Министерство образования и науки Российской Федерации

Севастопольский государственный университет

Институт информационных технологий и управления в технических системах

Кафедра ИС

ОТЧЕТ

по лабораторной работе №1

Исследование архитектуры однокристальной микро-ЭВМ

Выполнил:

ст. гр. ИС/б-21-о

Куркчи А. Э.

Проверил:

Кудрявченко И. В.

Севастополь

2016

1. ЦЕЛЬ РАБОТЫ

Изучить архитектуру однокристальной микро-ЭВМ (микроконтроллера) и исследовать работу функциональных узлов устройства при выполнении машинных команд различного типа. Приобрести практические навыки составления простейших программ на языке Ассемблера и отладки их в специализированной инструментальной среде.

1. ПОСТАНОВКА ЗАДАЧИ
   1. Изучить структуру и назначение функциональных блоков микроконтроллера.
   2. Ознакомиться с особенностями системы команд микроконтроллеров типа *AVR*.
   3. Ознакомиться со средой программирования и отладки программ типа *AVR* *Studio* *4* (или 5).
   4. Подготовить в редакторе *AVR Studio* программу на ассемблере согласно варианту задания (Приложение А).
   5. Записать в комментариях значение каждой команды.
   6. Выполнить ассемблирование программы.
   7. Запустить отладчик программы и исправить, при их наличии, синтаксические ошибки.
   8. Исследовать изменение содержимых рабочих регистров, указателя стека, флагов и ячеек памяти при пошаговом выполнении программы.
   9. Оформить отчет по лабораторной работе.
2. ОПИСАНИЕ УСТАНОВКИ

Используемый МК *ATmega16* имеет следующие характеристики:

* Тактовая частота – 16 МГц;
* 32 восьмиразрядных рабочих регистра общего назначения;
* 130 команд;
* 16 Кбайт внутрисистемная программируемая *Flash*-память;
* Встроенный сторожевой таймер, тактовый генератор;
* Питание 2.7 – 5.5 В.

1. КОД ПРОГРАММЫ

.include "m16def.inc"

.def temp**=**r16 ; Ассоциировать temp с регистром r16

.def count**=**r17 ; Ассоциировать count с регистром r17

rjmp init ; Перейти к метке init

init**:** ; Метка init

ldi temp**,**$80 ; Записать в temp значение 0x80

**out** SPL**,**temp ; Установить указатель стека в значение temp

ldi count**,**5 ; Записать в count 5

m1**:** ; Метка m1

**inc** r18 ; Инкрементировать значение регистра r18

**inc** r19 ; Инкрементировать значение регистра r19

**add** r18**,**r19 ; Добавить к регистру r18 значение регистра r19

**mov** r20**,**r18 ; Копирование регистра r18 в регистр r20

**push** r20 ; Записать значение регистра r20 в стек

**pop** r21 ; Извлечь из стека значение в регистр r21

subi r20**,**2 ; Вычесть из значения регистра r20 2

**dec** count ; Декрементировать count

brne m1 ; Если не ноль, перейти по метке m1

**ret** ; Возврат из подпрограммы

1. ОПИСАНИЕ РАБОТЫ ПРОГРАММЫ

В соответствии с заданием программа устанавливает указатель стека на адрес *0x80* и начинает выполнять цикл, установив количество итераций в 5. В каждой итерации инкрементируются регистры *r18* и *r19*, к регистру *r18* добавляется значение регистра *r19*. Значение регистра *r18* записывается в регистр *r20*, который записывается в стек. Значение из стека записывается в регистр *r21*, а от регистра *r20* отнимается число 2. Уменьшается количество оставшихся итераций и, если оно не достигло нуля осуществляется переход к началу следующей итерации, иначе происходит выход из подпрограммы. Пошаговое выполнение программы представлено на рисунках 1-7.

