Государственное образовательное учреждение высшего профессионального образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

1

В.Н. Бориков

ИНТЕГРИРОВАННАЯ ОТЛАДОЧНАЯ СРЕДА РАЗРАБОТКИ ПРИЛОЖЕНИЙ ДЛЯ МИКРОКОНТРОЛЛЕРОВ СЕМЕЙСТВА AVR ФИРМЫ *ATMEL*

Методические указания для проведения цикла практических занятий по курсу «Микропроцессоры в измерительных устройствах» для студентов направлений «Приборостроение» и «Метрология, стандартизация и сертификация»

Томск 2010

УДК 681.2:681.3-181.48

Бориков В.Н. Интегрированная отладочная среда разработки приложений для микроконтроллеров семейства AVR фирмы *ATMEL*: Методические указания для проведения цикла практических занятий по курсу «Микропроцессоры в измерительных устройствах» для студентов направлений «Приборостроение» и «Стандартизация и метрология». – Томск: Изд.ТПУ, 2010. – 30с.

Методические указания рассмотрены и рекомендованы к изданию методическим семинаром кафедры компьютерных измерительных систем и метрологии 31.08.2010 г.

1 Общие сведения

AVR Studio 4 - это интегрированная среда разработки (IDE, Integrated Development Environment), которая очень удобна для отладки AVR-приложений в операционных системах Windows 9x/Me/NT/2000/XP/. Эта среда предлагает интерфейс программного симулятора (имитатора) и внутрисхемного эмулятора для восьмиразрядных микроконтроллеров AVR RISC. Кроме того, AVR Studio поддерживает набор разработчика STK500, позволяющий программировать AVR-устройства, а также новый встроенный эмулятор JTAG.

Эмуляторов не являются составной частью AVR-Studio, поэтому в методических указаниях рассматривается только программный симулятор.

После установки AVR Studio по умолчанию на Рабочем столе Windows появится соответствующий ярлык. В любом случае, эту среду можно запустить по команде из меню **Пуск ►Все программы►Atmel AVR Tools.** При первом запуске AVR Studio на экране появится окно приветствия (рисунок 1).

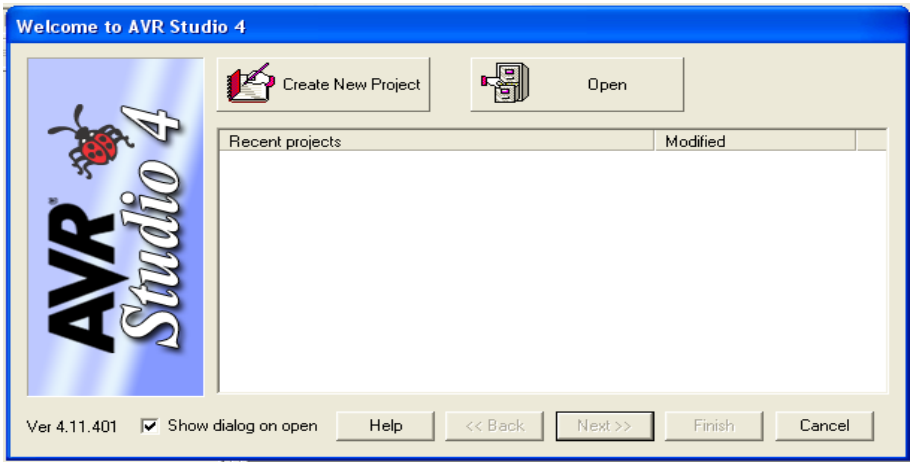

Рисунок 1 - Окно приветствия AVR Studio 4

С помощью этого окна можно выполнить одну из трех операций:

-создать новый проект "с нуля" - кнопка **Create New Project;**

-открыть какой-либо файл . cof или . hex с диска - кнопка **Open;**

-открыть один из проектов, которые использовались последними - для этого следует выбрать один из элементов в расположенном ниже списке.

Для примера, создадим новый проект. Для этого в окне приветствия нажмем кнопку **Create New Project**. В результате будет предложено выбрать платформу разработки и отладки программного обеспечения (рис. 2). В данном случае нас интересует интегрированный симулятор **AVR Simulator** и микроконтроллер

Atmega8.

Теперь можно нажать **Finish,** чтобы перейти к разработке и отладке программы.

Для того чтобы окно приветствия при последующих запусках AVR Studio не отображалось, в нем следует сбросить флажок **Show this dialog on open.** В таком случае все операции по созданию и загрузке проектов выполняются с помощью команд меню **File** и **Project,** а окно выбора эмулятора и типа микроконтроллеров, соответствующее рисунку 2, открывается по команде меню **Debug ► Select Platform and Device.**

