Министерство науки и образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования

«Московский государственный технический университет им. Н.Э. Баумана  
(национальный исследовательский университет)»  
(МГТУ им. Н.Э. Баумана)

****

Факультет ***Информатика и системы управления***

Кафедра ***Компьютерные системы и сети (ИУ6)***

проектирование регистрирующей мк-системы

Расчетно-пояснительная записка на курсовую работу

по дисциплине Микропроцессорные системы

Листов \_\_

Студент гр. ИУ6-71 **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ П. А. Краскова**

(Подпись, дата) (И.О. Фамилия)

Руководитель курсовой работы,

к.т.н., доцент  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В. Я. Хартов**

(Подпись, дата) (И.О. Фамилия)

Москва, 2017

РЕФЕРАТ

Расчётно-пояснительная записка 43 с., 17 рис., 8 табл., 10 источников, 2 прил.

МИКРОПРОЦЕССОР, МИКРОКОНТРОЛЛЕР , AT90S8515, ТАЙМЕР, РЕГИСТРАТОР

Объектом проектирования является разработка регистратора времени работы устройств.

Цель работы — создать рабочие программы, соответствующие требованиям ТЗ и продемонстрировать работу системы.

Задачи, решаемые в процессе проектирования: анализ альтернативных решений задачи; выбор схемотехнического решения и элементной базы; расчет потребляемой мощности устройства; разработка алгоритма управления и соответствующей программы микроконтроллера.

**Список используемых сокращений**

МК – микроконтроллер

ПЭВМ – персональная электронно-вычислительная машина

LCD – жидкокристаллический дисплей

PC (Program Counter) – программный счётчик

РОН (General purpose registers) – регистры общего назначения

АЛУ (ALU) – арифметико-логическое устройство

SREG (status register) – статусный регистр

MCUCR (MCU control register) – регистр управления

Содержание

[Введение 6](#_Toc501434740)

[1 Анализ требований 7](#_Toc501434741)

[1.1 Разработка структурной схемы 7](#_Toc501434742)

[1.2 Выбор микроконтроллера 8](#_Toc501434743)

[2 Разработка функциональной схемы 9](#_Toc501434744)

[2.1 Архитектура микроконтроллера AT90S8515. 9](#_Toc501434745)

[2.2 Пульт оператора 14](#_Toc501434746)

[2.3 Блок передачи в ПЭВМ 14](#_Toc501434747)

[2.4 Подключение цепей питания 15](#_Toc501434748)

[3 Разработка принципиальной схемы 16](#_Toc501434749)

[3.1 Конфигурация выводов микроконтроллера 16](#_Toc501434750)

[3.2 Подключение кнопок и ЖК-дисплея 16](#_Toc501434751)

[3.3 Подключение драйвера MAX202CSE 17](#_Toc501434752)

[3.4 Синтез принципиальной схемы 19](#_Toc501434753)

[4 Расчёт параметров 20](#_Toc501434754)

[4.1 Расчёт потребляемой мощности 20](#_Toc501434755)

[5 Описание алгоритмов работы программ 21](#_Toc501434756)

[6 Технологическая часть 27](#_Toc501434757)

[6.1 Отладка программы микроконтроллера 27](#_Toc501434758)

[6.2 Способы программирования микроконтроллера 29](#_Toc501434759)

[Заключение 35](#_Toc501434760)

[Список использованной литературы 36](#_Toc501434761)

ПРИЛОЖЕНИЕ А – Исходный текст программы микроконтроллера

ПРИЛОЖЕНИЕ Б – Спецификация радиоэлементов принципиальной схемы

# Введение

На основании учебного плана кафедры ИУ6 поставлена задача разработать регистратор времени работы устройств (сигналы приходят с 8 внешних устройств) на микроконтроллере Atmel AVR AT90S8515. В процессе выполнения работы проведён анализ технического задания, разработано программное обеспечение для оборудования.

Разработка регистратора времени состоит из трёх основных частей:

* аппаратной части – коммутации микроконтроллера, элементов пульта оператора и выходных буферов генератора;
* программной части – программа микроконтроллера, в соответствии с которой он обеспечивает генерацию сигналов, управление и отображение информации;
* технической части – отладка программы микроконтроллера.

Во время выполнения курсовой работы необходимо построить структурную, функциональную и принципиальную схемы, разработать алгоритмы работы системы, на основе которых программируется микроконтроллер, а также рассчитать потребляемую мощность.

# 1 Анализ требований

## 1.1 Разработка структурной схемы

Согласно заданию необходимо разработать регистратор времени работы 8 электроприборов в течение суток до полудня (с 0 до 12 часов) и после полудня (с 12 до 24 часов).

Таким образом, потребуется использовать внутренний таймер микроконтроллера. Для обеспечения дополнительной точности и надежности тактирования схемы, подключим внешний кварц с частотой 8MHz к разъемам XTAL1 и XTAL2.

Устройство будет управляться с пульта оператора, состоящего из 3 кнопок: «Вывести на экран», «Следующее устройство» и «Предыдущее устройство».

В качестве ЖК-дисплея используем LM032L. Он позволяет отображать 2 строки по 16 символов в каждой.

Для передачи данных в ПЭВМ будем использовать канал UART с применением драйвера MAX202CSE.

На рисунке 1 представлена структурная схема устройства.

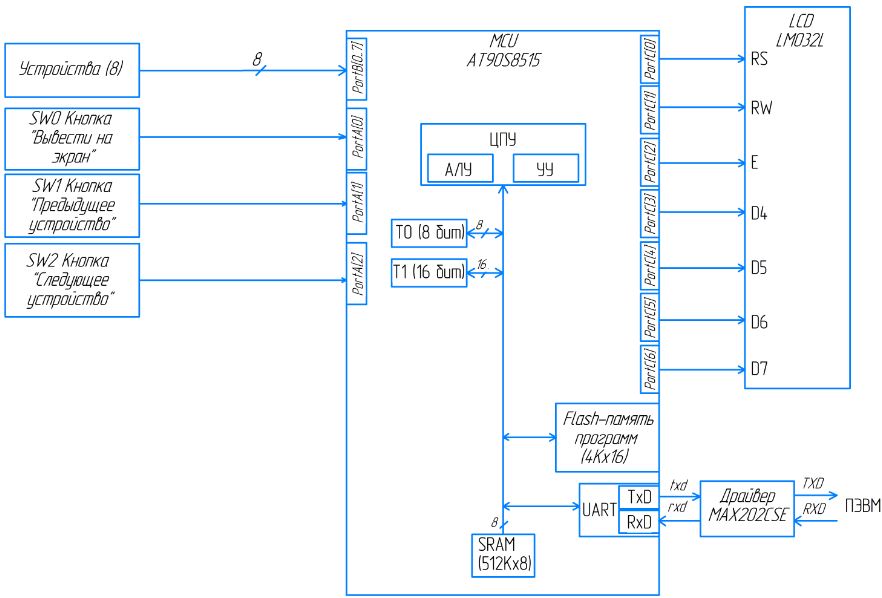


Рисунок 1 – Структурная схема устройства

По результатам анализа требований к микроконтроллерной системе, можно сформулировать перечень блоков, необходимых для реализации устройства соответствующего требованиям, указанным в техническом задании.