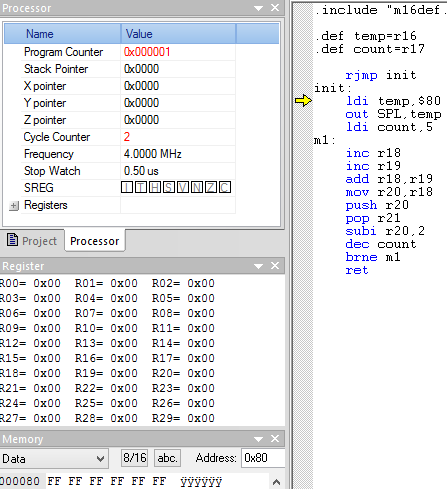
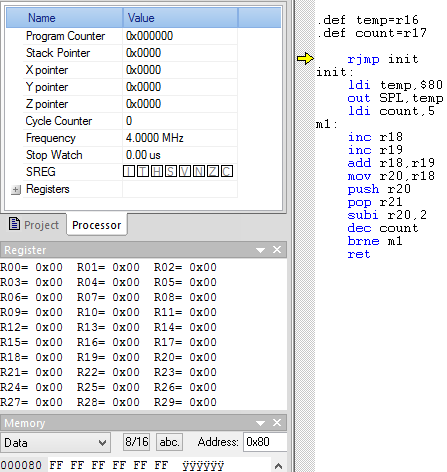


Рисунок 1 – Начальное состояние процессора и выполнение команды *rjmp*

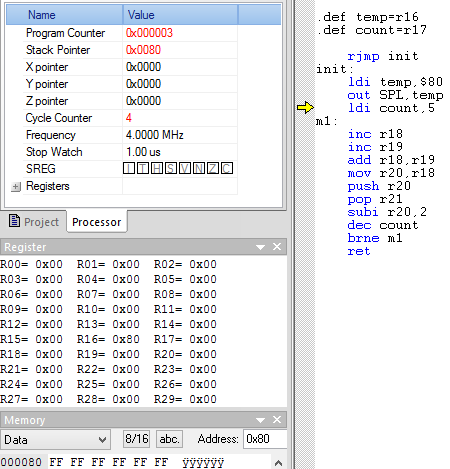
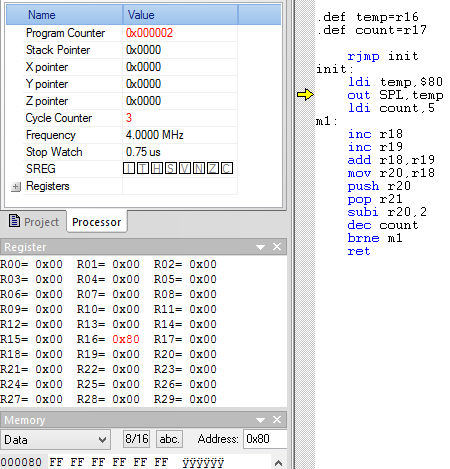


Рисунок 2 – Выполнение инструкций *ldi* и *out*

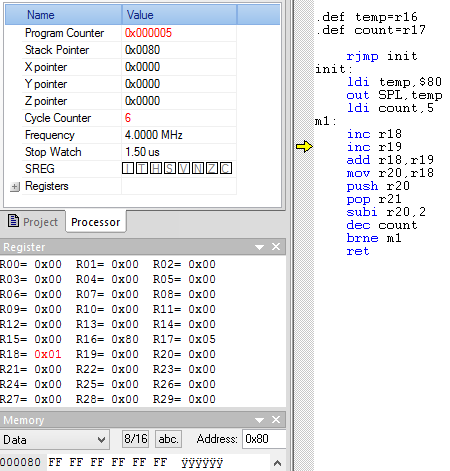
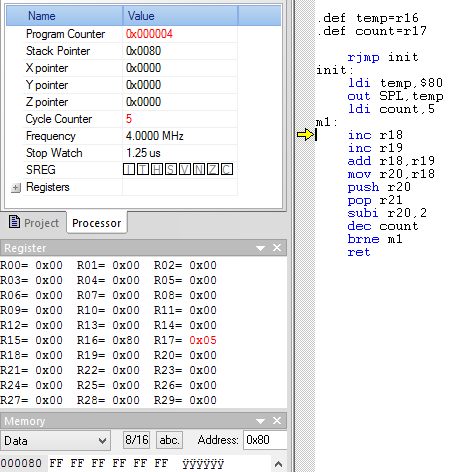