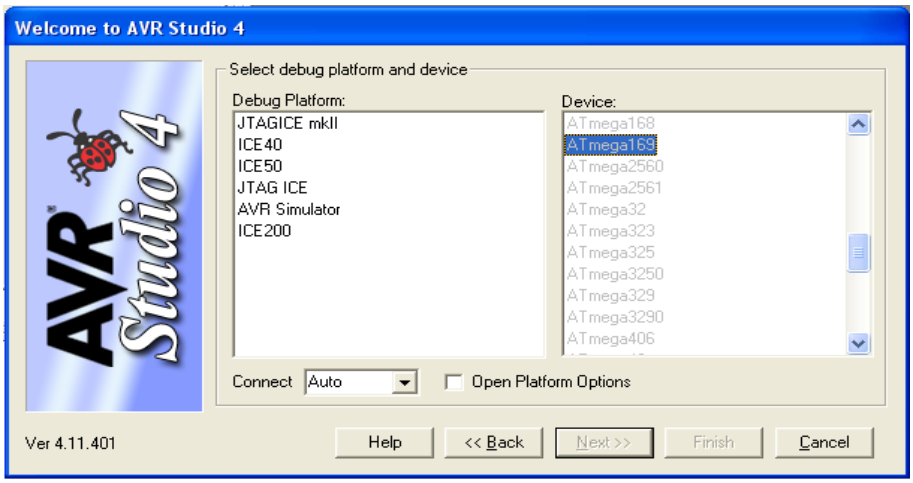

Рисунок 2 - Выбор эмулятора и типа микроконтроллеров

Главными окнами AVR Studio являются окно исходного кода и **Workspace** (рисунок 3).

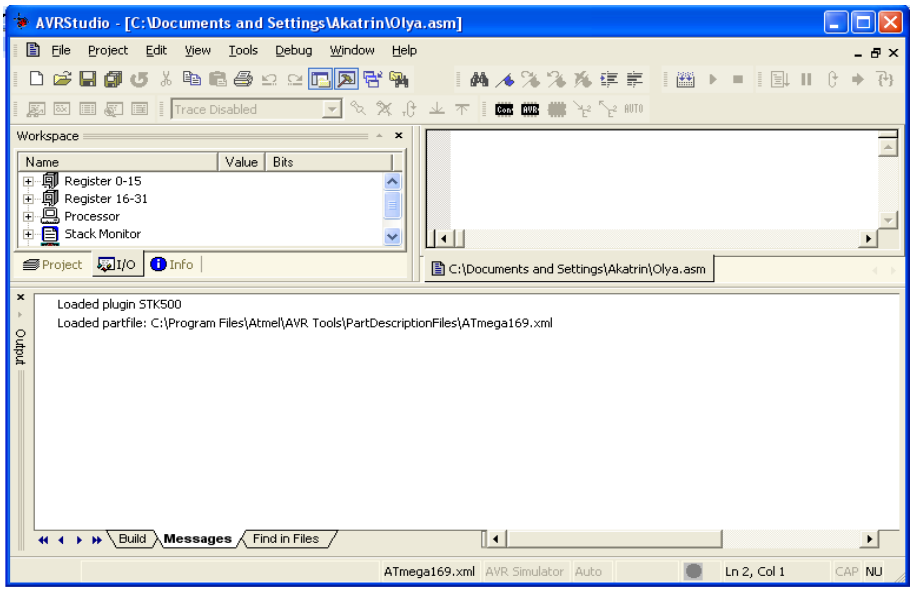

Рисунок 3 - Окно AVR Studio

В окне исходно кода строка, подлежащая исполнению первой, обозначена желтой стрелкой. Изменения следует вносить в среде Programmers Notepad, затем выполнять компиляцию и создание объектного файла, а затем - переключиться в AVR Studio. Если в AVR Studio был открыт тот же файл, то внешние изменения будут распознаны и на экране появится предложение автоматически обновить содержимое окна исходного кода. Эту же операцию можно выполнить и вручную с помощью кнопки панели инструментов **Reload Object File.**

2 Окно Workspace

Окно Workspace, расположенное на рисунке 3 слева от окна исходного кода предоставляет доступ к 32-м рабочим регистрам, к параметрам процессора (счетчик команд, указатель стека, регистры двойной длины X, Y и Z, частота генератора и др.), к значениям стека, а также портам ввода/вывода выбранного для симуляции или эмуляции микроконтроллера. К примеру, можно отредактировать содержимое какого-либо рабочего регистра как до, так и в процессе выполнения программы. Для этого следует выбрать требуемый элемент в окне **Workspace**, дважды щелкнуть мышью на значении в поле **Value** (рисунок 4) и ввести требуемое значение в диалоговом окне **Edit**.

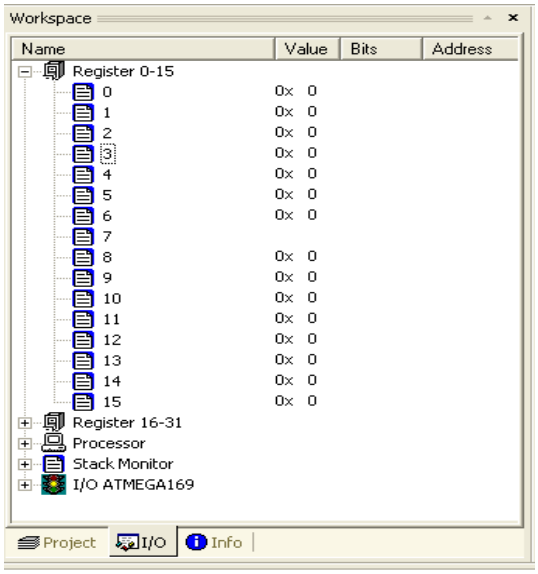

Рисунок 4 – Изменение значения в рабочем регистре 3

При этом значение можно вводить в шестнадцатеричном, десятичном, восьмеричном или двоичном формате, что выбирается с помощью

соответствующих переключателей.

Особый интерес на этапе отладки представляет собой ветвь параметров процессора (рисунок 5). Представленные здесь элемент позволяют просматривать и изменять содержимое программного счетчика, указателя стека, счетчика циклов во время выполнения программы, регистров двойной длины X, Y и Z, а также частоту кварцевого осциллятора и показания таймера отсчета времени.

