Лабораторная работа 1
Исследование программного ввода аналогового сигнала в ПЭВМ

Цель работы: получение навыков ввода аналогового сигнала в ПЭВМ с использованием платы аналогового ввода-вывода.

Основные вопросы, изучаемые в работе:

- структура и состав платы аналогового ввода-вывода L-154 фирмы L-Card;

- команды языка ассемблера, используемые для обращения к плате ввода-вывода;

- написание программы на языке ассемблер, осуществляющей ввод аналогового сигнала в ПЭВМ.

Описание платы L-154 фирмы L-Card

Плата L-154 является быстродействующим и надежным устройством для ввода и вывода аналоговой и цифровой информации в персональных IBM совместимых компьютерах. Данная плата предназначена для преобразования аналоговых сигналов в цифровую форму для ввода в ЭВМ, управления одним выходным аналоговым каналом (цифроаналоговый преобразователь), а также для ввода/вывода цифровых ТТЛ линий.

Стандартный комплект поставки позволяет:

- осуществлять многоканальный ввод с аналоговых каналов с частотой до 70 кГц на канал.

- осуществлять асинхронный ввод с различных аналоговых каналов

- управлять цифроаналоговым преобразователем.

- осуществлять работу в двух режимах: в программном режиме и в режиме генерации прерываний IRQ.

- управлять цифровыми линиями в асинхронном режиме.

Технические данные

Аналого - цифровой преобразователь (АЦП)

На плате имеется один АЦП, на вход которого с помощью коммутатора может быть подан один из 16 (32) аналоговых каналов с внешнего разъёма платы. Параметры АЦП приведены в табл. 1.1.

Таблица 1.1

Параметры АЦП

Количество каналов
дифференциальных 16
с общей землёй 32
Разрядность
12 бит
Время преобразования
1.7 мкс
Входное сопротивление
2 МОм

Диапазон входного сигнала

±5.12В, ±2.56В, ±1.024В

Максимальная частота преобразования

70 кГц

Интегральная нелинейность преобразования

±0.8 МЗР, макс. ±1.2 МЗР

Дифференц. нелинейность преобразования

±0.5 МЗР, макс. ±0.75

Отсутствие пропуска кодов

гарантировано 12 бит

Смещение нуля

±0.5 МЗР, макс. 1 МЗР

Цифроаналоговый преобразователь (ЦАП)

На плате установлен один ЦАП, который выдаёт постоянное напряжение в соответствии с записанным в него цифровым кодом. Параметры ЦАП приведены в табл. 1.2.

Таблица 1.2

Параметры ЦАП

Количество каналов

1

Разрядность

12 бит

Время установления

10 мкс

Выходной диапазон
±5.12 В

Цифровые входы и выходы

На плате имеется 8 цифровых входных ТТЛ линий и 8 выходных ТТЛ линий, с помощью которых можно управлять внешними устройствами, осуществлять цифровую синхронизацию ввода и т. п. Параметры цифровых линий приведены в табл. 1.3.

Таблица 1.3

Параметры цифровых линий

Входной порт
8 бит ТТЛШ
Выходной порт
8 бит ТТЛШ
Напряжение низкого уровня

мин. 0.0 В

макс. 0.4 В

Напряжение высокого уровня

мин. 2.5 В

макс. 5.0 В

Выходной ток низкого уровня (макс.)

4 мА

Входной ток высокого уровня (макс.)

0.4 мА

Счётчики-таймеры

На плате установлено три счётчика-таймера (одна микросхема 580ВИ53) с кварцевой стабилизацией 1 МГц, с помощью которых осуществляется программная синхронизация ввода и генерирование прерываний IRQ.

Первый и второй счетчики являются 16-битными каскадно соединёнными счётчиками. Счётный вход третьего канала выведен на внешний разъём для возможности внешней синхронизации процессов ввода и вывода.

Программирование платы

Адресное пространство платы

Управление всеми устройствами на плате (АЦП, ЦАП, таймеры, цифровые линии) осуществляется через порты ввода - вывода компьютера. Плата занимает 16 последовательных ячеек в пространстве ввода-вывода компьютера начиная с базового адреса, установленного на плате с помощью перемычек (табл. 1.4).