Рисунок 3 – Выполнение инструкций *ldi* и *inc*

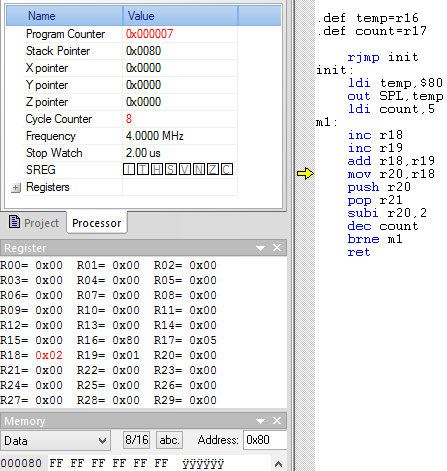
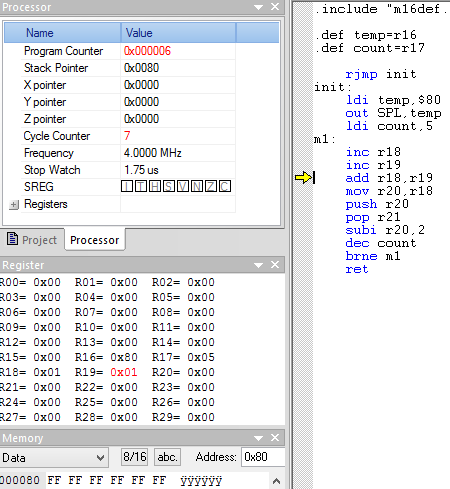


Рисунок 4 – Выполнение инструкций *inc* и *add*

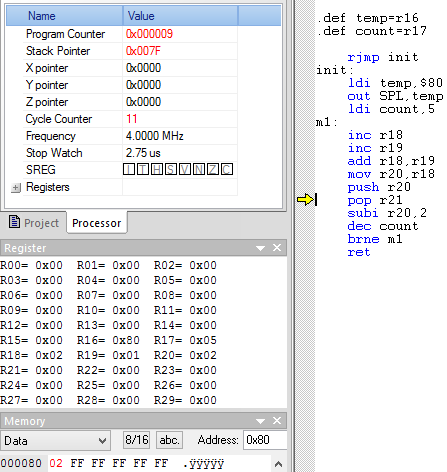
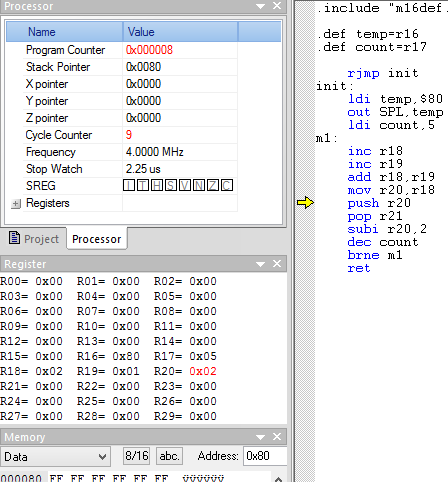


Рисунок 5 – Выполнение инструкций *mov* и *push*

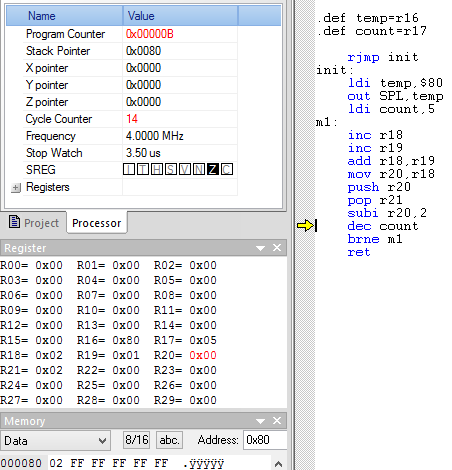
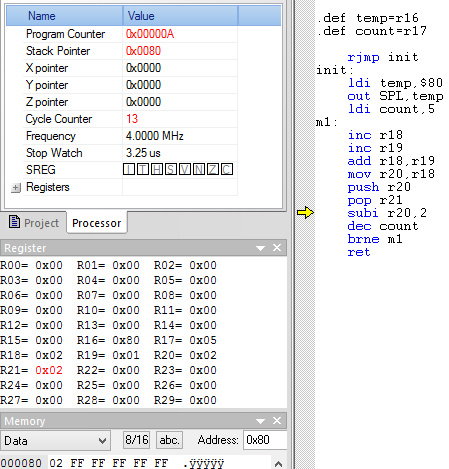


