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 3**

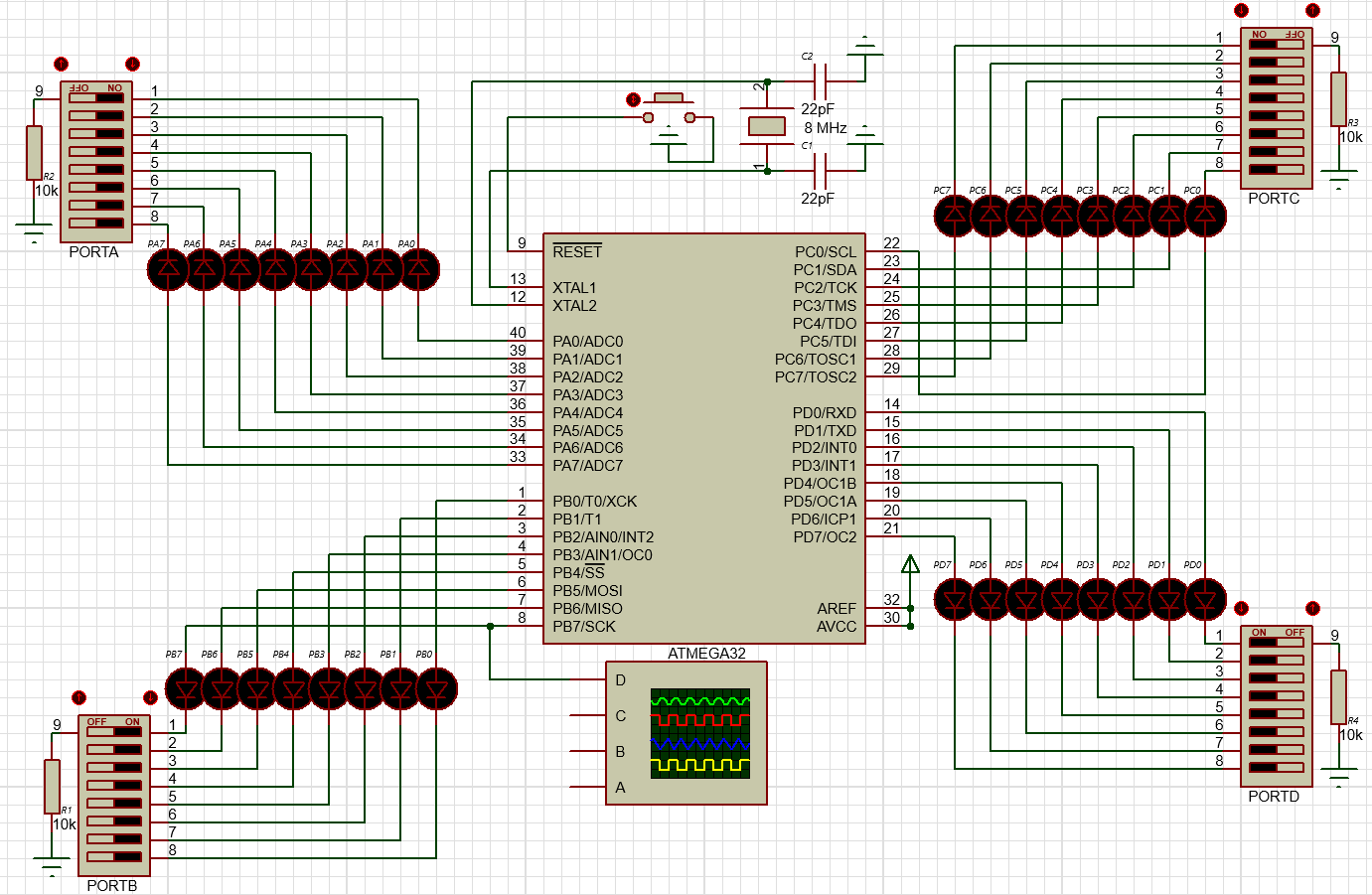
по дисциплине «Аппаратные средства вычислительной техники»

|  |  |  |  |
| --- | --- | --- | --- |
| Выполнили: | студенты группы 4831001/10003 |  | Г. А. Улановский |
|  |  | (подпись, дата) |  |
|  |  |  | Г. Г. Фидаров |
|  |  | (подпись, дата) |  |
|  |  |  |  |
|  |  |  |  |
| Проверил: | доцент, к.т.н. |  | П. О. Семенов |
|  |  | (подпись, дата) |  |