Таблица 1.4

Описание портов ввода-вывода PC

Адрес порта	Направление	Описание
BASE	Чтение	

(16-бит)

Результат аналого-цифрового преобразования (16 бит)

BASE

Запись

(16-бит)

12-битный код, устанавливаемый на ЦАП (младшие 12 бит)

BASE+2

Запись (8-бит)

Установка номера канала АЦП, режима подключения каналов, коэффициента усиления

BASE+2

Чтение (8-бит)

Вых. трёх счётчиков-таймеров и бит готовн.

BASE+3

Чтение (8-бит)

Значения 8 бит ТТЛ линий с внешнего разъёма

BASE+3

Запись (8-бит)

Установка 8 ТТЛ линий на внешнем разъёме

BASE+4

Запись (8-бит)

Старт аналого-цифрового преобразования

BASE+8

Чтение/Запись
(8-бит)

Программирование таймера A
(микросхема 580ВИ53)

BASE+9

Чтение/Запись
(8-бит)

Программирование таймера B
(микросхема 580ВИ53)

BASE+A

Чтение/Запись
(8-бит)

Программирование таймера C
(микросхема 580ВИ53)

BASE+B

Запись (8-бит)

Регистр управления таймера 580ВИ53

BASE+0xF

Запись (8-бит)

Включение / Отключение линии IRQ от шины компьютера

BASE - базовый адрес платы (в данной работе на плате установлен базовый адрес 310h).

Структура портов платы показана на рис. 1.1.

Рис. 1. 1 Структурная схема платы

Установка напряжения на ЦАП

На плате установлен один цифроаналоговый преобразователь, который устанавливает напряжение на линии “Выход ЦАП” на внешнем разъёме в соответствии с записанным в ЦАП цифровым кодом. На ЦАП передается 16-битный код, но поскольку код ЦАП – 12-битный, то старшие 4 бита не используются. Соответствие 12 - битного кода ЦАП напряжению приведено на рис. 1.2.

Рис. 1.2. Функция преобразования ЦАП

Для установки ЦАП в языке ассемблер используют команду записи в порт:

```
out  dx, ax
```

где dx и ax – 16-разрядные регистры общего назначения. В dx записывают адрес порта ЦАП (BASE), в ax – код, передаваемый на ЦАП (0…4095).

Пример установки кода на ЦАП:

```
mov  dx, 310H      ; адрес установки кода на ЦАП (BASE)

mov  ax, 0

out  dx, ax        ; установим минимальное напряжение

mov  ax, 4095
```

```
out  dx, ax        ; установим максимальное напряжение
```

Использование АЦП

Управление коммутатором и усилением

На плате установлен один АЦП с 32-канальным коммутатором, с помощью которого на вход АЦП подается один из 16- или 32-аналоговых каналов. После переключения номера канала в коммутаторе нельзя сразу давать старт АЦП, необходимо организовать задержку на установление аналогового канала. Эта задержка составляет 4 мкс.

Одновременно с номером аналогового канала осуществляется управление коэффициентом усиления и режимом подключения сигналов (т. е. используется 16-канальный дифференциальный режим или 32-канальный режим с общей землей). Формат байта установки режима АЦП представлен в табл. 1.5. Поле "Коэффициент усиления" (U2-U1) представлено в табл. 1.6. Поле "Номер канала" (C4-C1) в дифференциальном режиме показано в табл. 1.7.

Таблица 1.5

Бит
8
7
6
5

4

3

2

1

Имя

U2

U1

M32

C5

C4

C3

C2

C1

При дифференциальном подключении поле M32 должно быть равно 0 (соответственно бит C5 номера канала C5-C1 также должен быть равен нулю). В 32-канальном режиме бит M32 должен быть равен единице, при этом номер канала задаётся всеми битами C5-C1.

Таблица 1.6

Бит U2

Бит U1

Усиление

Диапазон входного сигнала АЦП

1

1

1

±5.12 В

1

0

2

±2.56 В

0

1

5

±1.024 В

0

0

Зарезервирован

Таблица 1.7

Номер канала АЦП
1
2
3
4
5
6

7
8
9
10
11
12
13
14
15
16

Бит 1

0

1

0

1

0

1

0

1

0

1

0

1

0

1

0

1

0

1

Бит 2

0

0

1

1

0

0

1

1

0

0

1

1

0

0

1

1

Бит 3

0

0

0

0

1

1

1

1

0

0

0

0

1

1

1

1

Бит 4

0

0

0

0

0

0

0

0

1

1

1

1

1

1

1

1

Соответствие кода с АЦП напряжению на входе в зависимости от коэффициента усиления АЦП показано на рис. 1.3.

