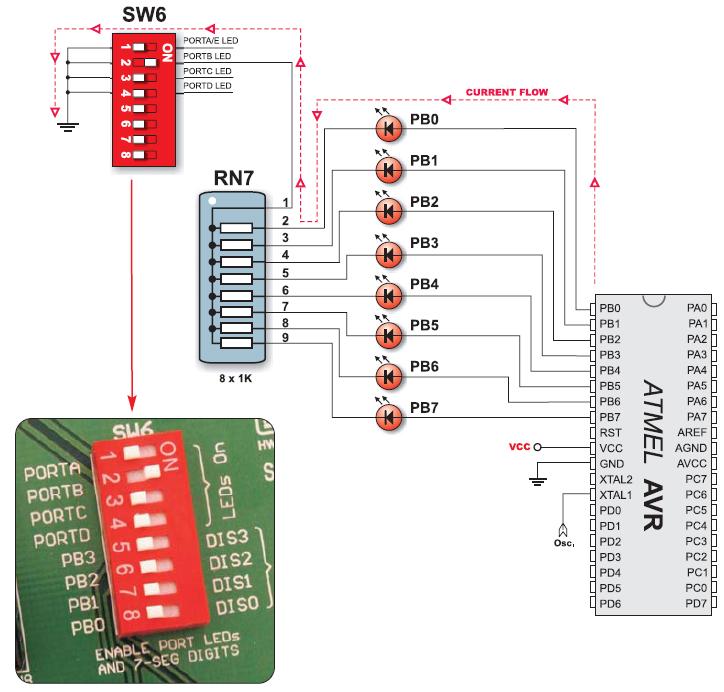


Рисунок 3 – Схема подключения светодиодов к выводам МК

Подключение светодиодов осуществляется группами одновременно к каждому из портов. Управление подключением осуществляется при помощи переключателя SW6.

Схема подключения кнопок к портам микроконтроллера представлена на рисунке 4.

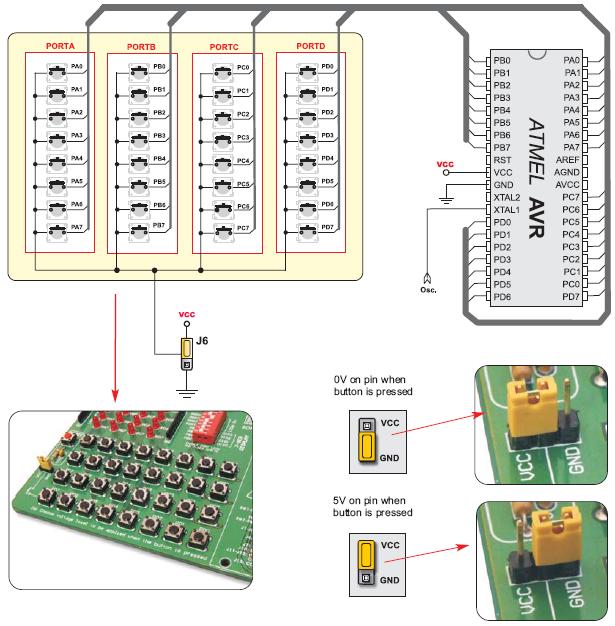


Рисунок 4 –Схема подключения кнопок к выводам МК

К каждому выводу микроконтроллера подключена кнопка. При нажатии кнопки на вывод микроконтроллера может быть сигнал либо нулевого, либо единичного уровня. Выбор уровня осуществляется при помощи джампера J6.

Схема подключения семисегментных индикаторов к портам микроконтроллера представлена рисунке 5.

