Министерство образования и науки Российской Федерации

Санкт-Петербургский политехнический университет Петра Великого

—

**Институт** **кибербезопасности и защиты информации**

Отчёт

по лабораторной работе №1

**Основы программирования МК ATmega32**

**Вариант 9(а)**

по дисциплине «Аппаратные средства вычислительной техники»

|  |  |  |  |
| --- | --- | --- | --- |
| Выполнили: | студенты группы 4851003/10002 |  | Билан Н. С. |
|  |  | (подпись, дата) |  |
|  |  |  |  |
|  |  |  |  |
|  |  |
|  |  |
| Проверил: | старший преподаватель |  | А.С. Макаров |
|  |  | (подпись, дата) |  |

# Формулировка задания

1. В базовом примере заменить фрагмент кода, выделенный зелёным, на фрагмент из своего варианта.
2. Создать новый проект в Atmel Studio на языке ассемблера, разместить исходный код из п.1 в файле проекта. Сформировать «.lss»-файл и «.hex»-файл. Записать «.hex»-файл в контроллер с помощью программы AVRFlash.
3. Изучить архитектуру и систему команд микроконтроллера ATmega32. Разобраться в алгоритме работы текущей программы. Убедиться в правильности работы программы.
4. Определить зависимость количества тактов, за которое выполняется заменённый блок кода, от констант x и y.
5. Вычислить значения констант x и y при которых блок кода «delay» будет выполняться ровно 0,1 секунды (при тактовой частоте 8 МГц). При невозможности обеспечения точной величины задержки необходимо дополнить блок кода соответствующим количеством команд NOP (на месте закомментированной команды NOP в базовом примере).
6. Изучить сформированный «.lss»-файл, выписать адреса всех меток программы, перечислить используемые форматы команд в части состава и размерности операндов.
7. Изучить структуру сформированного «.hex»-файла, определить количество записей в файле и количество машинных слов программы.
8. Взять команду ассемблера в соответствии с вариантом и описать порядок её выполнения внутри центрального процессора: определить этапы командного цикла, задействованные узлы МК, состав пересылаемых данных и управляющих сигналов.

# Схема лабораторной установки

Изображение выглядит как текст, внутренний

Автоматически созданное описание

# Блок-схема алгоритма работы программы



# Временные диаграммы логических сигналов на портах МК (фрагмент)

По оси Y указаны номера светодиодов, по оси X – время. Когда график находится на высоте светодиода, он горит.

# Результаты работы

### Определение зависимости количества тактов от констант x и y

LDI, INC, DEC, NOP – 1 такт;

RET – 4 такта;

BRNE – 2(1) такта;

Количество тактов N = 6 + 5x + 1282(256 - y)

Чтобы время выполнения было 0,01 с, нужно сделать так:

X = 102

Y = 194

### Анализ lss-файла

Адреса меток:

delay – 000002

delay\_sub - 000004

reset – 00000d

loop – 00001c

Используемые форматы команд:

1. **Пересылка:**

OUT:  bb4a, bb47, bb44, bb41, bf4e, bf4d, ba0b, ba18, ba25, ba32

LDI: e6d6, ece2, e041, e048, e54f

MOV: 2e04, 2e14, 2e24, 2e34

1. **Арифметические:**

INC: 95e3.

DEC: 95da.

1. **Битовые:**

SER: ef4f.

CLR: 2744.

1. **Сдвиг:**

LSR: 9436.

ROR: 9427, 9417, 9407.

1. **Изменение признаков:**

BST: fa00.

BLD: f837.

1. **Условные переходы:**

BRNE: f7e1, f7d1.

1. **Безусловные переходы:**

JMP: 940c

1. **Вызов подпрограммы:**

RET: 9508.

### Анализ hex-файла

:020000020000**FC**

:100000000C940D00D6E6E2EC0000DA950000E1F7**72**

:10001000E395D1F700000000089541E0042E4427**45**

:10002000142E242E342E4FEF4ABB47BB44BB41BB**9A**

:1000300048E04EBF4FE54DBF00FA369427941794**21**

:10004000079437F80BBA18BA25BA32BA0E940200**E0**

:02005000F3CF**EC**

:00000001**FF**

10 – количество байт данных в записи;

0010– адрес, с которого начинают размещаться данные этой записи;

00 – тип записи: 00 – запись, содержащая данные, 01 – запись, сигнализирующая о конце файла, 02 – запись адреса расширенного сегмента;

E395D1F700000000089541E0042E4427– Данные для загрузки в память;

**45** – контрольная сумма (рассчитывается так чтобы сумма всех байтов в записи была равна 0);

0000– расширенный сегментрый адрес, USBA.

Количество записей: 8

Количество машинных слов: 42

### Алгоритм выполнения команды ассемблера

Команда ассемблера: Команда: CLR R7 (0010.0100.0007.0007) (0010.01rd.rrrr.dddd) ((i) Rd 🡨 Rd E Rd)

Команда CLR производит очистку регистра.

rrrrr, ddddd – номер регистра, дублируется

Командный цикл:

1. Установка расположения команды на шину адреса

2. Извлечение команды из ПЗУП и запись в регистр команды

3. Декодирование команды – определение типа операции и формата

4. Извлечение адреса операнда (номер регистра общего назначения (РОН))

5. Очистка регистра

6. Увеличение значения счётчика команд

# Ответы на контрольные вопросы

1. Укажите, в чём проявляются признаки RISC-архитектуры в микроконтроллере ATmega32. В чём преимущества и недостатки приведённых особенностей?