## 1.2 Выбор микроконтроллера

Согласно заданию регистратор времени должен быть разработан на основе микроконтроллера семейства AVR. В таблице 1 представлено сравнение важных для нашей системы параметров МК AVR семейства classic [1].

Таблица 1 – Сравнение МК AVR семейства classic

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Тип | Напр. пита-ния (В) | Такт. Час-тота (МГц) | I/O | Flash | EEPROM | SRAM | Ин- тер- фей- сы | Тай-меры | I SP | Кор- пус |
| AT90LS2323 | 2.0-6.0 | 4 | 3 | 2K | 128 | 129 | - | 1x8bit | I | DIP8 SO8 |
| AT90S4433 | 4.0-6.0 | 8 | 20 | 4K | 256 | 128 | UART SPI | 6x10bit | I | DIP28 TQFP32 |
| AT90S2343 | 4.0-6.0 | 10 | 5 | 2K | 128 | 128 | - | 1x8bit | I | DIP8 SO8 |
| AT90S858515 | 2.7-6.0 | 8 | 32 | 8K | 512 | 512 | UART SPI | 2x8bit 1x16bit | I | DIP40 TQFP44 PLCC44 |

Часть МК из семейства classic были опущены в таблице 1, так как они включают в себя АЦП (аналого-цифровой преобразователь), который не был использован при разработке. Каждый из вышеуказанных МК включают в себя ISP (позволяет программировать микроконтроллер на целевой плате при основном напряжении питания), что позволит задать алгоритмы программы системы, интерфейс UART, который необходим для передачи данных в ПЭВМ.

Микроконтроллер AT90S8515 содержит наибольшее количество доступных линий ввода-вывода, обладает большей памятью FLASH, EEPROM и SRAM, поэтому для разработки регистратора времени был выбран данных МК.

# 2 Разработка функциональной схемы

На основе разработанной структурной схемы (раздел 1) и более подробного анализа выбранных компонентов МК (описанного ниже) разработана Схема электрическая функциональная.

Функциональная схема проекта состоит из структурных блоков:

1. Микроконтроллер Atmel AVR AT90S8515 (см. подраздел 2.1).
2. Пульт оператора (см. подраздел 2.2).
3. Блок передачи в ПЭВМ (см. подраздел 2.3).

## 2.1 Архитектура микроконтроллера AT90S8515.

Основным узлом разрабатываемого устройства является микроконтроллер AT90S8515 фирмы Atmel. Структурная схема МК приведена на рисунке 2 [6].

Ядро микроконтроллера AVR совмещает богатый набор инструкций с 32-мя регистрами общего назначения, каждый из которых непосредственно соединен с АЛУ, что позволяет двум независимым регистрам взаимодействовать в рамках одной инструкции, выполняющейся за один такт.

Микроконтроллер AT90S8515 поддерживается всеми основными средствами программной и аппаратной разработки, моделирования и отладки.

Перечислим особенности данного микроконтроллера, которые нашли применение в данной разработке:

* использование AVR RISC архитектуры, совмещающей высокую производительность с низким энергопотреблением. Большинство команд этого МК выполняются за один такт. Микроконтроллер функционирует на частотах до 8 МГц. AT90S8515 обеспечивает производительность 1 млн. операций в секунду на 1 МГц синхронизации за счет выполнения большинства инструкций за один машинный цикл и позволяет оптимизировать потребление энергии за счет изменения частоты синхронизации;
* 32 программируемых линии ввода-вывода (четыре 8-битных порта). Все 32 регистра непосредственно подключены к АЛУ (арифметико-логическое устройство), что позволяет указывать два регистра в одной инструкции и выполнить ее за один цикл. Данная архитектура обладает большей эффективностью кода и в 10 раз большей производительностью по сравнению с CISC микроконтроллерами;
* 8 – битный таймер с предделителем входной частоты;
* 16 – разрядный таймер с предделителем входной частоты, функцией сравнения, захвата и режимом ШИМ;
* энергопотребление на частоте 4 МГц при питающем напряжении 3 В составляет 3 мА;
* диапазон питающего напряжения 2,7 ÷ 6 В.

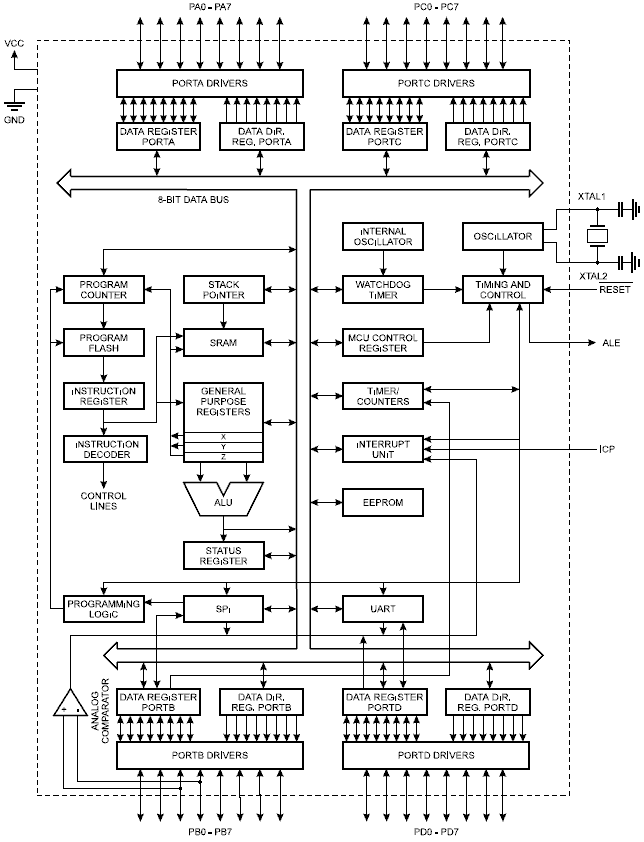


Рисунок 2 – Структурная схема микроконтроллера AT90S8515.

В таблице 2 представлено подробное описание используемых в проекте элементов МК AT90S8515. В таблице жирным шрифтом выделены элементы, входящие в ядро МК, курсивом – входящие в состав устройства управления.