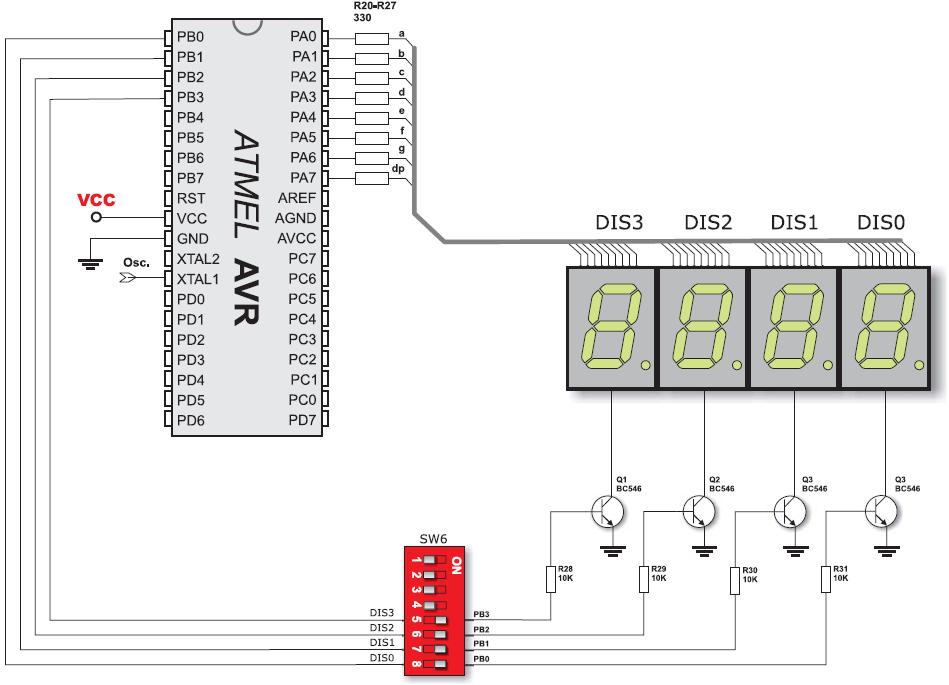


Рисунок 5 – Схема подключения семисегментных индикаторов к портам МК

В зависимости от типа макета семисегментные индикаторы подключаются либо к порту А, либо к порту С. Управление их работой происходит при помощи выводов порта В. Разрешение управления проводится группой переключателей SW6.

Загрузка исполняемой программы в микроконтроллер стенда проводится при помощи программы AVRFLASH Programmer. Перед программированием необходимо подключить стенд к USB-порту компьютера. Главное окно программы представлено на рисунке 6.

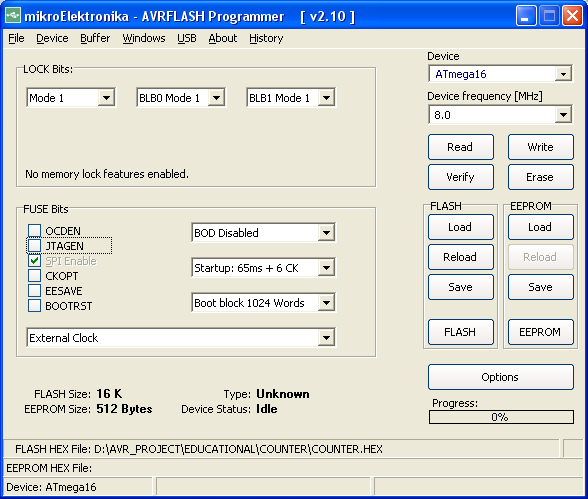


Рисунок 6 – Главное окно программы AVRFLASH Programmer

Перед программирование в правом внешнем углу окна необходимо выбрать тип микроконтроллера (ATMega16) и рабочую частоту устройства (8 МГц). Па панели FUSE Bits необходимо выставить параметры, которые указаны на рисунке. На панели FLASH при помощи кнопки Load выбирается hex-файл, который будет загружен в память программ. Далее производится запись, для чего необходимо нажать кнопку Write. Перед программированием обязательно провести стирание устройства при помощи кнопки Erase.

Практическая часть

Дополнительное задание

Описание:

Наша задача – при нажатии кнопки подсвечивать поочередно по парам диоды, подключенные к выходам PB0, PB2 и PB1, PB3. Для этого будем использовать переменную Counter, которая будет указывать, какое состояние у диодов должно быть на данный момент. При Counter = 0 диоды не будут гореть, при Counter = 1 горит первая пара, при Counter = 2 горит вторая пара. По достижению значения 3 происходит сброс счетчика до значения 1, и после второй пары снова загорается 1. Также введены задержки для подавления дребезга контактов при нажатии кнопок.

Вариант программы с комментариями будет представлен при защите.