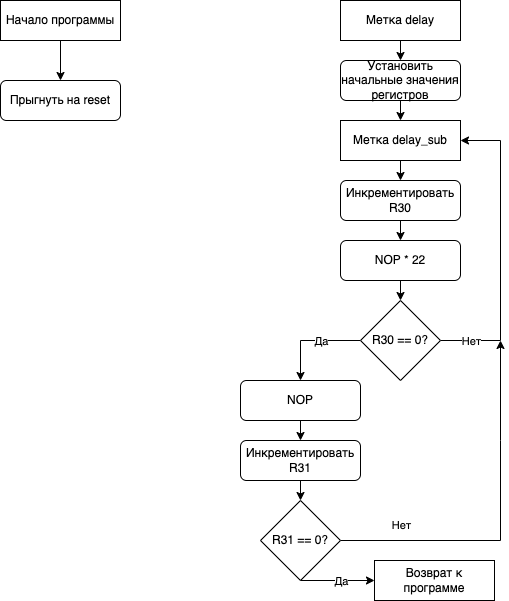
# Формулировка задания

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



# Блок-схема алгоритма работы программы



# Временные диаграммы логических сигналов на портах МК (фрагмент)

# Результаты работы

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

### Анализ lss-файла

### Анализ hex-файла

### Алгоритм выполнения команды ассемблера

# Ответы на контрольные вопросы

|  |  |
| --- | --- |
| Укажите, в чём проявляются признаки RISC-архитектуры в микроконтроллере ATmega32. В чём преимущества и недостатки приведённых особенностей? | Микроконтроллер ATmega32 имеет множество признаков RISC-архитектуры, вот некоторые из них:   1. Фиксированная длина команд: все инструкции ATmega32 имеют одинаковый размер в 16 бит, что делает их выполнение более быстрым и простым. 2. Ограниченный набор команд: ATmega32 имеет небольшой набор команд, которые выполняются быстро и эффективно. 3. Однородность регистров: все регистры в ATmega32 имеют одинаковый размер в 8 бит или 16 бит, что упрощает кодирование программ. 4. Использование стека: ATmega32 использует стек для передачи параметров между функциями, что упрощает программирование и позволяет эффективно использовать память.   Преимущества RISC-архитектуры в микроконтроллере ATmega32 включают:   1. Высокая производительность: RISC-архитектура позволяет быстро выполнять простые команды, что обеспечивает высокую производительность микроконтроллера. 2. Низкое энергопотребление: благодаря простоте и эффективности команд RISC-архитектуры, микроконтроллеры на ее основе потребляют меньше энергии. 3. Меньший размер чипа: меньший набор команд и однородность регистров позволяют уменьшить размер чипа, что снижает стоимость производства.   Недостатки RISC-архитектуры в микроконтроллере ATmega32 включают:   1. Ограниченный набор команд: ограниченный набор команд может усложнить разработку сложных алгоритмов. 2. Неэффективность при выполнении сложных команд: RISC-архитектура не всегда эффективна при выполнении сложных команд, которые требуют многократных обращений к памяти. 3. Ограничения на работу с памятью: доступ к памяти в RISC-архитектуре может быть ограничен, что усложняет работу с данными в некоторых случаях. |
| От чего зависит время выполнения команд SBIS и BRLT? Приведите примеры кода (до 3-10 команд каждый), приводящие к различной продолжительности выполнения указанных команд | Время выполнения команды SBIS и BRLT зависит от частоты тактового генератора микроконтроллера, на котором они выполняются. Также время выполнения этих команд может быть увеличено, если они используются в комбинации с другими командами, которые занимают время на выполнение.  Команда SBIS используется для проверки установки бита в регистре ввода-вывода. Время выполнения этой команды составляет один такт.  Пример кода, использующего команду SBIS:  SBIS PINB, 0  Команда BRLT используется для выполнения условного перехода на метку, если значение регистра меньше заданного. Время выполнения этой команды зависит от того, будет ли выполнен переход или нет. Если переход будет выполнен, то время выполнения составит два такта, если переход не будет выполнен, то время выполнения будет равно одному такту.  Пример кода, использующего команду BRLT:  LDI R16, 10  LDI R17, 20  CP R16, R17  BRLT label1  NOP  label1: NOP  В этом примере мы загружаем значения 10 и 20 в регистры R16 и R17, затем сравниваем их с помощью команды CP. Если значение в R16 меньше, чем значение в R17, то произойдет переход на метку label1, где будет выполнена команда NOP. Если же значение в R16 больше или равно значению в R17, то переход не будет выполнен и будет выполнена следующая команда NOP без дополнительных задержек. |
