**Глава 1. Начало работы на Турбо Ассемблере**

Если вы никогда ранее не программировали на языке Ассембле-

ра, то начните с данной главы. Возможно вам приходилось слышать,

что программирование на Ассемблере - это дело темное, доступное

только посвященным и мудрецам. Не верьте этому. Язык Ассемблера -

это не более чем человеческая форма языка самого компьютера, а

он, как и можно было предположить, в высшей степени логичен. Как

вы можете также догадаться, язык Ассемблера - это очень мощный

язык. Фактически, программирование на Ассемблере представляет со-

бой единственный способ реализации всего спектра возможностей

процессоров серии 80х86 фирмы Intel, являющихся "сердцем" всех

компьютеров семейства IBM PC и совместимых с ними компьютеров.

Вы можете писать программы целиком на языке Ассемблера или,

если захотите, использовать язык Ассемблера в программах, напи-

санных на Турбо С++, Турбо Паскале и других языках. В любом слу-

чае с помощью языка Ассемблера вы сможете разрабатывать компакт-

ные и быстрые программы. Наряду со скоростью большое значение в

программе на языке Ассемблера имеет также возможность управления

всеми аспектами работы компьютера, до последнего такта системного

таймера.

В данной главе вы познакомитесь с языком Ассемблера и опро-

буете уникальные свойства программирования на нем. Сначала вы

введете и запустите несколько работающих программ, написанных на

Ассемблере. Это даст вам возможность как почувствовать сам язык,

так и познакомиться с работой на Ассемблере.

Разумеется, мы не сможем при помощи этих нескольких глав

сделать вас крупным специалистом по программированию на языке Ас-

семблера; мы просто дадим вам введение в язык Ассемблера и помо-

жем написать первые программы. Мы очень надеемся, что у вас есть

хотя бы одна из множества замечательных книг, полностью посвящен-

ных программированию на языке ассемблера и архитектуре PC (см.

ссылки на литературу в конце данной книги). Кроме того, полезные

справочные материалы содержатся в "Техническом справочнике по IBM

DOS", "Техническом справочнике по интерфейсу BIOS" и "Техническом

справочнике по персональным компьютерам XT"; в этих руководствах

документируется интерфейс при программировании на языке ассембле-

ра с программным и аппаратным обеспечением системы персональных

компьютеров IBM.

Вы можете следовать данному руководству поэтапно, вводя все

примеры программ по мере их изучения, либо можете распаковать на-

ходящийся на диске файл с примерами (во время установки Турбо Ас-

семблера) и иметь все эти программы под рукой. (Независимо от ва-

шего решения, имена программ приводятся для удобства в самом

начале примеров этих программ).

**Установка Турбо Ассемблера в системе**

Перед тем, как познакомиться с программированием на Турбо

Ассемблере, вам нужно сделать следующее. Возьмите дистрибутивные

диски Турбо Ассемблера и сделайте для каждого из них (с помощью

утилиты DOS) рабочую копию. После этого исходные (дистрибутивные)

диски уберите в надежное место.

Если собираетесь использовать Турбо Ассемблер вместо MASM

(макроассемблер фирмы Microsoft), прочитайте Главу 2 и посмотри-

те, в чем поведение Турбо Ассемблера отличается от MASM.

На диске INSTALL находится программа INSTALL.EXE, которая

поможет вам установить Турбо Ассемблер версии 3.0 в вашей систе-

ме. При установке следует учесть два параметра:

1. Hard Disk Users (пользователям системы с жестким диском):

этот параметр позволяет выбрать подкаталоги, в которые

будут загружены файлы.

2. Floppy Disk Users (пользователям системы с гибкими диска-

ми): этот параметр установит файлы, необходимые для рабо-

ты Турбо Ассемблера, в системе с двумя дисководами для

гибких дисков. До начала работы подготовьте четыре отфор-

матированных диска.

Для начала установки перейдите в каталог с программой

INSTALL и наберите команду:

INSTALL

Вам будут даны инструкции относительно каждого приглашения в

рамке в нижней части экрана установки.

При запуске INSTALL в системе портативного компьютера или в

любой другой системе с жидкокристаллическим дисплеем запуск

INSTALL следует выполнять с параметром /b, который позволяет пе-

ревести систему в черно-белый режим:

INSTALL /b

**Текстовые справочные файлы**

В состав Турбо Ассемблера входят несколько справочных фай-

лов: FILELIST.DOC. README, HELPME!.TSM, MANUAL.TSM, H2ASH.TSM и

TCREF.TSM. Первые два файла содержатся на установочной дискете

"Installation Disk" и копируются в основной каталог языка. Другие

файлы находятся в подкаталоге DOC.