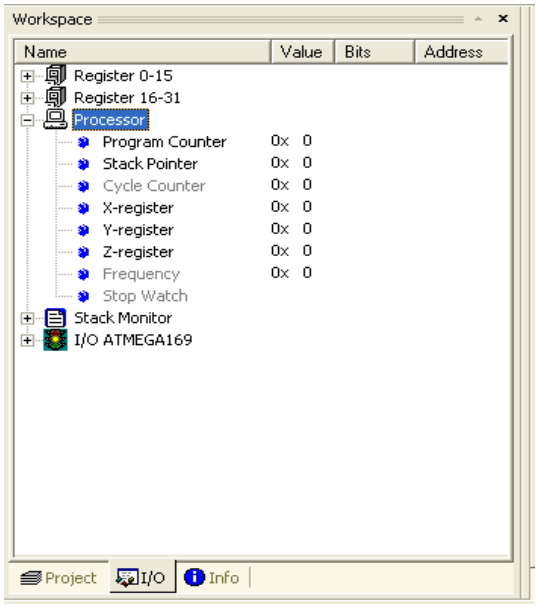

Рисунок 5 – Параметры процессора в окне Workspace

Значение счетчика команд можно изменить с помощью окна, представленного ранее на рис. 2. При этом откроется окно **Disassembler** с кодом программы на ассемблере и текущим будет выделена команда с соответствующим адресом (это же окно можно открыть в любой момент времени, выполнив команду меню **View ► Disassembler**).

Для сброса в нуль таймера отсчета времени следует дважды щелкнуть мышью на элементе **Stop Watch**. Подобный таймер удобно использовать для того, чтобы с точностью до сотых долей микросекунд определять, сколько времени ушло на выполнение некоторого фрагмента программы.

Самый нижний элемент в окне **Workspase** позволяет контролировать работу устройств ввода/вывода микроконтроллера, выбранного для эмуляции (рисунок 6).

К примеру, с помощью элемента **PORTD** мы можем увидеть программу в действии. Для этого достаточно запустить ее на выполнение командой меню **Debug ►Auto Step** (комбинация клавиш <Alt+F5>) или соответствующей кнопкой панели инструментов **Debug.**

Команда **Auto Step** отличается от команды **Run** (клавиша <F5>) тем, что в ходе выполнения программы после каждою шага обновляются окна AVR Studio (в случае команды **Run -** не обновляются).

Для того чтобы прервать выполнение программы, следует выполнить команду меню **Debug ► Break** (комбинация клавиш <Ctrl+F5>) или нажать соответствующую кнопку панели инструментов **Debug.**

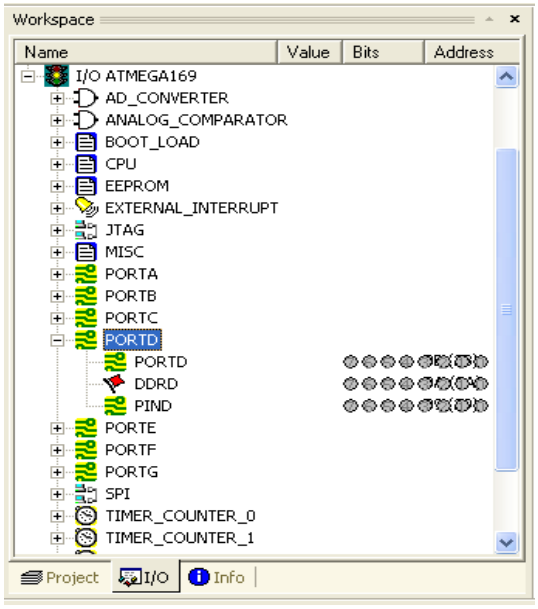

Рисунок 6 – Элементы контроля за работой устройства ввода/вывода микроконтроллера AVR

Поскольку симулятор выполняет программу в виртуальной среде, она работает значительно медленнее, чем в реальном микроконтроллере (особенно, выполнение функций задержки). Учитывайте это обстоятельство при отладке программ.

3 Окна Memory

Окно **Memory** позволяет пользователю при необходимости контролировать или изменять некоторую область памяти микроконтроллера AVR, например, память программ, память EEPROM, память данных, память ввода/вывода. Окно **Memory** находится в меню **View** (рисунок 7).

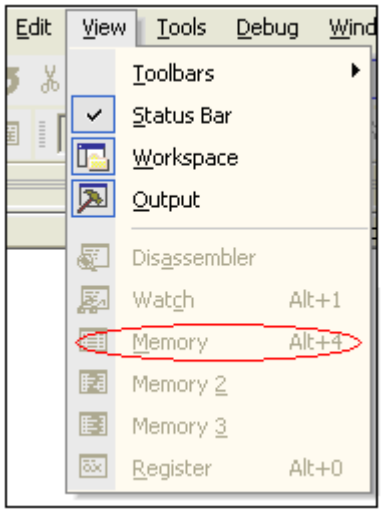

Рисунок 7

Одновременно могут быть открыты сразу несколько окон **Memory -** для этого используются команды меню (рисунок 8).

Тип области памяти выбирают с помощью раскрывающегося списка, расположенного в верхнем левом углу окна **Memory.** В примере на рисунке 8 для верхнего окна была выбрана память программ, а для нижнего - память данных.