RISC — [архитектура процессора](https://ru.wikipedia.org/wiki/%D0%90%D1%80%D1%85%D0%B8%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80%D0%B0), в которой быстродействие увеличивается за счёт упрощения [инструкций](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%86%D0%B8%D1%8F_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)), чтобы их декодирование было более простым, а время выполнения — меньшим. Для ATmega32 характерны такие особенности, как, например, большое количество регистров общего назначения. Большинство команд выполняются за один машинный такт с одновременным исполнением текущей и выборкой следующей команды. Команды имеют одинаковый формат и простую кодировку. К недостаткам RISC-архитектуры относят большую длину кода программы после компиляции

1. От чего зависит время выполнения команд SBIS и BRLT? Приведите примеры кода (до 3-10 команд каждый), приводящие к различной продолжительности выполнения указанных команд

SBIS - Пропуск следующей команды если бит в порту установлен if (P(b)=1) PC ← PC + 2 or 3.

BRLT - Условный относительный переход. Тестируется бит флага знака (S) регистра статуса и, если бит установлен, выполняется переход относительно состояния счетчика программ. if (N ⊕ V= 1) then PC ← PC + k + 1 (Бит S всегда равен исключающему ИЛИ между флагами N (отрицательный результат) и V (переполнение дополнения до двух))

1. Укажите команды и их аргументы в виде имён регистров общего назначения, портов ввода-вывода и констант в десятичной или шестнадцатеричной системе счисления для следующих машинных слов: 1001 0110 0111 1111, 1111 0011 1110 1101, 1010 1010 0011 1010 и 1001 1010 0011 0110?

ADIW - Сложение константы с регистровой парой - 1001.0110.KKdd.KKKK ( Rdh:Rdl ← Rdh:Rdl + K)

K = 111112 = 31

d = 112 = 3

BRHS - переход при установленном флаге полу-переноса (H=1) - 1111.00kk.kkkk.k101 ( if (H = 1) then PC ← PC + k + 1 )

k = 11111012 = 125

STD - косвенно-регистровая запись в ОЗУ со смещением - 10q0.qq1r.rrrr.1qqq ( (Y + q) ← Rr)

q = 1100102 = 50

r = 112 = 3

SBI - установить бит в порту ввода/вывода - 1001.1010.AAAA.Abbb ( I/O(P,b) ← 1 )

b = 1102 = 6

A = 1102 = 6

1. Приведите пример выполнения циклического сдвига вправо 8-разрядного числа (с переходом младшего разряда в старший) без использования флага Т.

Приведите пример из трёх машинных команд, обеспечивающих сложение 24-разрядного числа с 24-разрядной константой.

* 1. # циклический сдвиг вправо 8-разрядного числа

1. \_main
2. LDI R1, 0x01
3. AND R1, R0
4. LSR R1
5. LSR R0
6. loop
7. ROR RO
8. RJMP loop
9. # сложение 2х 24-битных чисел
10. summ
11. add r3, r0
12. adc r4, r1
13. adc r5, r2
14. Обоснуйте, чем вызваны ограничения допустимых значений номеров регистров и диапазонов констант в некоторых командах микроконтроллера ATmega32?

Для ATmega32 характерны инструкции с одинаковым форматом и простой кодировкой. Для однозначного декодирования вводятся дополнительные байты, из-за чего приходится вводить ограничения на размер аргументов.

# Выводы по лабораторной работе

# Приложение 1 Комментированный листинг программы для МК на языке ассемблера

\* lab\_1.asm

\*

\* Created: 08.04.2023 17:40:29

\* Author: scanel

\*/

.def TMP = R20

.org $000

JMP reset

delay: ; 4 + 5x + 1282(256 - y) + 2nop

LDI R29, 102 ; x xnew = 102

LDI R30, 194 ; y ynew = 194

delay\_sub:

NOP

DEC R29

NOP

BRNE delay\_sub

INC R30

BRNE delay\_sub

NOP

NOP

RET

;default setting

reset:

LDI TMP, 0x01

MOV R0, TMP

CLR TMP

MOV R1, TMP

MOV R2, TMP

MOV R3, TMP

;setting I/O ports

SER TMP ; 0xFF

OUT DDRA, TMP

OUT DDRB, TMP

OUT DDRC, TMP

OUT DDRD, TMP

;setting stack hight to the end of RAM

LDI TMP, HIGH(RAMEND) ; higher rank addr

OUT SPH, TMP

LDI TMP, LOW(RAMEND)

OUT SPL, TMP

;main cycle

loop:

;cyclic shift 32-bit number R0-R3

BST R0, 0 ;saving lower bit to T

LSR R3 ;logic shift to right� 1 put in C

ROR R2 ;again ... C put to 7th R2 bit, 0s bit put to C

ROR R1

ROR R0 ;again

BLD R3, 7 ;7th bit of R3 = T

; showing 32-bit numb R0-R3 to PORTA-PORTD

OUT PORTA, R0

OUT PORTB, R1

OUT PORTC, R2

OUT PORTD, R3

; PAUSE

CALL delay

RJMP loop

# Приложение 3 hex-файл

:020000020000**FC**

:100000000C940D00D6E6E2EC0000DA950000E1F7**72**

:10001000E395D1F700000000089541E0042E4427**45**

:10002000142E242E342E4FEF4ABB47BB44BB41BB**9A**

:1000300048E04EBF4FE54DBF00FA369427941794**21**

:10004000079437F80BBA18BA25BA32BA0E940200**E0**

:02005000F3CF**EC**

:00000001**FF**