Таблица 2 – Используемые элементы МК

|  |  |
| --- | --- |
| Наименование | Назначение |
| *Программный счётчик (Program Counter)* | Используется для указания следующей команды выполняемой программы. |
| **Память Flash (program Flash)** | Память программ. |
| *Регистры команд (instruction register)* | Содержит исполняемую в текущий момент (или следующий) команду, то есть команду, адресуемую счетчиком команд PC. |
| *Декодер/ дешифратор (instruction decoder)* | Блок, выделяющий код операции и операнды команды, а затем вызывающий микропрограмму, исполняющую данную команд. |
| *Сигналы управления (control lines)* | Поддерживают работу MCUCR. |
| *Регистр управления (MCU control register/MCUCR)* | Регистр, хранящий биты, контролирующие разные опции МК, в т.ч. прерывания. Используется для отслеживания прерывания по таймеру, позволяет измерять время. |
| *Указатель стека (stack pointer)* | Используется для работы со стеком. |
| **Оперативная статическая память (SRAM)** | Используется для хранения данных при выполнении программы. Данные для записи в ячейку SRAM поступают из РОН. Считываемые из ячейки памяти данные поступают в РОН. При выключении питания данные в памяти SRAM теряются. В SRAM находится стек, который используется при прерываниях, при вызове подпрограмм, при сохранении данных. |
| **Регистры общего назначения (General purpose registers)** | Предназначены для хранения операндов арифметико-логических инструкций, а также адресов или отдельных компонентов адресов ячеек памяти. |
| **Статусный регистр (status register)** | Предназначен для хранения результатов всех математических операций в условном виде (I,T,H,V,N,Z,С). |

Таблица 2 - Продолжение

|  |  |
| --- | --- |
| Наименование | Назначение |
| **АЛУ (ALU)** | Блок процессора, который под управлением устройства управления служит для выполнения арифметических и логических преобразований (начиная от элементарных) над данными, называемыми в этом случае операндами. |
| Логика программирования (programming logic) | Программа, которая зашита в микроконтроллер. |
| Последовательный синхронный порт (SPI) | Интерфейс для связи МК с другими внешними устройствами. В проекте используется исключительно для прошивки МК. В другое время не используется. |
| *Генератор (oscillator)* | Необходим для тактирования МК. |
| *Схема синхронизации (timing and control)* | Необходима для тактирования МК. |
| Таймеры (timers/counters) | Включает в себя 8-разрядный таймер Т0 и 16-разрядный таймер Т1. Так как разрядность таймера Т1 больше, чем у T0, а также так как регистратор времени должен считать по секундам, используется лишь таймер Т1. |
| Прерывания (interrupt unit) | Контроллер прерываний обрабатывает внешние прерывания и прерывания от периферийных устройств МК (таймеров, портов ввода/вывода). Прерывания позволяют получить информацию о том, на какие кнопки нажимает пользователь. |
| Последовательный асинхронный приёмопередатчик (UART) | Используется для передачи данных в ПЭВМ |
| Порты ввода/вывода (PortA[0..7], PortB[0..7], PortC[0..7], PortD[0..7]) | В проекте порт А используется для получения сигналов с кнопок, порт B для программирования МК, порт С для вывода сигналов на ЖК-дисплей, порт D для ввода вывода сигналов к ПЭВМ. |
| Энергонезависимая память (EEPROM) | Используется для долговременного хранения данных, которые могут изменяться в процессе работы МК. При выключении питания данные сохраняются. Согласно заданию на выполнение курсовой работы результаты регистрации сохраняются в EEPROM. |

В таблице 3 представлены элементы микроконтроллера, которые не были использованы в проекте.

Таблица 3 – Неиспользуемые элементы МК

|  |  |
| --- | --- |
| Наименование | Назначение |
| Аналоговый компаратор | Используется для сравнения аналоговых сигналов, поступающих на вход PB2 и PB3, формирует запрос прерывания ANA COMP при изменении знака разности, а также сигнал захвата для таймера. |
| Сторожевой таймер (watchdog timer) | Предназначен для защиты устройства от сбоев. |
| ШИМ (широтно-импульсная модуляция) | Способ регулирования напряжения. |
| Внутренний генератор (internal oscillator) | Предназначен для тактирования МК. |

Согласно подразделу 1.1 было принято решение не использовать внутренний генератор из-за низкой надежности, взамен которого были подключены внешние кварцевые резонаторы.

Поскольку разрабатываемое устройство является регистратором времени, требуется уделить особое внимание работе с таймером микроконтроллера. Как уже было сказано выше, с целью повышения надежности тактирования схемы, подключим внешний кварц.

По техническому заданию, требуется обеспечить точность измерения равную 1с. Будем использовать 16-разрядный таймер T1 микроконтроллера. Установим его частоту на 31250 KHz, установив начальное значение таймера (описано ниже). Поскольку нам требуется отсчитывать секунды, требуется посчитать какое количество тактов будет совершать таймер до переполнения и задать начальное значение Tнач.

Если за 1 секунду таймер совершает 31 250 000 тактов. Переполнение таймера наступает по достижению им значения равного FFFFh (65536). В таком случае, начальное значение будет равно:

Tнач = 65536 – 31250 + 1 = 85EEh

Запишем это значение в регистр TCNT1. Каждый раз, когда будет возникать переполнение таймера, будет проходить ровно 1 секунда.

## 2.2 Пульт оператора

Пульт оператора представляет собой набор кнопок и управляющей программы. Программа осуществляет опрос кнопок на нажатие, и при нажатии какой-либо из кнопок производит вызов соответствующей подпрограммы.

Можно выделить три управляющих сигнала:

1. «Вывести на экран» – переход в подпрограмму записанного времени работы устройств;
2. «Предыдущее устройство» – в случае, если поступает во время того, как выполняется подпрограмма вывода данных на дисплей, отображает сохраненные значения времени для предыдущего устройства. В противном случае игнорируется;
3. «Следующее устройство» – в случае, если поступает во время того, как выполняется подпрограмма вывода данных на дисплей, отображает сохраненные значения времени для следующего из 8 устройств. В противном случае игнорируется.

В основном цикле программы на LCD выводится текущее время.

Для упрощения отладки была добавлена одна кнопка на пульт оператора, позволяющая ускорить отсчет времени микроконтроллером. Это сделано для упрощения отладки и тестирования программы. Эта кнопка используется только как элемент эмуляции в Proteus и не фигурирует в конечном устройстве.

Для связи между микроконтроллером и данным модулем используется интерфейс UART.

## 2.3 Блок передачи в ПЭВМ

Блок передачи в ПЭВМ состоит из управляющей подпрограммы, драйвера MAX202CSE и выходной розетки RS-232C.

Для передающей и принимающей части интерфейса RS-232 применяем стандартные схемы сопряжения, также обеспечим необходимые уровни сигналов на линиях интерфейса при помощи резисторов R9 и R10.

Сигнал TxD с выхода микроконтроллера поступает на схему формирования уровней сигналов интерфейса RS-232C (микросхему MAX202CSE), далее через разъем, уже усиленным, он попадает на линию связи. Сигнал, поступающий с интерфейса RS-232C, поступает на преобразователь, а затем, ослабленный, в микроконтроллер на вход RxD.

Данные операции требуются для того, чтобы микроконтроллер мог посылать сигналы ПЭВМ с достаточным уровнем напряжения (в противном случае, ПЭВМ может просто не воспринимать их), а также затем, чтобы сигналы с высоким напряжением, которые поступают в микроконтроллер, не сожгли его и/или его входные порты.

Поскольку в данной работе мы используем асинхронный режим передачи данных, то каждый переданный или принятий байт будет обрамляться стартовым и стоповым битами, а также после него будет следовать некоторая программная задержка, чтобы избежать неправильной передаче байтов данных, или же потери некоторых из них.

Скорость передачи данных по интерфейсу RS-232C в данной работе составляет 9600 бит/с.

