ФГБОУ ВПО
«Воронежский государственный технический университет»

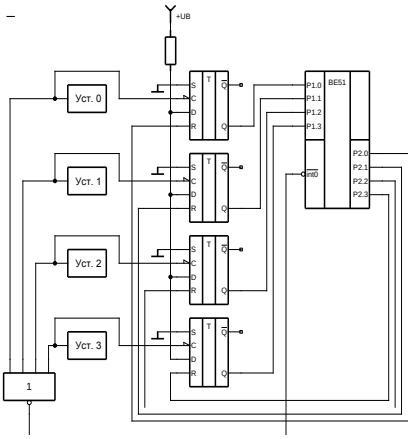
Кафедра автоматизированных и вычислительных систем

493 - 2015

РАЗРАБОТКА МИКРОПРОЦЕССОРНЫХ СИСТЕМ НА ОСНОВЕ ОДНОКРИСТАЛЬНЫХ МИКРОКОНТРОЛЛЕРОВ СЕМЕЙСТВА ATMEGA

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению лабораторных работ
по дисциплине «Микропроцессорные системы»
для студентов направления 09.03.01
«Информатика и вычислительная техника»
(профиль “Вычислительные машины, комплексы, системы и сети”) очной формы обучения



Воронеж 2015

Составитель канд. техн. наук Г.В. Петрухнова

УДК 681.32

Разработка микропроцессорных систем на основе однокристальных микроконттроллеров семейства ATMEGA: методические указания к выполнению лабораторных работ по дисциплине «Микропроцессорные системы» для студентов специальности 09.03.01 «Информатика и вычислительная техника» (профиль “Вычислительные машины, комплексы, системы и сети”) очной формы обучения / ФГБОУ ВПО сост. Г.В. Петрухнова. Воронеж, 2015. 47 с.

Методические указания содержат теоретические и практические сведения о разработке микропроцессорных систем на основе однокристальных микроконтроллеров семейства ATMEGA. В качестве среды программирования используется CodeVisionAVR.

Предназначены для бакалавров третьего курса.

Табл. 4. Ил. 21. Библиогр.: 4 назв.

Рецензент д-р техн. наук, проф. А.М. Литвиненко

Ответственный за выпуск зав. кафедрой д-р техн. наук, проф. С.Л. Подвальный

Печатается по решению редакционно-издательского совета Воронежского государственного технического университета

1. ЛАБОРАТОРНАЯ РАБОТА №1. ПОРТЫ ВВОДА/ВЫВОДА ОДНОКРИСТАЛЬНЫХ МИКРОКОНТРОЛЛЕРОВ

Цель лабораторной работы: получение навыков работы с микроконтроллерами семейства Atmega.

1.1. Краткие теоретические сведения

Стремительное развитие микропроцессорной техники требовало всё большей и большей степени интеграции микросхем. Именно поэтому были разработаны микросхемы, которые объединяют в себе сразу все элементы микропроцессорной системы. Такие микропроцессорные системы называются **микроконтроллерами**. Функции процессора в микроконтроллере заменяет **арифметико-логическое устройство (АЛУ)**. Кроме АЛУ, микроконтроллер (МК) содержит в своём составе тактовый генератор, память данных, память программ, порты ввода-вывода.

Все эти элементы соединены между собой внутренними **шинами данных и адреса**. С внешним миром микроконтроллер общается при помощи **портов ввода-вывода**. Любой микроконтроллер всегда имеет один или несколько портов. Кроме того, современные микроконтроллеры всегда имеют **встроенную систему прерываний**, а также встроенные программируемые таймеры, компараторы, цифро-аналоговые преобразователи и многое другое.

МК серии AVR относятся к **классу восьмиразрядных микроконтроллеров**. Они изготавливаются по КМОП-технологии, благодаря которой имеют достаточно высокое быстродействие и низкий ток потребления. Большинство команд микроконтроллера выполняется за один такт. Поэтому быстродействие МК может достигать 1 миллион операций в секунду при тактовой частоте 1 МГц.

Микроконтроллеры AVR имеют в своем составе **три вида памяти.**

Во-первых, это ОЗУ (оперативная память данных). В документации фирмы Atmel эта память называется SRAM.

Объем ОЗУ для разных МК варьируется от полного ее отсутствия до 2 Кбайт.

Второй вид памяти — это память программ. Она выполнена по Flash-технологии и предназначена для хранения управляющей программы. В фирменной документации она называется Flash-память. Объем программной памяти в разных микросхемах этой серии может составлять от 1 до 64

Третий вид памяти — это энергонезависимая память данных. Она выполнена по Flash-технологии и называется EEPROM. Основное назначение этого вида памяти — долговременное хранение данных. Данные, записанные в эту память, не теряются даже при выключенном источнике питания. Объем памяти EEPROM сравнительно небольшой. Для разных микросхем он составляет от 64 байт до 2 Кбайт

Порты ввода/вывода – это обязательный атрибут любого микроконтроллера. Их количество для каждой конкретной микросхемы разное. Все порты микроконтроллеров AVR восьмиразрядные, но в некоторых случаях отдельные разряды не используются. Это связано с ограниченным количеством выводов (ножек) у микросхемы.

Каждый порт имеет свое имя. Они именуются латинскими буквами от A до G.

Для управления каждым портом ввода/вывода используется три специальных регистра ввода/вывода. Это регистры PORTx, DDRx и PINx. Под «x» здесь подразумевается конкретная буква – имя порта. **Например**, для порта A имена регистров управления будут такими: PORTA, DDRA и PINA.

Назначение каждого из этих регистров следующее:

PORTx - регистр данных (используется для вывода информации);

DDRx- регистр направления передачи информации;

PINx – регистр ввода информации.

Отдельные разряды, приведенные выше, регистров также имеют свои имена. Разряды регистра PORTx обычно именуются как PORTx.n, где «n» - это номер разряда. **К при-**

меру, разряды регистра PORTA будут именоваться следующим образом: PORTA.0, PORTA.1, PORTA.2-PORTA.7.

Разряды порта DDRx именуются как DDxn (для порта A – DDRA.0, DDRA.1-DDRA.7).

Разряды порта PINx именуются как PINx.n (для порта A- PINA.0, PINA.1-PINA.7). **Для других портов** буква A заменяется соответственно на B, C, D, E, F, G.

Любой порт ввода/вывода МК серии AVR устроен таким образом, что каждый его разряд может работать как на ввод, так и на вывод. То есть он может быть входом, а может быть выходом. **Для переключения режимов работы** служит регистр DDRx. Каждый разряд регистра DDRx управляет своим разрядом порта. Если каком-либо разряде регистра DDRx записан ноль, то соответствующий разряд порта работает как вход. Если же в этом разряде единица, то разряд порта работает как выход. Для того, чтобы выдать информацию на внешний вывод МК, нужно в соответствующий разряд DDRx записать логическую единицу, а затем можно записать байт данных в регистр PORTx, либо записать логическую единицу в соответствующий разряд PORTx. Содержимое соответствующего бита этого байта тут же появится на внешнем выводе МК и будет присутствовать там постоянно, пока не будет заменено другим, либо пока данная линия порта не переключится на ввод.

Для того, чтобы прочитать информацию с внешнего вывода МК, нужно сначала перевести нужный разряд порта в режим ввода. То есть записать в соответствующий разряд регистра DDRx ноль. Только после этого на данный вывод МК можно подавать цифровой сигнал от внешнего устройства. Далее МК просто читает байт из регистра PINx. Содержимое соответствующего бита, прочитанного байта соответствует сигналу на внешнем выводе порта.

Порты ввода/вывода МК AVR имеют еще одну полезную функцию. В режиме ввода информации они могут при необходимости подключать к каждому выводу порта внутренний нагрузочный резистор. Внутренний резистор позволяет значи-

тельно расширить возможности порта. Такой резистор создает вытекающий ток для внешних устройств, подключенных между выводом порта и общим проводом. Благодаря этому резистору упрощается подключение внешних контактов и кнопок. Обычно контакты требуют внешнего резистора. Теперь без внешнего резистора можно обойтись. Включением и отключением внутренних резисторов управляет регистр PORTx, если порт находится в режиме ввода.

Таблица 1. Режимы работы регистров PORTx

DDxn	Pxn	Режим	Резистор	Примечание
0	0	Вход	Отключен	Вывод отключен от схемы
0	1	Вход	Подключен	Вывод является источником тока
1	0	Выход	Отключен	На выходе «0»
1	1	Выход	Отключен	На выходе «1»

1.2. Разработка программы в среде CodeVisionAVR

Задание 1. Согласно схеме на рис.1 и пояснениям, приведенным ниже, создайте проект в среде CodeVisionAVR для управления светодиодом при помощи тумблера и сохраните его в своей папке. Чтобы зажечь светодиод микроконтроллер должен подать на вывод PORTD.0 сигнал логического нуля.

Для создания программ на языке СИ будем использовать программную среду **CodeVisionAVR**. Это среда специально предназначена для разработки программ на языке СИ для микроконтроллеров серии AVR. Среда CodeVisionAVR не имеет своего отладчика, но позволяет отлаживать программы, используя возможности системы AVR Studio.

Отличительной **особенностью** системы CodeVisionAVR является наличие мастера-построителя программы. **Мастер-построитель** облегчает работу программи-

сту. Он избавляет от необходимости листать справочник и выискивать информацию о том, какой регистр за что отвечает и какие коды нужно в него записать. **Результат** работы мастера — это заготовка будущей программы, в которую включены все команды предварительной настройки, а также заготовки всех процедур языка СИ.