Рис. 1.3. Функции преобразования АЦП в зависимости
от установленного коэффициента усиления (КУ)

Для установки номера канала режима работы и коэффициента усиления АЦП, в языке ассемблер используют команду записи в порт:

```
out  dx, al
```

где в dx записывают адрес порта номера канала АЦП (BASE+2), в al – код, задающий режим АЦП (табл 1.5).

Старт АЦП и чтение результата

Запуск аналого-цифрового преобразования производится программно путем записи любого числа по адресу BASE+4. Для этого используется команда ассемблера:

```
out  dx, al
```

где в dx записывают адрес порта старта АЦП (BASE+4), в al – любое число.

После запуска АЦП ровно через 1.7 мкс результат преобразования можно будет прочитать по адресу BASE. Команда чтения результата АЦП:

```
in   ax, dx
```

где в dx записывают адрес порта АЦП (BASE), после выполнения команды в регистре ax будет код результата преобразования.

Для проверки окончания цикла преобразования задействован четвёртый бит в порте статуса. Нулевое состояние этого бита означает окончание преобразования.

Пример программы работы с АЦП:

```
; устан. 1-й кан. АЦП, включенный в дифференциальном режиме

mov  dx, 312H      ; адрес порта номера канала (BASE+2)

mov  al, 11000000b ; 1-й канал АЦП, усилен=1, диф. режим

out  dx, al        ; установим параметры работы АЦП



mov  dx, 303H      ; в качестве задержки на установление

; канала

in   al, dx        ; считаем цифровой порт
```

```
; дадим старт АЦП

mov  dx, 314H      ; адрес порта старта АЦП (BASE+4)

out  dx, al        ; дадим старт преобразования



; дождёмся завершения цикла преобразования

; для этого подождём бита готовности

mov  dx, 312H      ; порт с битом готовности (BASE+2)

wb_ready:              ; ждём нулевого состояния на бите

; готовности

in   al, dx

and  al, 00001000b ; если 4-й бит в al не равен 1
```

```
jnz  wb_ready      ; идем на метку

; чтение результата АЦП

mov  dx, 310H      ; адрес чтение кода с АЦП (BASE)

in   ax, dx        ; считаем код с АЦП

; в ax – код результата
```

Чтение цифровых ТТЛ линий

На внешнем разъёме платы имеются 8 цифровых ТТЛ входов, состояние которых может быть прочитано по адресу BASE+3. При каждом чтении цифровых линий генерируется цифровой строб на линии "Строб чтения цифрового порта" на внешнем разъёме платы. Формат чтения ТТЛ линий приведен в табл. 1.8.

Таблица 1.8

Бит
8

7

6

5

4

3

2

1

Имя

D8

D7

D6

D5

D4

D3

D2

D1

D8-D1 соответствуют восьми входным ТТЛ линиям с внешнего разъёма.

Пример чтения ТТЛ линий

```
mov  dx, 313H      ; адрес ТТЛ  линий (BASE+3)

in   al, dx        ; считаем ТТЛ линии
```

Установка цифровых ТТЛ линий

На внешнем разъёме платы имеются 8 цифровых ТТЛ выходов, состояние которых может быть изменено записью по адресу BASE+3. Формат установки ТТЛ линий приведен в табл. 1.9.

Таблица 1.9

Бит
8
7
6
5
4
3
2
1
Имя
D8
D7

D6

D5

D4

D3

D2

D1

D8-D1 соответствуют восьми выходным ТТЛ линиям с внешнего разъёма.

Пример установки ТТЛ линий

```
mov  dx, 313H      ; адрес ТТЛ  линий (BASE+3)

mov  ax, 0ffH      ; установим все единицы на ТТЛ

; выходах
```

```
out  dx, al        ; установим ТТЛ линии
```

Управление таймерами

Назначение

На плате установлена микросхема 580ВИ53 трёхканального таймера. С помощью трёх каналов таймера можно осуществлять программную синхронизацию процессов ввода и вывода, генерировать прерывания IRQ в компьютере. Входы разрешения всех каналов таймера всегда равны единице (т. е. разрешено). Канал А тактируется от установленного на плате кварца с частотой 1 МГц. Таймер В каскадно связан с таймером А, что позволяет работать с большими временными интервалами (до 0xFFFF*0xFFFF мкс). Счётный вход таймера С выведен на внешний разъём, что позволяет использовать программную синхронизацию процессов ввода от внешних цифровых сигналов.