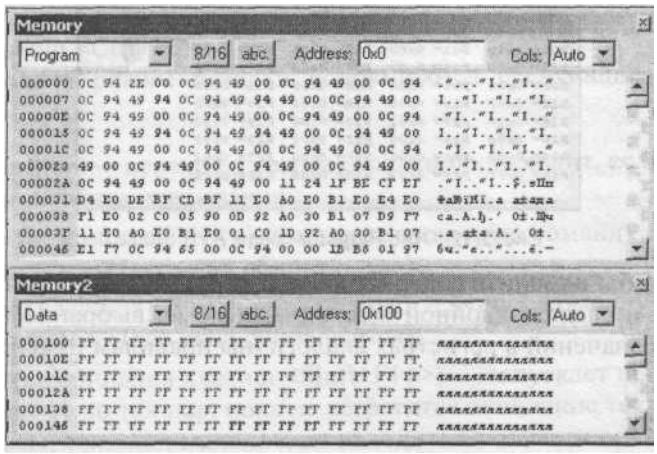

Рисунок 8 – Два окна Memory AVR-Studio

С помощью кнопки **8/16,** расположенной справа от раскрывающегося списка, а также команд контекстного меню **1 Byte** и **2 Byte** пользователь может переключаться между режимами разбивки дампа памяти на одно- и двухбайтные фрагменты, а с помощью кнопки **abc -** отобразить/скрыть дополнительную колонку с ASCII-значениями отображенной области памяти. Поле **Address** служит для отображения и ввода адреса ячейки памяти, на которой в данный момент установлен курсор. Количество колонок со значениями ячеек памяти выбирается с помощью раскрывающегося списка **Cols** (если в этом списке выбран элемент **Auto,**

то количество колонок выбирается автоматически в зависимости от ширины окна **Memory).**

Содержимое ячейки памяти в окне **Memory** можно легко изменить. Введенные значения расцениваются как шестнадцатеричные числа, при этом новое значение попадает в ячейку памяти сразу же после каждого нажатия клавиши. Если это нежелательно (например, в случае применения эмулятора при доступе на запись к регистру UDR приемопередатчика UART инициируется новая передача, хотя в этом случае байт еще неполный), то новое значение можно альтернативно вводить в специальном диалоговом окне, которое открывается по двойному щелчку мыши на соответствующей ячейке памяти. В этом случае вводимое значение записывается в ячейку только тогда, когда в окне **Edit** будет нажата кнопка ОК (клавиша <Enter>).

Значения, которые изменились с момента последней операции отладки, отображаются красным шрифтом. Адреса восьмиразрядных ячеек памяти и ASCII-символы выделены серым фоном, а адреса 16-разрядных ячеек памяти - голубым фоном.

4 Окно Register

Окно открывается по соответствующей команде меню **View** (или при нажатии комбинации клавиш <Alt+0>).

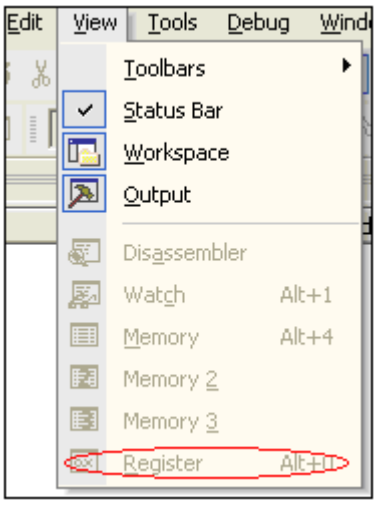

Рисунок 9

Окно **Register** показывает содержимое 32-х рабочих регистров микроконтроллера, которое обновляется после выполнения каждой команды (рисунок

10).

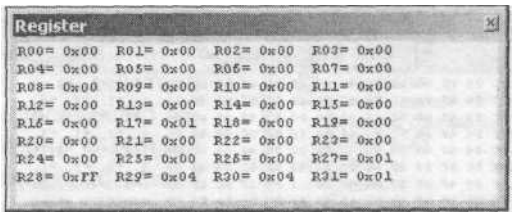

Рисунок 10 – Окно Register AVR Studio

Значения, изменившиеся в результате последней отладочной операции (например, при пошаговом проходе или в результате достижения точки останова) отображаются красным шрифтом.

Для того чтобы изменить содержимое регистра, следует остановить выполнение программы и сделать двойной щелчок мышью на выбранном регистре. Правила изменений значений в регистрах аналогичны правилам изменения значений в окнах **Memory**.

5 Окно Watch

Окно открывается по соответствующей команде меню **View** (или при нажатии комбинации клавиш <Alt+l>), рисунок 11.

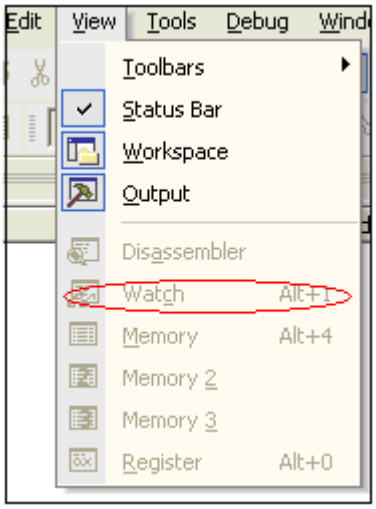

Рисунок 11

Окно **Watch** (рисунок 12) служит для отображения типов, значений и адресов таких объектов как, например, переменные в программе на С или на Ассемблере. Это окно состоит из четырех колонок. В первой указано имя объекта, за которым ведется наблюдение, во второй - тип объекта (Integer, unsigned Char и т.д.), в третьей - текущее значение, а в четвертой - адреса объектов. Пользователь может

добавлять новые объекты в окно Watch по команде контекстного меню Add Item или же по команде контекстного меню Add to Watch окна исходного кода. В последнем случае в список Watch будет добавлен элемент кода, на котором установлен курсор.