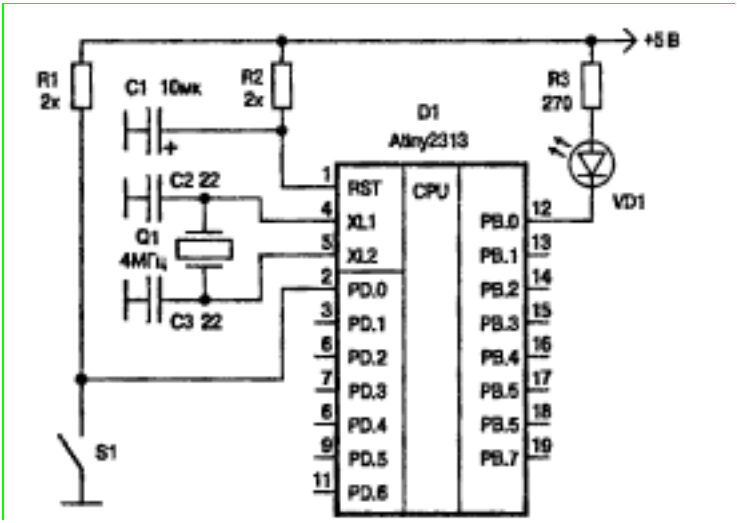


Рис. 1. Функциональная схема подключения светодида и тумблера к МК

Программа CodeVisionAVR, как и любая современная среда программирования, работает не просто с текстом программы, а с **Проектом.** Проект, кроме текста программы, содержит ряд вспомогательных файлов, содержащих дополнительную информацию, такую, как тип используемого МК, тактовую частоту кварца и т. п. Эта информация необходима в процессе трансляции и отладки программ. За исключением текста программы, все остальные файлы проекта создаются и изменяются автоматически. Задача программиста лишь написать текст программы, для которого в проекте отводится отдельный файл с расширением «с».

Для создания нового проекта выберем в меню «File» пункт «New». Откроется небольшой диалог, в котором нужно выбрать тип создаваемого файла. Предлагается два варианта:

– «Source» (Исходный текст программы);
– «Project» (Проект).

Выбираем Project и нажимаем «Ok». Появляется окно с вопросом «You are about to create a new project. Do you want to use the CodeWizardAVR?». В переводе на русский это означает: «Вы создаете новый проект. Будете ли вы использовать построитель CodeWizardAVR?». Выбираем «Yes», после чего открывается окно построителя (см. рис. 2). Окно имеет множество вкладок, каждая из которых содержит элементы выбора режимов.

Все эти управляющие элементы позволяют **настроить параметры создаваемой заготовки программы**. Сразу после запуска мастера все параметры принимают значения по умолчанию (все внутренние устройства выключены, а все порты ввода/вывода настроены на ввод, внутренние нагрузочные резисторы отключены). Это соответствует начальному состоянию МК после системного сброса.

Пройдемся немного по вкладкам мастера и выберем необходимые параметры.

Первая вкладка называется «Chip». На этой вкладке можно выбрать общие параметры проекта. Используя выпадающий список «Chip», выберем тип МК - ATMega328P-PU. При помощи поля «Clock» выбираем частоту кварцевого резонатора 12 МГц. При помощи поля «Crystal Oscillator Divider» выбирается **коэффициент деления тактового генератора**. Этот параметр требует пояснений. Дело в том, что выбранный нами МК ATMega328P-PU имеет систему предварительного деления тактовых импульсов. Если частота тактового генератора не устраивает, можно поделить ее, и МК будет работать на другой, более низкой частоте. Коэффициент деления может изменяться от 1 до 256. Выберем его равным единице (без деления).

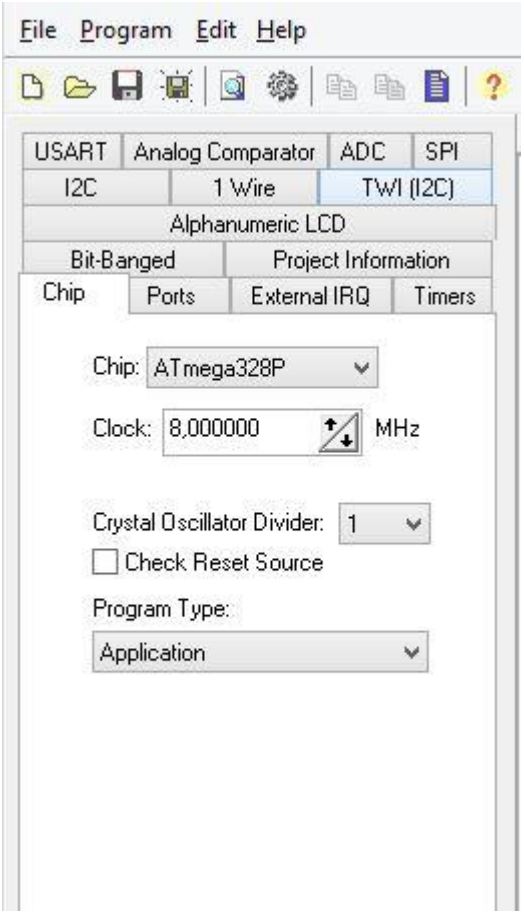


Рис. 2. Окно мастера ATMega328P-PU

Без изменений оставим флажок «Check Reset Source» (Проверка источника сигнала сброса). Включение данного флажка добавляет к создаваемой программе процедуру, связанную с определением источника сигнала системного сброса.

Покончив с общими настройками, перейдем к **вкладке Порты**. Эта вкладка позволяет настроить все имеющиеся порты ввода/вывода. На вкладке «Ports» мы видим еще три

вкладки поменьше (см. рис. 3). По одной вкладке для каждого порта. Выбираем вкладку «Port B».

На вкладке видим два столбца с параметрами. **Столбец «Data direction» (Направление передачи данных)** позволяет настроить каждую линию порта либо на ввод, либо на вывод. По умолчанию каждый параметр имеет значение «In» (вход). Поменяем для каждого разряда это значение на «Out» (Выход).

Для того, чтобы поменять значение разряда, достаточно щелкнуть по полю с надписью «In» один раз мышью, и параметр сменится на «Out». Повторный щелчок заставит вернуться к «In». Каждое поле столбца «Data direction» определяет, какое значение будет присвоено соответствующему разряду регистра DDRB в нашей будущей программе.

Второй столбец на той же вкладке **называется «Pullup / Output Value» (Включение нагрузки / Выходное значение)**. Этот столбец определяет, какое значение будет присвоено каждому из разрядов регистра PORTB. В нашем случае порт PB работает на вывод. Поэтому содержимое регистра PORTB определяет выходное значение всех разрядов порта. По умолчанию все они имеют значение «0». Но по условиям задачи они должны быть равны единице (при старте программы светодиод должен быть отключен). Поэтому изменим все нули на единицы. Для этого также достаточно простого щелчка мыши. В результате всех операций вкладка «Port B» будет выглядеть так, как это показано на рис. 3.

Теперь перейдем к **настройке последнего порта**. Для этого выберем вкладку «Potr D». На вкладке увидим такие же органы управления, как на вкладке «Port B» (см. рис. 4). По условиям задачи порт PD МК должен работать на ввод. Поэтому состояние элементов первого столбца не будем менять.

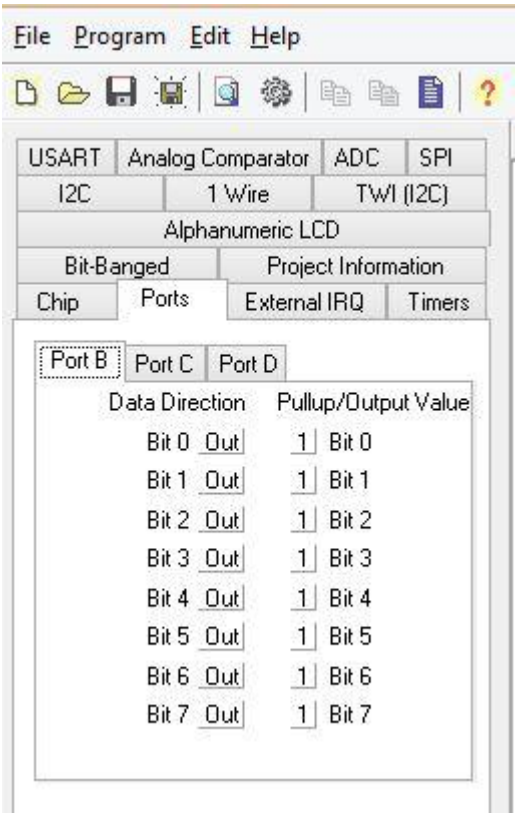


Рис. 3. Настройка порта B

Нужно включить внутренние нагрузочные резисторы для каждого из входов. Для этого необходимо изменить значения элементов второго столбца. Так как порт PD работает в режиме ввода, элементы в столбце «Pullup / Output Value» принимают значение «T» или «P».

«T» (Terminate) означает отсутствие внутренней нагрузки, а **«P» (Pull-up)** означает, что нагрузка включена. Включим нагрузку для каждого разряда порта PD, изменив значение поля с «T» на «P». В результате элементы управления будут выглядеть так, как показано на рис. 4.

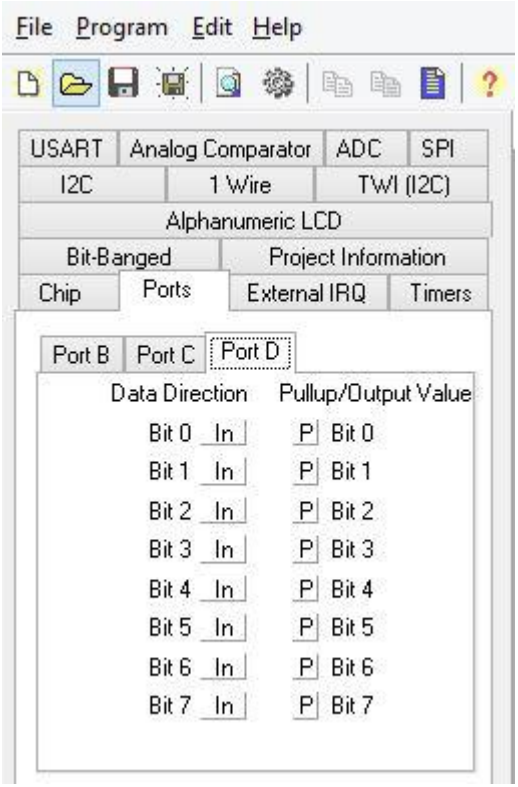


Рис.4. Настройка порта D

Для данной программы настройки можно считать оконченными. Остальные системы МК оставим без изменений.

Воспользуемся еще одним **полезным свойством мастера программ**. Откроем вкладку «Project Information» (см. рис. 5). В поле «Project Name» можно занести название проекта. **Поле «Version»** предназначено для номера версии. В **поле «Date»** помещают дату разработки программы. В **полях «Author»** и **«Company»** помещается, соответственно, имя автора и название компании. В **поле «Comments:»** можно поместить любые необходимые комментарии. Вся эта информация будет автоматически помещена в заголовок будущей программы.