## 2.4 Подключение цепей питания

Перезагрузка микроконтроллера AT90S8515 происходит при:

1. Включении питания. Процессор сбрасывается при подаче питания на выводы VCC и GND.
2. Подаче низкого уровня на вывод RESET на время более двух периодов тактовой частоты.

В AT90S8515 встроена схема контроля питающего напряжения, которая запускает микроконтроллер только при безопасном уровне питающего напряжения.

На входах микроконтроллера высокий уровень сигнала является логическим «0», а низкий уровень является логической «1».

# 3 Разработка принципиальной схемы

## 3.1 Конфигурация выводов микроконтроллера

Основным узлом разрабатываемого устройства является микроконтроллер AT90S8515 фирмы Atmel. Конфигурация выводов микроконтроллера для варианта корпуса PDIP приведена на рисунке 3 [6].

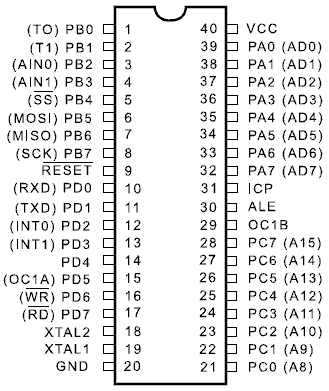


Рисунок 3 – Конфигурация выводов AT90S8515 PDIP

## 3.2 Подключение кнопок и ЖК-дисплея

В качестве кнопок используем миниатюрные кнопки без фиксации – например, PS800L. Они достаточно удобны, легко монтируются, недороги и надежны.

Для отображения результатов работы устройств и текущего времени используем LCD дисплей на 2 строки по 16 символов в каждой – LM032L.

В основном цикле программы на LCD отображается текущее время в формате ЧЧ:ММ:СС (часы, минуты, секунды). При запросе с пульта оператора на LCD выводится записи из памяти микроконтроллера о времени работы устройств.

Подключим LCD-дисплей LM032L. Он поддерживает прием данных двумя методами:

1. 8-битным, когда используются все 8 портов (D0-D7)
2. 4-битным, когда используются 4 порта (D4-D7)

Используем второй метод подключения.

Ниже приведены таблица пинов микросхемы LM032L (таблица 4). На вход Vdd требуется подать напряжение от 1.5 до 5.25 В. Подключим его к питанию платы (+5 В). Вход Vss требуется заземлить.

Таблица 4 – Входы микросхемы LM032L

|  |  |  |
| --- | --- | --- |
| № входа | Символ | Назначение |
| 1 | Vss | Земля |
| 2 | Vdd | Питание |
| 3 | Vo | Настройка контраста |
| 4 | RS | Выбор регистра; 0→Регистр инструкций, 1→Регистр данных |
| 5 | R/W | Сигнал чтения/записи; 1→Чтение, 0→ Запись |
| 6 | E | Разрешение |
| 7 | DB0 | Двунаправленная шина данных. Передача данных осуществляется один раз при использовании портов DB0 - DB7, или два раза в случае использования только портов DB4-DB7. В последнем случае вначале осуществляется передача старших битов, затем младших. |
| 8 | DB1 |
| 9 | DB2 |
| 10 | DB3 |
| 11 | DB4 |
| 12 | DB5 |
| 13 | DB6 |
| 14 | DB7 |

## 3.3 Подключение драйвера MAX202CSE

В качестве преобразователей уровней напряжения при связи устройства с ПЭВМ используем микросхему-драйвер фирмы Maxim серии MAX202 (рисунок 4).

Она содержит преобразователь напряжения +5 В в напряжение +10 В, инвертор, преобразующий напряжение +10 В в напряжение -10 В, и преобразователь уровней сигналов последовательного интерфейса.

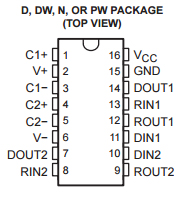


Рисунок 4 – Конфигурация выводов микросхемы MAX202CS.

Ниже представлена типичная схема подключения драйвера MAX202 [7] (рисунок 5). Помимо прочего, требуется разместить на плате 4 дополнительных конденсатора емкостью 0.1 мкФ. Рекомендуется выбирать конденсаторы, работа которых не будет сильно зависеть от перепадов температуры. Используем конденсаторы X7R 0.1 мкф / 16 В. Они обладают высокой надежностью и могут работать при температуре от -55°С до + 125°С.

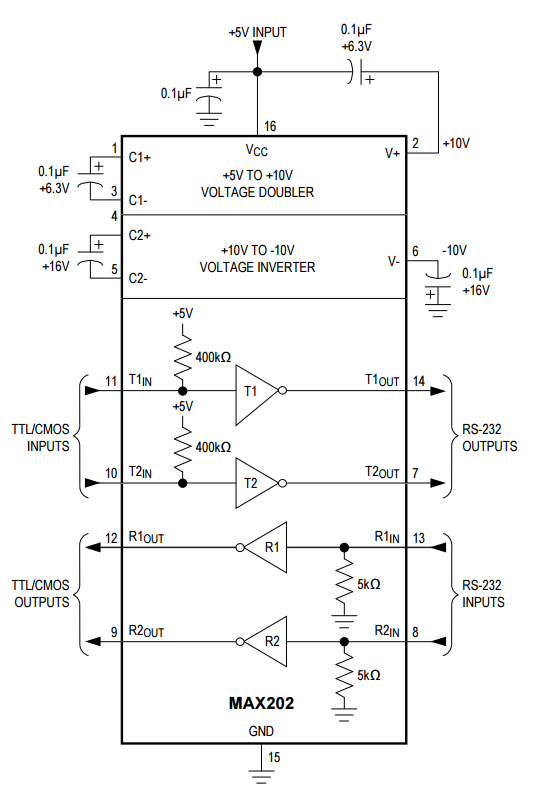


Рисунок 5 – Типовая схема подключения драйвера серии MAX202.

Описание драйвера представлено в таблице 4.

Таблица 4 – Описание драйвера MAX202CSE

|  |  |
| --- | --- |
| Техническая характеристика | Описание |
| Корпус | SOIC-16-3.9 SO16-150 |
| Количество передатчиков | 2 |
| Количество приёмников | 2 |
| Максимальная скорость передачи данных | 150 кбит/с |
| Напряжение питания | 5 В |
| Особенности | 0.1 мкФ |
| Т раб. | 0..70°С |

## 3.4 Синтез принципиальной схемы

По результатам анализа возможностей реализации требований к конечному устройству можно сформулировать перечень структурных элементов, из которых состоят блоки принципиальной схемы:

1. Пульт управления состоит из 3 кнопок с инверсной логикой;
2. Блок индикации - 1 LCD;
3. Блок передачи в ПЭВМ состоит из драйвера и розетки.
4. Блок приема сигнала состоит из розетки с внешними сигналами от 8 устройств.

Схема включения кнопки с инверсной логикой (рисунок 6).