Установка режимов таймеров и опрос их текущего состояния достигается при записи соответствующего кода в регистр управления и двойным опросом соответствующего счетчика.

Опрос состояния таймеров

Опрос состояния счётчиков осуществляется одним из двух способов.

Выходы всех трёх каналов выведены на порт BASE+0x2, поэтому возможно отслеживание состояния соответствующих битов этого порта.

Текущее состояние счётчика можно отследить путём опроса соответствующего порта таймера. Формат чтения порта BASE+0x2 приведен в табл. 1.10.

Таблица 1.10

Бит

8

7

6

5

4

3

2

1

Имя

X

X

X

X

D4

D3

D2

D1

D4 бит готовности от АЦП;

D3 выход третьего канала таймера;

D2 выход второго канала таймера;

D1 выход первого канала таймера.

Работа с прерываниями (адрес BASE+0xF)

Прерывания IRQ могут генерироваться выходом второго таймера, который каскадно соединён с первым таймером (т. е. интервал между генерируемыми прерываниями равен

произведению интервалов, записанных в первый и второй таймеры). Номер генерируемого прерывания устанавливается перемычкой на плате (см. описание перемычек). При включении компьютера генерирование прерываний запрещено.

Запись по адресу BASE+0xF единицы приводит к разрешению генерирования прерываний платой. Запись по адресу BASE+0xF нуля приводит к запрещению генерирования прерываний платой (т. е. выход второго таймера будет отключён от линии прерывания компьютера).

Порядок выполнения работы

Однократный асинхронный ввод с АЦП

Обобщенный алгоритм, реализующий аналого-цифровое преобразование и вывод результата в единицах напряжения, может быть представлен следующей блок-схемой (рис. 1.4):

Рис. 1.4. Блок-схема алгоритма ввода данных с АЦП

Данный алгоритм предлагается реализовать с помощью программы на языке ассемблер. Для этого запустите программу Norton Commander, ярлык которой расположен на рабочем столе Windows. Войдите в каталог C:Students, с помощью команды <Shirf+F4> создайте новый текстовый файл, для которого введите имя, например lab1.asm. Введите следующий текст программы. Заполните пропуски соответствующими командами ассемблера, установив дифференциальный режим работы АЦП, коэффициент усиления

равный единице, номер канала равный номеру вашей бригады (комментарии после точки с запятой можно не вводить):

```
assume cs:text,ds:data,ss:stk

stk segment stack

dw 100 dup(?)

stk ends



text    segment 'code'

include mac.mac

assume cs:text,ds:data,ss:stk



main    proc
```

```
mov  ax,data

mov  ds,ax




; установка номера канала

; коэффициента усиления и режима работы АЦП

; ...




; ожидание установления номера канала

; ...




; дать старт преобразования
```

```
; ...




; ожидание прихода бита готовности

; ...




; считать результат преобразования из порта АЦП

; ...




; привести результат преобраз. в единицы милливольт

sub  ax, 2048          ; ax: -2048 … 2048

mov  bx, 10

imul bx                ; ax=ax*10
```

```
mov  bx, 4

idiv bx                ; ax=ax/4


call Write             ; вывод результата


; завершение работы программы

mov   ax, 4c00h

int   21h

main endp


; локальная функция вывода числа в десятичном виде на экран
```

```
Write  proc near

; вызов макроса, преобразующего двоичное число из AX

; в строку символов десятичных цифр и знака ‘-’

OutToStr ax, ResOut

mov  ah, 09h

mov  dx, offset    ResOut   ; вывод строки на экран

int  21h

ret

Write  endp



text ends
```

```
data segment       ;сегмент данных

Result   db   1 DUP(0) ; перем. для хранения результата

ResOut   db   7 DUP(0) ; строковая переменная

data ends

end  main
```

Сохраните набранный текст клавишей <F2>, покиньте текстовый редактор клавишей <ESC>. Проассемблируйте программу командой TASM имя_файла.asm. Если компилятор обнаружил ошибки, исправьте строки текста программы, в которых обнаружены ошибки, проассемблируйте программу заново. Если ошибок нет, скомпонуйте исполняемую программу, для чего наберите tlink имя_файла.obj. Запустите появившийся файл имя_файла.exe. Чтобы посмотреть выводимый программой текст нажмите клавиши «CTRL+O». Данная программа должна выводить на экран значения напряжения, поданного на заданный канал АЦП (значение напряжения выводится в милливольтах).