Рисунок 6 – Выполнение инструкций *pop* и *subi*

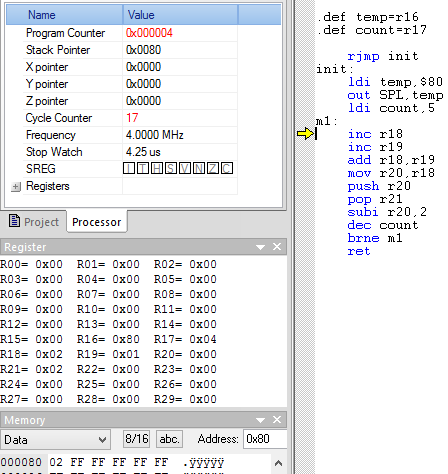
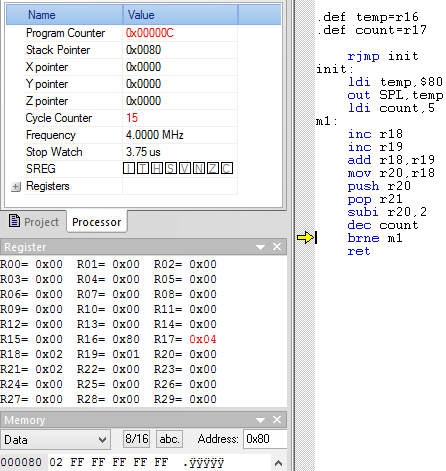


Рисунок 7 – Выполнение инструкций *dec* и *brne*

Программа выполняется за 64 цикла процессора. В результате регистры *r18* и *r19* соответственно равны *0x14* и *0x05*, а регистры *r20* и *r21* равны *0x12* и *0x14*. В стеке по адресу *0x80* хранится значение *0x14*, а флаг нуля установлен в 1. Результат выполнения представлен на рисунке 8.

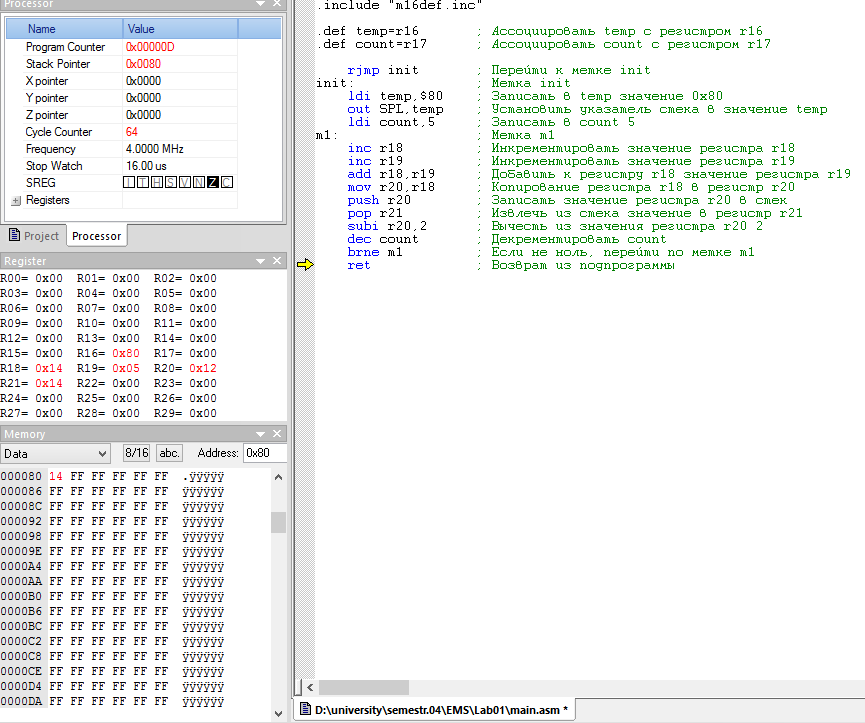


Рисунок 8 – результат выполнения программы

ВЫВОДЫ

В ходе выполнения данной лабораторной работы была исследована архитектура однокристальной микро-ЭВМ (микроконтроллера) и исследована работа функциональных узлов устройства при выполнении машинных команд различного типа. Приобретены практические навыки составления простейших программ на языке Ассемблера и отладки их в специализированной инструментальной среде.