Элементы окна Watch можно удалять по одному с помощью команды контекстного меню Remove Selected Item или все сразу по команде контекстного меню Remove All Items.

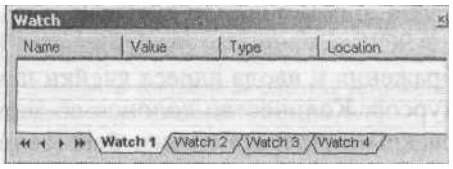

Рисунок 12 - ОкноWatch среды AVR Studio

6 Отладка программы

Под отладкой подразумевают пошаговое выполнение программы с контролем содержимого регистров микроконтроллера (проверка на низком уровне) и переменных (проверка на программном уровне). Для отладки программ в AVR Studio используют команды меню Debug и кнопки одноименной панели инструментов.

Прежде, чем рассмотреть эти команды имеет смысл разъяснить такое понятие как "точка прерывания". Точка прерывания (breakpoint) - это строка исходного кода, на которой работа программы приостанавливается. Таких точек (обозначаются коричневым кружком слева от строки) может быть установлено столько же, сколько эффективных строк в программе. Для установки/удаления точки прерывания в текущей строке служит команда меню **Debug ► Toggle Breakpoint** (клавиша <F9>) или соответствующая кнопка панели инструментов **Debug**. Для удаления всех расставленных в программе точек прерывания служит команда меню **Debug ► Remove Breakpoints** или кнопка **Clear all breakpoints** панели инструментов Debug. Для последовательного перехода от одной точки прерывания к другой используется команда меню **Debug ► Next Breakpoint** или комбинация клавиш <Ctrl+F9>.

Для перехода в режим отладки используются следующие команды меню

Debug (рисунок 13):

Рисунок 13 – Программы для перехода в режим отладки

1 Run, Auto Step - переход в режим отладки происходит, если встречается точка прерывания;

2 Step Into (клавиша <F 11>) - выполняет текущую команду с заходом в подпрограммы (все окна обновляются); I

3 Step Over (клавиша <F10>) - выполняет текущую команду без захода в подпрограммы (все окна обновляются);

4 Step Out (комбинация клавиш <Shift+Fl 1>) - запускает программу и выполняет ее до тех пор, пока не встретится окончание текущей подпрограммы; если ход выполнения находится в области основной программы, то программа будет выполняться до тех пор, пока не будет остановлена пользователем командой **Break** или не встретит точку прерывания;

5 Run To Cursor (комбинация клавиш <Ctrl+F10>) - запускает программу, которая выполняется до тех пор, пока не будет достигнута позиция курсора в окне исходного кода; если встречается точка останова, то выполнение программы не останавливается; если позиция курсора не достигается никогда, то программа выполняется до тех пор, пока не будет остановлена командой **Break.** После выполнения команды все окна обновляются.

7 Настройка параметров имитатора

Для того чтобы выбрать имитацию работы конкретного микроконтроллера, а также его рабочую частоту, служит диалоговое окно **Simulator Options** (рисунок 14), которое открывается по команде меню **Debug ► AVR Simulator Options** (комбинация клавиш <Alt+O>).

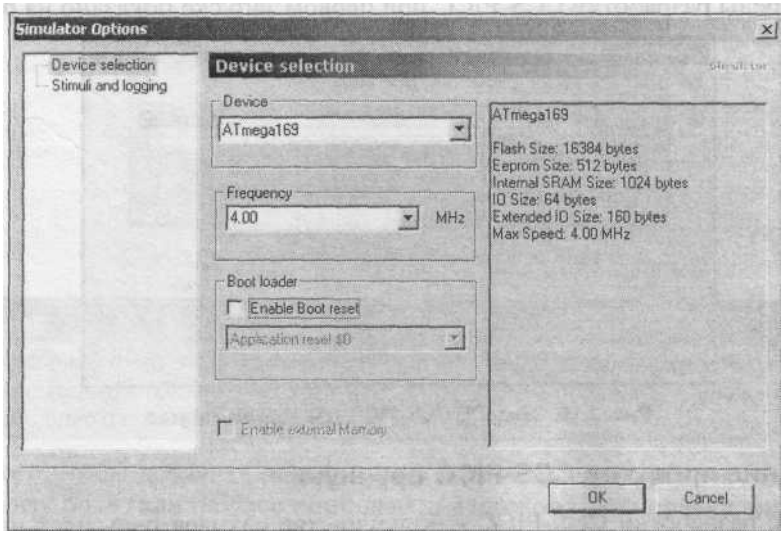

Рисунок 14 - Диалоговое окно Simulator Options

Допустимые параметры выбранного микроконтроллера отображаются в текстовом поле, расположенном справа.

Лабораторная работа №1

1 Цель работы:

- изучение AVR Studio;
- изучение принципов написания программ на языке ассемблера;
- изучение простейших команд ATmega8.

2 Программа выполнения работы.