После того, как все параметры выставлены, можно приступить непосредственно **к процессу генерации программы.** Для этого нужно выбирать в меню мастера пункт «Generate, Save and Exit», как это показано на рис. 6. Процесс генерации начнется с запроса имени файла будущей программы. Для этого откроется стандартный диалог сохранения файла, в котором нужно выбрать каталог, а затем указать имя файла программы.

Рекомендуется для каждого проекта создавать свой отдельный каталог. Назовем каталог именем «Progl».

Однако процесс генерации проекта на этом не заканчивается. Сразу после того, как файл программы будет записан, откроется новый диалог, который запросит имя для файла проекта. **Файл проекта** предназначен для хранения параметров конкретного проекта.

Кроме типа используемой микросхемы и частоты задающего генератора, файл проекта используется для хранения вспомогательной информации, такой как наличие и расположение точек останова, закладок и т. д. В качестве имени файла проекта удобнее всего использовать то же самое имя, что и для текста программы. То есть «Progl». Файлу проекта автоматически присваивается расширение «.prj».

Когда файл проекта будет записан, диалог записи файла откроется в третий раз. Теперь необходимо **выбрать имя для файла данных построителя**. В этот файл будут записаны текущие значения всех параметров мастера. То есть значения всех управляющих элементов со всех его вкладок - и те, которые изменили в процессе настройки, и те, которые остались без изменений (по умолчанию).

Эти данные могут понадобиться, если потребуется заново пересоздать проект. Используя файл данных построителя, можно в любой момент восстановить значения всех его элементов, немного подкорректировать их и создать новый проект. Файлу данных построителя присвоим то же самое имя, что обоим предыдущим («Progl»). Новый файл получит расширение «.cwp» (Code Wizard Project).

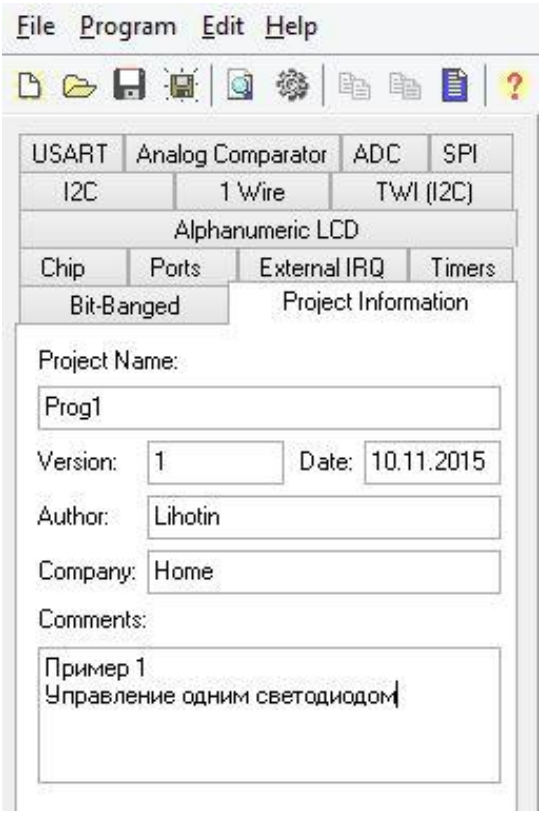


Рис.5. Занесение информации для заголовка программы

Рис. 6. Запуск процесса генерации программы

После того, как и этот файл будет записан, процесс генерации проекта завершается. На экране появляются два новых окна. В одном окне открывается **содержимое файла**

«Progl. c». Втрое окно — это **файл комментариев**. В тело цикла while {} программы добавим оператор

```
while (условие)
{
PORTB=PIND;
};
```

1.3. Описание программы

Начинается программа с заголовка. В начале заголовка мастер поместил информацию о том, что программа создана при помощи CodeWizardAVR. Далее в заголовок включен блок информации из вкладки «Project Information». Далее сообщается тип МК, его тактовая частота, модель памяти, размер используемой внешней памяти и размер стека.

В строке 1 находится инструкция **include**, присоединяющая файл описаний. После команды include мастер поместил сообщение для программиста, которое предупреждает о том, что именно в этом месте нужно поместить описание всех глобальных переменных. В данном конкретном случае глобальные переменные не нужны.

Функция main содержит в себе набор команд, настройки системы и заготовку главного цикла программы. Лишние команды при желании можно убрать.

Что касается настройки портов PB и PD, то в PORTB и DDRB присваивается значение 0x7F. А в в регистр PORTD записывается 0x7F, а в регистр DDRD — ноль.

После инициализации всех регистров начинается основной цикл программы. **Основной цикл** — это обязательный элемент любой программы для МК. Поэтому мастер всегда создает заготовку этого цикла. То есть создает цикл, тело которого пока не содержит ни одной команды.

В том месте, где нужно расположить команды, составляющие тело цикла, мастер помещает специальное сообщение, приглашающее вставить код программы: «Place your code here» («Пожалуйста, вставьте ваш код»).

В процессе расширенной трансляции программы формируется не только HEX-файл, но и файл той же программы, переведенный на язык Ассемблер, а также специальный файл в COF-формате, предназначенный для передачи программы в AVR Studio для отладки. После создания и расширенной трансляции проекта его директория будет содержать файлы со следующими расширениями:

prj — файл проекта Code Vision AVR;

txt — файл комментариев. Это простой текстовый файл, который заполнялся по своему усмотрению;

c — текст программы на языке СИ;

asm — текст программы на Ассемблере (сформирован Code Vision);

cof — формат для передачи программы в другие системы для отладки;

eep — содержимое EEPROM (формируется одновременно с HEX-файлом);

hex — результат трансляции программы;

inc — файл-дополнение к программе на Ассемблере с описанием всех зарезервированных ячеек и определением констант;

1st — листинг трансляции программы на Ассемблере;

map — распределение памяти микроконтроллера для всех переменных программы на СИ;

obj — объектный файл (промежуточный файл, используемый при трансляции);

rom — описание содержимого программной памяти (та же информация, что и в HEX-файле, но в виде таблицы);

vec — еще одно дополнение к программе на Ассемблере, содержащее команды переопределения векторов прерываний;

cwp — файл построителя проекта. Содержит все параметры, которые были введены в построитель.

Все перечисленные выше файлы имеют одинаковые имена, соответствующие имени проекта. Кроме перечисленных выше файлов, директория проекта может содержать не-

сколько файлов с р асширением типа c~, pr~ или cw~. Это страховочные копии соответственно файлов c, prj и cwp. То же самое, что файл bak для текстовых файлов.

Программа Code Vision AVR разработана румынской фирмой «HP Infotech», специализирующейся на разработке программного обеспечения. Инсталляционный пакет свободно распространяемой версии программы, рассчитанной на создание программ, результирующий код которых не превышает 2 Кбайт, можно скачать в Интернете по адресу

http://www.hpinfotech.ro/html/download.htm.

Там же можно скачать архив с полной и облегченной коммерческими версиями той же программы, защищенные паролем. Эти версии платные. Условия предоставления права на использование этих программ можно прочитать на той же самой странице.

1.4. Интерфейс программы Code Vision AVR

1.4.1. Окно номер 1

Интерфейс программы Code vision avr показан на рис. 7. Он напоминает интерфейс AVR Studio, он гораздо проще. Основная панель Code Vision тоже разделена на три окна. **Окно номер 1** имеет три вкладки разного назначения. «Корешки» этих вкладок расположены в верхней части окна.

Первая вкладка называется «Navigator». В окне этой вкладки показана структура текущего открытого проекта. Структура включает в себя список всех файлов, из которых состоит проект, а также список найденных ошибок и предупреждений, который появляется здесь после трансляции программы. В данном случае под файлами проекта понимаются не все те файлы, которые были перечислены в предыдущем разделе, а только исходные файлы (тексты программ на языке Си и файл описания). Щелчок мышью по имени любой из функций в дереве проекта приведет к тому, что окно с текстом программы, содержащей эту функцию, переместится на передний

план, и текстовый курсор установится в начало выбранной функции.

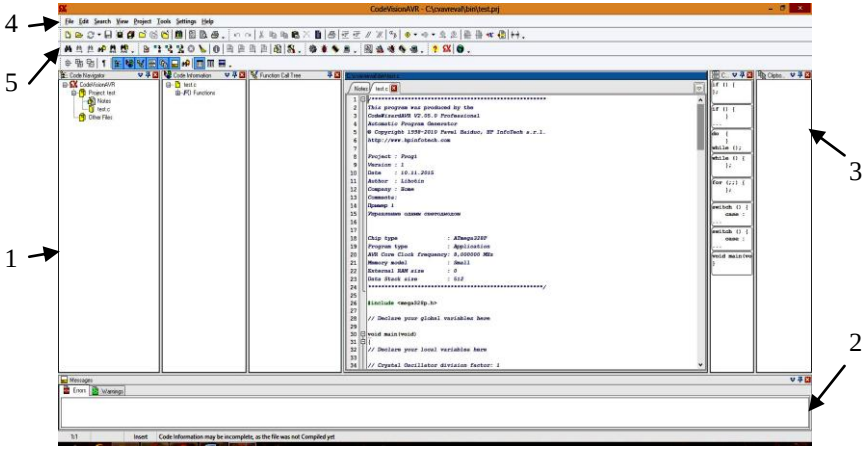


Рис. 7. Основное окно программы Code Vision

Вторая вкладка окна 1 называется «Code Templates» (Шаблоны Кода). Эта вкладка задумана как помощь программисту. Она содержит шаблоны нескольких основных конструкций языка СИ. В частности, шаблоны операторов for, while, if и так далее. В любой момент программист может открыть эту вкладку, выбрать нужную структуру и просто «перетянуть» ее при помощи мыши в основной текст программы (в окно 3). При этом перетянутый текст будет скопирован. Нужно лишь заполнить полученный таким образом шаблон командами, и фрагмент программы готов. При желании можно пополнить набор шаблонов.Для того, чтобы ввести новый элемент в список шаблонов, сначала наберите его текст в текстовом окне программы (окно 3), выделите его и «перетяните» при помощи мыши в окно шаблонов. Записанный таким образом шаблон останется в окне шаблонов даже после загрузки нового проекта. С этого момента можно использовать его в любой другой программе. Лишние шаблоны можно удалить. Для этого достаточно щелкнуть по ненужному шаблону

правой кнопкой мыши и в появившемся меню выбрать пункт «Delete».