Подайте на используемый канал напряжение +5В с источника питания стенда (рис. 1.5).

Запустите программу заново, убедитесь, что на экран выводится данное значение (в милливольтах).

Соберите следующую схему (рис. 1.6):

Рис. 1.5. Подключение канала АЦП к источнику напряжения на лабораторном стенде

Рис. 1.6. Подключение канала АЦП к регулируемому источнику напряжения

Запускайте программу несколько раз, изменяя положение ручки потенциометра. Убедитесь в изменении значения выводимого напряжения. Результат покажите преподавателю.

Однократный ввод последовательности каналов АЦП

Измените текст программы так, чтобы при ее выполнении на экран выводился результат преобразования по всем 16-ти дифференциальным каналам АЦП. Для этого введите в программу цикл в соответствии с алгоритмом (рис. 1.7).

Рис. 1.7. Блок-схема алгоритма АЦ преобразования
последовательности каналов АЦП

В качестве переменной цикла можно использовать, например, регистр общего назначения cl. Тогда блок подготовки цикла будет представлять собой команду обнуления cl:

```
mov  cl, 0
```

Установка режима работы АЦП в этом случае может выглядеть следующим образом:

```
mov  dx, 312H      ; адрес порта номера канала (BASE+2)

mov  al, 11000000b ; усилен = 1, диф. режим

or   al, cl        ; установка номера канала из cl

out  dx, al        ; установим параметры работы АЦП
```

Модификация переменной цикла и условие продолжения цикла могут быть представлены командами:

```
inc  cl        ; увеличение на 1 значения в cl

cmp  cl, 15    ; cl=15 ?

jle  имя_метки ; если меньше или равно перейти на метку
```

Откомпилируйте и запустите исправленную программу. Данная программа выводит на экран результат преобразования по 16-ти дифференциальным каналам. Соберите следующую схему (рис. 1.8).

Рис. 1.8. Подключение нескольких каналов АЦП, включенных через резистивный делитель напряжения к источнику напряжения на лабораторном стенде

Запустите программу. Убедитесь, что значения напряжений с 5-го по 16-й каналы равномерно уменьшаются. Результат покажите преподавателю.

Контрольные вопросы

1. Укажите назначение и состав платы ввода и вывода аналоговой и цифровой информации L-154 (фирмы LCard).
2. Что собой представляет функция преобразования ЦАП, какой диапазон входного кода?
3. Приведите команды установки кода на ЦАП.
4. Что собой представляет функция преобразования АЦП?
5. Приведите команды установки режима работы АЦП.
6. Какие команды используются для запуска АЦП и чтения результата преобразования?
7. Как осуществляется чтение и установка цифровых ТТЛ линий?
8. Каково назначение таймеров на плате ввода-вывода?
9. Какие существуют способы опроса состояния таймеров?

Лабораторная работа □2
Исследование программного вывода аналогового сигнала с ПЭВМ через ЦАП

Цель работы: получение навыков использования ЦАП и аппаратного таймера, в составе платы аналогового ввода-вывода, для вывода аналогового сигнала с ПЭВМ с заданной частотой дискретизации.

Основные вопросы, изучаемые в работе:

- использование ЦАП и аппаратного таймера платы аналогового ввода-вывода L-154 фирмы L-Card;

- команды языка ассемблера, используемые для обращения к плате ввода-вывода;

- написание программы на языке ассемблер, осуществляющей вывод аналогового сигнала с ПЭВМ с заданной частотой дискретизации.

Использование ЦАП для вывода аналогового сигнала

Цифроаналоговый преобразователь (ЦАП)

На плате L-154 фирмы L-Card, используемой в данной лабораторной работе установлен один ЦАП, параметры и программирование которого описаны в лабораторной работе №1.

Непрерывный вывод на ЦАП

В работе предлагается реализовать цифровой генератор сигналов специальной формы на базе ПЭВМ, со встроенным ЦАП. Работа генератора в общем виде происходит по алгоритму, изображенному на рис. 2.1. Здесь сначала осуществляется формирование массива отсчетов в памяти компьютера, задающего форму выводимого сигнала. Затем отсчеты циклически берутся из массива последовательно и передаются на ЦАП, формируя на его выходе желаемый сигнал. Частота вывода отсчетов определяется скоростью выполнения команд, составляющих тело цикла, и непосредственно при программировании не задается.