Рисунок 6 – Схема включения кнопки

Также было принято решение переименовать название кнопок (из раздела 1) на более компактное:

1. Out – Вывести на экран.
2. Prev – Предыдущее устройство.
3. Next – Следующее устройство.

На плате будет размещен разъем XP1 (розетка), из которого по 8-проводной шине в МК поступают данные о работе внешних устройств.

Таким образом, была разработана Схема электрическая принципиальная.

# 4 Расчёт параметров

## 4.1 Расчёт потребляемой мощности

Оценим мощность, потребляемую микроконтроллером в пиковом режиме. Воспользуемся графиком тока через вывод микроконтроллера в зависимости от частоты и питающего напряжения, приведенного на рисунке 7.

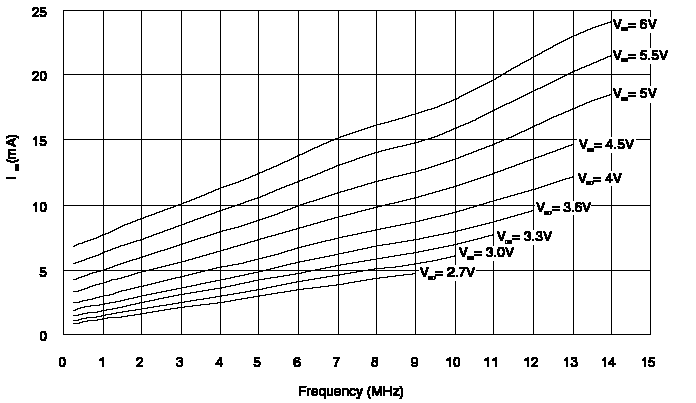


Рисунок 7 – График зависимости тока через вывод МК в зависимости от частоты и питающего напряжения.

По графику определяем ток = 12 мА.

Потребляемая мощность определим как:

PAT90S8515 = Icc \* Uпит \* N,

где N – число задействованных выводов, равное 18. Получаем:

PAT90S8515 =

Мощность LCD:

PLM032L = IDD \* Uпит = ;

Мощность драйвера MAX202:

PMAX202CSE = Icc \* Uпит = = 0,04 Вт;

Суммарная потребляемая мощность:

Pобщ = PAT90S8515 + PLM032L + PMAX202CSE = 0,36 + 0,015 + 0,04 Вт = 0,415 Вт

# 5 Описание алгоритмов работы программ

Алгоритм основной программы микроконтроллера для разрабатываемого устройства представлен на рисунке 8. Основная программа состоит из бесконечного цикла и совершает следующую последовательность действий:

1. В зависимости от переменной loc\_time[3] выводит одно из времен на ЖК-дисплей:
   1. Локальное время (часы);
   2. Время одного из устройств.
2. Производит опрос трех кнопок:
   1. кнопка «Out» – переключает переменную loc\_time[3] так, чтобы отображалось время работы первого устройства;
   2. кнопка «Prev» – если на экран выведено время работы устройства, переключает loc\_time[3] так, чтобы показывалось предыдущее устройство (отнимает от нее 1);
   3. кнопка «Next» – если на экран выведено время работы устройства, переключает loc\_time[3] так, чтобы показывалось следующее устройство (прибавляет к ней 1).
3. Цикл опроса портов B. В случае, если на каком-то порту подан 0, записывает определенную переменную, которая будет подсчитывать время работы этого устройства во время прерываний таймера (прибавлять 1 секунду каждый раз при заходе в прерывание). Счет времени прекращается, в случае если на порт будет подана 1.
4. Проверяет часы на предмет 12 или 24 часов. В случае если на часах 12:00 или 24:00 (00:00), передает данные в ПЭВМ, после чего очищает все записанные данные и продолжает работу.

Предустановка МК состоит из:

1. Инициализации портов ввода/вывода, а именно:
   1. PORTB устанавливается на ввод. В него будут приходит данные о включении внешних устройств;
   2. инициализируются порты RxD и TxD;
   3. PORTA инициализируется для соединения с ЖК-дисплеем.
2. Установки таймера Т1.
3. Инициализации UART.

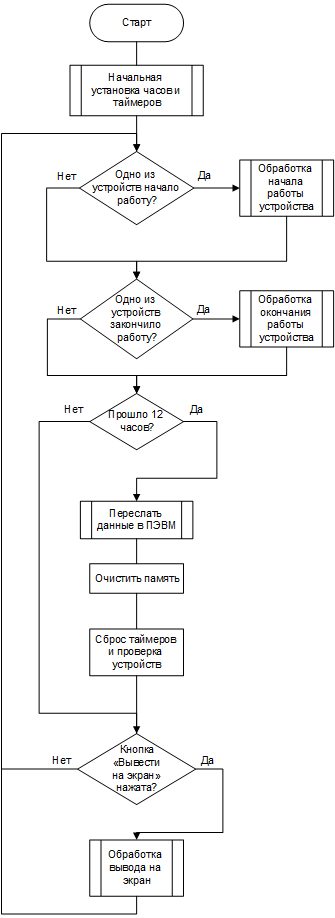


Рисунок 8 – Алгоритм основной программы

На рисунке 9 представлен алгоритм подпрограммы начальной установки часов и таймеров, которая включает в себя:

1. Инициализацию переменных, которые отвечают за установку «часов».
2. Инициализацию переменной, отвечающей за хранение времени работы устройств.
3. Инициализацию переменной для хранения показателей выбранного при помощи кнопок «Next» и «Prev» устройства.
4. Инициализацию таймера в соответствии с технической спецификацией к ATmega8515 [2].

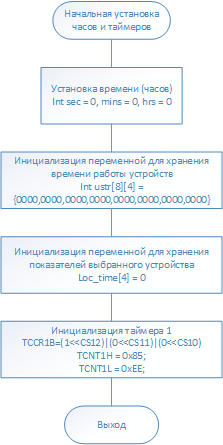


Рисунок 9 – Алгоритм начальной установки часов и таймеров

На рисунке 10 представлен алгоритм подпрограммы обработки начала работы устройства, который состоит из:

1. Определения номера устройства, активированного пользователем.
2. Запуска соответствующего таймера.



Рисунок 10 – Алгоритм обработки начала работы устройства

На рисунке 11 представлен алгоритм подпрограммы обработки окончания работы устройства, включающий в себя:

1. Определение номера устройства, отключённого пользователем.
2. Приостановки соответствующего таймера.
3. Сохранение значения таймера в соответствующую ячейку памяти.



Рисунок 11 – Алгоритм обработки окончания работы устройства

Алгоритм подпрограммы пересылки данных в ПЭВМ показан на рисунке 12, где переменная i отвечает за каждое из внешних устройств, переменная j – за каждый символ строки udr\_buf, хранящей время работы устройства. Алгоритм состоит из цикла с предусловием, благодаря которому происходит пересылка в ПЭВМ времени работы каждого из 8 устройств. Пробел (UDR=’ ‘) был добавлен, чтобы разделить время работы устройств. В подпрограмму, реализующую данный алгоритм, также была добавлена задержка в 5 микросекунд, чтобы передача новой информации в ПЭВМ не осуществлялась раньше окончания передачи текущих данных.