2.1Самостоятельно ознакомиться с методичкой по работе с AVR Studio;

2.2 Самостоятельно найти стандарты по оформлению алгоритмов программного обеспечения;

2.3 Составить алгоритм программы;

2.2 Написать программу на языке ассемблера ATmega8;

2.3 Произвести отладку программы в AVR Studio;

2.4 Самостоятельно проверить выполнение программы;

2.5 Составить отчет по лабораторной работе (СТО 2.03.05-2006).

3 Задание:

Написать программу с использованием AVR Studio и проверить по шагам ее работу.

```
        ldi     R16,55      ; Непосредственная загрузка числа
        ldi     R17,66      ; Непосредственная загрузка числа
k:      nop                 ; Нет операций
        mov     R0,R17      ; Персылка данных из регистра R17 в R0
        cp      R16,R0      ;Сравнение двух регистров R16-R0
        brlo    stop        ;Переход на метку stop если меньше
        brsh    k           ;Преход на метку k если больше или равно
stop:   rjmp    stop        ; Безусловный переход на метку k
```

Лабораторная работа №2

ПРОГРАМИРОВАНИЕ ПРОСТЕЙШИХ ПРОГРАММ

1 Цель работы:

- изучение команд ветвления микроконтроллера ATmega8;

- организация циклов;

- овторение знаний по программированию в отладочной системе AVR Studio;

Изучение методов сортировки двоичных сигналов;

2 Программа выполнения работы.

2.1 Для выполнения работы ознакомиться самостоятельно с методами сортировки чисел;

2.2 Написать программу на языке ассемблера ATmega8;

2.3 Произвести отладку программы в AVR Studio;

2.4Самостоятельно проверить выполнение программы;

2.5 Составить отчет по лабораторной работе;

3 Задание:

Занести в регистры R0, R1, R2, R3, R4, R5 микроконтроллера ATmega8 двоичные числа. Произвести сортировку массива, находящегося в регистрах R0-R5 по убыванию или по возрастанию.

Лабораторная работа №2*

ПРОГРАМИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ ПОДПРОГРАММ

1 Цель работы:

- изучение подпрограмм микроконтроллера ATmega8;
- изучение косвенной адресации через регистр Z
- повторение организации циклов;
- повторение знаний по программированию в отладочной системе AVR Studio;
- изучение методов сортировки двоичных сигналов;

2 Программа выполнения работы.

2.1 Для выполнения работы воспользоваться знакомым из курса ИИС «пузырьковым» методом сортировки чисел;

2.2 Написать программу на языке ассемблера ATmega8 нахождения минимального или максимального члена массива;

2.3 Написать программу сортировки массива используя подпрограмму нахождения максимального или минимального элемента массива;

2.4 Произвести отладку программы в AVR Studio;

2.5 Самостоятельно проверить выполнение программы;

2.6 Сдать программу преподавателю;

2.7 Составить отчет по лабораторной работе;

3 Задание:

Занести в регистры R0, R1, R2, R3, R4, R5 микроконтроллера AT901200 двоичные числа. Произвести сортировку массива, находящегося в регистрах R0-R5 по убыванию или по возрастанию с использованием подпрограммы нахождения минимального или максимального члена массива.

Лабораторная работа №3

ИЗУЧЕНИЕ МАТЕМАТИЧЕСКИХ И ЛОГИЧЕСКИХ ОПЕРАЦИЙ И ВЛИЯНИЕ ИХ НА ФЛАГИ МИКРОКОНТРОЛЛЕРА

1 Цель работы:

- изучение математических команд ассемблера микроконтроллера AVR;
- изучение логических команд ассемблера микроконтроллера AVR;

2 Программа выполнения работы.

2.1 Провести исследование с помощью AVR Studio выполнение команд ADD, ADC, SUB, SBC, SUBI, SBCI, ADIW, SBIW.

2.2 С помощью окна состояния процессора AVR Studio исследовать влияние команд на состояние флагов; Обратить внимание на сложение и вычитание чисел $09+$01; $09+$07, $10-$01; $10-$07.

2.3 С помощюь AVR Studio исследовать выполнение команд ANDI, ORI, LSL, LSR;

2.4Результаты исследований занести в отчет.

2.5 На основе исследований составить алгоритм сложения или вычитания двух десятичных чисел микроконтроллера AVR ATmega8;

2.6 Написать программу на языке ассемблера и отладить ее для различных чисел, результат суммирования или вычитания которых не превышал числа +/- 99.

2.7 Сдать программу преподавателю;

2.8 Составить отчет по лабораторной работе.

3 Задание:

Занести в регистр R0 микроконтроллера AVR ATmega8 первый разряд одного десятичного числа (единицы). Занести в регистр R1 микроконтроллера AVR ATmega8 второй разряд этого десятичного числа (десятки). Занести в регистр R2 микроконтроллера AVR ATmega8 первый разряд второго десятичного числа (единицы). Занести в регистр R3 микроконтроллера AVR ATmega8 второй разряд этого второго десятичного числа (десятки). Произвести математическое сложение или вычитание двух десятичных чисел и представить результат

вычисления в десятичной форме.

