Министерство образования и науки Украины

ХАРЬКОВСКИЙ НАЦИОНАЛЬНЫЙ
АВТОМОБИЛЬНО-ДОРОЖНЫЙУНИВЕРСИТЕТ

ЛАБОРАТОРНЫЙ ПРАКТИКУМ

по дисциплинам
«Бортовые вычислительные комплексы»
и «Компьютерное управление рабочими процессами»

для студентов специальностей 8.090210, 8.090211, 8.090214,
8.092201, 8.092501

(Часть 1)

Харьков
2009

Учебное издание

ЛАБОРАТОРНЫЙ ПРАКТИКУМ

по дисциплинам «Электротехника, электроника
и микропроцессорная техника» и «Электроника
и микросхемотехника» (раздел «Электроника»)
для студентов специальностей 8.090210, 8.090211,
8.090214, 8.090214, 8.092201, 8.092501

Составители: Богаевский Александр Борисович
Двадненко Владимир Яковлевич
Дзюбенко Александр Андреевич
Боженов Владимир Сергеевич

Ответственный за выпуск А. В. Бажинов

Редактор О. В. Акперова

Компьютерная верстка С. П. Рожкова

Подписано к печати 15.12.2008 г.
Формат 60x84 1/8. Бумага офсетная. Гарнитура School Book C.
Печать RISO

Министерство образования и науки Украины

ХАРЬКОВСКИЙ НАЦИОНАЛЬНЫЙ
АВТОМОБИЛЬНО-ДОРОЖНЫЙУНИВЕРСИТЕТ

ЛАБОРАТОРНЫЙ ПРАКТИКУМ

по дисциплинам
«Бортовые вычислительные комплексы»
и «Компьютерное управление рабочими процессами»

для студентов специальностей 8.090210, 8.090211, 8.090214,
8.092201, 8.092501

(Часть 1)

Утверждено методическим
советом университета,
протокол №3 от 08.12.2008

Харьков
2009

Составители: А. Б. Богаевский
В. Я. Двадненко
А. А. Дзюбенко
В. С. Боженов

Кафедра автомобильной электроники

Теперь нужно загрузить в микроконтроллер программу формирования ШИМ-сигнала. Нажимаем кнопку «Download» и выбираем в выпавшем окошке папку «ASM51» и транслированный файл нашей программы (рис. 10). Нажимаем «Открыть» — программа загружена.

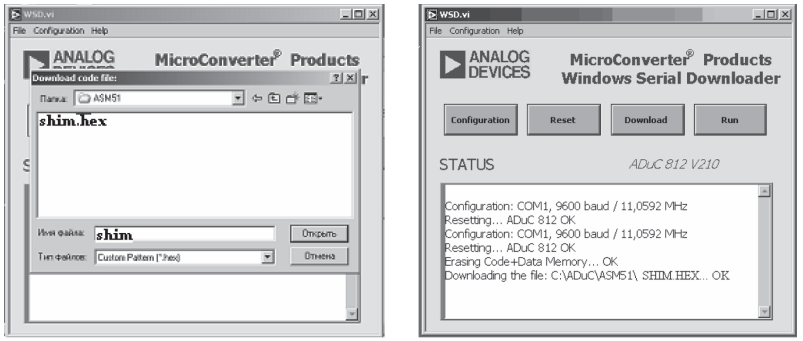


Рисунок 10

Нажатием кнопки «Run» включаем микроконтроллер на отработку программы управления. Ее реализацию можно увидеть, наблюдая на экране осциллографа изображение сигнала.

2.6 Меняя значения регистров R0, R1 для данного стенда набирать R1,R0 соответственно 0E,02 (медленно вращается), затем 0D,0F(средние обороты) и 00,0F (практически максимальные обороты).

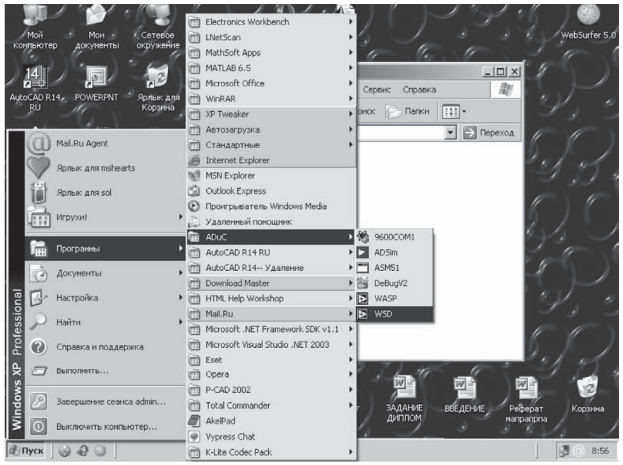


Рисунок 8

2.5 Проверяем работоспособность программы в составе аппаратных средств.

В меню «Пуск», «Программы» находим пакет «ADuC», выбираем приложение «WSD» (рис. 8).

Далее делаем перезагрузку микроконтроллера: в появившемся окне (рис. 9), нажимаем кнопку «Reset».

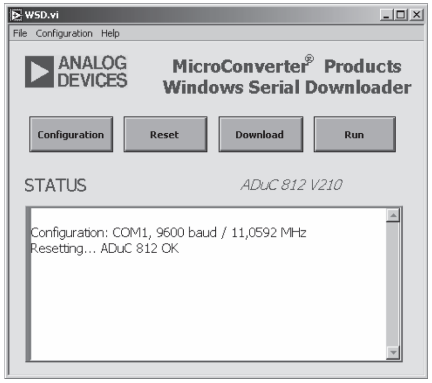


Рисунок 9

ОБЩИЕ ПОЛОЖЕНИЯ

Лабораторные работы помогают студентам ознакомится с работой наиболее распространенных программно-аппаратных бортовых блоков транспортных машин и позволяют экспериментально проверить основные положения теории, приобрести опыт по сборке процессорных схем.

Непосредственное участие в экспериментах вырабатывает у студентов практические навыки по методике их проведения и обработке полученных результатов, по которым они должны научиться оценивать электрические характеристики компонентов БВК.

В каждой работе конкретно сформулирована ее цель, приведены краткие теоретические положения, порядок эксперимента и контрольные вопросы.

Для проведения лабораторной работы требуется определенная теоретическая подготовка, во время которой необходимо рассмотреть вопросы, подлежащие исследованию.

Прежде чем приступать к экспериментальной части работы, нужно внимательно изучить методические рекомендации, четко представить цель опыта и характер исследуемых явлений, детально ознакомится со схемой и элементами лабораторного стенда, а также последовательного выполнения лабораторной работы.

Все работы выполняются на специализированных лабораторных стендах, обеспечивающих их последовательное проведение.

ПОРЯДОК ВЫПОЛНЕНИЯ И ОФОРМЛЕНИЯ ЛАБОРАТОРНЫХ РАБОТ

После включения напряжения, не записывая показания приборов, следует убедиться на опыте в возможности получения заданных пределов изменения режима исследуемой схемы, качественно определить характер соответствующей зависимости и установить диапазоны изменения измеряемых величин. На основании этого предварительного опыта необходимо выбрать интервалы, через которые следует записывать показания приборов на различных участках исследуемой зависимости. При проведении опыта необходимо убедиться в том, что по точкам, полученным на основании измерений, можно построить кривую исследуемой зависимости со всеми ее характерными особенностями: максимумы, минимумы, резкое изменение углов наклона и т.п.

При записи показаний приборов нужно обязательно указывать единицы измеряемых величин. Если цена делений прибора непосредственно на его шкале не указана, то в примечании к таблице отчета следует записать цену деления прибора. В этом случае для каждого режима записывают две цифры: число делений по шкале прибора и вычисленное с учетом цены деления значение измеряемой величины.

Приборы с неравномерной шкалой не пригодны для измерений в начальной части шкалы между нулем и первым делением. При таких показаниях в отчете указывают, что показание прибора меньше значения, соответствующего первому делению шкалы прибора.

Все записи в отчете следует делать четко и аккуратно. Если при измерениях, или записи допущена ошибка, то неправильные результаты зачеркивают, а правильные значения записывают в новую строку.

2.1 Подключаем осциллограф к выводу микроконтроллера P2.0

2.2 На плате быстрого старта необходимо соединить вывод 0 порта 2 с входом выходного каскада оптронной развязки. Для этого необходимо замкнуть перемычкой выводы J5.1 и J12.1 (см. схему лабораторной работы №1).

2.3 Подсоединим плату «быстрого старта» микроконтроллера ADuC812 к COM-порту ПК, с помощью интерфейсного кабеля.(рис 1).

2.4 Подключим шлейф и разъем идущие от микроконтроллера ADuC к стенду микропроцессорного управления электрическим двигателем (рис. 2).

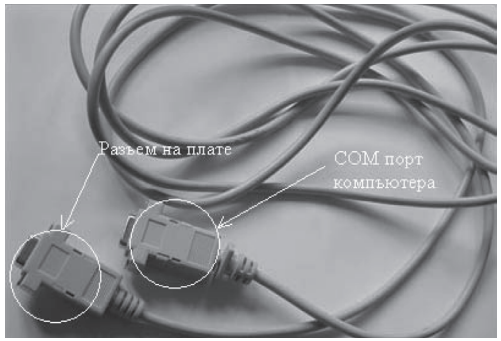


Рисунок 6 — Внешний вид кабеля COM порта

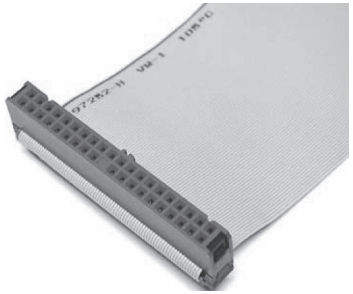


Рисунок 7 — Внешний вид шлейфа соединяющего разъем стенда с контроллером

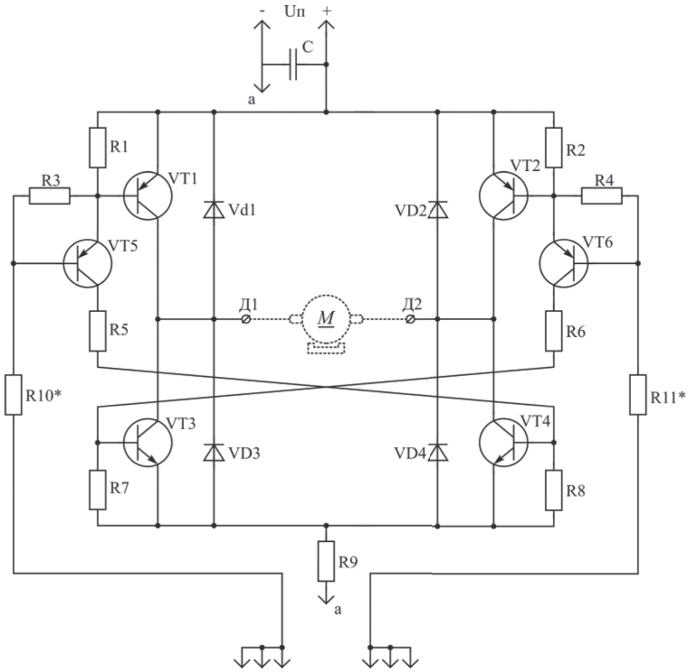


Рисунок 4 — Электрическая схема транзисторного ключа

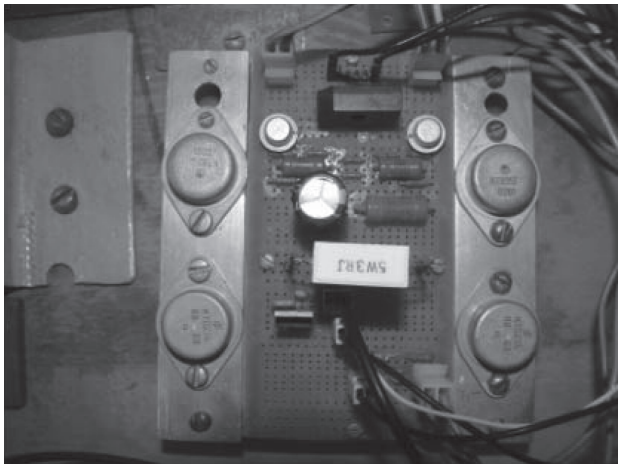


Рисунок 5 —Внешний вид электрической схемы транзисторного ключа

После окончания каждого этапа работы, до разборки схемы, следует оценить правильность полученных результатов, обработать данные эксперимента и представить этот материал преподавателю для проверки, после которой можно разобрать схему и приступить к следующей части работы. По окончании всей работы преподаватель визирует данные эксперимента каждого студента и делает отметку о выполнении студентом лабораторной работы в журнале.

К очередному занятию в лаборатории каждый студент должен представить полностью оформленный отчет по предыдущей лабораторной работе.