Практическая часть

Пункт 1 – выполнение дополнительного задания по вариантам

По условию задания, необходимо поочередно включать диоды, подключенные к портам PA0, PA3 и PA1, PA2. Время переключения должно случайным образом варьироваться от 4 до 12 секунд.

Программа начинается с инициализации портов и некоторых регистров, а также объявления переменных

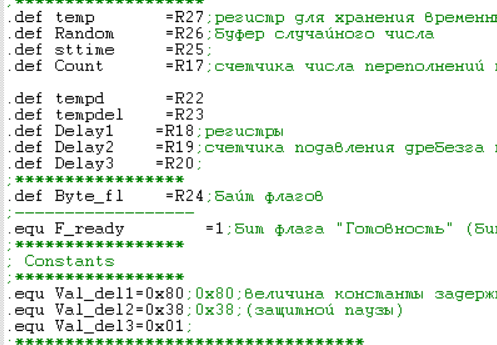


Рисунок 1 – Объявление переменных и констант

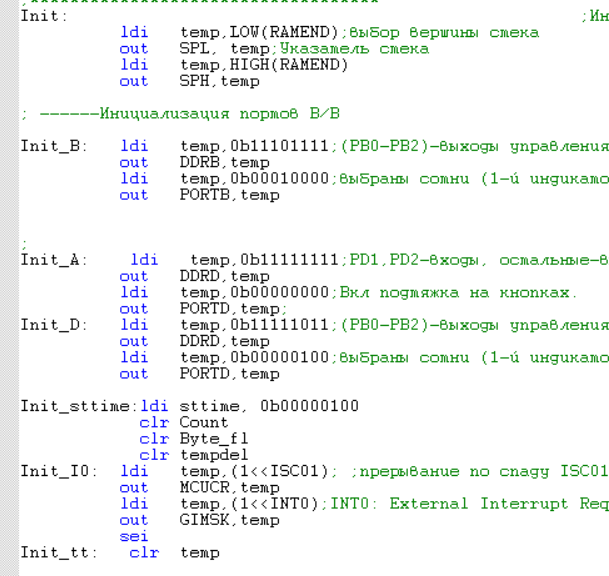


Рисунок 2 – Блок инициализации МК

После инициализации прописываем основной цикл программы, включающий в себя переменную count, которая будет указывать, какую пару светодиодов необходимо включить в данный момент времени. За каждую итерацию ее значение будет увеличиваться на 1, а после достижения 3 сбрасываться снова до 1, указывая на первую пару.

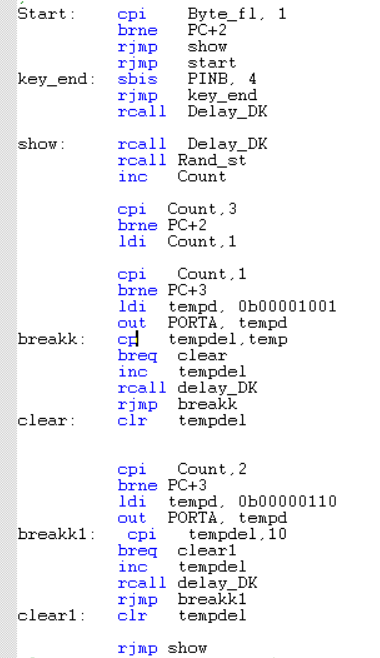


Рисунок 3 – Основной цикл программы

Для случайного времени переключения нам необходима программа получения псевдослучайного числа в диапазоне от 90 до 256

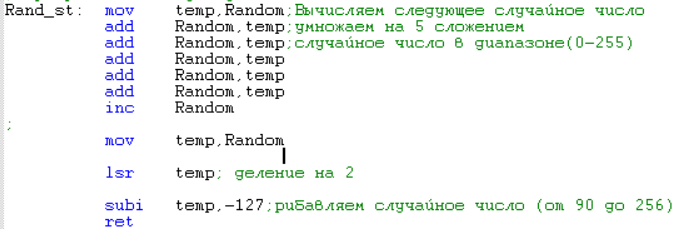


Рисунок 4 – получение псевдослучайного числа от 90 до 256

Для задержки будем вызывать подпрограмму Delay\_DK рандомное кол-во раз при помощи числа в регистре temp. Остановку и начало работы реализуем через вектор внешнего прерывания INT0