Третья вкладка окна номер 1 называется «Clipboard History» (История значений буфера обмена). Ее структура такая же, как структура вкладки «Code Templates». Но вместо шаблонов вкладка содержит все, что в процессе редактирования попадало в буфер обмена. Если снова понадобилось какое-либо из этих значений, в любой момент можно «перетянуть» его при помощи мышки в программу.

Окно номер 1 при необходимости можно быстро убрать с экрана. Для этого достаточно нажать кнопку, которая расположена в левой части окна 5 на панели инструментов. При повторном нажатий на эту кнопку окно появляется вновь.

1.4.2. Окно номер 2

Окно номер 2 содержит всего одну вкладку. Она называется **«Messages» (Сообщения).** Сюда выводятся все сообщения об ошибках и предупреждения в процессе трансляции. Для перехода к месту программы, где найдена ошибка, достаточно выполнить двойной щелчок мышкой по соответствующему сообщению об ошибке в окне 2.

1.4.3. Окно номер 3

Окно номер 3 может содержать одно или несколько окон с текстом программ на языке Си. Кроме того, там же появляется окно «Project Notes» - окно комментариев к текущему проекту. Все окна, появляющиеся в окне 3, обладают свойствами текстового редактора, где поддерживаются функции выделения фрагментов, их перетаскивания, копирования, вставки, поиска, замены и т. д.

Так как система Code Vision AVR не поддерживает функцию отладки, то отсутствуют все команды, связанные с этим режимом. Отсутствует также механизм закладок.

1.5. Настройка компилятора

Перед тем, как приступать к трансляции проекта, необходимо настроить **параметры компилятора**. Снова открываем окно «Configure project». В окне проекта открываем вкладку «C Compiler». На этой вкладке имеется множество параметров, определяющих стратегию компиляции. Установим только главные из них.

Сначала нужно выбрать тип микросхемы. Для этого служит выпадающее меню с заголовком «Chip:». Затем нужно выбрать частоту тактового генератора. Данные о частоте будут использованы транслятором при формировании процедур задержки. Выбор частоты производится при помощи другого выпадающего меню, озаглавленного «Clock:». При желании, можно **выбрать способ оптимизации**. Для такого выбора служит поле «Optimize for:» («оптимизировать по:»). Предлагаются два способа:

- оптимизация по размеру («Size»);
- оптимизация по скорости («Speed»).

Оптимизация по размеру заставляет компилятор создавать результирующий код программы, минимальный по длине. Оптимизация по скорости позволяет создать более длинную, но зато более быстродействующую программу.

После того, как все параметры установлены, необходимо нажать кнопку «Ок» в нижней части окна «Configure project». Окно проекта закроется. Теперь можно **приступать к компиляции**. Директивы режима компиляции приведены на рис. 8. Если программа достаточно большая, то перед компиляцией можно проверить ее на ошибки. Для этого служит директива «Проверка синтаксиса». При выборе директивы «Компиляция» проверка синтаксиса производится автоматически.

В процессе компиляции создается объектный файл в HEX- формате, а также файл, содержащий данные для EEPROM (расширение eep). Директива «**Построить проект**» запускает процедуру полного построения проекта.

Таблица 5.5. *Директивы работы с программой Code Vision AVR*

Название	Пункт меню «Project»	Кнопка	Горячая клавиша	Описание
Проверка синтаксиса	Check Sintax		-	Проверить синтаксис программы в текущем окне
Компиляция	Compile		F9	Скомпилировать программу
Построить	Make		Shift+F9	Построить проект
Конфигурация	Configure		-	Открыть окно конфигурации проекта

Рис.8. Директивы работы с программой Code Vision AVR

1.6. Программа для загрузки прошивки в ATMega 328P-PU

Программа ARP/Arduino Uploader, по сути, является графической оболочкой программы AVR Dude, настроенной специально для прошивки МК, и использует тот же принцип загрузки прошивки, что и среда Arduino. Загрузка происходит с помощью загрузчика Boot Loader, записываемого на все платы Arduino по умолчанию, по протоколу STK500. Главное окно программы ARP/Arduino Uploader представлено на рис. 9.

1.7. Порядок загрузки прошивки

На панели «File» необходимо кликнуть по кнопке «Browse». Откроется диалоговое окно, в котором нужно указать требуемый файл прошивки.

На панели Setup в выпадающем списке «COM Port» необходимо выбрать COM Port, соответствующий подключенной плате Arduino. Далее в выпадающем списке «Microcontroller» выбрать нужный тип МК (для Arduino Uno - это m328p).

В поле параметров «AVR Dude Params», передаваемых программе AVR Dude, значение скорости передачи данных нужно изменить с 19200 на 115200 для правильной связи с загрузчиком Arduino.

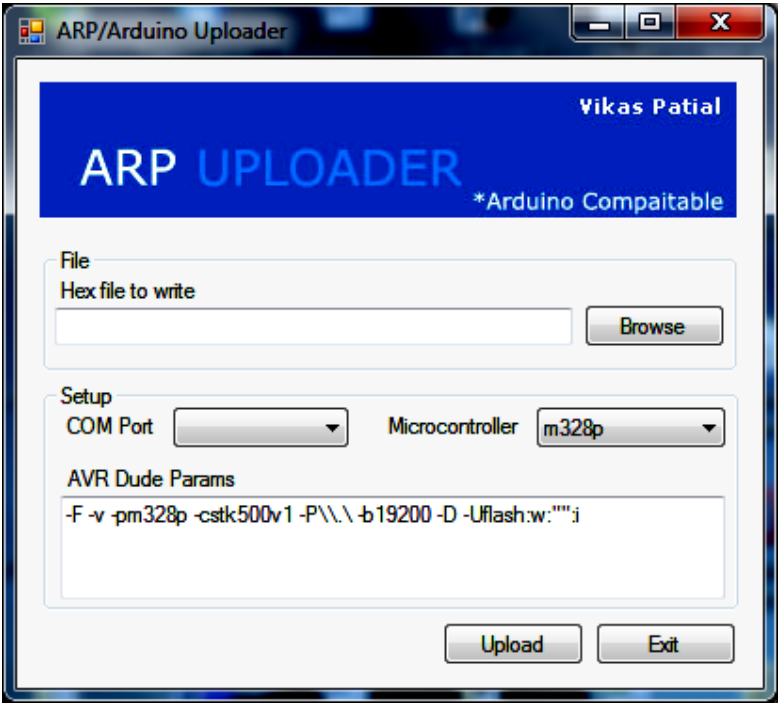


Рис. 9. Главное окно программы ARP/Arduino Uploader

1.8. Набор внешних устройств

На лабораторных работах будем реализовывать различные алгоритмы управления внешними устройствами, которые представляют собой небольшие платы с разъемами ВН-10. Эти разъемы имеют 10 выводов (8 линий порта, питание, земля). Распиновка разъемов периферии и всех портов выполнена унифицировано в соответствии с табл. 2

В ходе данной лабораторной работы из набора периферийных устройств потребуются светодиодная линейка и тумблерный регистр.

Схема подключения светодиодов к выводам МК показана на рис. 10. В этой схеме светодиоды будут светиться, когда МК выдает сигнал «1» (высокое напряжение). Когда вывод

работает в качестве входа данных или на него выводится «0», то светодиод будет выключен.

Таблица 2. Распиновка разъемов ВН-10

Номер ножки порта	Наименование вывода
1	Px0
2	Px1
3	Px2
4	Px3
5	Px4
6	Px5
7	Px6
8	Px7
9	VCC
10	GND

Схема подключения светодиодов к выводам МК показана на рис. 10. Схема подключения тумблеров к выводам МК показана на рис. 11.

Задание 2. Используя светодиодную линейку и тумблерный регистр проверьте работу выше разработанной программы. Для этого подключите светодиоды к порту B, а тумблеры к порту D. После создания проекта загрузите прошивку в МК ATMega 328P-PU с помощью программы ARP/Arduino Uploader. Продемонстрируйте работу программы.

Управляющая программа МК жестко зависит от схемы электрической принципиальной разрабатываемого устройства. В процессе отладки программы производится поиск ошибок не только в программном коде, но и в аппаратной части разрабатываемого устройства. При этом не исключены как ошибки монтажа устройства, так и самой его схемы.

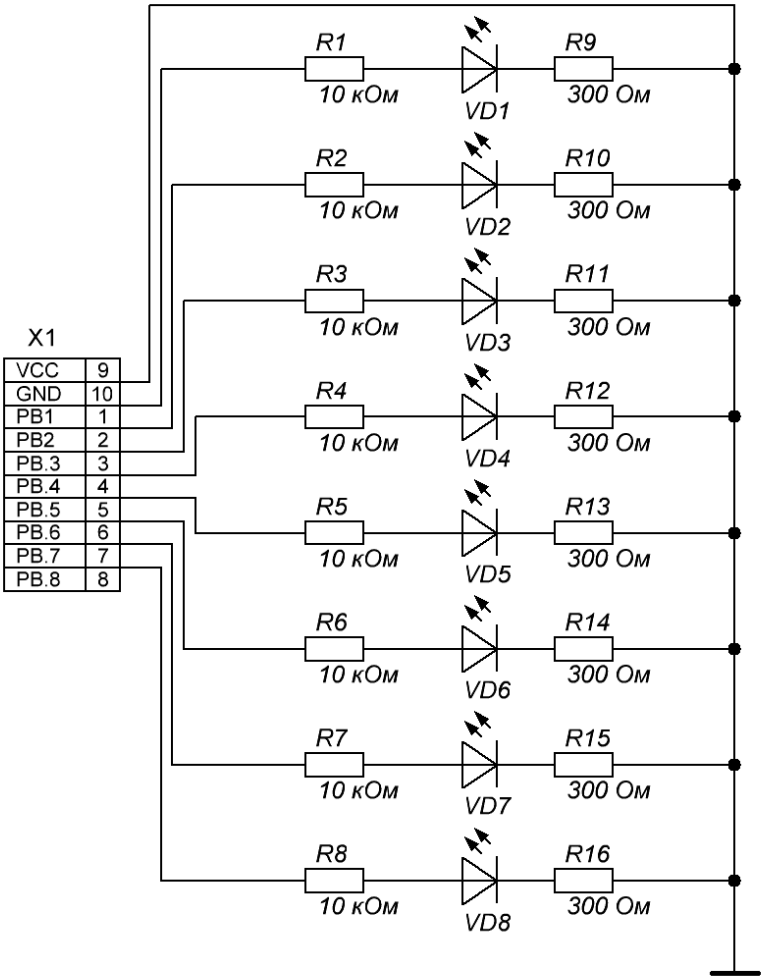


Рис. 10. Схема подключения светодиодов к порту МК

Светодиоды и тумблеры можно подключить к одному порту через общую шину. Функциональная схема и соответствующий программный модуль приведены на рис. 12.