Пример:

R0=07

R1=03

R2=09

R3=01

R4 =56

37+19=56

Лабораторная работа №3*

КОНТОЛЬНОЕ ЗАДАНИЕ ДЛЯ ПРОМЕЖУТОЧНОЙ ПРОВЕРКИ ЗНАНИЙ ПРОСТЕЙШИХ КОМАНД ЯЗЫКА АССЕМБЛЕРА МИКРОКОНТРОЛЛЕРА AVR

1 Цель работы:

-контролирование знаний логических команд ассемблера микроконртоллера AVR;

- контролирование знаний простейших операций;

- контролирование знаний составления алгоритмов;

-контролирование знаний команд работы с флагами микроконтроллера;

- контролирование знаний работы с AVR Studio-

2 Программа выполнения работы.

2.1 Составить алгоритм выполнения программы микроконтроллера AVR 90S1200;

2.2 Написать программу на языке ассемблера и отладить ее для различных чисел.

2.3 Сдать программу преподавателю;

2.4 Составить отчет по лабораторной работе.

3 Задание (кроме А, Б, З):

Занести в регистры R0, R1, R2, R3, R4, R5 микроконтроллера AVR ATmega8 массив данных. Выполнить одну из задач:

А) Нахождение минимального члена массива;

Б) Нахождение максимального члена массива;

В) Преобразование шестнадцатиричного числа в двоично-десятичное;

Г) Преобразование двоично-десятичного числа в шестнадцатиричное.

Д) Нахождение в массиве одинаковых чисел и подсчет их количества.

Е) Обнаружение в массиве нуля и подсчет их количества;

Ж) Вычитание двоично-десятичных чисел;

З) Сложение двоично-десятичных чисел;

И) Нахождение количества 0 в байте;

К) Нахождение количества 1 в байте;

Лабораторная работа №4

ПРОГРАМИРОВАНИЕ ПАРАЛЛЕЛЬНОГО ПОРТА ВВОДА-ВЫВОДА МИКРОКОНТРОЛЛЕРА AVR

1 Цель работы:

- контролирование знаний по архитектуре параллельного порта ввода-вывода микроконтроллера AVR;

- контролирование знаний программирования порта ввода-вывода микроконртоллера AVR;

- изучение команд работы с портами ввода-вывода микроконтроллера AVR;

- контролирование знаний составления алгоритмов;

- контролирование знаний работы с AVR Studio;

- изучение работы программатора.

2 Программа выполнения работы:

2.1Изучить принципиальную схему ввода-вывода на микроконтроллере AVR ATmega8;

2.2Составить алгоритм выполнения программы микроконтроллера AVR ATmega8;

2.3Написать программу на языке ассемблера и отладить ее для различных чисел.

2.4 Сдать программу преподавателю;

2.5 Составить отчет по лабораторной работе.

3 Задание:

Написать программу ввода-вывода двоичной информации. Проверить ее работу с использованием отладчика и на устройстве ввода-вывода. Выполнить одну из индивидуальных задач:

1) Отображение состояния порта Д в порт В;

2) Отображение состояния порта Д в порт В, если нажата кнопка Д1;

3) Вывод числа 0x55, принажатии кнопки Д1, в противном случае 0xAA;

4) Зажечь все диоды, если на входе присутствует комбинация

переключателей 0b0010;

5) Вывод в порт B числа 0x101010, если присутствует комбинация переключателей 0x0101;

6) Вывод в порт B числа 0x101010, если присутствует комбинация переключателей 0x0101 и нажата кнопка D1;

7) Получить нормальное 4 битовое число в порту B, соответствующее комбинации кнопок Д;

8) Генератор случайных чисел по нажатию кнопки Д.

Усложненные задания (разрешается не описывать программатор и устройство):

8) Зажечь любой двоичный код на светодиодах. Изменить код в сторону увеличения при нажатии кнопки Д1;

9) Зажечь любой двоичный код на светодиодах. Изменить код в сторону уменьшения при нажатии кнопки Д1;

10) Зажечь крайний светодиод на индикаторе и произвести зажигание соседнего светодиода при нажатии кнопки Д1 справа налево;

11) Зажечь крайний светодиод на индикаторе и произвести зажигание соседнего светодиода при нажатии кнопки Д1 слева направо;

12) Зажечь светодиоды в соответствии с комбинацией кнопок Д и при нажатии кнопки Д! Произвести сдвиг двоичного числа влево;

13) Зажечь светодиоды в соответствии с комбинацией кнопок Д и при нажатии кнопки Д! Произвести сдвиг двоичного числа вправо;

14) Зажечь крайние светодиоды и при нажатии кнопки Д1 производить попарный сдвиг в стороны сближения и удаления.

Лабораторная работа №5

ПРОГРАМИРОВАНИЯ УСТРОЙСТВА ФОРМИРОВАНИЯ ВРЕМЕННЫХ ЗАДЕРЖЕК НА МИКРОКОНТРОЛЛЕРЕ AVR

Цель работы:

- контролирование знаний по архитектуре микроконтроллера AVR;
- контролирование знаний по архитектуре таймера микроконтроллера AVR;
- контролирование знаний программирования таймера микроконртоллера AVR;
- изучение команд работы с таймером микроконтроллера AVR;
- повторение знаний программирования параллельного порта ввода-вывода;
- контролирование знаний составления алгоритмов;
- контролирование знаний работы с AVR Studio;
- контролирование знаний по работе с программатором.

2 Программа выполнения работы:

2.1 Изучить принципиальную схему устройства на микроконтроллере AVR ATmega8;

2.2 Составить алгоритм выполнения программы микроконтроллера AVR ATmega8;

2.3 Написать программу на языке ассемблера и отладить ее для различных задач.

2.4Сдать программу преподавателю;

2.5Составить отчет по лабораторной работе.