Лабораторный отчет должен содержать:

- наименование и цель работы;
- перечень приборов, применяемых в работе, и их характеристики;
- схемы электрической цепи, вычерченные в соответствии с ГОСТами;
- краткое описание работы, необходимые формулы и расчетные зависимости;
- экспериментальные данные и результаты их обработки, сведенные в таблицы;
- графики и диаграммы, выполненные в координатных сетках размером не менее 100 х 150 мм. По осям указывают обозначения величин и единицы их измерения. На графиках следует четко показать все точки, соответствующие экспериментальным данным. Необходимо, чтобы расчеты, графики и диаграммы имели наименования и краткие пояснения. Студенты, пришедшие на занятия неподготовленными или не составившими отчет по предыдущей работе, к выполнению следующей работы не допускаются.

ПРАВИЛА ТЕХНИКИ БЕЗОПАСНОСТИ ПРИ ВЫПОЛНЕНИИ ЛАБОРАТОРНЫХ РАБОТ

При выполнении лабораторных работ во избежание несчастных случаев студенты обязаны строго соблюдать правила техники безопасности, которые содержат следующие основные требования:

1. Перед сборкой системы необходимо убедиться, что все выключатели источников энергии стенда отключены.
2. При сборке схемы обратить внимание на надежность электрического контакта между отдельными элементами и соединительными проводниками.
3. Включить выключатели источников электроэнергии стенда можно только после проверки схемы преподавателем и в его присутствии.
4. Не прикасаться к оголенным частям электрической цепи, находящимся под напряжением.
5. Не производить какие-либо изменения и не устранять неисправности в схеме при включенных выключателях источников электроэнергии стенда.
6. При исчезновении напряжения в питающей сети немедленно отключить все выключатели источников электроэнергии стенда.
7. При обнаружении любой неисправности в схеме немедленно отключить все выключатели источников электроэнергии стенда и сообщить об этом преподавателю.
8. В случае поражения током немедленно обесточить схему и оказать первую помощь пострадавшему.
9. Не оставлять включенной схему во время перерыва.
10. После окончания работы и проверки данных, полученных преподавателем, выключатели источников электроэнергии стенда отключить, а схему разобрать.

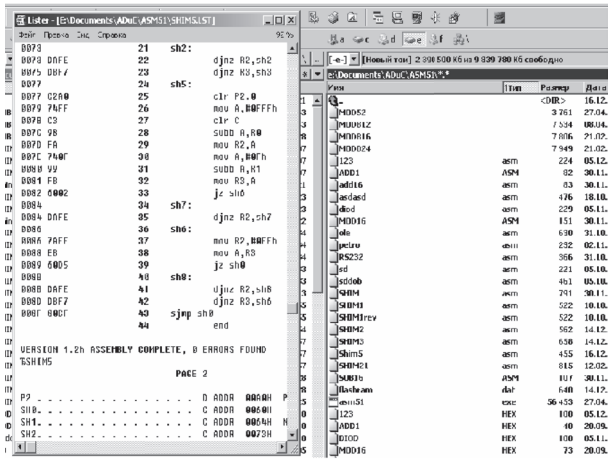


Рисунок 2

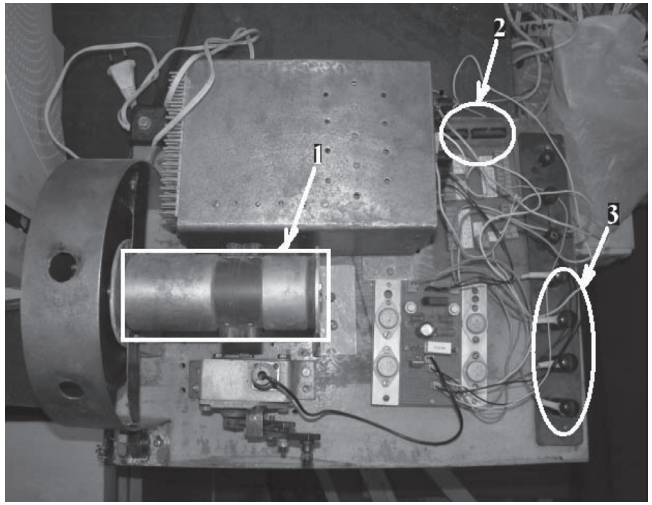


Рисунок 3 — Электромашинный стенд с устройствами согласования: 1 — электрический двигатель постоянного тока; 2 — место подключения шлейфа соединяющего микроконтроллер с електромеханическим стендом; 3 — выводы питания електродвигателя (2 нижиние) и тахогенератора (2 верхние)

мируется импульс. Под меткой sh5 выполняется обнуление **(clr P2.0)** и получение длительности паузы, путем вычитания длительности импульса от максимально возможного значения периода (Tmax = 0FFFh). Под метками sh7, sh6 и sh8 выполняется обнуление регистров **R2** и **R3** так как в них занесена длительность паузы(длительность паузы определена разностью между длительностью периода и длительностью импульса). По метке **sjmp sh0** выполняется прыжок в начало программы(программа циклически повторяется).

2 Подготовка стенда

Внимание!!! Подключение питания только после проверки правильности подключения всех приборов преподавателем.

Ретранслируем программу, для этого в командной строке Total Commander пишем »asm51.exe shim.asm», нажимаем Enter (рис. 1).

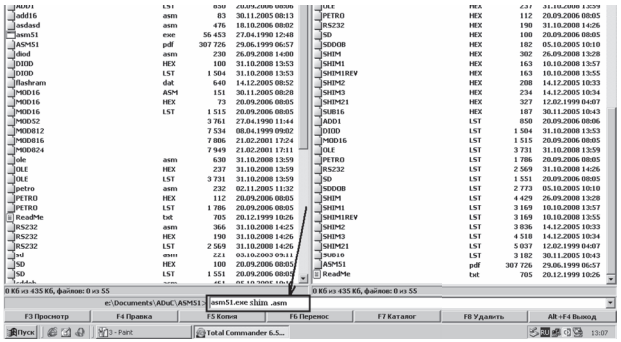


Рисунок 1

После этого в папке появляется ещё 2 файла: **shim.LST** и **shim.HEX** — в первом из них записывается информация об ошибках, допущенных при составлении программы, а во втором коды для прошивки микроконтроллера. Выделяем **shim.LST** и нажимаем на клавиатуре F4, после чего наблюдаем количество ошибок. Если ошибок нет, то можно загружать программу в микроконтроллер.

Ответственность за соблюдение правил техники безопасности возлагается на студентов, работающих в лаборатории, а контроль за их выполнением — на преподавателя.

Лабораторная работа №1

«ПРОГРАММНОЕ УПРАВЛЕНИЕ СВЕТОДИОДОМ С ИСПОЛЬЗОВАНИЕМ МИКРОКОНТРОЛЛЕРА ADuC812»

Цель работы: получение практических навыков по управлению светодиодом с помощью программируемой логики от микроконтроллера ADuC812.

Оборудование: плата «быстрого старта» микроконтроллера ADuC812, персональный компьютер, пакет прикладных программ «ADuC», соединительные провода.

Общие теоретические сведения

Микросхема ADuC812 является одним из представителей микроконвертеров фирмы Analog Devices.

Схема «быстрого старта» ADuC812 (рис. 1) содержит светодиод, включенный через линию порта P3.4 с обратной логикой (рис. 2). Наличие сигнала на этой линии микроконтроллера (МК) должно инициировать потушенное состояние светодиода. Для осуществления такого управления перед светодиодом стоит инвертирующий триггер Шмидта (на схеме 74HC14), изменяющий сигнал из МК на обратный (0 на 1 и наоборот).

1.2 Описание работы программы.

Чтобы компилятор программы корректно воспринимал названия регистров R2 и R3 общего назначения, которые использованы в программе, подключается файл **mod812**, функция которого заключается в присвоении определённых адресов памяти для каждого из регистров.

Далее выполняется уход из зоны памяти, скачек на sh0 вызван тем, что запрещено занимать зону, отведенную для обслуживания прерывания (60 ячеек). Если не осуществить этот переход, то программа может работать некорректно в случае, когда выполняется управление по прерываниям. Здесь такое управление не задействовано, но всегда необходимо учитывать все возможные ситуации, например, текст данной программы будет взят как подпрограмма более сложной программы, и если не учесть диапазона памяти для их обслуживания, то ошибка неизбежна!!!

В метке **sh0** записываются данные в **R1** и **R0 —** это информация о длительности импульса (в данном случае длительность составляет 00FFh).

Команда **setb P2.0** выполняет запись единицы в 0-й бит порта **P2**. Записываем данные о длительности импульса в регистры **R2** и **R3**. На основе регистров **R3, R2** организуется счетчик, причем регистр R3 является старшим, а регистр **R2** — вспомогательным. **djnzRn**,<metka> — команда декремент и переход на метку если не равен нулю, выполняет вычитание «1» из указанной части, если результат не равен «0».

jz sh3 — проверка содержимого **R3** (если 0, переход на sh5). **djnz R2,sh4** команда декремент(обнуление ,отниманием единицы, регистра R2). Под меткой sh3 выполняется заполнение R2 в случае ненулевого значения R3 (см **sh1**) и проверка R3 (если 0, переход на sh5),а когда **R3** не равно 0 командами **djnz R2,sh2, djnz R3,sh3** фор-

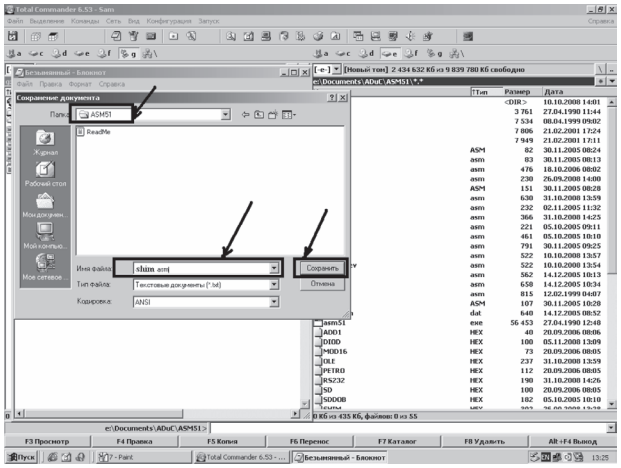


Рисунок 8

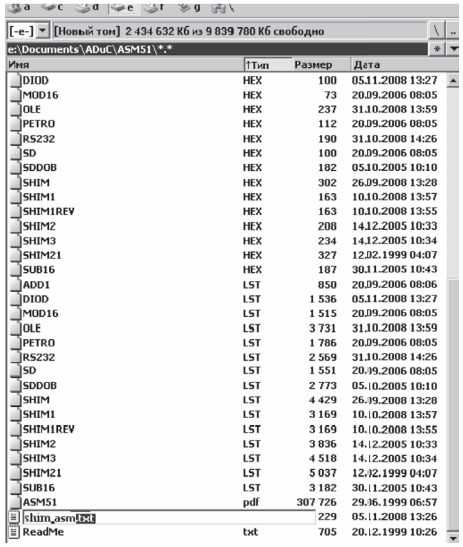


Рисунок 9

SHIM.asm.txt, нажимаем на него левой кнопкой мышки и удаляем **«.txt»**, жмём **Enter** (рис. 9).

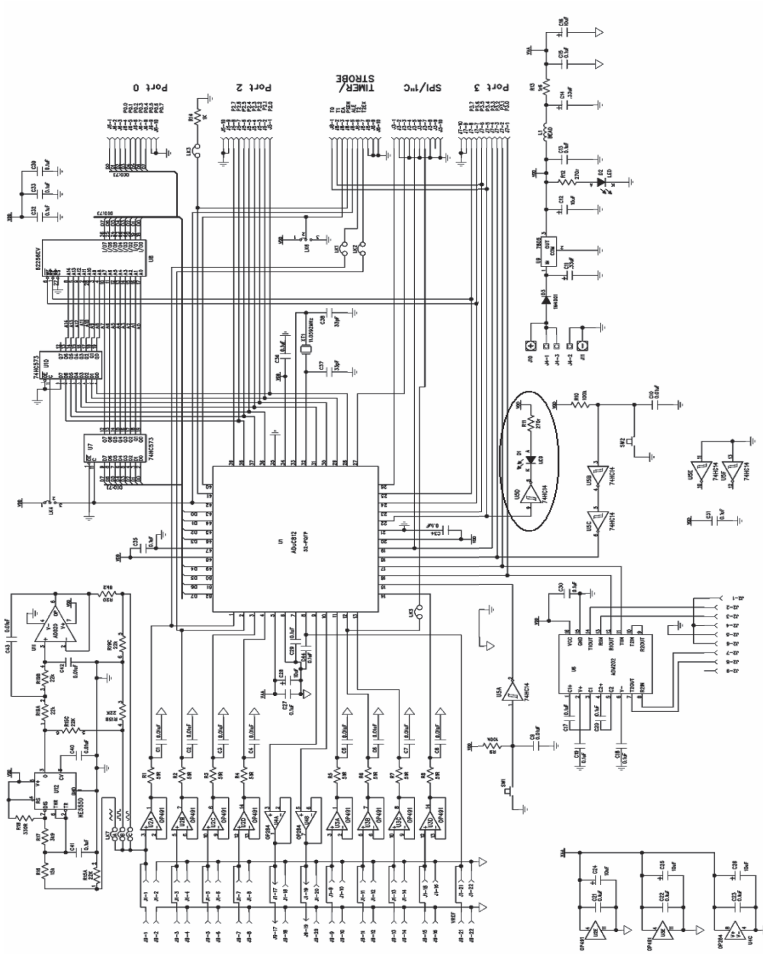


рис. 1 — Схема платы “быстрого старта” микроконтроллера ADuC812

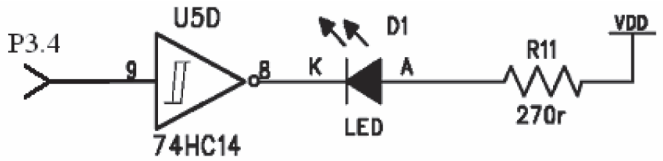


рис. 2 — Включение светодиода в схему платы “быстрого старта”

Порядок выполнения работы

1. Составление программы по управлению светодиодом.