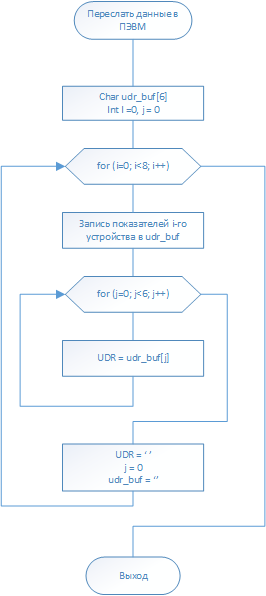


Рисунок 12 – Алгоритм пересылки данных в ПЭВМ

На рисунке 13 представлен алгоритм подпрограммы обработки вывода данных на экран. Он включает в себя:

1. Первичную очистку экрана.
2. Вывод надписи «Time» на первой строке, а также номера выбранного устройства при помощи кнопок «Next» и «Prev».
3. Вывод времени работы выбранного пользователем устройства.



Рисунок 13 – Алгоритм обработки вывода на экран

# 6 Технологическая часть

## 6.1 Отладка программы микроконтроллера

Программа была отлажена с использованием программного Code Vision AVR и ISIS Proteus. Данный пакет специально предназначен для разработки программного обеспечения для микроконтроллеров AVR. Он позволяет тщательно отладить программу, предоставляя визуальные средства отображения состояний всех регистров микроконтроллера.

При пошаговой отладке программы, можно определить точные моменты наступления тех или иных событий по шкале реального времени, что позволяет оценить точность измерения реакции оператора.

Отладка с помощью пакета программ ISIS Proteus позволяет получить наглядный результат моделирования системы. Внешний вид собранной модели в Proteus (рисунок 14).

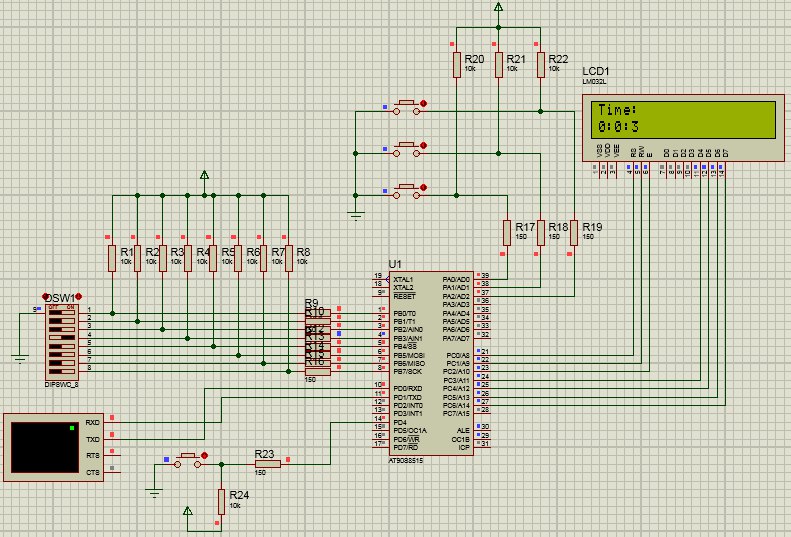


Рисунок 14 – Моделирование системы в ISIS Proteus.

Для моделирования вывода данных в ПЭВМ используется инструмент системы Proteus - Virtual Terminal. Этот инструмент позволяет эмулировать простейший терминал, который позволяет получать и передавать данные по портам TxD и RxD.

При работе в ISIS Proteus, при указании файла .cof в качестве программного файла микроконтроллера, становится доступна пошаговая отладка (рисунок 15).

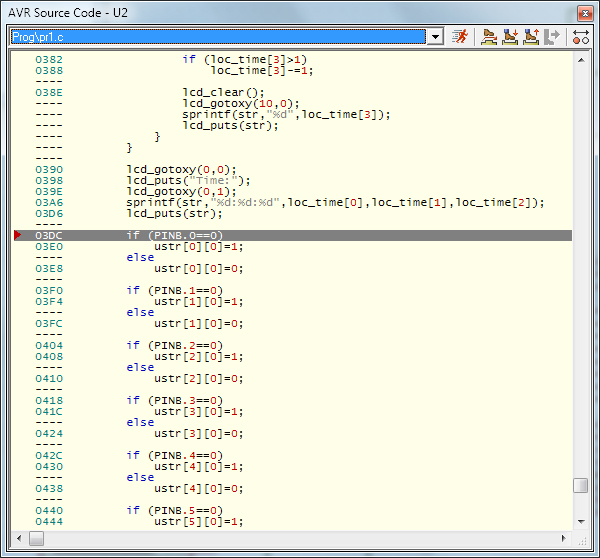


Рисунок 15 – Окно пошаговой отладки в Proteus

В этом окне можно выполнять пошаговую отладку с заходом и без заходов в процедуры, посмотреть на какой строке кода находится выполняемая программа в данный конкретный момент.

Это сильно облегчает задачу программирования микроконтроллера, позволяя находить проблемные места в коде и баги, оперативно их устраняя. Ниже приведено окно состояния переменных программы в среде Proteus (рисунок 16).

Это окно имеет три столбца:

1. Name – имя переменной, заданное программистом в коде программы;
2. Address – адрес переменной во внутренней памяти МК. Можно посмотреть 16-ричное значение этой переменной, в окно просмотра внутренней памяти МК, найдя ее по этому адресу;
3. Value – значение переменных.

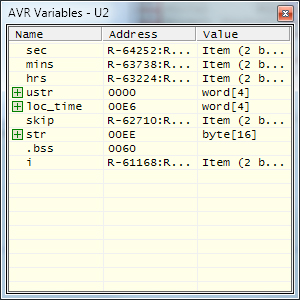


Рисунок 16 – Окно переменных в ISIS Proteus

## 6.2 Способы программирования микроконтроллера

Для программирования МК ATmega8515 могут быть использованы два способа:

1. параллельное программирование с использованием дополнительного источника питания +12В;
2. последовательное программирование при низком уровне напряжения (по интерфейсу SPI);

Рассмотрим последовательное программирование по интерфейсу SPI.  
Самым распространенным и удобным интерфейсом для прошивки AVR является **SPI** (Serial Peripheral Interface). Для подключения по SPI нужно всего четыре провода, не считая земли:

1.SCK — тактовый сигнал, синхронизирует все операции обмена данными.

2. MOSI (Master Out Slave In) — линия данных от ведущего устройства к ведомому.

3. MISO (Master In Slave Out) — линия данных, наоборот, от ведомого устройства к ведущему.

4. RESET — для разрешения прошивки по SPI нужно подать логический «0» на этот вывод.

Взаимодействие устройств по интерфейсу SPI требует установки одного из устройств в режим ведущего, а остальных – в режим ведомого. При этом ведущее устройство отвечает за выбор ведомого и инициализацию передачи.

SPI является синхронным интерфейсом: все операции синхронизированы фронтами тактового сигнала (SCK), который вырабатывается ведущим устройством. Максимальная скорость передачи ограничена величиной 1/4 тактовой частоты контроллера. На минимальную же скорость нет никаких ограничений: без тактового сигнала обмен данными «замораживается», и интерфейс может оставаться в статическом состоянии сколь угодно долго.