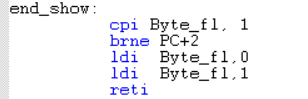


Рисунок 5 – Подпрограмма обработки прерывания

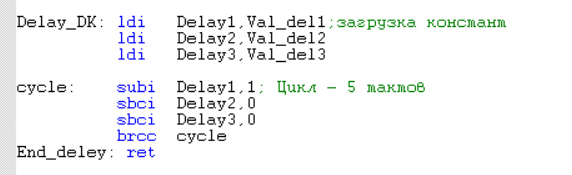


Рисунок 5 – задержка для подавления дребезга контактов

Пункт 2 – выполнение основного задания лабораторной работы номер 2.

Лабораторная работа включала несколько задач

1 – исправление ошибки

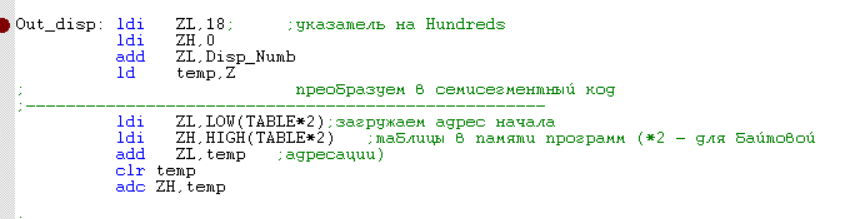


Рисунок 7 – код с исправленной ошибкой

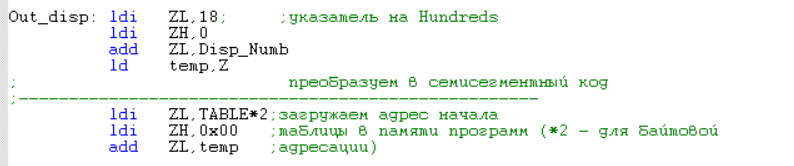


Рисунок 8 – исходный код

Ошибка заключалась в неправильной записи 2-ух байтовой адресации в регистры ZH и ZL.

2 – тест программы на плате

При нажатии кнопки раньше светодиода или ненажатии кнопки после загорания светодиода в течение опред времени на семисегментом индикаторе возникает условное обозначение ошибки (BAD). В ином случае на индикаторах увидим время нашей реакции

3 – Доработка до 2-ух байтовой адресации

Добавляем дополнительную переменную Disp\_CountH, за счет которой будет реализовывать расширение адресации еще на 1 байт

Задаем ей начальное значение времени включения Val\_dispCountH=255,

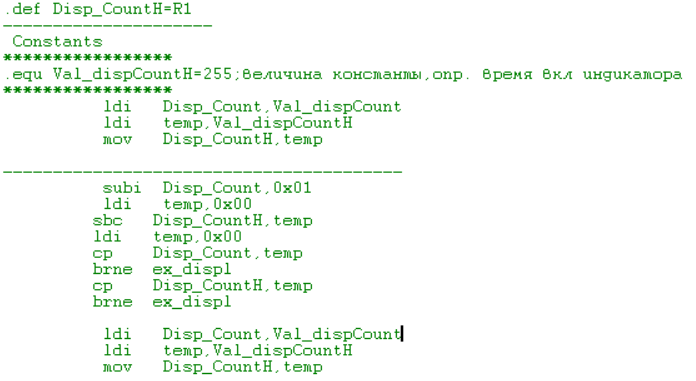


Рисунок 9 – код для расширения адресации до 2 байт

5 - Экспериментально подберите константу Val\_dispCount, определяющую время включения индикатора, когда глаз не воспринимает мигание индикаторов.