**Файл README**

Перед тем, как начать работу с Турбо Ассемблером, не за-

будьте прочесть файл README. В нем находится последняя информация

о программе, а также исправления и/или дополнения к руководствам.

Там вы также найдете полный список всех поставляемых на дистрибу-

тивных дисках файлов и их описания. В ответ на подсказку DOS вве-

дите:

README

Вы можете также использовать программу README для чтения

файлов документации (с расширением .DOC)

**Файл HELPME!.TSM**

На установочной дискете содержится также файл HELPME!.TSM. В

нем можно найти ответы на вопросы, с которыми обычно сталкиваются

пользователи. Он содержит также вопросы и ответы, касающиеся раз-

личных режимов Турбо Ассемблера.

**Файл H2ASH.TSM**

-----------------------------------------------------------------

Данный файл содержит подробную информацию об утилите-преоб-

разователе файлов .h в .ahs. Краткое описание этой утилиты можно

найти в Приложении D.

**Файл TCREF.TSM**

-----------------------------------------------------------------

В данном файле описывается использование для создании перек-

рестных ссылок идентфикаторов и идентификации использующих их мо-

дулей утилиты TCREF.

**Оперативный справочник**

Оперативную справочную информацию по Турбо Ассемблеру можно

получить с помощью резидентной с памяти справочной программы-ути-

литы TASMHELP. Например, если вы используете Турбо отладчик, мож-

но загрузить TASMHELP, затем загрузить Турбо отладчик и получать

во время отладки справочную информацию по Турбо Ассемблеру. Либо

вы можете использовать TASMHELP для поиска информации по языку

при написании исходного кода.

Так как TASMHELP - это резидентная в памяти программа, ее

нужно загрузить в память перед запуском редактора. Чтобы загру-

зить TASMHELP, в ответ на подсказку командной строки DOS C: набе-

рите:

TASMHELP

После этого вы можете запустить редактор, как обычно. Нахо-

дясь в редакторе, нажмите для вызова TASMHELP оперативную клавишу

- клавишу 5 на дополнительной клавиатуре (справа). Если при этом

курсор у вас находится на пустой строке, TASMHELP выведет таблицу

справочных экранов. Для перемещения курсора по ключевым словам

используйте клавишу Tab и нажмите клавишу Enter.

Программа TASMHELP работает точно также, как справочная ути-

лита компилятора THELP.

Подробнее о TASMHELP и ее параметрах командной строки можно

узнать в документации по компилятору, где описывается работа с

THELP.

**Ваша первая программа на Турбо Ассемблере**

В программировании первой программой традиционно является

программа, выводящая на экран сообщение "Привет!". Не будет иск-

лючением и наша программа, поскольку это является хорошей отправ-

ной точкой.

Войдите в текстовый редактор (один из тех редакторов, кото-

рые формируют файлы в коде ASCII) и введите следующие строки

программы под названием HELLO.ASM:

.MODEL SMALL

.STACK 100h

.DATA

Message DB 'Привет!',13,10,'$'

.CODE

mov ax,@Data

mov ds,ax ; установить регистр DS таким

; образом, чтобы он указывал

; на сегмент данных

mov ah,9 ; функция DOS вывода строки

mov dx,OFFSET Message ; ссылка на сообщение "Привет!"

int 21h ; вывести "Привет!" на экран

mov ah,4ch ; функция DOS завершения

; программы

int 21h ; завершить программу

END

После того, как вы введете эту программу, сохраните ее на

диске.

Если вы знакомы с языками Си, С++ или Паскаль, вам может по-

казаться, что эта версия программы несколько длинновата. Это

действительно так, программы на Ассемблере длиннее, поскольку

каждая инструкция Ассемблера выполняет меньше функций, чем инс-

трукция Паскаля, С++ или языка Си. С другой стороны, вам предос-

тавляется свобода, и вы можете комбинировать эти инструкции Ас-

семблера так, как захотите. Это означает, что в отличие от языков

Си и Паскаль, Ассемблер позволяет вам программировать компьютер

таким образом, что он будет делать все, на что способен. Часто

это стоит нескольких дополнительных строк.