Чтобы составить программу, необходимо: открыть Total Commander (рис. 3), найти на одном из локальных дисков компью-

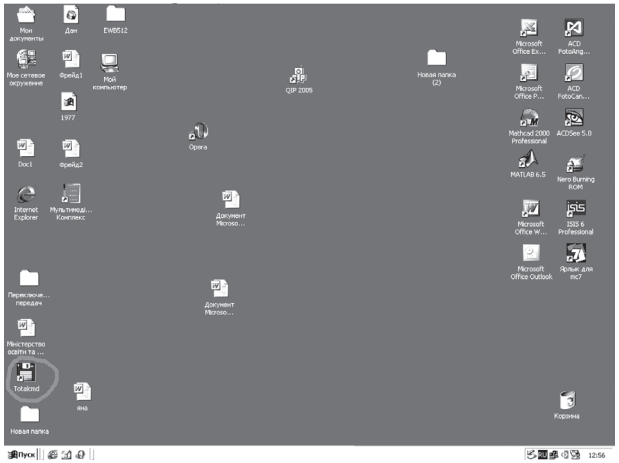


Рисунок 3

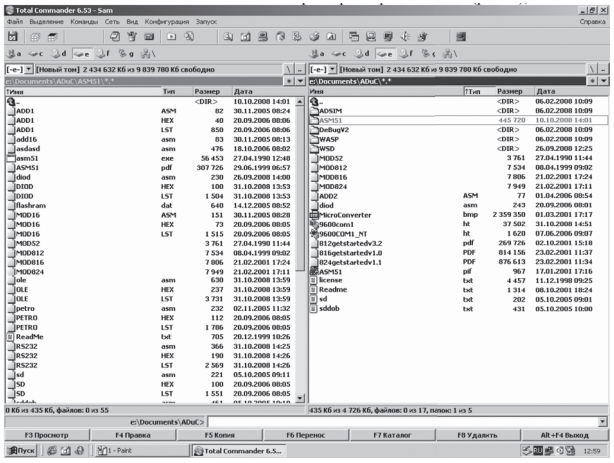


Рисунок 4


```
sh6:
        mov R2,#0FFh  ; заполнение R2 в случае ненулевого значения
                      R3
        mov A,R3
        jz sh0        ; в случае нулевого значения R3 переходим в
                      начало программы, т.к. пауза отсутствует
sh8:
        djnz R2,sh8   ; формирование паузы
        djnz R3,sh6
sjmp sh0              ;прыжок на начало (зацикливание програм-
                      мы)
end                   ;метка окончания программы.
```

Сохраняем результат, выполняя команду Файл → Сохранить как... (рис. 7), указываем путь к своей папке **ASM51** (с файлом

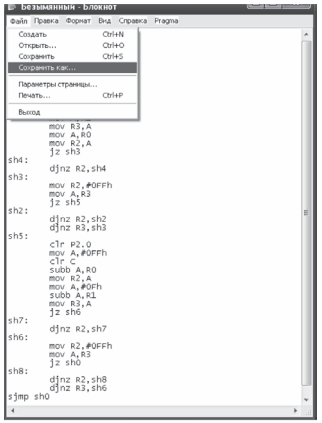


Рисунок 7

asm51.exe), записываем имя (имя файла не должно содержать более 8 символов) и нажимаем кнопку **Сохранить** (рис. 8), изменяем расширение файла, для чего выделяем сохраненный нами

```
        mov R3,A
        mov A,R0
        mov R2,A
        jz sh3          ; проверка R2 (если 0, переход на sh3)
sh4:
        djnz R2,sh4; начало формирования импульса (младший ре-
                        гистр)
sh3:
        mov R2,#0FFh ; заполнение R2 в случае ненулевого значения
                        R3
        mov A,R3
        jz sh5          ; проверка R3 (если 0, переход на sh5)
sh2:
        djnz R2,sh2     ; формирование импульса
        djnz R3,sh3
sh5:
        clr P2.0        ; очистка порта P2.0
        mov A,#0FFFh ; получение длительности паузы, путем вычи-
                        тания длительности импульса от максимально
                        возможного значения периода (Tmax = 0FFFh)
        clr C
        subb A,R0
        mov R2,A
        mov A,#0Fh
        subb A,R1
        mov R3,A
        jz sh6          ; проверка R3 (если 0, переход на sh6)
sh7:
        djnz R2,sh7     ; начало формирования паузы (младший ре-
                        гистр)
```

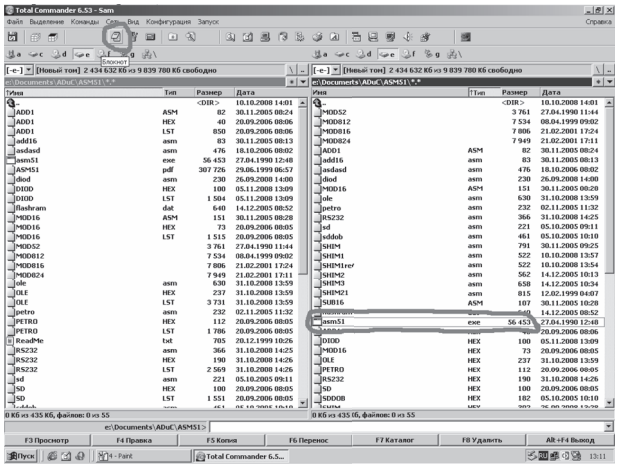


Рисунок 5

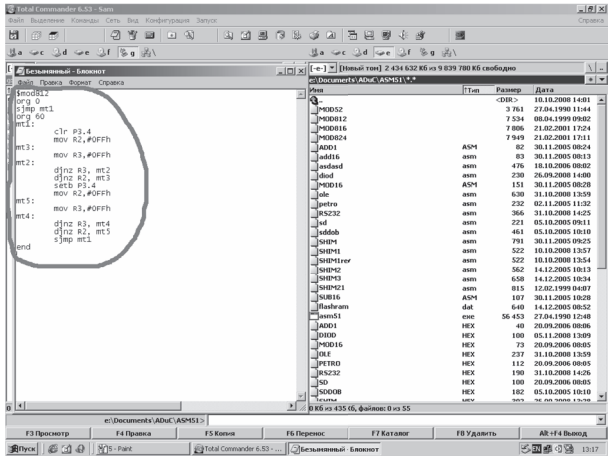


Рисунок 6

тера директорию ASM51 (рис. 4), войти в неё, убедится, что она содержит файл asm51.exe, и создать там текстовый документ в редакторе Блокнот (рис. 5), в открывшемся окне записать следующий ниже текст программы (рис. 6):

```
$mod812          ;подключение файла mod812
org 0            ;определение адреса последующей команды
     sjmp mt1    ;переход на метку mt1 — скачок в зону памя-
                 ти, свободную от ;обслуживания прерываний
org 60
mt1:
     clr P3.4    ;очистка (запись нулевого значения) порта
                 P3.4 — диод не ;светится
     mov R2,#0FFh ;установка старшего разряда времени, в тече-
                 ние которого диод ;будет потушен
mt3:
     mov R3,#0FFh ;установка младшего разряда времени, в тече-
                 ние которого диод ;будет потушен
mt2:
     djnz R3, mt2 ;цикл убывания младшего разряда
     djnz R2, mt3 ;цикл убывания старшего разряда
     setb P3.4   ;взведение (запись единицы) порта P3.4 —
                 диод светится
     mov R2,#0FFh ;формирование старшего разряда времени, в
                 течение которого диод ;будет светится
mt5:
     mov R3,#0FFh ;формирование младшего разряда времени, в
                 течение которого ;диод ;будет светится
mt4:
     djnz R3, mt4 ;цикл убывания младшего разряда
     djnz R2, mt5 ;цикл убывания старшего разряда
     sjmp mt1    ;переход на метку mt1 — зацикливание про-
                 граммы
     end
```

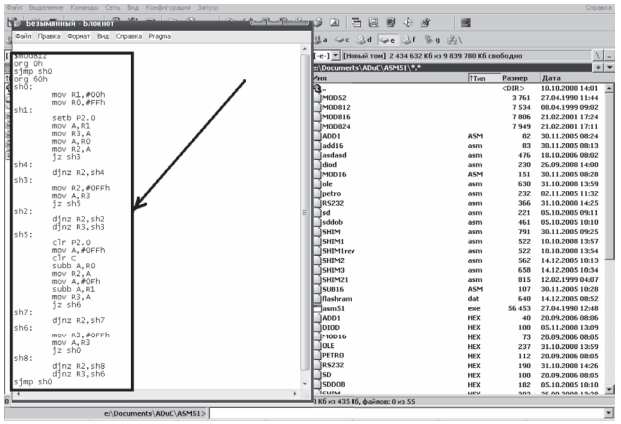



Рисунок 6

Программа формирования ШИМ-сигнала

```
$mod812          ; подключение файла mod812
org 0h           ;определение адреса последующей команды
     sjmp sh0    ;скачек на sh0 вызван тем, что запрещено за-
                 нимать зону, отведенную для обслуживания
                 прерывания (60 ячеек)
org 60h
sh0:
     mov R1,#00h ; информация о длительности импульса (в
                 данном случае длительность составляет 00F-
                 Fh)
   mov R0,#0Fh
sh1:
     setb P2.0   ; возведение бита на выводе P2.0
     mov A,R1    ; копирование данных о импульсе для даль-
                 нейшей работы с ними (информация о дли-
                 тельности импульса должна остаться неиз-
                 менной в регистрах R0 и R1)
```

тера директорию ASM51 (рис.4), войти в неё, убедится, что она содержит файл asm51.exe, и создать там текстовый документ в редакторе Блокнот (рис. 5), в открывшемся окне (рис. 6) записать текст программы.

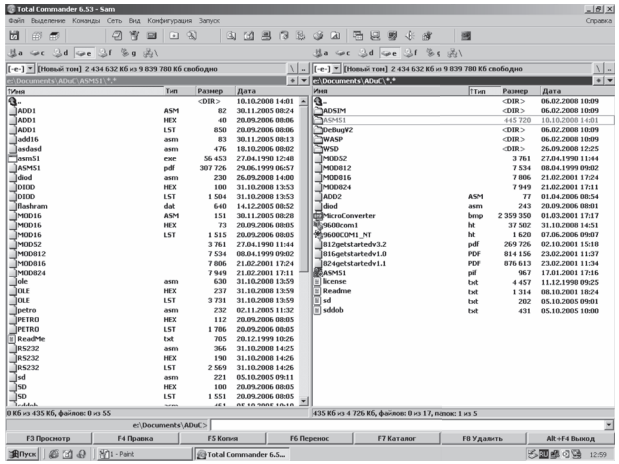


Рисунок 4

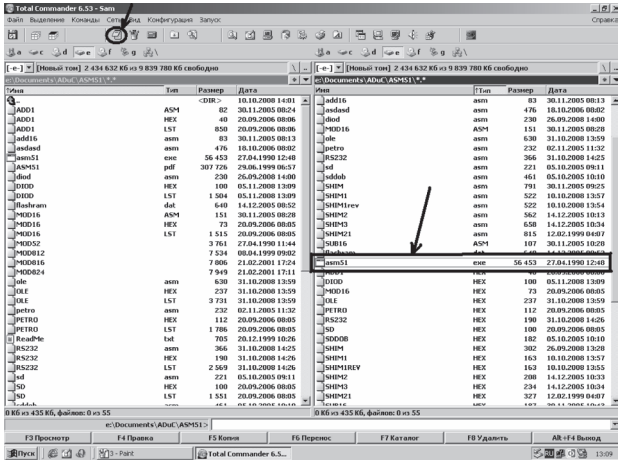


Рисунок 5

Полученный результат сохраняем, выполняя команду **Файл —> Сохранить как...** (рис. 7), указываем путь к своей папке ASM51 (с файлом asm51.exe), записываем имя и нажимаем кнопку Сохранить (рис. 8), изменяем расширение файла, для чего выделяем сохраненный нами diod.asm.txt, нажимаем на него левой кнопкой мышки и удаляем «.txt», жмём Enter (рис. 9).