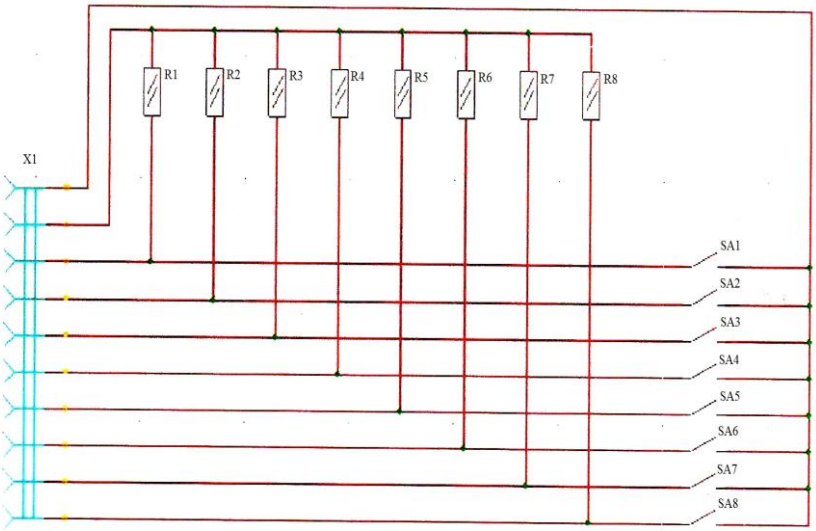


Рис. 11. Схема подключения тумблеров к порту МК

При начальной установке все биты портов D и C устанавим в 1, т.е. на вывод. Следовательно, регистр, к которому подключены тумблеры, отключается от шины, т.к. $\overline{OE} = 1$, а регистр, к которому подключены светодиоды, будет работать в режиме записи $(\overline{OE} = 0, stb = 1)$. На выходах D0:D7 этого регистра установятся единичные сигналы, и светодиоды окажутся погашенными (отсутствует разность потенциалов). Составим фрагмент программы, который обеспечивает ввод восьмибитного кода с тумблерного регистра и выдачу его на светодиоды. Не забудьте объявить переменную x как

unsigned char x;

Обратите внимание, что многоточие в фрагменте программы означает, что часть программы с настройками пропущена для наглядности.

-

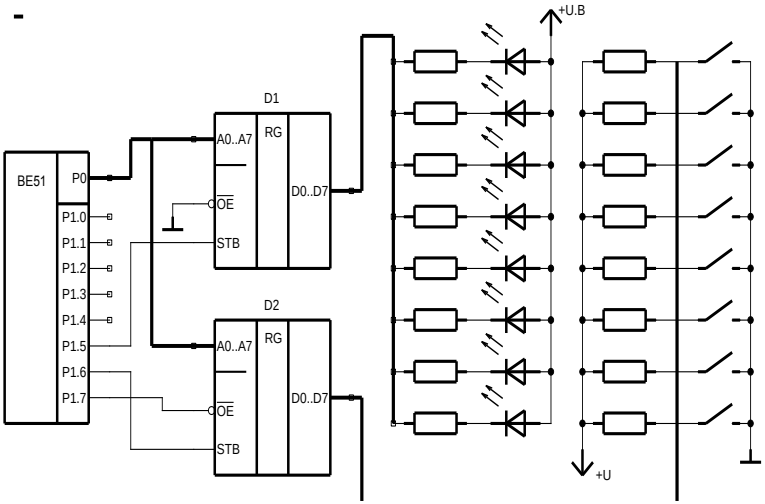


Рис. 12. Пример схемы простейшего устройства, построенного с использованием микроконтроллера

```
....
main()
{
…
//схему D1 переводим в режим чтения
PORTC.5 = 0;
//схему D2 переводим в режим записи
PORTC.6 = 1;
//разрешаем информационную связь со схемой D2, и тем
//самым в регистр D2 и в порт PD записываем состояние
//тумблеров
PORTC.7 = 0;
//настраиваем порт D на ввод
DDRD=0x0;
//состояние порта  D сохраняем в переменной x
x = PIND;
//переводим схему D2 в состояние чтения
PORTC.6 = 0;
```

```
//отключаем схему D2 от шины
PORTC.7 = 1;
//настраиваем порт D на вывод
DDRD=0xFF;
//инвертируем состояние x и записываем в порт D;
//схема D1 находится в состоянии записи;
//данные с D запишутся в регистр и выведутся
//на светодиоды
PORTD = ~ x;
//схему D1 переводим в режим записи
PORTC.5 = 1;
//бесконечный цикл
while(1);
}
```

Задание 2.

Разработайте МПС на базе МК ATMega16 обеспечивающую ввод восьмибитного кода с тумблерного регистра и выдачу его на светодиоды в соответствии с вариантом задания. Шина данных (ШД) – это шина, подключенная к порту A (рис. 12), управляющие сигналы (УС) – линии, подключенные к порту C. Напишите и отладьте соответствующий программный модуль.

Варианты заданий:

а) шина данных (ШД)– порт C, управляющие сигналы (УС) – порт A;

б) ШД – порт B, УС – порт A;

в) ШД – порт B, УС – порт D;

г) ШД – порт D, УС – порт A;

д) ШД – порт D, УС – порт B.

Задание 3. Подключите к МК ATmega16 два адаптера параллельного ввода/вывода КР580ВВ55. Напишите программу инициализации адаптеров для случая работы всех каналов первого адаптера на ввод в режиме 0, а всех каналов второго адаптера на вывод в режиме 0. Отладьте программный модуль.

Справочный материал для выполнения задания приведен на рис. 13, рис. 14 и в табл. 3.

Схема строится по аналогии с рис. 12. Один из портов выбирается для подключения шины данных, по которой будет передаваться информация в один из портов адаптера. Еще один порт МК используется для подключения сигналов, управляющих работой адаптера.

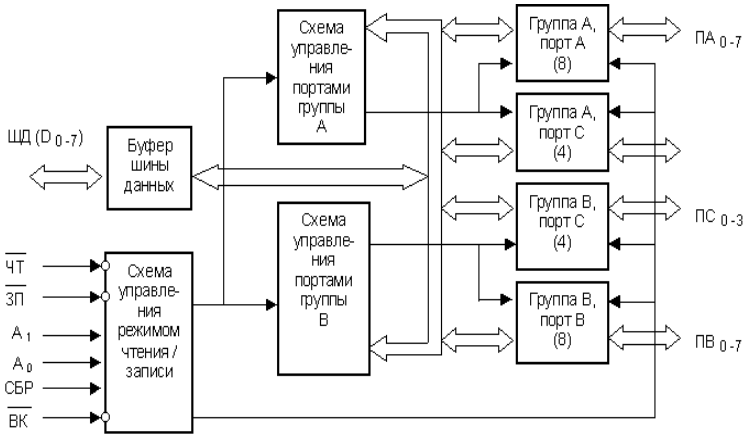


Рис. 13. Программируемый периферийный адаптер К580ВВ55

Настройка адаптера осуществляется программно посредством управляющего слова. В начале необходимо составить управляющее слово в соответствии со структурой, приведенной на рис. 14, Программа должна записать его в регистр управляющего слова (РУС). Для этого необходимо установить соответствующие управляющие сигналы контроллера как показано в предпоследней строчке табл. 3.

Домашнее задание

1. Составьте отчет, в который включите программы и схемы, над которыми Вы работали, выполняя задания 1, 2, 3.

2. Изучите систему прерываний МК семейства AVR.

Таблица 3. Основные операции ППА

Действие	Сигналы управления					Операции
	A_1	A_0	$\overline{ЧТ}$	$\overline{ЗП}$	$\overline{ВК}$	
Чтение	0	0	0	1	0	$ПА \rightarrow ШД$
	0	1	0	1	0	$ПВ \rightarrow ШД$
	1	0	0	1	0	$ПС \rightarrow ШД$
Запись	0	0	1	0	0	$ШД \rightarrow ПА$
	0	1	1	0	0	$ШД \rightarrow ПВ$
	1	0	1	0	0	$ШД \rightarrow ПС$
	1	1	1	0	0	$ШД \rightarrow РУС$
Отключение	–	–	–	–	–	ШД и порты отключены

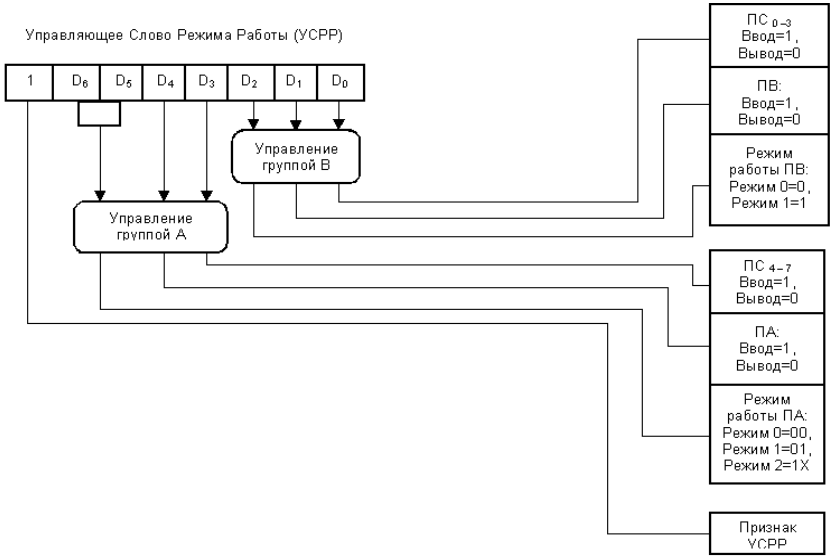


Рис. 14. Управляющее слово РЕЖИМ РАБОТЫ (УСРР) адаптера КР580ВВ55

ЛАБОРАТОРНАЯ РАБОТА №2.
ПОДСИСТЕМА ВНЕШНИХ ПРЕРЫВАНИЙ

Цель лабораторной работы: получение навыков работы с системой прерываний микроконтроллеров семейства Atmega.