Рис. 2.1. Блок-схема алгоритма вывода сигнала

специальной формы с ПЭВМ

Непрерывный вывод на ЦАП с синхронизацией от таймера

Для временной синхронизации выводимых отсчетов используется специальное средство – аппаратный таймер. Фактически на плате L-154 установлено 3 счетчика-таймера, работающих с тактовой частотой 1МГц (см. лаб. раб. №1). Программируя таймеры, мы можем устанавливать различные коэффициенты деления частоты и получать последовательность импульсов желаемой частоты с их выхода. Эти импульсы можно использовать для синхронизации выводимых отсчетов. При этом различают программную синхронизацию и синхронизацию по прерыванию. В первом случае программа периодически опрашивает состояние выхода таймера, ожидая появления очередного импульса. Во втором случае таймером вырабатывается прерывание, реагируя на которые исполняемая программа совершает необходимые действия.

В данной работе реализуется программная синхронизация выводимых отсчетов. Работа программы описывается алгоритмом, представленным на рис. 2.2.

Рис. 2.2. Блок-схема алгоритма

вывода сигнала с синхронизацией от таймера

Для конфигурирования таймера на выдачу непрерывной последовательности импульсов заданной частоты, может быть использована следующая подпрограмма:

;********** локальная подпрограмма установки частоты ввода

```
; ax -> Rate, bx -> Timer

set_rate  proc near

mov     ax,<число>      ; <число> – период

; следования импульсов в мкс

mov     bx, 0           ; установка номера таймера

push    cx

push    ax

push    bx

mov     ax, bx

mov  bl,30h

mov  cl,6
```

```
shl  al,cl

or   bl,al



mov  ax,6     ; считаем устанавливаемый режим таймера

or   al,bl

mov  dx,31bh

out  dx,al

pop   bx



mov  dx,318h

add  dx, bx
```

```
pop   ax

out   dx,al



push  dx

mov  dx, 314h      ; delay

in   al, dx

pop  dx



mov  al,ah

out  dx,al
```

```
pop   cx

ret

endp
```

Ожидание сигнала с таймера в случае программной синхронизации предлагается реализовать в соответствии с алгоритмом рис. 2.3. Здесь анализируется состояние бита 1 в байте состояния таймеров (бит 1 соответствует таймеру А, см. рис. 1.1). Сначала программа дожидается нулевого состояния, затем ожидает появления единичного состояния бита 1. Используемое в алгоритме условие реализуется командой:

```
and al, 00000001b
```

Рис. 2.3. Блок-схема алгоритма ожидания сигнала с таймера

Порядок выполнения работы

В данной работе вы пишите программу на языке ассемблер, осуществляющую вывод массива отсчетов через ЦАП (моделирование генератора сигналов специальной формы).

Введите следующий текст программы, реализующий непрерывный вывод на ЦАП, в соответствии с алгоритмом рис. 2.1.

```
assume cs:text,ds:data,ss:stk

stk segment stack

dw 100 dup(?)

stk ends

text    segment 'code'

assume cs:text,ds:data,ss:stk

main    proc

mov  ax, data
```

```
mov   ds, ax




call Signal        ; вызов подпрограмм записи в массив

; формы вых. сигнала

mov  di, 0

repeat:

mov  ax, massive[di]    ; код ЦАП 0…4095

mov  dx, 310h

out  dx, ax             ; Вывод на ЦАП

inc  di

inc  di
```

```
cmp  di, 512          ; достигнут ли конец массива

jl   S1               ; если нет, то переход.

mov  di, 0            ; возврат указателя                    ; к первому элементу
массива

; проверка нажатия клавиши

S1:  mov  ah, 06h  ; DOS 06h: считать символ из STDIN

mov  dl, 0ffh

int  21h

cmp  al, 1bh  ; Если введен символ 1bh (ESC), то выход

je   key_ESC

jmp  repeat   ; переход к началу цикла



; завершение работы программы
```

```
key_ESC:

mov   ax, 4c00h

int   21h

main endp



Signal Proc Near   ; подпрограмма записи в массив                    ; формы вых. сигнала

; формирование первого участка сигнала

mov  cx,128   ; половина объема массива

mov  di,0     ; обнуляем указатель номера ячейки

mov  ax,0     ; текущее значение элемента массива

ms1: mov  massive[di],ax     ; запись в ячейку массива               ; значения из ax
```

```
inc  di

inc  di    ; увеличиваем номер ячейки массива

add  ax,32 ; изменяем текущее значение отсчета

loop ms1   ; цикл по элементам массива

dec  ax    ; максимальное значение кода 4095,а не 4096

; формирование второго участка сигнала

mov  cx,128   ; половина объема массива

ms2: mov  massive[di],ax     ; запись в ячейку массива                    ; значения из ax

inc  di

inc  di       ; увеличиваем номер ячейки массива

sub  ax,32    ; изменяем текущее значение отсчета

loop  ms2     ; цикл по элементам массива
```

```
ret

Signal EndP

text ends

data segment        ;сегмент данных

Massive  dw   256 dup (0)   ; массив отсчетов                                  ; из 256 элементов

data ends

end  main
```