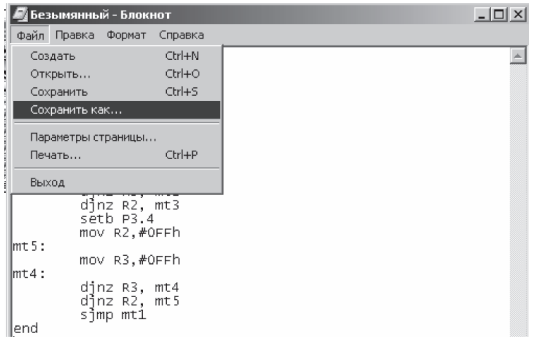


Рисунок 7

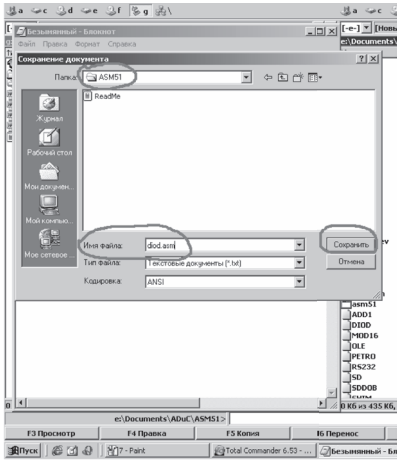


Рисунок 8

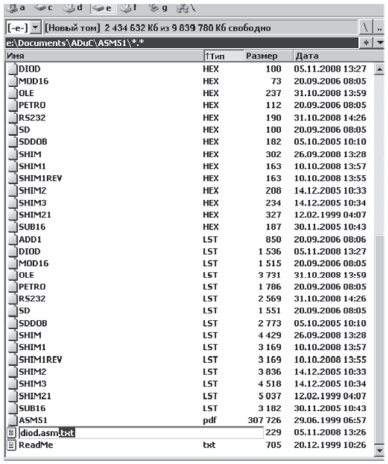
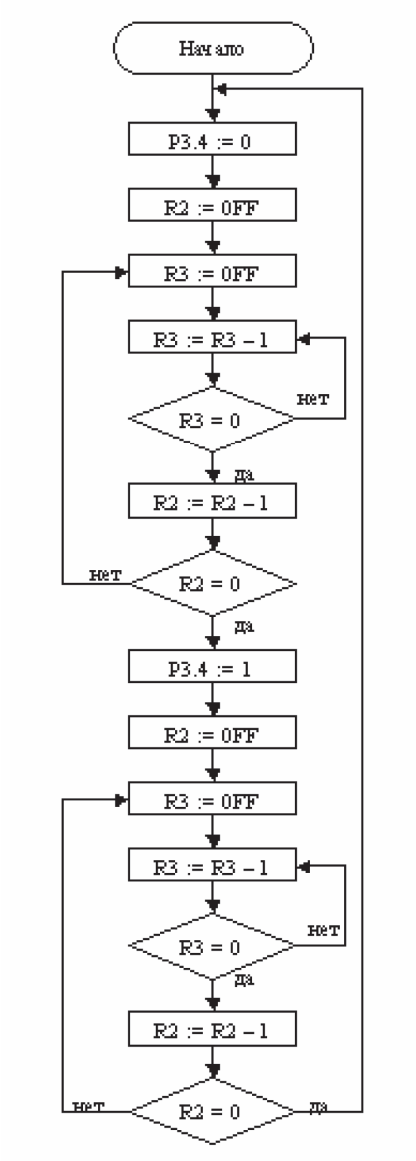


Рисунок 9

1.1 Алгоритм программы.



Восстанавливается непрерывный аналоговый сигнал арифметическим усреднением импульсов за много периодов при помощи простейшего фильтра низких частот. Хотя обычно даже этого не требуется, так как электромеханические составляющие привода обладают индуктивностью, а объект управления (ОУ) — инерцией, импульсы с выхода ШИП сглаживаются и ОУ, при достаточной частоте ШИМ-сигнала, ведёт себя как при управлении обычным аналоговым сигналом.

Информация о длительности импульса храниться в регистрах R0 и R1. Она остается неизменной и модифицируется только оператором. Текущая информация о длительности импульса и паузы будет храниться в R3 и R2.

Длительность паузы получаем путем вычитания длительности импульса от максимально возможного значения периода **(0FFFh).**

Порядок выполнения работы

1 Составление программы по формированию ШИМ сигнала.

1.1 Чтобы составить программу, необходимо:открыть Total Commander (рис. 3), найти на одном из локальных дисков компью-

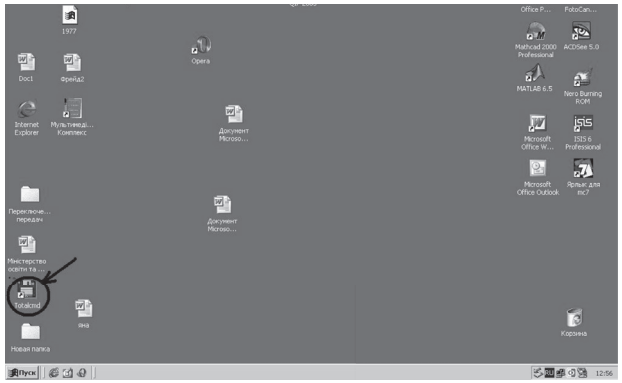


Рисунок 3

В цифровой технике, выходы которой могут принимать только одно из двух значений, приближение желаемого среднего уровня выхода при помощи ШИМ является совершенно естественным. Схема настолько же проста: пилообразный сигнал генерируется N-битным счётчиком. Цифровые устройства (ЦШИП) работают на фиксированной частоте, обычно намного превышающей реакцию управляемых установок (передискретизация). В периоды между фронтами тактовых импульсов, выход ЦШИП остаётся стабильным, на нём действует либо низкий уровень либо высокий, в зависимости от выхода цифрового компаратора, сравнивающего значение счётчика с уровнем приближаемого цифрового сигнала V(n). Выход за много тактов можно трактовать как череду импульсов с двумя возможными значениями 0 и 1, сменяющими друг-друга каждый такт T. Частота появления единичных импульсов получается пропорциональной уровню приближаемого сигнала ~V(n). Единицы, следующие одна за другой, формируют контур одного, более широкого импульса. Длительности полученных импульсов переменной ширины ~V(n), кратны периоду тактирования T, а частота равна 1/(T*2N). Низкая частота означает длительные, относительно T, периоды постоянства сигнала одного уровня, что даёт невысокую равномерность распределения импульсов.

Описанная цифровая схема генерации подпадает под определение однобитной (двухуровневой) импульсно-кодовой модуляции (ИКМ). 1-битную ИКМ можно рассматривать в терминах ШИМ как серию импульсов частотой 1/T и шириной 0 либо T. Добиться усреднения за менее короткий промежуток времени позволяет имеющаяся передискретизация. Высоким качеством обладает такая разновидность однобитной ИКМ, как импульсно-плотностная модуляция (pulse density modulation), которая ещё именуется импульсно-частотной модуляцией.

1.2 Описание работы программы.

Чтобы компилятор программы корректно воспринимал названия регистров R2 и R3 общего назначения, которые использованы в программе, подключается файл **mod812**, функция которого заключается в присвоении определённых адресов памяти для каждого из регистров.

Далее выполняется уход из зоны памяти, предназначенной для обслуживания прерываний, с помощью команды **sjmp mt1.** Если не осуществить этот переход, то программа может работать некорректно в случае, когда выполняется управление по прерываниям. Здесь такое управление не задействовано, но всегда необходимо учитывать все возможные ситуации, например, текст данной программы будет взят как подпрограмма более сложной программы, в которой будут использоваться прерывания, и если не учесть диапазона памяти для их обслуживания, то ошибка неизбежна!!!

Команда **clr P3.4** выполняет запись нуля в 4-й бит порта P3, являющегося параллельным (каждый из 8-ми бит имеет собственный вывод на плате). На схеме электрической принципиальной видно, что у микросхемы 52 «ножки», 16-й, 17-й, 18-й, 19-й выводы соответствуют 0-му, 1-му, 2-му и 3-му биту P3, с 22 по 25 — с 4-го по 7-й бит. К 5-му выводу (соответствующему 4-му биту) присоединяется цепь с диодом. В этой цепи сигнал, полученный на выходе микроконтроллера, проходя через микросхему 74HC14, инвертируется, то есть при низком уровне сигнала (нуль) на входе 74HC14 на его выходе получаем высокий уровень (единицу) и наоборот, что обеспечивает приемлемый для МК ток, а, следовательно, и тепловой режим его работы. Светодиод анодом подключен к «+» питания, а катодом — к выходу 74HC14. Резистор R11 ограничивает ток таким образом, чтобы не превысить максимально допустимый ток

через светодиод и обеспечить достаточную яркость его мерцания (ID1»10 мА).

На основе регистров R2 и R3 организуется программный счётчик, работающий следующим образом:

1) программа постепенно уменьшает содержимое регистра R3, являющегося младшим разрядом установленного времени потушенного состояния диода, и сравнивает его с нулём (**«djnz R3, mt2»** — кусок программы, где метка **mt2** зацикливается сама на себе);
2) когда содержимое регистра R3 достигает нулевого значения, программа уменьшает на единицу содержимое регистра R2 — старшего разряда времени — и переходит на метку **mt3** (кусок программы **«djnz R2, mt3»**), в которой в регистр R3 записывается его начальное значение;
3) таким образом, получается, что после каждого уменьшения на единицу старшего разряда повторяется цикл для младего — **«djnz R3, mt2»**; например, для десятиразрядного числа 57 программа работает следующим образом: сначала уменьшается младший разряд, то есть 7, 6, 5,..., 2, 1, 0, потом старший разряд уменьшается на единицу (устанавливается 4), а в младший снова записывается число 7, которое опять снижается до нуля, после — в старшем — 3, в младшем — 7, и так далее, пока в обоих разрядах не установятся нулевые значения;
4) время, в течение которого в обоих разрядах будут достигнуты нули, и будет временем погашенного состояния диода; для случая, приведенного в программе (с максимальными значениями в обоих разрядах), это время будет составлять $t \approx (257 \cdot 256 \cdot 2 + 2 \cdot 1) \cdot (12{,}0)/11{,}059\text{МГц} \approx 0{,}142\text{с}$ (в числителе — число, показывающее количество машинных циклов, за которое выполнится фрагмент программы с потушенным (зажженным)

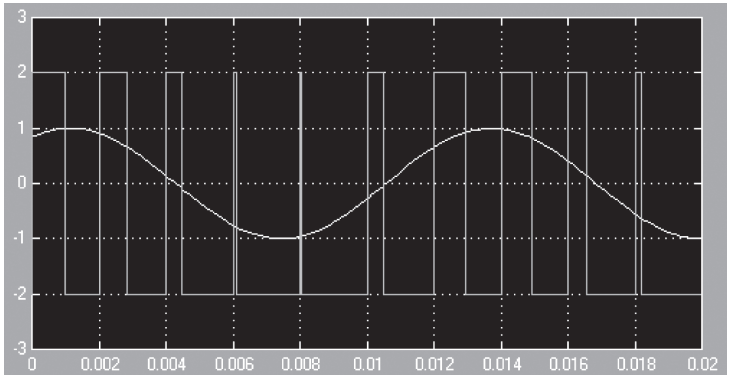
транзистора крайне мало, и, следовательно, падение напряжения на нём близко к нулю — выделяемая мощность так же мала.

$$Rtr \to \infty \leftrightarrow P = \frac{U^2}{R} \to 0$$

$$Rtr \to 0 \leftrightarrow P = I^2 R \to 0$$

Как работает ШИМ.

ШИМ есть импульсный сигнал постоянной частоты и переменной скважности, то есть отношения длительности импульса к периоду его следования. С помощью задания скважности (длительности импульсов) можно менять среднее напряжение на выходе ШИМ генерируется аналоговым компаратором, на отрицательный вход которого подаётся опорный сигнал в виде «пилы» или «треугольника», а на положительный — собственно сам модулируемый непрерывный аналоговый сигнал. Частота импульсов соответствует частоте «зубъев» пилы. Ту часть периода, когда входной сигнал выше опорного, на выходе получается единица, ниже — нуль.



лителей мощности, который достигается за счёт использования их исключительно в ключевом режиме. Это значительно уменьшает выделение мощности на силовом преобразователе (СП).

Применение

При широтно-импульсной модуляции в качестве несущего колебания используется периодическая последовательность прямоугольных импульсов, а информационным параметром, связанным с дискретным модулирующим сигналом, является длительность этих импульсов. Периодическая последовательность прямоугольных импульсов одинаковой длительности имеет постоянную составляющую, обратно пропорциональную скважности импульсов, то есть прямо пропорциональную их длительности. Пропустив импульсы через ФНЧ с частотой среза, значительно меньшей, чем частота следования импульсов, эту постоянную составляющую можно легко выделить, получив постоянное напряжение. Если длительность импульсов будет различной, ФНЧ выделит медленно меняющееся напряжение, отслеживающее закон изменения длительности импульсов. Таким образом, с помощью ШИМ можно создать несложный ЦАП: значения отсчётов сигнала кодируются длительностью импульсов, а ФНЧ преобразует импульсную последовательность в плавно меняющийся сигнал.