Изменяем значения переменных Val\_dispCount и Val\_dispCountH, уменьшая их, пока отображение на индикаторе не потеряет сплошность



Рисунок 10 – Предельные значения констант

При значениях меньше данных уже можно наблюдать мигания на семисегментных индикаторах

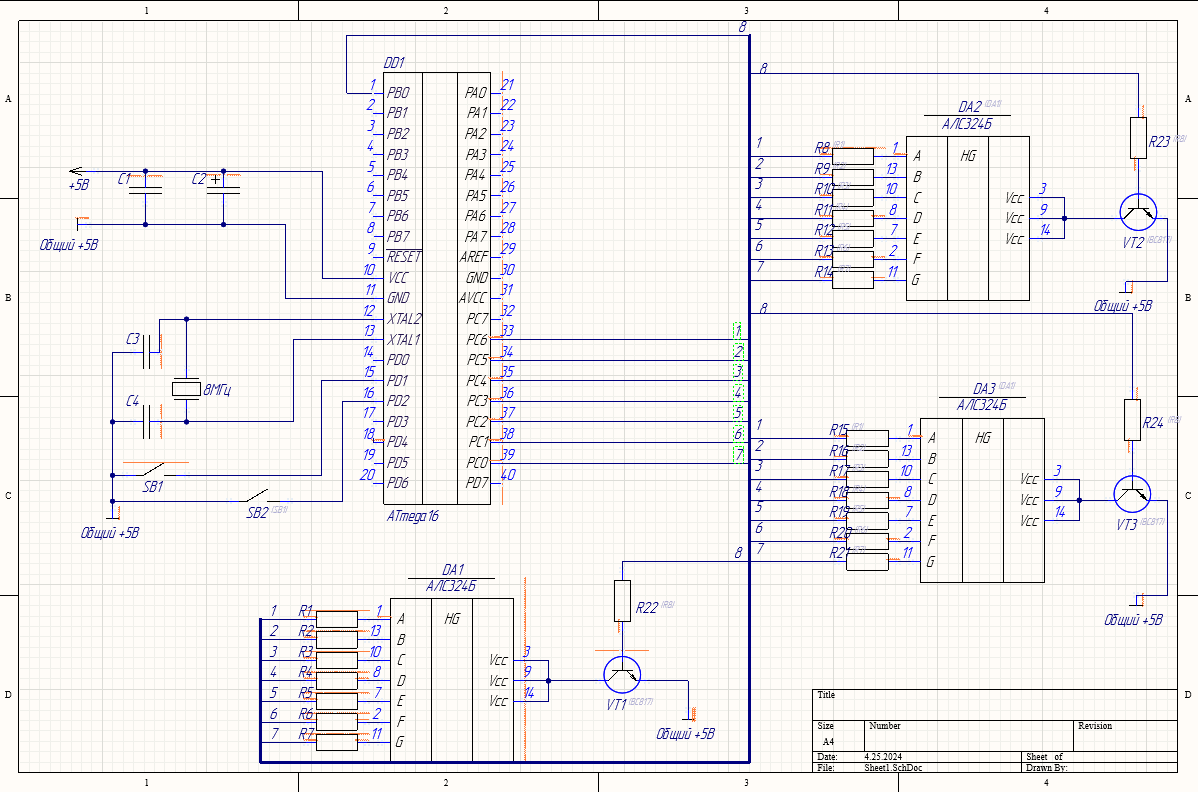


Рисунок 11 – схема Э3 для системы

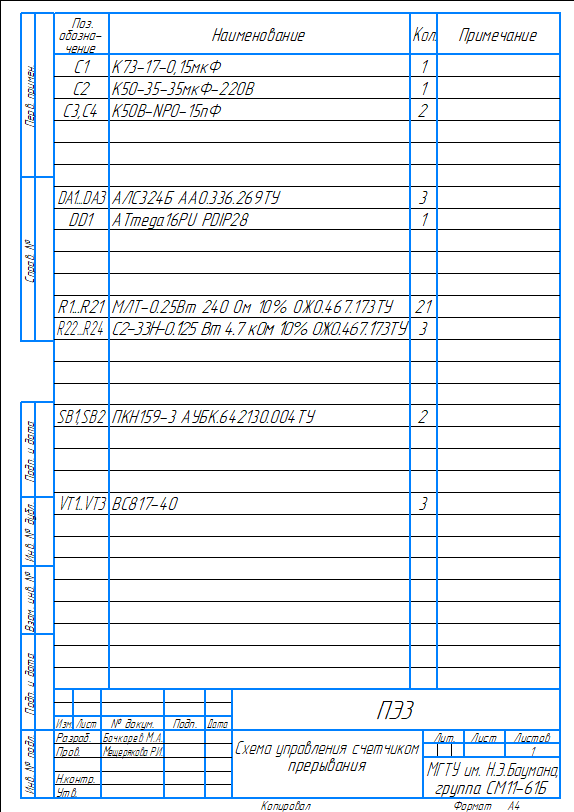


Рисунок 12 – перечень элементов

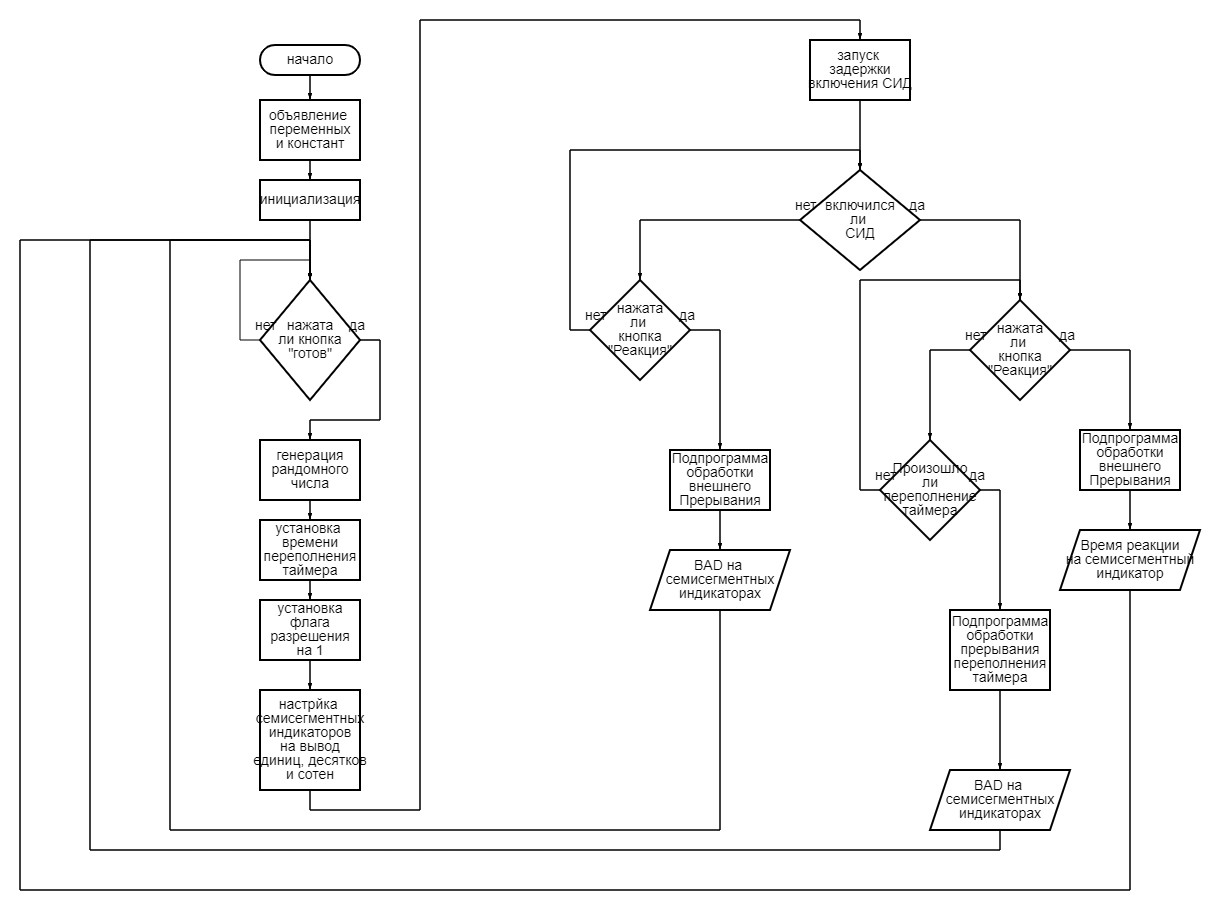


Рисунок 12 – Блок-схема основной программы