Для того, чтобы определить конфигурацию внешних прерываний необходимо вызвать закладку **External IRQ** в среде **Code Vision**. Отметив соответствующее окошко **INTx Enabled** (Разрешить прерывание x), можно разрешить соответствующее внешнее прерывание. Далее появиться окошко **Mode** (Способ) с выпадающим списком, в котором можно выбрать способ инициирования прерывания:

- **Low level** (Низкий уровень) – прерывание будет инициировано по низкому уровню на соответствующем входе;

- **Falling Edge** (Спадающий фронт) – прерывание будет инициировано по спадающему фронту на соответствующем входе;

- **Rising Edge** (Нарастающий фронт) – прерывание будет инициировано по нарастающему фронту на соответствующем входе.

Для каждого разрешенного внешнего прерывания **CodeWizardAVR** определит программу обслуживания прерывания ext_intx_isr, где x – номер внешнего прерывания.

В таблице векторов прерываний **INT0** имеет адрес $0002, **INT1** - адрес $0004, **INT2** - адрес $0024. В случае последнего прерывания обнаружение фронтов сигналов происходит асинхронно, т.е. не требует наличия тактового импульса.

Для выполнения лабораторной работы из набора периферийных устройств нам потребуется плата со светодиодной линейкой (рис. 10), плата с кнопками, схема которой представлена на рис. 15, и плата с восьмисегментными индикаторами.

Для устранения дребезга кнопки подключаются к порту МК через триггеры.

Схема подключения индикаторов к порту МК представлена на рис. 16, коды символов – в табл. 4.

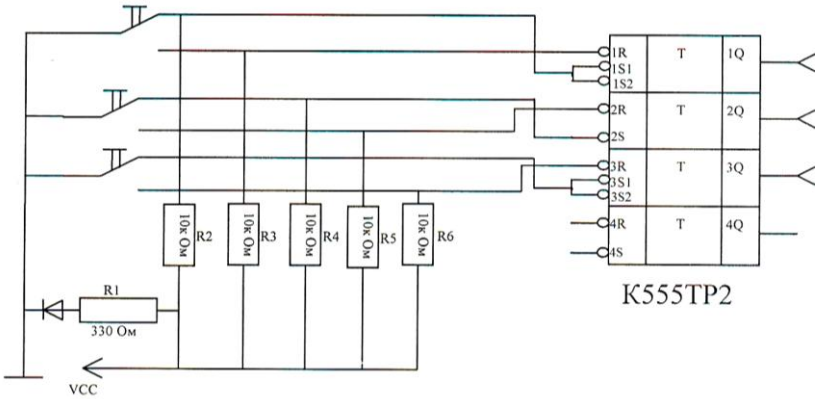


Рис. 15. Схема электрическая принципиальная платы с кнопками

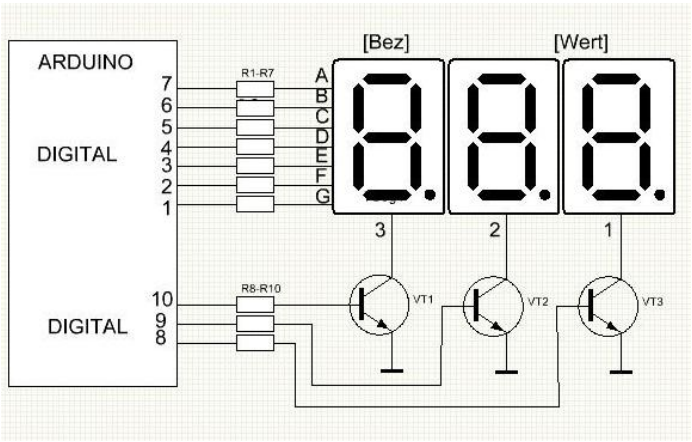


Рис. 16. Схема подключения восьмисегментных индикаторов к МК

Разработаем структуру МПС и напишем программный модуль, обеспечивающий изменение состояния светодиода по запросу от специальной кнопки. Функциональная схема МПС представлена на рис. 17. К выводу PORTD.2 (INT0) подключена кнопка. Для подавления дребезга использован RS-

триггер. По нажатию кнопки на вывод PORTD.2 подается сигнал низкого уровня, что приводит к вызову программы-обработчика прерывания. Соответствующий программный модуль представлен ниже. Для запуска обработчика прерывания необходимо нажатие кнопки.

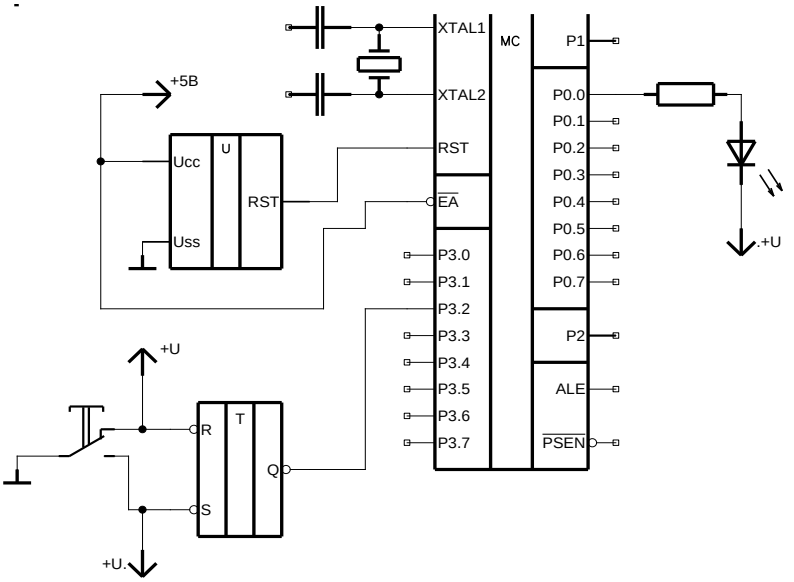


Рис. 17. Функциональная схема МПС

```
interrupt [EXT_INT0] void ext_int0_isr(void)
{
cha
//изменение состояния светодиода
X= PORTB;
PORTB=~X;
}

main()
{
```

```
//настройки
…
while(1);//бесконечный цикл
…
}
```

Ключевое слово interrupt используется для объявления функции, использующейся как обработчик прерывания.

Задания на лабораторную работу

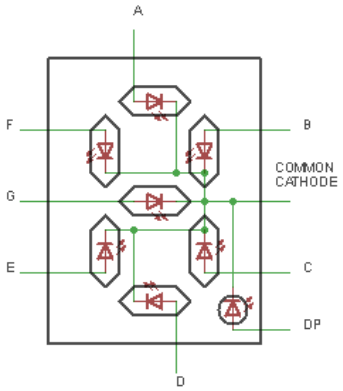
Задание 1. Проверьте работу составленного выше программного модуля на базе платы Arduino.

Задание 2. Разработайте структуру МПС на базе МК ATMega328P-PU и напишите программный модуль, обеспечивающие выдачу на семисегментный индикатор шестнадцатиричного кода по запросу от специальной кнопки.

При выводе данных на индикаторы необходимо учитывать соответствие выводов индикаторов и номеров контакта шлейфа (см. рис. 18).

Домашнее задание

Составьте отчет, в который включите программы и схемы, над которыми Вы работали, выполняя задания 1, 2.



1	2	3	4	5	6	7	8	9	10	11	12	13	14
D	C	E	DP	B	G	F	A			4	3	2	1

Рис. 18. Соответствие выводов индикаторов и номеров контакта шлейфа

ЛАБОРАТОРНАЯ РАБОТА №3. ДИНАМИЧЕСКИЙ ДИСПЛЕЙ. ПРОЦЕДУРА ПОЛЛИНГА.

Цель лабораторной работы: закрепление полученных навыков работы с микроконтроллерами семейства Atmega.

Задание 1. Разработайте структуру МПС на базе МК ATMega328P-PU и напишите программный модуль, обеспечивающие по запросу от специальной кнопки:

– ввод 8- битного двоичного кода с тумблерного регистра;

– преобразование его в двоично-десятичное представление;

– выдачу двоично-десятичного кода на семисегментные индикаторы.

Задание 2. Организуйте подсистему внешних прерываний в системе на базе МК ATMega328P-PU. Количество источников прерывания - 4 (например 4 кнопки). Подключите к МПС 4 светодиода. Требуется выполнить следующие задачи:

а) проанализируйте функциональную схему МПС, представленную на рис. 19 и разработайте свою в соответствии с заданием;

б) будем считать, что каждому источнику запроса прерывания (кнопке) соответствует свой светодиод. Напишите программный модуль, реализующий изменение состояния светодиода, соответствующего устройству (соответствие - по желанию разработчика программы);

в) отладьте программный модуль.

Варианты задания 3:

1. Кнопку подключить к INT0, светодиоды - к PORTD, источники запроса на прерывание - к PORTB.
2. Кнопку подключить к INT0, светодиоды - к PORTB, источники запроса на прерывание - к PORTD.
3. Кнопку подключить к INT1, светодиоды - к PORTD, источники запроса на прерывание - к PORTB.

4. Кнопку подключить к INT1, светодиоды - к PORTB, источники запроса на прерывание - к PORTD.

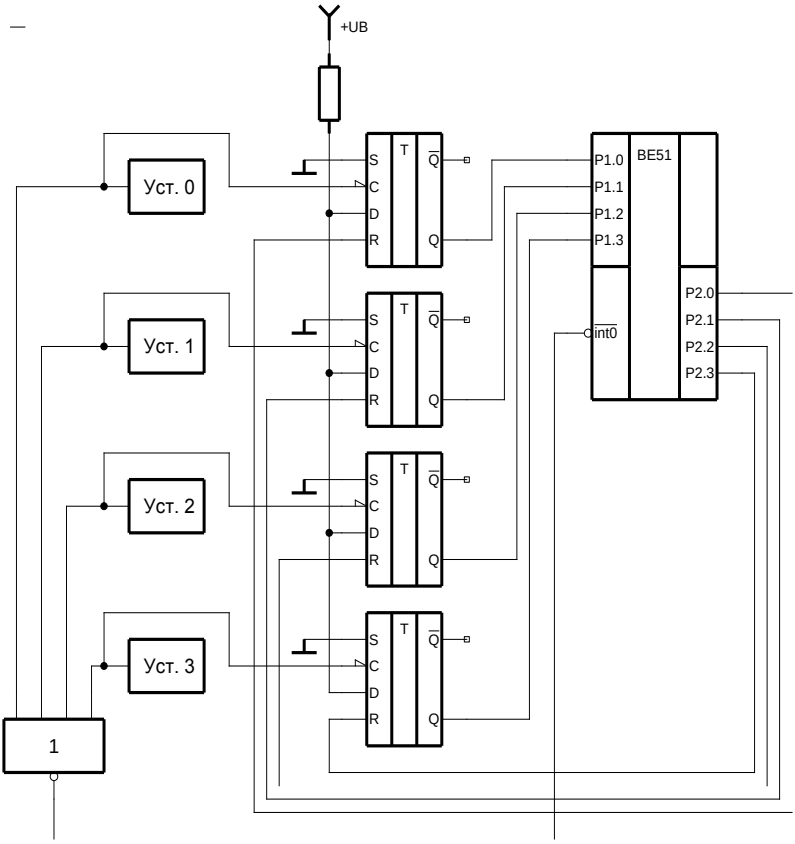


Рис. 19. Функциональная схема подсистемы внешних прерываний

Краткие пояснения к алгоритму решения задачи

1. При начальной установке все биты порта PORTC устанавливаются в 1. С порта PORTC единичные сигналы поступают на входы R каждого из D-триггеров и сбрасывают их в состояние «0». Это состояние будет сохраняться до тех пор, пока на входе R присутствует единичный сигнал. Следовательно, функция main() должна начинаться с установки в 0 выводов микроконтроллера, соединенных с входами R D-триггера, и установки управляющих битов для работы подсистемы внешних прерываний.