ШИМ использует транзисторы (могут быть и др. элементы) не в активном (правильнее будет сказать — линейном), а в ключевом режиме, то есть транзистор всё время или разомкнут (выключен), или замкнут (находится в состоянии насыщения). В первом случае транзистор имеет бесконечное сопротивление, поэтому ток в цепи не течёт, и, хотя всё напряжение питания падает на транзисторе, то есть КПД=0 %, в абсолютном выражении выделяемая на транзисторе мощность равна нулю. Во втором случае сопротивление

состоянием диода, оно состоит из 65792 команд, выполняемых за машинных цикла, и 2 команд длительностью в 1 машинный цикл, а также частота кварцевого резонатора, при к4оторой длительность машинного цикла равна 1 мкс, а в знаменателе — тактовая частота кварцевого генератора, устанавливающая время выполнения 1 машинного цикла.

Командой **setb P3.4** в 4 бит порта P3 записывается 1, после выполнения которой диод светится.

Аналогичным образом на регистрах R2 и R3 (но не обязательно с теми же значениями) программно формируется счётчик для времени светящегося состояния диода — метки **mt4**, **mt5** соответственно.

Команда **sjmp mt1** осуществляет переход на метку **mt1** в начало алгоритма, тем самым зацикливая программу.

2. Работа программы в составе аппаратных средств.

Проделаем операции, в результате которых, будет возможно посмотреть как работает плата «быстрого старта» с созданной нами и загруженной в микроконтроллер программой.

Ретранслируем программу, для этого в командной строке Total Commander пишем «**asm51.exe diod.asm**», нажимаем Enter (рис. 10),

После этого в папке появляется ещё 2 файла: **diod.LST** и **diod.HEX** — в первом из них записывается информация об ошибках, допущенных при составлении программы, а во втором коды для прошивки микроконтроллера.

Выделяем **diod.LST** и нажимаем на клавиатуре **F4**, после чего наблюдаем количество ошибок (рис. 11).

Если ошибок нет, то можно загружать программу в микроконтроллер.

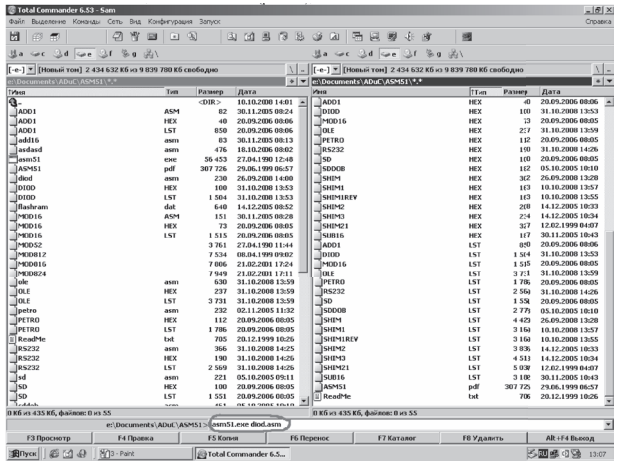


Рисунок 10

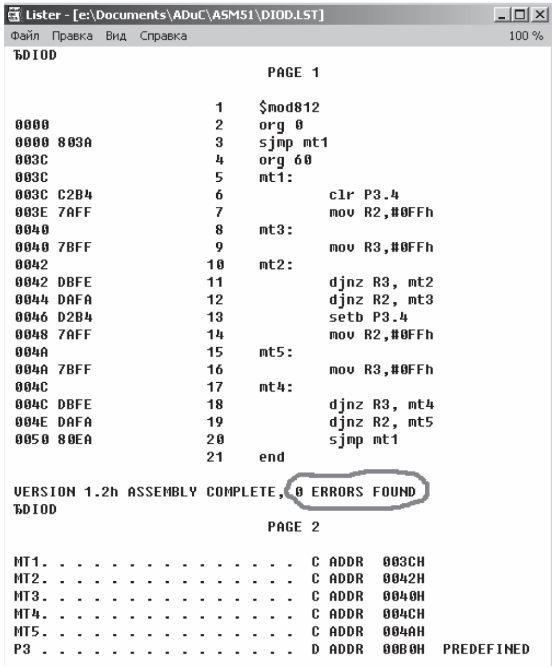


Рисунок 11

руководствоваться указанными физическими способностями МК. Максимальное значение периода ШИМ — 0FFFh, оно постоянно.

Широтно-импульсная модуляция

Широтно-импульсная модуляция (ШИМ, англ. Pulse-width modulation (PWM)) — приближение желаемого сигнала (многоуровневого или непрерывного) к действительным бинарным сигналам (с двумя уровнями — вкл/выкл), так, что, в среднем, за некоторый отрезок времени, их значения равны. Формально, это можно записать так:

$$\int_{t_1}^{t_2} x(t)dt = \sum A^*\Delta T_i$$

где x(t) — желаемый входной сигнал в пределе от t_1 до t_2, а ΔT_i — продолжительность i -го ШИМ импульса, каждого с амплитудой A. ΔT_i подбирается таким образом, что суммарные площади (энергии) обеих величин приблизительно равны за достаточно продолжительный промежуток времени, равны так же и средние значения величин за период:

$$\frac{\int_{t_1}^{t_2} x(t)dt}{t_2 - t_1} = \frac{\sum A^*\Delta T_i}{t_2 - t_1}$$

Управляемыми «уровнями», как правило, являются параметры питания силовой установки, например, напряжение импульсных преобразователей /регуляторов постоянного напряжения/или скорость электродвигателя. Для импульсных источников x(t)= Uconst стабилизации.

ШИП — широтно-импульсный преобразователь, генерирующий ШИМ-сигнал по заданному значению управляющего напряжения. Основное достоинство ШИМ — высокий КПД его уси-

Лабораторная работа №3

ФОРМИРОВАНИЕ ШИРОТНО-ИМПУЛЬСНОГО СИГНАЛА ДЛЯ УПРАВЛЕНИЯ ЕЛЕКТРОДВИГАТЕЛЕМ С ИСПОЛЬЗОВАНИЕМ МИКРОКОНТРОЛЛЕРА ADUC812.

Тема: Формирование широтно-импульсного сигнала с использованием микроконтроллера ADuC812.

Цель работы: Получение практических навыков по формированию ШИМ-сигнала от микроконтроллера ADuC812 и его регулированию.

Оборудование: плата «быстрого старта» микроконтроллера ADuC812, осциллограф, персональный компьютер, пакет прикладных программ «ADuC», соединительные провода, электромеханический стенд с устройствами согласования.

Общие теоретические сведения

Используемый ADuC 812 содержит 12-разрядный АЦП и ЦАП, поэтому при формировании ШИМ-сигнала (см.рис.1) необходимо

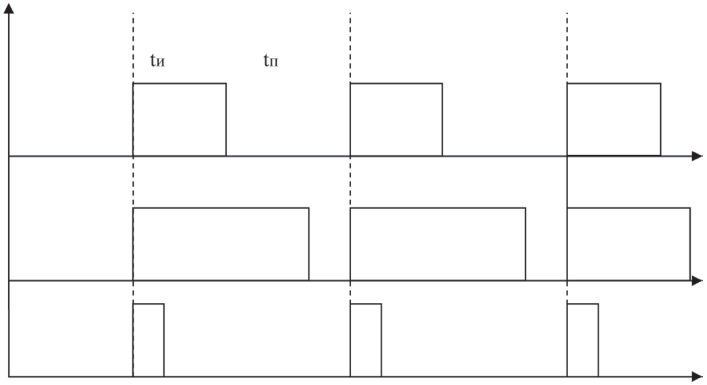


Рисунок 1 — Пример ШИМ-сигнала

Подсоединим плату «быстрого старта» микроконтроллера ADuC812 к COM-порту ПК, с помощью интерфейсного кабеля.

В меню «Пуск», «Программы» находим пакет «ADuC», выбираем приложение «WSD» (рис. 12).

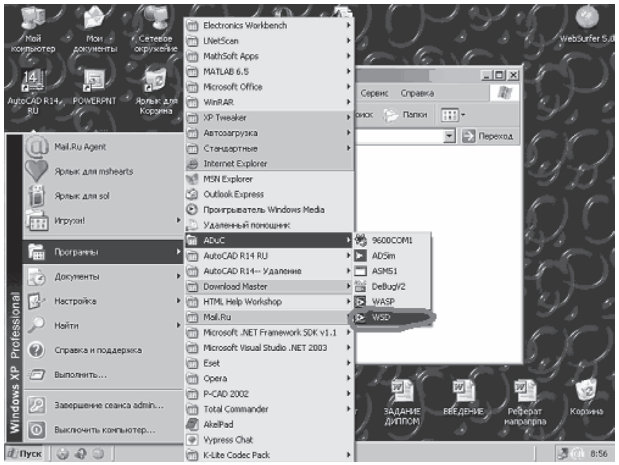


Рисунок 12 — Запуск приложения «WSD»

Далее делаем перезагрузку микроконтроллера: в появившемся окне (рис. 13) нажимаем кнопку «Reset», причём необходимо обратить внимание на положение ключа на плате «быстрого старта» — ребро ключа должно совпадать с зелёной точкой, а не с красной.

Если положение ключа правильное, а сброс не выполняется, то необходимо нажать кнопку на плате, находящуюся возле ключа, и снова «Reset».

Об удачном выполнении перезагрузки свидетельствует запись, показанная на рис. 13.

Теперь нужно загрузить в микроконтроллер программу управления светодиодом. Нажимаем кнопку «Download», выбираем в выпавшем окошке папку «ASM51», выделяем транслированный файл

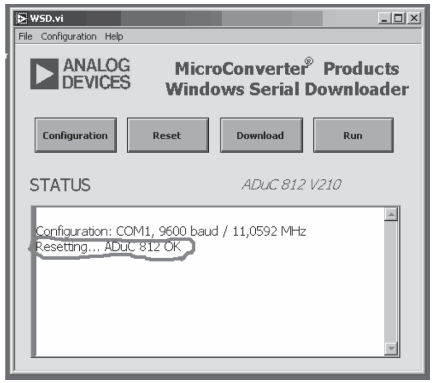


Рисунок 13 — Сброс микроконтроллера

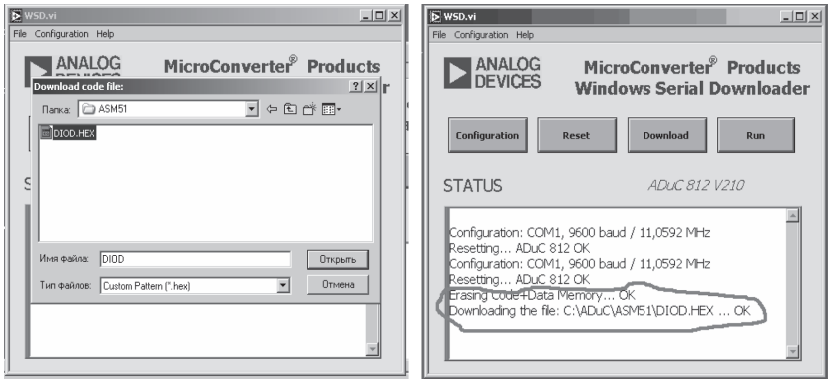


Рисунок 14 — Загрузка файла программы управления

нашей программы с расширением HEX. Нажимаем «Открыть» — программа загружена (рис. 14),

Нажатием кнопки «Run» включаем микроконтроллер на отработку программы управления. Ее реализацию можно увидеть, наблюдая за светодиодом микросхемы.

Меняя значения содержимого регистров R2, R3 наблюдаем за изменением длительности засвеченного и незасвеченного состояний светодиода.

Абсолютная погрешность определяется как модуль разности между результатом измерения и истинным (действительным) значением измеренной величины:

$$\Delta = \left| X_д - X_в \right|$$

Абсолютная погрешность имеет размерность измеренной величины, т.е. выражается в единицах измеренной величины.

Относительная погрешность определяется как отношение абсолютной погрешности к действительному значению измеренной величины :

$$\mathrm{d} = \frac{\Delta}{X_д} 100\%$$

Относительная погрешность выражается в относительных единицах или в процентах.

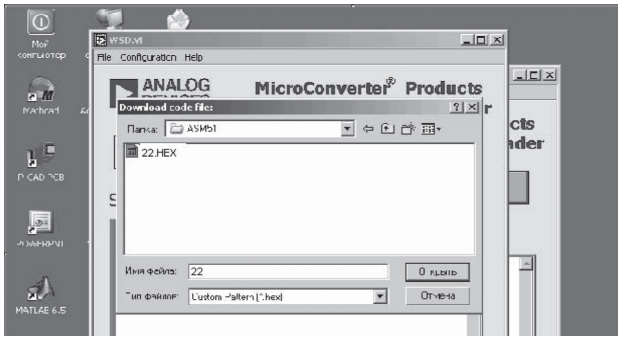


Рисунок 16

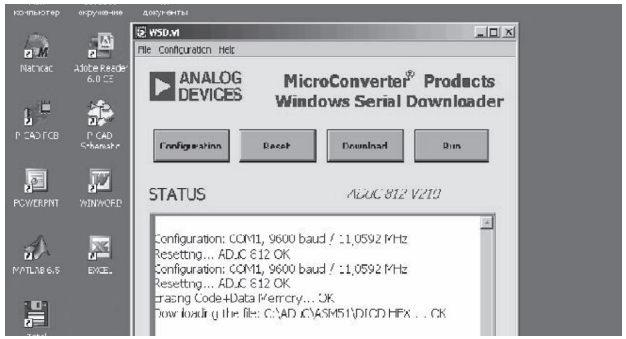


Рисунок 17

А также подключить вольтметр в режиме измерения постоянного напряжения к выходам ЦАП№1 микроконтроллера, так же при нажатой и отпущенной до конца педали. Посчитать погрешность измерений.