3 Задание:

Написать программу формирования временных задержек. Проверить ее работу с использованием отладчика и на устройстве ввода-вывода с микроконтроллером AVR. Выполнить одну из индивидуальных задач:

1) Секундомер на основе программной реализации задержки с отображением информации на светодиодном индикаторе;

2) Секундомер на основе аппаратной реализации задержки с отображением информации на светодиодном индикаторе;

3) Генератор прямоугольных импульсов с частотой генерации 1 кГц и скважностью 2;

4) Генератор прямоугольных импульсов с частотой генерации 2 кГц и скважностью 4;

5) Синтезатор музыкальной мелодии «сирена»;

6) Устройство отсчеты 10 минут;

7) Часы с переключением режима часы, минуты, секунды.

Лабораторная работа №6

КОНТРОЛЬНАЯ РАБОТА ПО РАЗРАБОТКЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ СЕКУНДОМЕРА

1 Цель работы:

- контролирование знаний по микроконтроллеру AVR;
- контролирование знаний по ассемблеру микроконтроллера AVR;
- контролирование знаний составления алгоритмов;
- контролирование знаний работы с AVR Studio;
- контролирование знаний по работе с программатором.

2 Программа выполнения работы:

2.1 Разработать алгоритм выполнения программы микроконтроллера AVR ATmega8;

2.2 Написать программу на языке ассемблера и отладить ее для различных задач.

2.3 Сдать программу преподавателю;

2.4 Составить отчет по лабораторной работе.

3 Задание:

Написать программу секундомера. Информацию выводить на светодиодную матрицу. Разработать однозначный интерфейс вывода информации о секундах. Запуск и останов секундомера производить путем нажатия кнопки D1.

Лабораторная работа №7

ИЗУЧЕНИЕ ПРЕРЫВАНИЙ МИКРОКОНТРОЛЛЕРА AVR Studio

1 Цель работы:

- контролирование знаний по архитектуре микроконтроллера AVR;
- контролирование знаний по архитектуре таймера микроконтроллера AVR;
- контролирование знаний программирования таймера микроконртоллера AVR;
- изучение команд работы с таймером микроконтроллера AVR;
- контролирование знаний по архитектуре сторожевого таймера микроконтроллера AVR;
- контролирование знаний программирования сторожевого таймера микроконртоллера AVR;
- изучение команд работы со сторожевым таймером микроконтроллера AVR;
- контролирование знаний по архитектуре аналогового компаратора микроконтроллера AVR;
- контролирование знаний программирования аналогового компаратора микроконртоллера AVR;
- изучение команд работы с аналоговым компаратором микроконтроллера AVR;
- контролирование знаний по архитектуре внешнего прерывания микроконтроллера AVR;
- контролирование знаний программирования внешнего прерывания микроконртоллера AVR;
- изучение команд работы с внешним прерыванием микроконтроллера AVR;
- повторение знаний программирования параллельного порта ввода-вывода;
- контролирование знаний составления алгоритмов;
- контролирование знаний работы с AVR Studio;
- контролирование знаний по работе с программатором.

2 Программа выполнения работы:

2.1 Изучить принципиальную схему устройства на микроконтроллере AVR ATmega8;

2.2 Составить алгоритм выполнения программы микроконтроллера AVR ATmega8;

2.3 Написать программу на языке ассемблера и отладить ее для различных задач.

2.4 Сдать программу преподавателю;

2.5 Составить отчет по лабораторной работе.

3 Задание:

Написать программу обработки одного из 4-х прерываний микроконтроллера: таймер, внешнее прерывание, сторожевой таймер и компаратор. Выбор вида прерывания осуществляет преподаватель, а задание, доказывающее получение знаний по обработке прерываний, формирует студент. Одно из требований к программе – визуальное доказательство возникновения и обработки прерывания (светодиодные индикаторы и переключатели).

Примеры индивидуальных задач:

1) Секундомер на основе аппаратной реализации задержки с отображением информации на светодиодном индикаторе;

2) Генератор прямоугольных импульсов с частотой генерации 1 кГц и скважностью 2;

3) Генератор прямоугольных импульсов с частотой генерации 2 кГц и скважностью 4;

4) Устройство отсчета 10 минут;

5) Часы с переключением режима часы, минуты, секунды.

6) Сброс микроконтроллера по прошествии 2-х секунд.

7) Сброс процессора при не нажатии кнопки в течении определенного времени;

8) Сброс процессора при выполнении определенных условий от комбинации кнопок.

9) Сброс процессора при нажатии кнопки D1;

10) Визуальное отображение информации об уменьшении напряжения;

11) Визуальное отображение информации об увеличении напряжения;

12) Визуальное отображение информации об изменении напряжения;

13) Контроль напряжения питания в пределах 10%;

14) Визуальное отображение низкого уровня на входе прерывания;

15) Визуальное отображение высокого уровня на входе прерывания;

16) Визуальное отображение изменения информации на входе прерывания;

17) Запуск секундомера по внешнему прерыванию;

18) Подсчет количества импульсов на входе прерывания.

ИНТЕГРИРОВАННАЯ ОТЛАДОЧНАЯ СРЕДА РАЗРАБОТКИ ПРИЛОЖЕНИЙ ДЛЯ МИКРОКОНТРОЛЛЕРОВ СЕМЕЙСТВА AVR ФИРМЫ *ATMEL*

Методические указания

Составитель: Валерий Николаевич Бориков