| 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? | Для того, чтобы определить команды и их аргументы из машинных слов, необходимо знать формат команд и их кодирование. В микроконтроллере ATmega32 команды кодируются в 16-битные слова и имеют следующий формат:   |  |  |  | | --- | --- | --- | | Opcode | Rd | K | | 15-12 | 11-8 | 7-0 |    Машинное слово: 1001 0110 0111 1111   * Opcode: 1001 (SBC) * Rd: 0110 (R6) * K: 0111 1111 (0x7F)   Значит, команда будет иметь вид: SBC R6, #0x7F   Машинное слово: 1111 0011 1110 1101   * Opcode: 1111 (LPM) * Rd: 0011 (R3) * K: 1110 1101 (0xED)   Значит, команда будет иметь вид: LPM R3, Z   Машинное слово: 1010 1010 0011 1010   * Opcode: 1010 (LDX) * Rd: 1010 (X) * K: 0011 1010 (0x3A)   Значит, команда будет иметь вид: LDX X, #0x3A   Машинное слово: 1001 1010 0011 0110   * Opcode: 1001 (SBIC) * Rd: 1010 (A) * K: 0011 0110 (0x36)   Значит, команда будет иметь вид: SBIC A, 6 |
| 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. | Пример выполнения циклического сдвига вправо 8-разрядного числа (с переходом младшего разряда в старший) без использования флага Т:  ; Предположим, что число находится в регистре R16  ; Циклический сдвиг вправо с переходом младшего разряда в старший  LSR R16 ; сдвиг вправо на 1 разряд, младший бит сохраняется в флаге C  ROL R16 ; циклический сдвиг вправо на 1 разряд, флаг C устанавливается в значение младшего бита  Пример из трёх машинных команд, обеспечивающих сложение 24-разрядного числа с 24-разрядной константой:  ; Предположим, что число находится в регистрах R17:R16:R15  ; Константа находится в памяти по адресу 0x123456  LDI R18, high(0x123456) ; загрузка старшего байта адреса константы в регистр R18  LDI R19, low(0x123456) ; загрузка младшего байта адреса константы в регистр R19  LDI R20, 0 ; загрузка младшего байта результата в регистр R20  ADD R15, R19 ; прибавление младшего байта адреса константы к младшему байту числа  ADC R16, R18 ; прибавление старшего байта адреса константы и флага переноса к среднему байту числа  ADC R17, R20 ; прибавление флага переноса к старшему байту числа  Здесь используется инструкция LDI для загрузки адреса константы в регистры R18 и R19, а затем инструкция ADD с флагом переноса (ADC) для сложения 24-разрядного числа, находящегося в регистрах R17:R16:R15, с 24-разрядной константой, находящейся в памяти по заданному адресу. При этом результат сохраняется в тех же регистрах R17:R16:R15.  Начало формы  Конец формы |
| Обоснуйте, чем вызваны ограничения допустимых значений номеров регистров и диапазонов констант в некоторых командах микроконтроллера ATmega32? | Ограничения допустимых значений номеров регистров и диапазонов констант в некоторых командах микроконтроллера ATmega32 обусловлены ограничением размера инструкции и количеством битов, выделенных для закодирования номера регистра или значения константы. |

# Выводы по лабораторной работе

# Приложение 1 Комментированный листинг программы для МК на языке ассемблера

# Приложение 2 Фрагмент lss-файла

# Приложение 3 hex-файл