Данные измерений занести в таблицу.

Таблица 13. Данные измерений

Показание вольтметров	Педаль отпущена полностью	Педаль нажата полностью	Погрешность измерений
Показание вольтметра V1, В.			
Показание вольтметра V2, В.			

Лабораторная работа №2

ПРЕОБРАЗОВАНИЕ АНАЛОГОВОЙ ИНФОРМАЦИИ В ЦИФРОВУЮ С ПОМОЩЬЮ АЦП И ПРЕОБРАЗОВАНИЕ ЦИФРОВОЙ ИНФОРМАЦИИ В АНАЛОГОВУЮ ЧЕРЕЗ ЦАП

Цель: Получить практические навыки по работе с АЦП и ЦАП с применением микроконтроллера ADuC812.

Задание: Проверить функционирование каналов АЦП и ЦАП на примере микроконтроллера ADuC812.

Приборы и материалы: микроконтроллер ADuC812, персональный компьютер, пакет прикладных программ «ADuC», функциональный узел «электронная педаль», соединительные провода.

Общие сведения

Микроконтроллер ADuC812 состоит из:

1. ПАМЯТЬ.

- 8Кбайт FLASH памяти программ;
- 640 байт FLASH памяти данных;
- внутренний источник программирования типа «зарядовый насос» (внешний источник не нужно);
- 256 байт внутренней памяти данных;
- 16 Мбайт адресного пространства внешней памяти данных.

2. ПИТАНИЕ.

- Допускает напряжение питания 3V или 5V;
- режимы: нормальный, холостой и очередной.

3. ВСТРОЕННАЯ ПЕРИФЕРИЯ.

- Последовательный UART;

- 2-ведущий (I2C) и SP;
- сторожевой таймер (WDT);
- монитор источника питания.

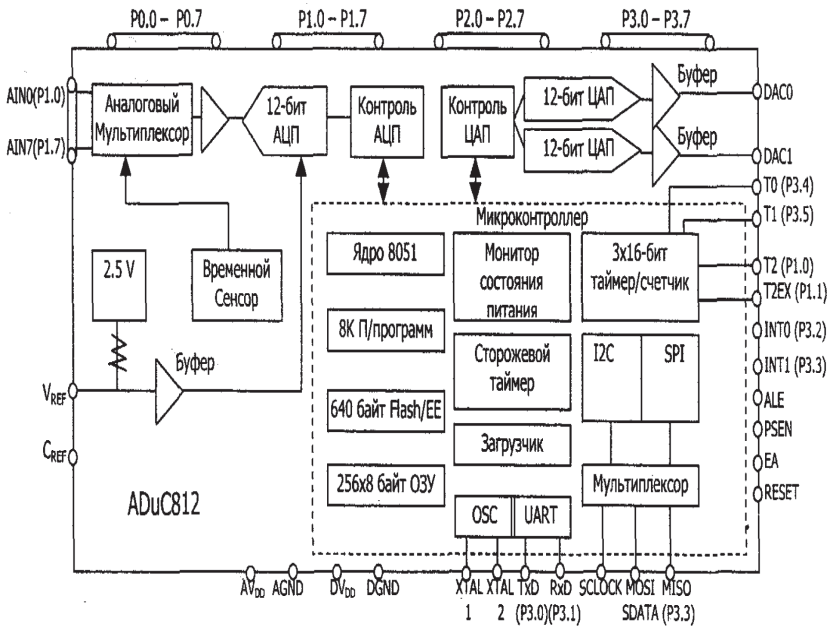


Рисунок 1 — Функциональная блок-схема микроконтроллера ADuc812

Функциональная блок-схема микроконтроллера ADuC812 представлена ниже на рисунке 1. А также ниже показано расположение контактов микроконтроллера ADuC812 в таблице 1.

Таблица 1 — Расположение контактов микроконтроллера Aduc812

№	Наименование контакта	№	Наименование контакта
1	P1.0/ADC0/T2	27	SDATA/MOSI
2	P1.1/ADC1/T2EX	28	P2.0/A8/A16

Для этого, в меню ПУСК, в списке программ выбрать ADuC и далее в открывшемся меню выбрать WSD. (см. рисунок 14).

Далее делаем перезагрузку микроконтроллера: в появившемся окне (рисунок 15), нажимаем кнопку «Reset».

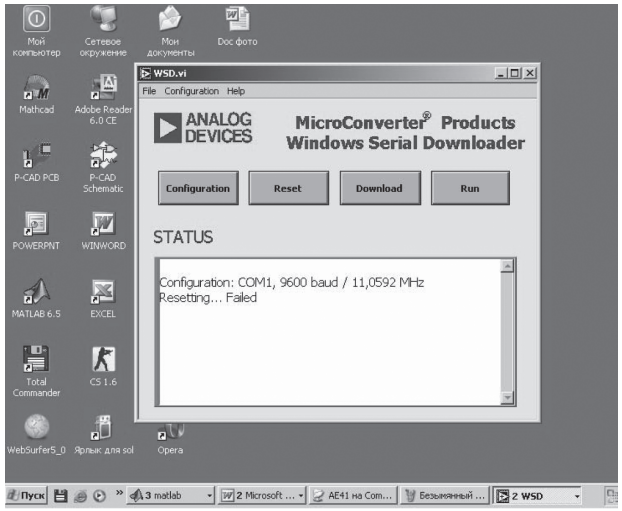


Рисунок 15

Загрузить в буферную память программатора данные из подготовленного ранее файла. Для этого нужно. Нажать на клавишу Download, далее появится диалоговое окно, как на рисунке 16. Выбрать свой скомпилированный файл с расширением (.hex), находящийся в папке ASM51 из директории ADuC, и нажать на кнопку открыть. Нажатием кнопки «Run» включаем микроконтроллер на отработку программы управления (рисунок 17).

Программа загружена в микроконтроллер.

Далее нужно провести измерения напряжения, подключая вольтметр к выходу электронной педали, при отпущенной педали и при максимально нажатой.

ADC0-ADC7 — аналоговые входы. 8 однофазных входов.

Выбор канала осуществляется через регистр ADCCON2;

AGND — аналоговая земля. Общая точка для аналоговых соединений.

На рисунке 10 обозначены: V1 — вольтметр подключаемый к выходу электронной педали; V2 — вольтметр подключаемый к выходу ЦАП микроконтроллера ADuC 812.

Для этой лабораторной работы необходимо соединить между собой одноименные выводы разъемов J9 и J14 ADC0, AGND, DAC0. (J9.1 соединяем с J14.1 и J9.17 соединяем с J14.21).

После того, как схема будет собрана, необходимо подать на нее питание. Затем необходимо загрузить ранее подготовленную программу в микроконтроллер.

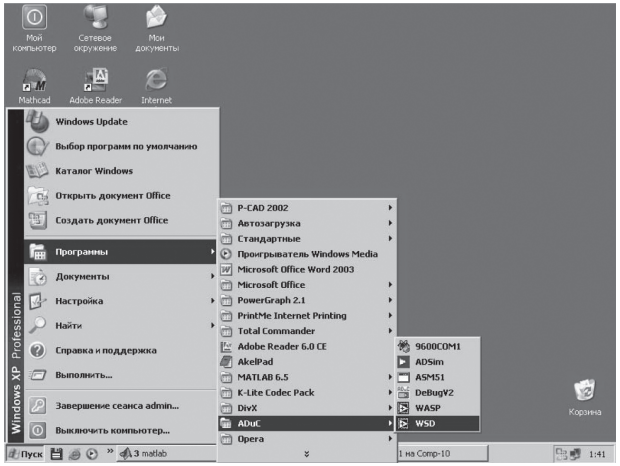


Рисунок 14

Подготовить HEX-файл с исходными данными для программирования.

Запустить программу, обслуживающую программатор.

Продолжение таблицы 1 — Расположение контактов микроконтроллера Aduc812

№	Наименование контакта	№	Наименование контакта
3	P1.2/ADC2	29	P2.1/A9/A17
4	P1.3/ADC3	30	P2.2/A10/A18
5	Avdd	31	P2.3/A11/A19
6	AGND	32	XTAL1 (in)
7	Cref	33	XTA2 (out)
8	Vref	34	DVdd
9	DAC0	35	DGND
10	DAC1	36	P2.4/A12/A20
11	P1.4/ADC4	37	P2.5/A13/A21
12	P1.5/ADC5/SS/	38	P2.6/A14/A22
13	P1.6/ADC6	39	P2.7/A15/A23
14	P1.7/ADC7	40	EA//Vpp
15	RESET	41	PSEN/
16	P3.0/RxD	42	ALE
17	P3.1/TxD	43	P0.0/AD0
18	P3.2/INT0/	44	P0.1/AD1
19	P3.3/INT1//MIS0	45	P0.2/AD2
20	DVdd	46	P0.3/AD3
21	DGND	47	DGND
22	P3.4/T0	48	DVdd
23	P3.5/T1/CONVST/	49	P0.4/AD4
24	P3.6/WR/	50	P0.5/AD5
25	P3.7/RD/	51	P0.6/AD6
26	SCLOCK	52	P0.7/AD7

Порядок выполнения работы.

1.Подготовка к инициализации регистров специальных функций ЦАП и АЦП.

Для того, чтобы проверить функционирование каналов АЦП и ЦАП микроконтроллера ADuc812 необходимо составить управляющую программу, с помощью которой собственно это и удастся проделать.

На рисунке 1, показана функциональная блок-схема микроконтроллера ADuc812, из которой можно определить, что архитектура контроллера содержит два цифро-аналоговых преобразователя ЦАП№0 и ЦАП№1 и 8 канальный аналого-цифровой преобразователь, который конфигурируется с помощью регистров специальных функций ADCCON1, ADCCON2, ADCCON3.

Чтобы начать составлять управляющую программу, сначала необходимо выбрать какой канал ЦАП будем проверять. Зададимся выбором канала ЦАП№1. Следующим этапом необходимо провести инициализацию регистров специальных функций АЦП и ЦАП, собственно, что сейчас и будем делать.

В таблице 2 представлено размещение битов регистра ЦАП микроконтроллера ADuc812.

Таблица 2 — Размещение битов регистра SFR-DACCON

Размещение битов	Мнемоника	Описание
DACCON.7	MODE	Бит устанавливает режим работы обоих ЦАП. Если=1, то 8-битный запись (запись 8-ми битов у DACxSFR) Если=0, то 12-битный.
DACCON.6	RNG1	Бит выбора диапазона АЦП1. Если=1, то диапазон ЦАП1 0...Vdd. Если=0, то диапазон ЦАП1 0...Vref.
DACCON.5	RNG0	Бит выбора диапазона ЦАП0. Если=1, то диапазон ЦАП0 0...Vdd. Если=0, то диапазон ЦАП0 0...Vref.

Ниже приведены таблицы функционального назначение выводов разъемов, которые могут понадобится лишь в тех случаях когда выводы подписаны, или для получения дополнительной информации по лабораторной работе.

Таблица 11. Функциональное назначение выводов разъема J9

Вывод	Функциональное назначение	Вывод	Функциональное назначение
1	ADC0	12	AGND
2	AGND	13	ADC6
3	ADC1	14	AGND
4	AGND	15	ADC7
5	ADC2	16	AGND
6	AGND	17	DAC0
7	ADC3	18	AGND
8	AGND	19	DAC1
9	ADC4	20	AGND
10	AGND	21	VREFIN
11	ADC5	22	AGND

Таблица 12. Функциональное назначение выводов разъема J14

Вывод	Функциональное назначение	Вывод	Функциональное назначение
1	ADC0	12	AGND
2	AGND	13	ADC6
3	ADC1	14	AGND
4	AGND	15	ADC7
5	ADC2	16	AGND
6	AGND	17	VREFIN
7	ADC3	18	AGN0
8	AGND	19	DAC1
9	ADC4	20	AGND
10	AGND	21	DAC0
11	ADC5	22	AGND

где: DAC0 — выходное напряжение с ЦАП№0;

DAC1 — выходное напряжение с ЦАП№1;

Подключим 2 шлейфа идущие от микроконтроллера ADuC и двухпроводный кабель (белый и красный провод) идущие от микроконтроллера ADuC (рис13.1) к макету (рис. 13.2)

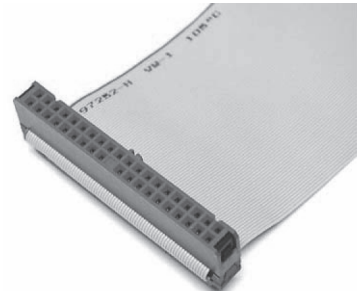


Рисунок 13.1

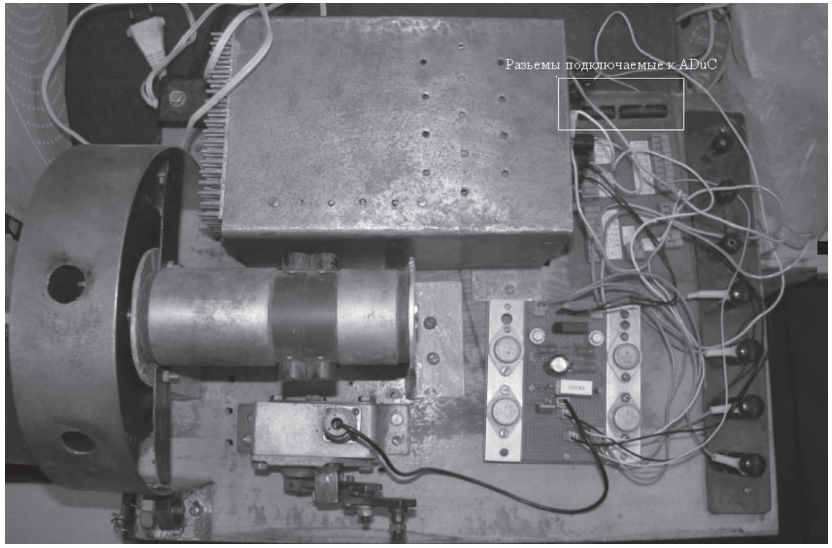


Рисунок 13.2

Внимание! Все подключения нужно проводить при непосредственном руководстве преподавателя.