Теперь сохраните набранный текст клавишей F2, покиньте текстовый редактор клавишей ESC. Откомпилируйте программу командами *tasm имя_файла.asm* и затем *tlink имя_файла.obj*
. Если компилятор обнаружил ошибки, исправьте строки текста программы, в которых обнаружены ошибки, откомпилируйте программу заново. Если ошибок нет, запустите появившийся файл имя_файла.exe. Соберите следующую схему (рис. 2.4):

Рис. 2.4. Схема подключения осциллографа к выходу ЦАП

Запустите программу и добейтесь устойчивого изображения на экране осциллографа. Данная программа позволяет получить с выхода ЦАП треугольный сигнал. Результат покажите преподавателю.

В случае успешной работы измените программу для осуществления непрерывного вывода с синхронизацией от таймера, в соответствии с алгоритмом рис. 2.2.

Примечания:

- вызов подпрограммы конфигурирования таймера реализуйте в подпрограмме main, до метки repeat с помощью команды:
  call set_rate

- текст подпрограммы set_rate добавьте после подпрограммы main;

- установите период следования импульсов с таймера 50 мкс;

- текст программы, реализующий алгоритм рис. 2.3, добавьте после метки repeat.

Сохраните изменения, затем откомпилируйте и запустите на выполнение полученную программу. Данная программа должна выводить на экран следующий сигнал (рис. 2.5).

Рис. 2.5. Сигнал, формируемый программой

Настройте осциллограф для получения устойчивого изображения данного сигнала. Определите амплитуду и период сигнала. Результат покажите преподавателю.

В случае успешного выполнения предыдущего задания измените программу так, чтобы она выводила сигнал, изображенный на рис. 2.6 (в соответствии с номером бригады). Результат покажите преподавателю.

Рис. 2.6. Формы сигналов для самостоятельного задания

Контрольные вопросы

1. Приведите функцию преобразования ЦАП, какой диапазон выходного сигнала у данного ЦАП.
2. Какие команды языка ассемблер используются для установки кода на ЦАП?
3. Приведите алгоритм вывода сигнала специальной формы без синхронизации.
4. Как обеспечить постоянную частоту вывода отсчетов?
5. Как осуществить программную синхронизацию ввода с использованием аппаратного таймера?
6. Приведите алгоритм вывода сигнала специальной формы с синхронизацией от таймера.
7. Как реализуется ожидание сигнала с таймера в случае программной синхронизации вывода отсчетов?

Лабораторная работа 3
Использование языка программирования высокого уровня Delphi при проектировании систем сбора аналоговых данных

Цель работы. ознакомиться со средствами среды Delphi, позволяющими производить автоматизированный обмен и визуальное отображение данных с платой ввода-вывода L-154.

Основные вопросы, изучаемые в работе:

- изучение среды Delphi;

- ввод и отладка исходного текста программы;

- тестирование приложения;

- модификация программы.

Изучение среды Delphi

Прикладные программы, или приложения Delphi создаются в интегрированной среде разработки (IDE — Integrated Development Environment). Пользователь-ский интерфейс включает в себя ряд окон, содержащих различные элементы управле-ния (рис. 3.1.). С помощью средств интегрированной среды можно осуществлять проектирование интерфейсной части приложения, а также писать программный код и связывать его с элементами управления. В интегрированной среде разра-ботки проходят все этапы создания приложения, включая отладку.

После загрузки интерфейс Delphi, в зависимости от версии, может выглядеть по-разному, но обычно он включает следующие окна:

– главное окно (Delphi - Project1);

– окно Инспектора объектов (Object Inspector);

– окно Формы, или Конструктора формы (Forml);

– окно Редактора кода (Unit1.pas);

– окно Проводника кода (Exploring Unitl.pas).