2. Когда устройство выставляет запрос на прерывание – этот запрос фиксируется в D-триггере и поступает на соответствующий вход микроконтроллера, а также подается на элемент ИЛИ. С выхода этого элемента на вход запроса прерывания INT0 микроконтроллера подается 0, что приводит к автоматческому запуску обработчика прерываний.

3. В обработчике прерываний в порядке приоритета (по усмотрению разработчика) просматриваются биты порта PD и находится устройство, обратившееся с запросом на прерывание. Далее необходимо запустить функцию, которая изменит состояние соответствующего светодиода, сбросит состояние триггера в 0 и переведет его в режим хранения.

Домашнее задание

Составьте отчет, в который включите схемы и программы, над которыми Вы работали, выполняя задания 1, 2.

ПРИЛОЖЕНИЕ 1
ЭЛЕКТРОННЫЙ КОНСТРУКТОР ARDUINO UNO

1. Основные характеристики

Плата Arduino Uno состоит из МК Atmel AVR ATmega328, а также элементов обвязки для программирования и интеграции с другими схемами. На плате присутствуют линейные стабилизаторы напряжения +5 и +3,3 В. Тактирование осуществляется на частоте 16 МГц кварцевым резонатором. В МК предварительно прошивается загрузчик BootLoader, поэтому внешний программатор не нужен. В качестве преобразователя USB-Serial используется МК ATmega8U2. Улучшена стабильность цепей сброса МК.

2. Основные требования к функциональности конструктора Arduino Uno

Основные характеристики изделия:

1) относительно низкое энергопотребление;

2) модульность платы – возможность выбора портов и подключаемой периферии пользователем;

3) возможность конфигурации МК - переключение осуществляется с помощью четырехрежимного реле;

4) средства отладки – реализация прошивки микроконтроллера платы;

5) максимальная универсальность элементов платы – подключение модулей так, чтобы их можно было использовать независимо от данного комплекса, а вновь разработанные – средствами комплекса;

7) возможность гибкого управления питанием, регулирование напряжения питания с помощью стандартного блока питания в непересекающихся диапазонах от 0 до 12В;

8) возможность управления устройствами комплекса через порты RS-232 и USB.

3. Плата Arduino Uno

Плата Arduino Uno построена на основе микроконтроллера ATmega328P-PU. Платформа имеет 14 цифровых

входов/выходов, 6 аналоговых входов(которые можно использовать как цифровые), кварцевый генератор с частотой 16 МГц, разъем USB, силовой разъем, разъем ICSP и кнопку перезагрузки. Для работы необходимо подключить платформу к компьютеру посредством кабеля USB, либо подать питание при помощи адаптера AC/DC или батареи.

Соответствие колодок платы Arduino Uno пинам МК ATMega 328P-PU представлено на рис. П1.1.

	-			PC5 ADC5/SCL/PCINT13
IOREF	IOREF			PC4 ADC4/SDA/PCINT12
RESET	RESET		AREF	AREF
3.3V	3.3V		GND	GND
5V	5V		13	PB5 SCK/PCINT5
GND	GND		12	PB4 MISO/PCINT4
GND	GND		11	PB3 MOSI/OC2A/PCINT3
Vin	Vin		10	PB2 SS/OC1B/PCINT2
			9	PB1 PCINT1/OC1A
			8	PB0 PCINT0/CLK0/ICP1
			7	PD7 PCINT23/AIN1
			6	PD6 PCINT22/OC0A/AIN0
ADC0/PCINT8 PC0	A0		5	PD5 PCINT21/OC0B/T1
ADC1/PCINT9 PC1	A1		4	PD4 PCINT20/XCK/T0
ADC2/PCINT10 PC2	A2		3	PD3 PCINT19/OC2B/INT1
ADC3/PCINT11 PC3	A3		2	PD2 PCINT18/INT0
ADC4/SDA/PCINT12 PC4	A4		1	PD1 PCINT17/TXD
ADC5/SCL/PCINT13 PC5	A5		0	PD0 PCINT16/RXD

Рисунок П1.1. Соответствие колодок платы Arduino Uno пинам микроконтроллера ATMega 328P-PU**3.**

4. Питание

Arduino Uno может получать питание через подключение USB или от внешнего источника питания. Источник питания выбирается автоматически.

Внешнее питание (не USB) может подаваться через преобразователь напряжения AC/DC (блок питания) или аккумуляторной батареей. Преобразователь напряжения подключается посредством разъема 2.1 мм с центральным положительным полюсом. Провода от батареи подключаются к выводам Gnd и Vin разъема питания.

Платформа может работать при внешнем питании от 6В до 20В. При напряжении питания ниже 7В, вывод 5V может выдавать менее 5В, при этом платформа может работать нестабильно. При использовании напряжения выше 12В регулятор напряжения может перегреться и повредить плату. Рекомендуемый диапазон от 7В до 12В.

5. Выводы питания

VIN. Вход используется для подачи питания от внешнего источника (в отсутствие 5В от разъема USB или другого регулируемого источника питания). Подача напряжения питания происходит через данный вывод.

5V. Регулируемый источник напряжения, используемый для питания МК и компонентов на плате. Питание может подаваться от вывода VIN через регулятор напряжения, или от разъема USB, или другого регулируемого источника напряжения 5В.

3V3. Напряжение на выводе 3.3В, генерируемое встроенным регулятором на плате. Максимальное потребление тока 50 мА.

GND. Выводы заземления.

6. Связь

На платформе Arduino Uno установлено несколько устройств для осуществления связи с компьютером, другими устройствами Arduino или МК. ATmega328P поддерживает по-

следовательный интерфейс UART TTL (5В). Установленный на плате МК ATmega8U2 направляет данный интерфейс через USB, программы на стороне компьютера "общаются" с платой через виртуальный COM порт. Прошивка ATmega8U2 использует стандартные драйвера USB COM, поэтому никаких сторонних драйверов не требуется, но на Windows для подключения требуется файл arduino.inf. Светодиоды RX и TX на платформе будут мигать при передаче данных через микросхему FTDI или USB подключение (но не при использовании последовательной передачи через выводы 0 и 1). ATmega328 поддерживает интерфейсы I2C (TWI) и SPI.

7. Программирование

Платформа может программироваться посредством ПО Arduino. Из меню Tools > Board выбирается «Arduino Uno» (согласно установленному микроконтроллеру). Также можно с помощью программы ARP/Arduino Uploader загрузить прошивку, написанную в любой другой среде программирования (Code Vision, AVR Studio, WinAVR, …).

Микроконтроллер ATmega328P поставляется с записанным загрузчиком, облегчающим запись новых программ без использования внешних программаторов. Связь осуществляется оригинальным протоколом STK500.

Имеется возможность не использовать загрузчик и запрограммировать микроконтроллер через выводы ICSP (внутрисхемное программирование).

8. Автоматическая (программная) перезагрузка

Uno разработана таким образом, чтобы перед записью нового кода перезагрузка осуществлялась самой программой Arduino на компьютере, а не нажатием кнопки на платформе. Одна из линий DTR микросхемы ATmega8U2, управляющих потоком данных (DTR), подключена к выводу перезагрузки микроконтроллера ATmega328P через 100 нФ конденсатор. Активация данной линии, т.е. подача сигнала низкого уровня, перезагружает микроконтроллер. Программа Arduino, исполь-

зуя данную функцию, загружает код одним нажатием кнопки Upload в самой среде программирования. Подача сигнала низкого уровня по линии DTR скоординирована с началом записи кода, что сокращает таймаут загрузчика.

Функция имеет еще одно применение. Перезагрузка Uno происходит каждый раз при подключении к программе Arduino на компьютере с ОС Mac X или Linux (через USB). Следующие полсекунды после перезагрузки работает загрузчик. Во время программирования происходит задержка нескольких первых байтов кода во избежание получения платформой некорректных данных (всех, кроме кода новой программы). Если производится разовая отладка скетча, записанного в платформу, или ввод каких-либо других данных при первом запуске, необходимо убедиться, что программа на компьютере ожидает в течение секунды перед передачей данных.

На Arduino Uno имеется возможность отключить линию автоматической перезагрузки разрывом соответствующей линии. Контакты микросхем с обоих концов линии могут быть соединены с целью восстановления. Линия маркирована «RESET-EN». Отключить автоматическую перезагрузку также можно, подключив резистор 110 Ом между источником 5 В и данной линией.

9. Токовая защита разъема USB

В Arduino Uno встроен самовостанавливающийся предохранитель (автомат), защищающий порт USB компьютера от токов короткого замыкания и сверхтоков. Хотя практически все компьютеры имеют подобную защиту, тем не менее, данный предохранитель обеспечивает дополнительный барьер. Предохранитель срабатывает при прохождении тока более 500 мА через USB порт и размыкает цепь до тех пор, пока нормальные значения токов не будут восстановлены.

ПРИДОЖЕНИЕ 2
ХАРАКТЕРИСТИКИ МИКРОКОНТРОЛЛЕРА ATMega328P-PU

ATMega328P-PU – малопотребляющий CMOS - 8-битный микроконтроллер, основанный на AVR-усовершенствованной RISC-архитектуре. Выполняя команды за один такт процессора, ATMega328P-PU достигает производительности, приближающейся к 1 миллиону команд в секунду на 1МГЦ, и позволяет системному проектировщику оптимизировать потребление мощности в компромиссе со скоростью обработки.