Передача по SPI осуществляется в полнодуплексном режиме, по одному биту за такт в каждую сторону. По возрастающему фронту сигнала SCK ведомое устройство считывает очередной бит с линии MOSI, а по спадающему — выдает следующий бит на линию MISO (см. рисунок 17).

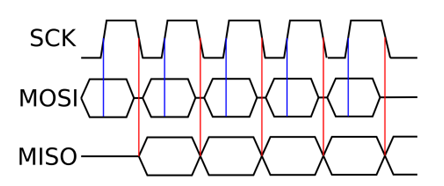
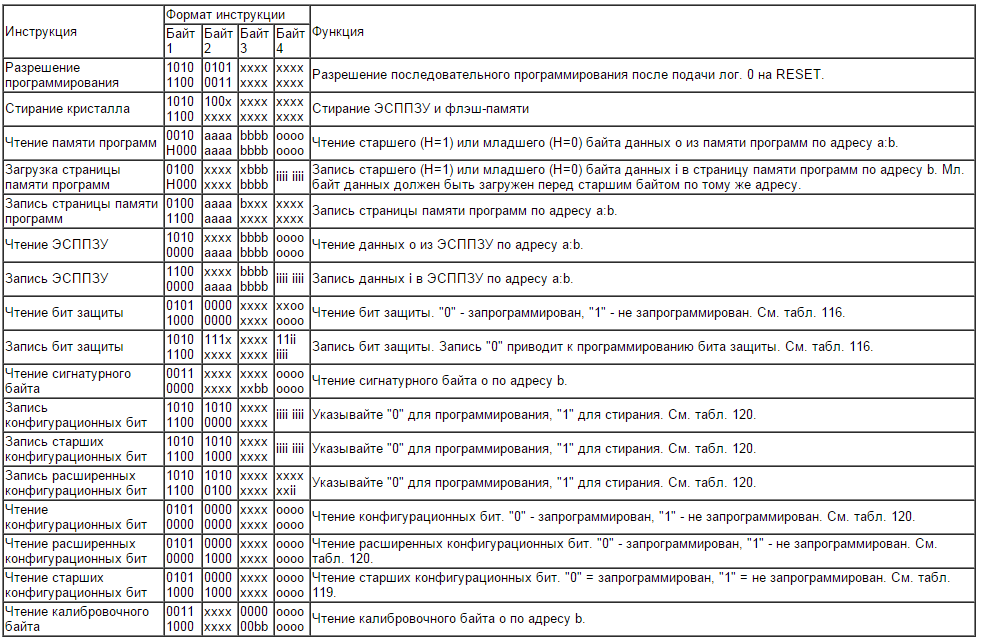


Рисунок 17 – Временная диаграмма работы SPI

Программирование микроконтроллера по SPI осуществляется путем посылки 4-байтовых команд (см. таблицу 6) на вывод MOSI МК, в который один или два байта определяют тип операции, остальные – адрес, записываемый байт, установочные биты и биты защиты, пустой байт. При выполнении операции чтения считываемый байт снимается через вывод MISO. Так же можно запрограммировать память данных EEPROM. В каждой команде указывается адрес записываемой ячейки и записываемое значение.

Полное описание форматов команд, используемых для программирования МК, можно найти в технической спецификации к ATmega8515.

Таблица 6 – Формат байтов команд для программирования

 Примечание:

* a - адрес старших разрядов;
* b - адрес младших разрядов;
* H - 0 - мл. байт, 1 - ст. байт;
* o - вывод данных;
* i - ввод данных;
* x - произвольное значение.

Операции необходимые выполнять для прошивки МК:

* перевод контроллера в режим программирования;
* (опционально) чтение идентификатора устройства;
* стирание;
* запись во flash;
* (опционально) проверка записанного.

При программировании микроконтроллера производится установка конфигурационных битов (FUSE Bits). При выполнении операции стирания памяти, выполняемой перед перепрограммированием, конфигурационные биты и байты идентификации остаются без изменений, в отличие от ячеек Flash памяти и памяти EEPROM, которым присваивается значение «1».

Конфигурационные биты доступны только при программировании. ATmega8515 имеет 2 конфигурационных бита:

* SPIEN – при программировании в «1» запрещает параллельное программирование;
* FSTRT – задает время задержки старта AVR-микроконтроллера после сброса.

Рассмотрим каждый шаг подробно.

**Включение режима программирования**

Режим программирования включается подачей «0» на RESET. Но есть некоторые тонкости. Atmel рекомендует сначала выставить на выводах RESET и SCK низкий уровень, а только потом подавать на контроллер питание. Если такой возможности нет, нужно после включения питания подать «0» на SCK, а затем положительный импульс на RESET:

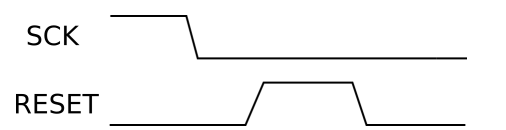
****

Рисунок 18 – Перевод МК в режим программирования

Далее нужно передать команду на собственно включение режима программирования: 10101100 01010011 xxxxxxxx xxxxxxxx

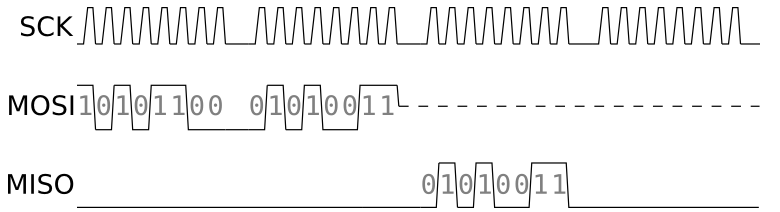
****

Рисунок 19 – Команда «Program Enable»

Биты, обозначенные как x, могут быть любыми. Во время передачи третьего байта контроллер должен переслать обратно второй байт (01010011). Если это произошло, значит, все хорошо, команда принята, контроллер ждет дальнейших инструкций. Если ответ отличается, нужно перезагрузить МК и попробовать все сначала.

**Проверка идентификатора**

Прежде чем что-либо писать в память МК, нужно убедиться, что перед нами нужная модель. Каждая модель контроллера имеет свой трехбайтный идентификатор (Signature). Прочитать его можно командами вида 00110000 000xxxxx xxxxxxbb xxxxxxxx. Вместо bb (третий байт команды) следует подставить 00 для первого байта идентификатора, 01 — для второго и 10 — для третьего. Соответствующий байт идентификатора будет передан контроллером при отправке 4-го байта команды.