Продолжение таблицы 2 — Размещение битов регистра SFR-DACCON

Размещение битов	Мнемоника	Описание
DACCON.4	CLR1	Бит стирания ЦАП1. Если=1, то выход ЦАП1 соответствует коду. Если=0, то выход ЦАП1=0В.
DACCON.3	CLR0	Бит стирания ЦАП0. Если=1, то выход ЦАП0 соответствует коду. Если=0, то выход ЦАП0=0В.
DACCON.2	SYNC	Бит синхронизации ЦАП0/1. Если=1 ,то выходы ЦАПов изменяются сразу ,как только данные поступают в регистры DACxSFRs. Пользователь может сразу обновить выходы обоих ЦАПов путем предварительной записи данных у DACx/H при SYNC=0. Выходы обоих ЦАПов сразу обновятся теперь при установке SYNC=1.
DACCON.1	PD1	Бит включения ЦАП1. Если=1, то ЦАП1 включено. Если=0, то ЦАП1 выключено.
DACCON.0	PD0	Бит включения ЦАП1. Если=1, то ЦАП0, включено. Если=0, то ЦАП0 выключено.

Следовательно, чтобы включить выбранный нами ЦАП№1, нужно взвести некоторые биты регистра DACCON как показано в таблице 3.

Таблица 3 — Составление управляющего слова для работы регистра DACCON

MODE	RNG1	RNG0	CLR1	CLR0	SYNC	PD1	PD0
0	0	0	1	0	1	1	0

Полученное управляющее слово 00010110 необходимо перевести в шестнадчатиричную систему исчисления, и согласно таблице 10 это будет число 16h. Командой mov DACCON,#16h полученная

информация 16h будет запрограммирована через регистр DACCON и соответственно будет включен в работу ЦАП№1. В таблице 4 представлено размещение битов регистра ADCCON1 микроконтроллера ADuc812. Регистр ADCCON1 управляет преобразованием, временем переключения, режимами преобразования и энергопотреблением устройства.Следовательно, для того, чтобы выбрать нормальный режим работы АЦП необходимо составить управляющее слово. Для этого нужно взвести 7 бит регистра ADCCON1 в единицу, как показано в таблице 5, а остальные биты, будут не задействованы.

Таблица 4 — Распределение битов SFR-регистра ADCCON1

Размещение битов	Мнемоника	Описание
ADCCON1.7 ADCCON1.6	MD1 MD0	(MD0 MD1) биты режима выбирают режимы работы АЦП в такой способ: MD1 MD0 Режим АЦП 0 0 Дежурный 0 1 Нормальный. 1 0 Дежурный, если не исполняется цикл преобразований. 1 1 Холостой, если не исполняется цикл преобразований.
ADCCON1.5 ADCCON1.4	CK1 CKO	Биты деления тактовой частоты, выбирают коэффициент деления основной частоты микропроцессора для получения тактовой частоты АЦП. Цикл преобразования АЦП занимает 16 тактов, на приложение числа тактов переключения. Коэффициент выбирается согласно: СК1 СК0 Делитель для MCLK 0 0 1 0 1 2 1 0 4 1 1 8

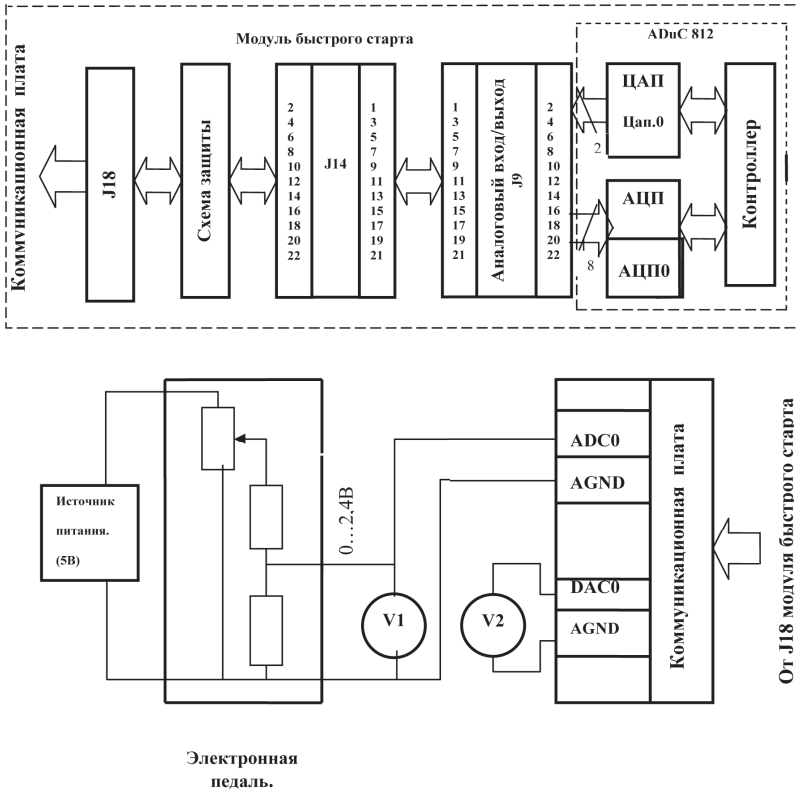


Рисунок 11 — Функциональная схема лабораторной работы

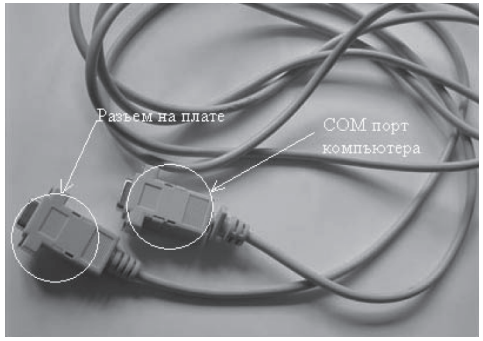


Рисунок 12

ями .obj, out, cod и некоторыми другими, однако их форматы разнообразны и сложны.

Рисунок 10

Обобщенная схема процедуры программирования микроконтроллера разделяется на 4 этапа (рис. 10).

Вначале в текстовом редакторе набирается исходный листинг программы, после чего полученному файлу приписывается расширение «.asm».

Далее запускается программа-компилятор, которая преобразует этот файл в HEX–коды прошивки. Завершает процедуру запуск программы обслуживания контроллера ADuC812. В результате работы которой, собственно и прошивается ПЗУ микроконтроллера.

4. Собрать функциональную схему как показано на рисунке 11.

Внимание! Все подключения нужно проводить при непосредственном руководстве преподавателя.

Устанавливаем напряжение на вольтметре 2,5 В. Измерять выходное напряжение нужно с ЦАП№1, потому что ЦАП№0 физически не подключён. Поэтому «-« вольтметра подключаем к «земле», а «+» к выходу ЦАП№1. Тоже самое проделываем с АЦП№0.

Перед началом работы нужно сделать следующие присоединения. Для этого нужно подключить интерфейсный провод с одной стороны к разъему COM-порт компьютера, а другим концом соединить с платой быстрого старта (рис.12).

Продолжение таблицы 4 — Распределение битов SFR-регистра ADCCON1

Размещение битов	Мнемоника	Описание
ADCCON1.3 ADCCON1.2	AQ1 AQ0	Биты задержки переключения, выбирают время, необходимое для перезарядки УВХ при переключении мультиплексора: AQ1 AQ0 Число тактов задержки АЦП 0 0 1 0 1 2 1 0 3 1 1 4
ADCCON1.1 ADCCON1.0	T2C EXC	Бит запуска преобразования от Таймера 2. Если бит установлен, то сигнал переполнения Таймера 2 используется для запуска АЦП. Бит разрешения внешнего запуску. Если установлен, то контакт 23 (CONVST/) будет использоваться как сигнал запуска.

Таблица 5 — Составление управляющего слова для работы регистра ADCCON1

MD1	MD0	CK1	CK0	AQ1	AQ0	T2C	EXC
0	1	0	0	0	0	0	0

Полученное управляющее слово 01000000 необходимо перевести в шестнадцатеричную систему исчисления, и согласно таблице 10, это будет 40h.

Командой **mov ADCCON1,#40h** полученная информация, будет запрограммирована через регистр ADCCON1 и соответственно АЦП будет работать в нормальном режиме.

В таблице 6 представлено размещение битов регистра ADCCON2 микроконтроллера ADuc812.

Регистр ADCCON2 управляет выбором номера канала и режимами преобразования.

Этот регистр в данной программе не используется. По этому, при составлении управляющего слова, все биты регистра ADCCON2 необходимо записать в состояние все нули, как показано в таблице 7.

Таблица 6 — Распределение битов SFR-регистра ADCCON2

Размещение битов	Мнемоника	Описание
ADCCON2.7	ADC1	Бит прерываний АЦП устанавливается аппаратно после окончания однократного цикла преобразования АЦП или после окончания передачи блока в режиме КПД. ADC1 очищается аппаратно при переходе по вектору на Процедуру Обслуживания Прерываний.
ADCCON2.6	DMA	Бит разрешения режима КПД. Устанавливается пользователем для начала операции КПД со стороны АЦП.
ADCCON2.5	CCONV	Бит циклического преобразования. Устанавливается пользователем для установки АЦП в режим безостановочного циклического преобразования. В этом режиме АЦП выполняет преобразование в соответствии с типом синхронизации и конфигурацией каналов, выбранных в других SFR.
ADCCON2.4	SCONV	Бит запуска однократного преобразования. Устанавливается пользователем для однократного запуска АЦП. Бит сбрасывается автоматически после завершения преобразований.
ADCCON2.3 ADCCON2.2 ADCCON2.1 ADCCON2.0	CS3 CS2 CS1 CS0	Биты выбора входных каналов (CS3... CS0). Разрешают пользователю выполнять выбор номера канала АЦП под управлением программы. Преобразование будет вырабатываться для канала, номер которого определен битами. В режиме КПД выбор номера канала вырабатываются с 10 канала, записанного во внешней памяти. CS3 CS2 CS1 CS0 CH# 0 n2 n1 n0 Номер входного канала (n2 n1 n0) 1 0 0 0 Температурный сенсор. (внутренний) 1 X X X Другие комбинации. 1 1 1 1 Останов КПД.

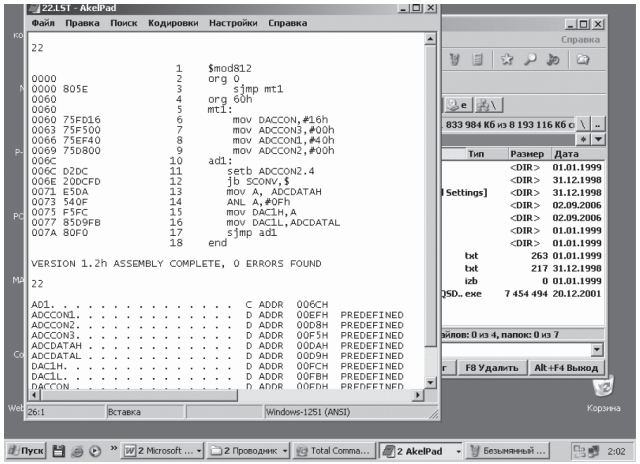


Рисунок 9

Первоначально HEX–формат предназначался для хранения восьмиразрядных данных. Коды большей разрядности разбивают на соответствующее число байт, которые записываются в файл в порядке от младшего к старшему.

HEX-файлы — конечный продукт ассемблеров, трансляторов языков высокого уровня и других средств разработки программ. Однако, на разных этапах своей работы все эти программные средства создают и используют большое число других файлов, содержащих промежуточные результаты трансляции, данные о размещении памяти микро контроллера программы в целом, ее фрагментов и переменных, об установленных (в том числе по умолчанию режимах трансляции, отчеты о ходе работы и много другой полезной информации).