Встроенная в чип внутрисистемная программируемая флэш-память позволяет перепрограммировать память программ через SPI-(serial peripheral interface) последовательный интерфейс или через обычный программатор.

ATMega328P-PU поддерживается программными, инструментальными средствами разработки, включая: Си-компиляторы, макроассемблеры, программные отладчик/симуляторы, внутрисхемные эмуляторы и оценочные комплекты.

Главными особенностями являются:

1) AVR RISC-архитектура, которая является быстродействующей и малопотребляющей:

– 131 команда - большинство выполняется за 1 так ЦПУ;

– 32 х 8 регистра общего назначения (32 х 8-ми разрядных регистра);

– производительность до 20 миллионов команд в секунду при частоте 20 МГЦ;

2) память программ (Flash память):

– объем внутрисистемной энергонезависимой Flash памяти программ составляет 32 КБ;

– Flash память программ допускает 10000 циклов записи/стирания;

3) энергонезависимая память EEPROM:

– объем EEPROM памяти составляет 1Кб;

– EEPROM допускает 100000 циклов записи/стирания;

4) объем внутренней SRAM памяти составляет 2Кб;

5) программирование защитной блокировки для Flash программ и EEPROM-данных;

6) периферийные особенности:

– два 8-битных таймер/счетчика с отдельными предделителями частоты и режимами сравнения;

– один 16-битный таймер/счетчик с отдельным предделителем частоты и режимом сравнения;

– 6 ШИМ каналов;

– 6 каналов 10-ти битного АЦП в PIP корпусе;

– встроенный в чип аналоговый компаратор;

– программируемый сторожевой таймер с отдельным генератором;

– USI - универсальный последовательный интерфейс;

– TWI интерфейс, совместимый с Philips I^2C;

– программируемый интерфейс USART;

7) дополнительные особенности МК:

– сброс по включению питания;

– программирование через SPI порт;

– внешние и внутренние источники прерывания;

– 6 режимов сна (Idle,ADC Noise Reduction, Power-Save, Power-down, Standby, Extemded Standby;

– программируемая схема защиты от пониженного напряжения питания Brown-out Detector (BOD);

– внутренний калиброванный генератор;

8) порты ввода/ вывода и корпуса:

– 23 программируемых линий ввода – вывода;

– корпус PDIP с 28 ножками, корпус TQFP с 32 ножками, корпуса QFN/MLF с 28 ножками;

9) напряжения питания: 1.8 - 5.5V;

10) таблица производительности: при 1.8 – 2.7V допускается 4 - 10МГц; при 2.7 - 4.5V допускается 0 - 20 МГц, при 4.5 - 5.5V допускается 20МГц;

11) типичное потребление энергии:

– активный режим: 1 МГЦ, 1.8V: 200 μA; 32kHz, 1.8V: 75 μA (при включенном генераторе);

– экономичный режим: .8V: <0.1 μA;

Диаграмма составляющих блоков приведена на Рис.П.2.1.

AVR ядро комбинирует богатый набор инструкций с 32 регистрами общего назначения. Все 32 регистра непосредственно связаны с арифметико-логическим устройством (ALU), что позволяет 2 независимым регистрам обращаться к одной инструкции и выполнять ее за 1цикл ЦПУ.

RISC архитектура более эффективна по коду, достигая прироста производительности в десять раз по сравнению с обычными CISC микроконтроллерами.

Холостой Режим (Idle mode) останавливает ЦПУ, но позволяет памяти SRAM, таймерам/счетчикам и системе прерывания продолжать функционировать.

Режим энергосбережения (Power Down mode) сохраняет содержимое регистров, но останавливает генератор, запрещает все другие функции контроллера до следующего прерывания или аппаратного сброса.

Режим сна (Standby mode) оставляет включенным только генератор, пока остальные функции контроллера выключены, что позволяет осуществлять экономию энергии и одновременно с этим быстро запускать контроллер в работу.

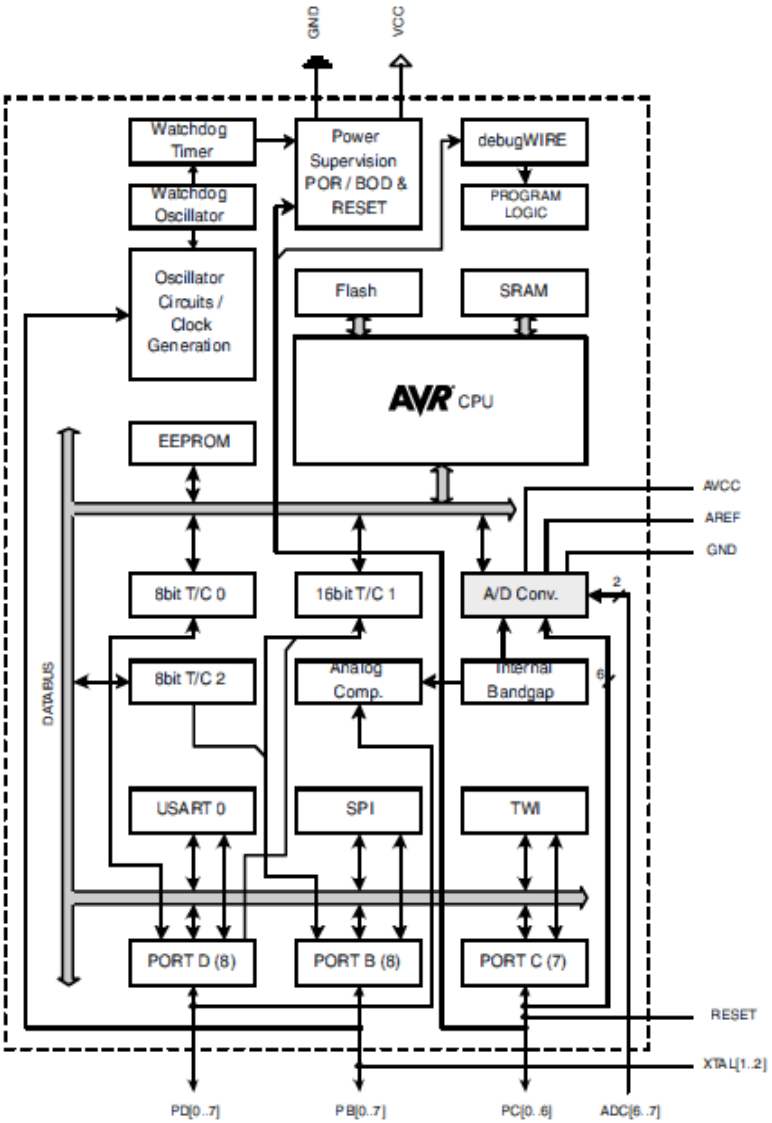


Рисунок П.2.1. Диаграмма блоков ATMega328P-PU.

Все порты МК (B, C, D) являются двунаправленными портами ввода/вывода с внутренними подтягивающими

(нагрузочными) резисторами (выбираемыми для каждого вывода). Выходные буферы портов имеют одинаковые характеристики с высокой нагрузочной способностью. Если выводы портов замыкаются на землю и включены нагрузочные резисторы, то выводы являются источниками тока. Сразу после сброса контроллера выводы портов находятся в высокоимпедансном состоянии (третьем состоянии) даже, если генератор не запущен.

Таблица П2.1. Функции выводов МК ATMega328P

VCC	«Цифровое» напряжение питания
GND	«Земля»
Port B (PB7..PB0)	8-битный порт B
Port C (PC6..PC0)	7-битный порт C
PortD (PD7..PD0)	8-битный порт D
RESET	Вывод сброса контроллера. Низкий уровень на этом выводе, длящийся дольше, чем минимальная длительность импульса, генерирует сброс, даже если ЦПУ не запущен. Более короткие импульсы не генерируют сброс в большинстве случаев. RESET имеет альтернативную функцию – PC6.
XTAL1	Этот вывод является входом инвертирующего усилителя и входом для внутренней схемы вырабатывающей тактовый сигал. XTAL1 имеет альтернативную функцию -PB6.
XTAL2	Этот вывод является выходом инвертирующего усилителя генератора. XTAL2 имеет альтернативную функцию – PB7.

ПРИМЕНЯЕМЫЕ УСЛОВНЫЕ ОБОЗНАЧЕНИЯ

БИС – большая интегральная схема,
МПС – микропроцессорная система,
МК – микроконтроллер,
ШД – шина данных,
УС – управляющий сигнал,
ППА – программируемый периферийный адаптер,

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Микропроцессорные системы [Текст]: учеб. пособие для вузов / Е.К. Александров, Р.И. Грушвицкий, М.С. Куприянов, О.Е. Мартынов и др.; под общ. ред. Д.В. Пузанкова. - СПб.: Политехника, 2012.

2. Петрухнова, Г.В. Архитектура и эволюция микропроцессоров [Текст] /Г.В. Петрухнова. - Воронеж, ВГТУ, 2011.

3. Белов, А.В. Самоучитель разработчика устройств на микроконтроллерах AVR [Текст] /А.В. Белов. - СПб: Наука и техника, 2008.

4. Естифеев, А.В. Микроконтроллеры семейств Tiny и Mega фирмы Atmel [Текст] /А.В. Естефеев. - М.: «Додэка - XXI», 2010.

СОДЕРЖАНИЕ

РАЗРАБОТКА МИКРОПРОЦЕССОРНЫХ СИСТЕМ НА ОСНОВЕ ОДНОКРИСТАЛЬНЫХ МИКРОКОНТРОЛЛЕРОВ СЕМЕЙСТВА ATMEGA

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
к выполнению лабораторных работ
по дисциплине «Микропроцессорные системы»
для студентов направления 09.03.01
«Информатика и вычислительная техника»
(профиль “Вычислительные машины, комплексы, системы и сети”) очной формы обучения

Составитель
Галина Викторовна Петрухнова

В авторской редакции

Подписано в печать .11.2015.
Формат 60x84/16. Бумага для множительных аппаратов.
Усл. печ. л. 3,3 Уч.-изд. л. 3,1. Тираж экз. "С"
Зак. №

ФГБОУ ВПО
«Воронежский государственный технический университет»
394026 Воронеж, Московский просп., 14