Последние два окна находятся позади окна Формы, причем окно Проводника кода

пристыковано слева к окну Редактора кода, поэтому оба этих окна имеют общий заголовок Unitl.pas.

Рис. 3.1. Вид интегрированной среды разработки Delphi

На экране кроме указанных окон могут присутствовать и другие окна, отобра-жаемые при вызове соответствующих средств. Окна Delphi можно перемещать, изменять их размеры и убирать с экрана (кроме главного окна), а также состыковывать между собой.

Несмотря на наличие многих окон, Delphi является однодокументной средой и позволяет одновременно работать только с одним проектом приложения. Название проекта приложения выводится в строке заголовка главного окна в верхней части экрана.

Главное окно Delphi включает:

– главное меню;

– панели инструментов;

– палитру компонентов.

Главное меню содержит набор команд для доступа к функциям Delphi. Панели инструментов находятся под главным меню в левой части главного окна и содержат кнопки для вызова наиболее часто используемых команд главного меню, например FileOpen (ФайлОткрыть) или RunRun (Выполнение Выполнить).

Палитра компонентов находится под главным меню в правой части главного окна и содержит компоненты, размещаемые в создаваемых формах. Компоненты являются блоками, из которых конст-руируются формы приложения. Все компоненты разбиты на группы, каждая из которых в Палитре компонентов располагается на отдельной странице, а сами компоненты представлены значками. Нужная страница Палитры компонентов выбирается щелчком мыши на ее значке.

Разработка приложений начинается с визуального конструиро-вания формы и заключается в размещении на форме необходимых элементов. Кроме того, в окне Инспектора объектов задаются значения свойств компонентов.

Окно Редактора кода (Unitl.pas) после запуска системы программирования на-ходится под окном Формы и представляет собой обычный текстовый редактор, с помощью которого осуществляется редактирование текста модуля и других текстовых файлов приложения. Каждый редактируемый файл находится в окне Редактора кода на отдельной странице, доступ к которой осуществляется щелчком на соответст-вующем значке. Первоначально в окне Редактора кода на странице Code содер-жится одна вкладка Unit1 исходного кода модуля формы Form1 разрабатывае-мого приложения.

Переключаться между окнами Формы и Редактора кода удобно с помощью кла-виши <F12>.

Окно Проводника кода (Exploring Unitl.pas) пристыковано слева к окну Редак-тора кода. В нем в виде дерева отображаются все объекты модуля формы.

Состав проекта приложения, создаваемый в среде Delphi, состоит из нескольких элементов, объе-диненных в проект. В состав проекта входят следующие элементы (в

скобках указаны расширения имен файлов):

– код проекта (dpr);

– описания форм (dfm – для Windows, xfm – кроссплатформенный вариант);

– модули и модули форм (pas);

– параметры проекта (dof – для Windows, kof – для Linux);

– параметры среды (cfg);

– описание ресурсов (res).

Порядок выполнения работы

Ввод и отладка исходного текста программы

Запустите редактор Delphi (ПускПрограммыBorland Delphi 5 Delphi 5). Разместите на форме следующие компоненты, располагающиеся на панели компонентов (Component Palette) (рис. 3.2):

– компонент Chart (вкладка Additional);

– компонент Button (вкладка Standart);

– компонент Edit (вкладка Standart);

– компонент Label (вкладка Standart).

Рис. 3.2. Интерфейсная часть приложения

Для настройки компонента Chart1 необходимо двойным щелчком по элементу открыть окно настройки диаграммы (Editing Chart1). С помощью окна настройки добавить новую серию данных (кнопка Add, расположенная Chart-Series), выбрать тип графика FastLine в окне TeeChart Gallery. Выключить отображение легенды и названия графика, убрав отметки свойства Visible, располагающиеся на вкладках Chart-Legend и Chart-Titles.

Для компонента Button1 задайте для свойства Caption значение «Start». Изменение свойств произведите на панели Object Inspector. При этом компонент предварительно должен быть выбран.

Для компонента Editl задайте для свойства Text значение «1». Данный компонент будет определять номер ка-нала.

Для компонента Label1 задайте для свойства Caption значение «Номер канала 1-16:».

Перейдите в окно редактора приложения и введите следующий текст программы (комментарий, расположенный после символов "//" можно опустить):