Кроме файла с расширением (.lst)- полного текстового отчета о выполнении трансляции программы, больше всего информации о ней содержит так называемые «объектные файлы» с расширени-

Ретранслируем программу, для этого в командной строке Total Commander пишем »asm51.exe /имя файла/.asm», нажимаем Enter (рис. 8),после того, как нажали на клавишу enter, идет трансляция программы.

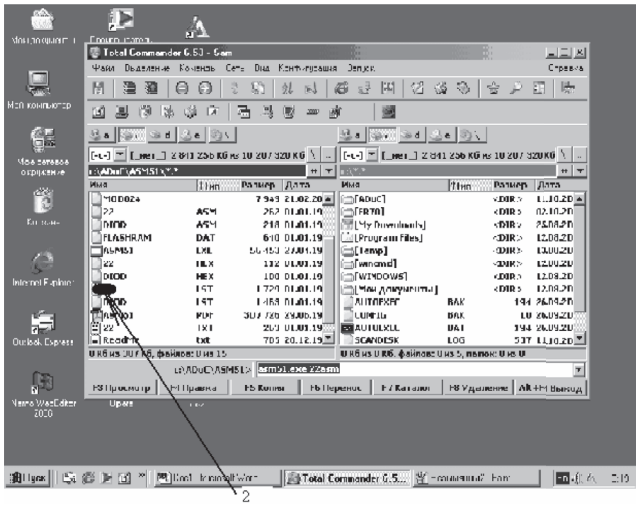


Рисунок 8

После этого находим в папке ASM51 имя своего файла, но с расширением LST (2),выделяем его и нажимаем клавишу F4. Появится диалоговое окно, как показано на (рис. 9).

Если не было допущено ошибок, то на экране монитора должны появится сообщения. Если будут допущены ошибки, то их можно посмотреть во вновь созданном файле с расширением (.lst). Попутно в ту же директорию будет записан файл с HEX-кодами прошивки микроконтроллера под названием /имя файла/.hex.

Теперь, имея HEX-файл, можно с помощью программатора занести коды прошивки в ПЗУ микроконтроллера.

Таблица 7 — Составление управляющего слова для работы регистра ADCCON2

ADC	DMA	CCONV	SCONV	CS3	CS2	CS1	CS0
0	0	0	0	0	0	0	0

Полученное управляющее слово 00000000 необходимо перевести в шестнадцатеричную систему исчисления, и согласно таблице 10, это будет 00h.

Командой **mov ADCCON2,#00h** полученная информация, будет запрограммирована через регистр ADCCON2 и соответственно этот регистр не содержит взведенных битов, так как они находится в состоянии все нули.

В таблице 8 представлено размещение битов регистра ADCCON3 микроконтроллера ADuc812.

Регистр ADCCON3 выдает индикацию занятости АЦП для прикладных программ.

Этот регистр в данной программе не используется. По этому, при составлении управляющего слова, все биты регистра ADCCON3 необходимо взвести в состояние все нули, как показано в таблице 9.

Таблица 8 — Распределение битов SFR-регистров

Размещение битов	Мнемоника	Описание
ADCCON3.7 ADCCON3.6 ADCCON3.5 ADCCON3.4 ADCCON3.3 ADCCON3.2 ADCCON3.1 ADCCON3.0	BUSY RSVD RSVD RSVD RSVD RSVD RSVD RSVD	Биты занятости АЦП только для чтения. Устанавливаются на время или преобразования калибровки АЦП. Автоматически снимаются после завершения циклов или преобразований калибровки. Биты ADCCON3.0-ADCCON3.6 — резервные. Эти биты читаются нулями, их следует записывать только нулями.

Таблица 9 — Составление управляющего слова для работы регистра ADCCON3

BUSY	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
0	0	0	0	0	0	0	0

Полученное управляющее слово 00000000 необходимо перевести в шестнадцатеричную систему исчисления, и согласно таблице 10, это будет 00h.

Командой **mov ADCCON3,#00h** полученная информация, будет запрограммирована через регистр ADCCON3 и соответственно этот регистр не содержит взведенных битов, так как они находится в состоянии все нули.

После того, как АЦП сконфигурировано с помощью регистров ADCCON 1-3, он начнет преобразовывать аналоговые входные сигналы и выдавать 12 битные выходные коды в регистр ADCDATAH(L). В четырех верхних битах регистра ADCDATAH будет записываться код выбору канала результата.

Таблица 10 — Системы исчисления

10	2	8	16
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Чтобы компилятор программы корректно воспринимал название регистров специального назначения, общих регистров, которыми мы оперируем в программе, необходимо в самом начале текста

(рис. 6), изменяем расширение файла, для чего выделяем сохраненный нами asm51.asm.txt, нажимаем на него левой кнопкой мышки

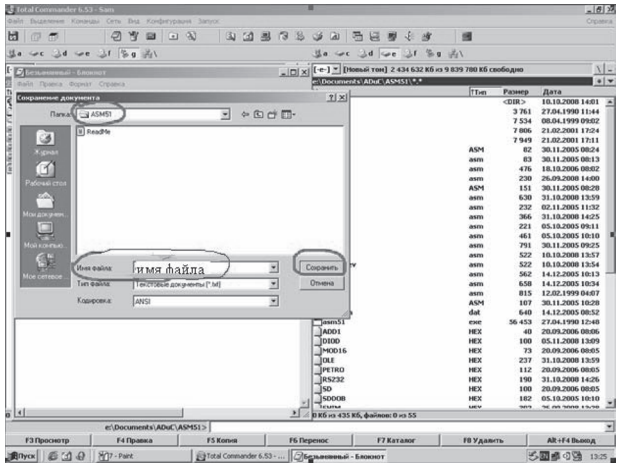


Рисунок 6

и удаляем «.txt», жмём Enter (рис.7), перед именем файла пробел обязателен.

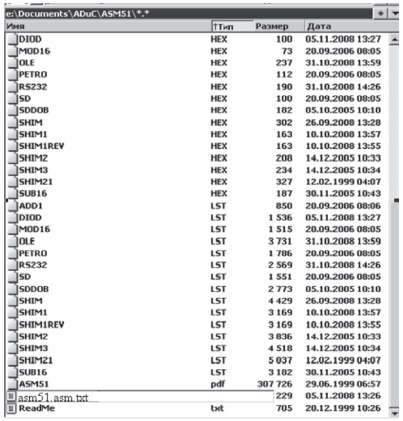


Рисунок 7

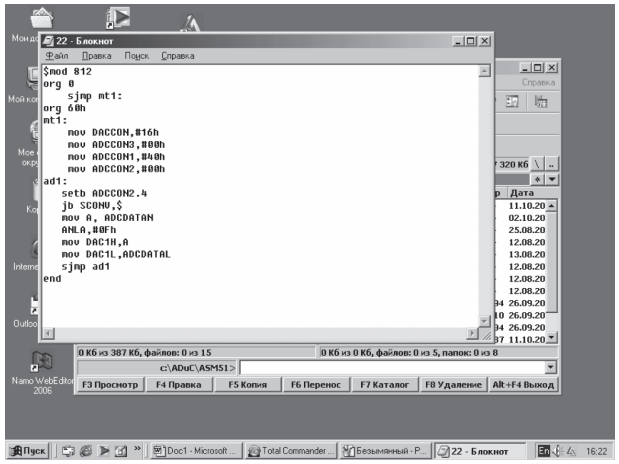


Рисунок 4

текст программы из листинга 1, в любом текстовом редакторе, DOS, Windows, значения не имеет. Воспользуемся блокнотом.

Сохраняем полученный результат, выполняя команду Файл→ Сохранить как... (рис. 5), указываем путь к своей папке ASM51 (с файлом asm51.exe), записываем имя и нажимаем кнопку Сохранить

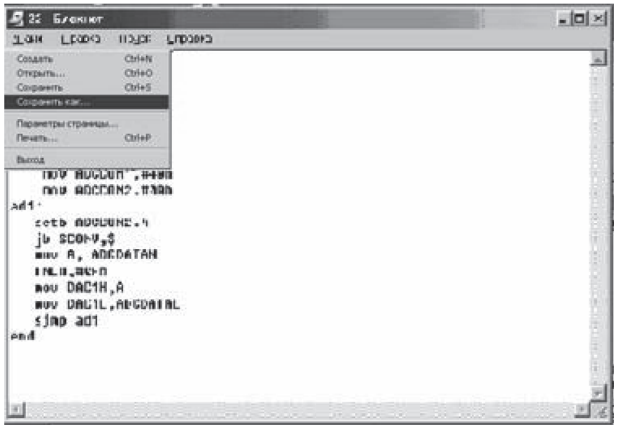


Рисунок 5

программы подключить файл mod812, в котором каждому имени регистра назначаем определенный адрес.

Промежуток в памяти программы от org 0 до org 60 в контроллере ADuC 812 занимают вектора прерываний. Поэтому их нужно пропустить.

Оператор $ в рассматриваемой программе — это оператор зацикливания на себя.

Командой **setb ADCCON2.4** устанавливается бит запуска однократного прерывания в 1. Необходимо для однократного запуска АЦП. Бит сбрасывается после окончания преобразования.

Командой mov DAC1H,A происходит пересылка содержимого в старший регистр ЦАП№1.

Jb SCONV,$ — команда зацикливания на саму себя до выполнения условия.

ANLA,#0Fh — команда логическое «И».

Командой **mov A,ADCDATAH** происходит пересылка из старшего информационного байта АЦП в аккумулятор.

mov DAC1L, mov DAC1L, ADCDATAL — загрузка содержимого информационных регистров ЦАП информации о значении измеряемого напряжения.

2. Написание программы в любом текстовом редакторе из листинга 1.

Листинг 1. Это подпрограмма предназначена для чтения информации через АЦП,конечным результатом которой будет выдача этой же информации через ЦАП.

$mod812	; подключение файла $mod812
org 0h	; определение адреса последующей команды
sjmp mt1:	; скачек на mt1 вызван тем, что запрещено занимать зону

```
org 60h          ; отведенную для обслуживания прерываний (60 ячеек)
mt1:             ; метка инициализации регистров ЦАП и АЦП
    mov DACCON,#16h      ; включение ЦАП№1 микроконтроллера
    mov ADCCON3,#00h     ; этот регистр не используется в данном случае
    mov ADCCON1,#40      ; включение нормального режима работы контроллера
    mov ADCCON2,#00h     ; этот регистр не используется в данном случае
ad1:             ; метка включения однократного преобразования 7-го бита АЦП
setb ADCCON2.4   ; включение бита запуска однократного прерывания
jb SCONV,$       ; зацикливание на саму себя до выполнения условия
    mov A, ADCDATAH      ; пересылка из старшего информационного байта в аккумулятор
ANL A,#0Fh       ; использование команды логическое «И»
    mov DAC1H,A  ; пересылка содержимого в старший регистр ЦАП№1
    mov DAC1L, ADCDATAL  ; загрузка содержимого в информационные регистры ЦАП
sjmp ad1         ; переход на метку ad1
end              ; завершение программы
```

3. Процедура компиляции программы.

1. Чтобы составить программу, необходимо:

Открыть Total Commander (рис.2), Найти на одном из жестких

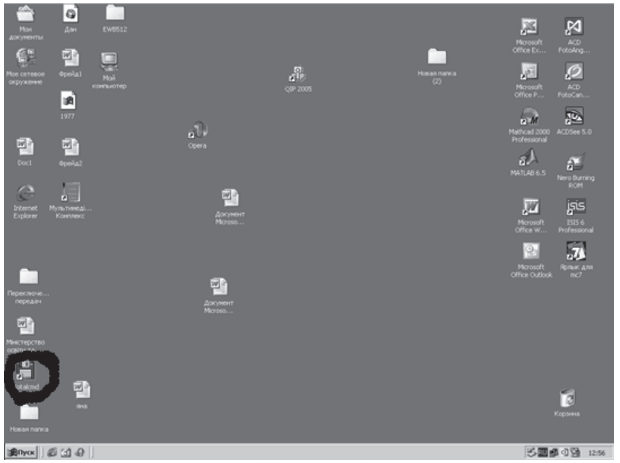


Рисунок 2

дисков компьютера папку с названием ADuC. Открываем ее. Далее находим ASM51 и входим в эту папку(рис.3) и создаем там текстовый документ в редакторе Блокнот (рис.4), Необходимо набрать

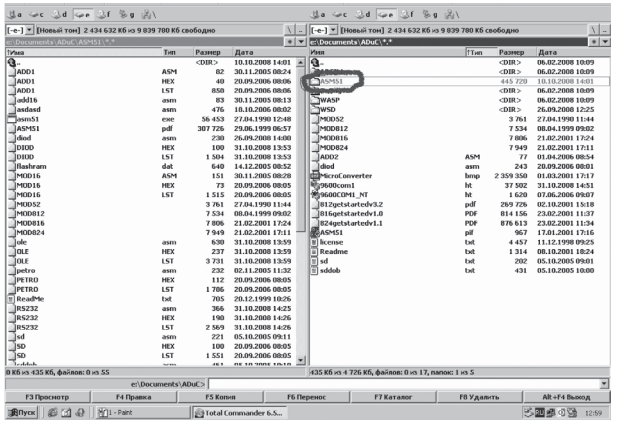


Рисунок 3