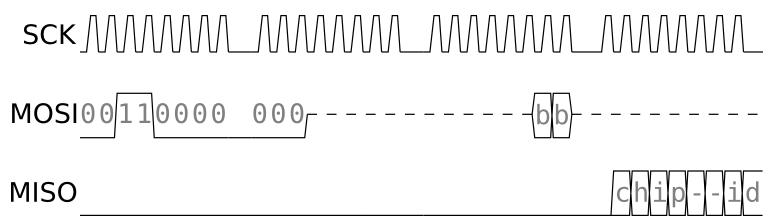
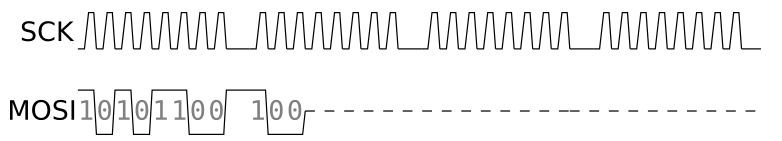


Рисунок 20 - Команда «Read Signature Byte»

**Очистка контроллера**

  
Рисунок 21 – Команда «Chip Erase»

Следующим шагом будет очистка памяти МК, которая осуществляется посылкой команды «Chip Erase» 10101100 100xxxxx xxxxxxxx xxxxxxxx.

Этой командой выполняется стирание содержимого Flash и EEPROM (все ячейки будут содержать FF), а также снятие lock-битов, если они установлены.  
 **Запись во flash-память**

Память программ (Flash) в AT90S8515 состоит из 512 двухбайтных слов (1К байт). Адрес слова имеет разрядность 9 бит. Flash-память разделена на страницы, каждая страница имеет размер 16 слов (всего получается 32 страницы). Запись во flash осуществляется в два этапа.

Сначала необходимо загрузить данные в буфер страницы, для этого используется команда «Load Program Memory Page» (см. рисунок 22) 01000000 000xxxxx xxxxbbbb iiiiiiii — для загрузки младшего байта слова, и 01001000 000xxxxx xxxxbbbb iiiiiiii — для загрузки старшего. Сначала всегда должен загружаться младший байт слова, а затем — старший байт того же слова. Где, bbbb — адрес слова на странице, iiiiiiii — загружаемый байт.

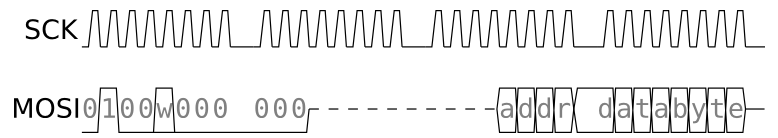


Рисунок 22 – Команда «Load Program Memory Page»

После того, как буфер страницы загружен, нужно выполнить команду «Write Program Memory Page» 01001100 0000000a bbbbxxxx xxxxxxxx для записи страницы непосредственно в память контроллера. Младший бит второго байта и старшие 4 бита третьего a:bbbb — пятибитный номер страницы для записи.

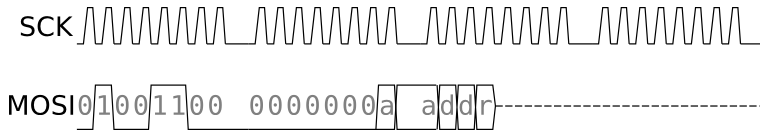


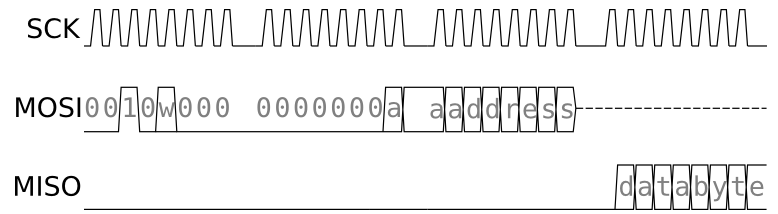
Рисунок 23 – Команда «Write Program Memory Page»

Адрес любого байта памяти программ состоит из 10 бит: ppppp:bbbb:w, где:

ppppp — номер страницы (используется в команде «Write Program Memory Page»);

bbbb — адрес слова на странице (в команде «Load Program Memory Page»);

w — бит, определяющий старший или младший байт в слове (зашифрован в первом байте команды «Load Program Memory Page»).

**Чтение flash**  
   
 Рисунок 24 – Команда «Read Program Memory»

После записи прошивки в МК нужно проверить записанное, так как никакой проверки целостности данных не выполнялось. Единственный способ проверки состоит в том, чтобы прочитать весь объем flash-памяти и сравнить с оригиналом. Чтение выполняется побайтно. Команда «Read Program Memory» выглядит так:

00100000 0000000a bbbbbbbb xxxxxxxx — для чтения младшего байта слова;

00101000 0000000a bbbbbbbb xxxxxxxx —для чтения старшего байта слова.

Младший бит второго байта и весь третий байт a:bbbbbbbb — адрес слова в памяти. Прочитанный байт возвращается во время передачи 4-го байта команды.

**Завершение программирования**

Пожалуй, самая простая операция. Чтобы завершить программирование и перевести МК в рабочий режим, достаточно подать на RESET логический уровень «1». Контроллер запустится и будет работать по новой программе.

# Заключение

Таким образом, разработан курсовой проект по дисциплине «Микропроцессорные системы» ­ – регистратор времени работы восьми электроприборов. Управление осуществляется с пульта оператора, оснащенного 3-я кнопками. Большая часть программы автоматизирована. Пульт оператора служит исключительно для просмотра результатов работы МК.

Параметры измерений соответствуют техническому заданию. Благодаря использованию современной элементной базы было разработано точное, надежное устройство с небольшой потребляемой мощностью.

Основная аппаратная часть устройства реализована в пределах используемого микроконтроллера, обладающего значительными возможностями ввода-вывода и счета времени.

Устройство может применяться в составе тестирующих и измерительных комплексов, устройств регистрации и учета работ других устройств.

Созданное устройство соответствует заданию на разработку курсовой работы.

# Список использованной литературы

1. AVR микроконтроллеры: очередной этап на пути развития [Электронный ресурс].   
   – URL: <https://www.kit-e.ru/articles/micro/2002_03_98.php> (дата обращения 20.09.17).
2. Производитель МК ATMEGA – компания Atmel [Офиц. сайт]. URL: http://www.atmel.com/ (дата обращения: 11.10.2017).
3. Хартов, В.Я. Микропроцессорные системы: учебник для студ. технических ВУЗов. – Издательство: Академия, 2014. – 368с.
4. Хартов, В.Я. Микроконтроллеры AVR. Практикум для начинающих: учеб. пособие. – 2- изд., испр. И доп. – М.: Издательство: МГТУ им. Н.Э. Баумана, 2012. – 280с.
5. Г. Р. Аванесян, В.П. Левшин. Интегральные микросхемы ТТЛ, ТТЛШ. Справочник. – Издательство: Машиностроение, 1993 г.
6. Документация на микроконтроллер AT90S8515 [Электронный ресурс]. – URL: <http://www.atmel.com/Images/DOC0841.PDF> (дата обращения 17.09.2017).
7. Документация на драйвер MAX202CSE [Электронный ресурс]. – URL: <https://datasheets.maximintegrated.com/en/ds/MAX200-MAX213.pdf> (дата обращения 01.10.2017).
8. ГОСТ 2.743-91 Обозначения условные в графических схемах. Элементы цифровой техники.
9. ГОСТ 2.701-84 Правила выполнения схем.
10. ГОСТ 2.702-75 Правила выполнения электрических схем.