**Ассемблирование вашей первой программы**

После того, как вы сохранили файл HELLO.ASM, вы захотите за-

пустить программу. Однако, перед тем, как вы сможете ее запус-

тить, потребуется преобразовать программу в выполняемый вид. Как

показано на Рис. 1.1, где изображен полный цикл создания програм-

мы (редактирование, ассемблирование, компоновка и выполнение),

это потребует двух дополнительных шагов - ассемблирования и ком-

поновки.

На этапе ассемблирования ваш исходный код (текст программы)

превращается в промежуточную форму, которая называется объектным

модулем, а на этапе компоновки один или несколько модулей комби-

нируются в выполняемую программу. Ассемблирование и компоновку вы

можете выполнять с помощью командной строки.

Для ассемблирования файла HELLO.ASM наберите:

TASM hello

Создание новой программы

¦

--------------------------------¦

¦ ¦

¦ Редактирование

¦ ¦

¦ ­

¦ -----------------------------------------------¬

¦ ¦ Исходный файл Ассемблера HELLO.ASM ¦

¦ L---------------------T-------------------------

¦ ¦

¦ Ассемблирование

¦ ¦

¦ ­

¦ ----------------------------------------------¬

¦ ¦ Объектный файл HELLO.OBJ ¦

¦ L---------------------T------------------------

¦ ¦

¦ Компоновка

¦ ¦

¦ ­

¦ ----------------------------------------------¬

¦ ¦ Выполняемый файл HELLO.EXE ¦

¦ L---------------------T------------------------

¦ ¦

¦ Выполнение

¦ -----------------------¬ ¦

L---+ Если нужны изменения ¦-----

L-----------------------

Рис. 1.1 Редактирование, ассемблирование, компоновка и вы-

полнение.

и нажмите клавишу Enter. Если вы не задали другое имя, файл

HELLO.ASM будет ассемблирован в файл HELLO.OBJ. (Заметим, что

расширение имени файла вводить не требуется. Турбо Ассемблер под-

разумевает в этом случае, что файл имеет расширение .ASM.) На эк-

ране вы увидите следующее:

Turbo Assembler Version 3.0 Copyright (C) 1988,1991 (1)

by Borland International Inc.

Assembling file: HELLO.ASM (2)

Error messages: None (3)

Warning messages: None (4)

Passes: 1 (5)

Remaining memory: 266K (6)

1 - Турбо Ассемблер, версия 3.0; авторские права фирмы

Borland, 1991 г.; 2 - ассемблирован файл HELLO.ASM; 3 - сообщения

об ошибках: нет; 4 - предупреждающие сообщения: нет; 5 - число

проходов: 1; 6 - остается памяти: 266К

Если вы введете файл HELLO.ASM в точности так, как показано,

то вы не получите никаких предупреждающих сообщений или сообщений

об ошибках. Если вы получаете такие сообщения, они появляются на

экране наряду с номерами строк, указывающими строки, где содер-

жатся ошибки. При получении сообщений об ошибках проверьте исход-

ный код (текст) программы и убедитесь, что он выглядит точно так,

как исходный код в нашем примере, а затем снова ассемблируйте

программу.

**Компоновка программы**

После ассемблирования файла HELLO.ASM вы продвинулись только

на один шаг в процессе создания программы. Теперь, если вы ском-

понуете только что полученный объектный код в выполняемый вид, вы

сможете запустить программу.

Для компоновки программы используется программа TLINK,

представляющая собой поставляемый вместе с Турбо Ассемблером ком-

поновщик. Введите командную строку:

TLINK HELLO

Здесь опять не требуется вводить расширение имени файла.

Компоновщик TLINK по умолчанию предполагает, что этим расширением

является расширение .OBJ. Когда компоновка завершится (самое

большее через несколько секунд), компоновщик автоматически прис-

воит файлу с расширением .EXE имя, совпадающее с именем вашего

объектного файла (если вы не определили другое имя). При успешной

компоновке на экране появляется сообщение:

Turbo Linker Version 3.0 Copyright (c) 1988, 1991 by Borland

International Inc.

В процессе компоновки могут возникнуть ошибки (в данной

программе это маловероятно). Если вы получили сообщения об ошиб-

ках компоновки (они выводятся на экран), измените исходный код

программы так, чтобы он в точности соответствовал тексту програм-

мы в приведенном выше примере, а затем снова выполните ассембли-

рование и компоновку.

**Запуск вашей первой программы**

Теперь программу можно запустить на выполнение. Для этого в

ответ на подсказку операционной системы DOS введите hello. На эк-

ран выведется сообщение:

Привет!

Пока это все. Вы только что создали и выполнили свою первую

программу на Ассемблере!.

**Что происходит?**

Теперь, когда вы получили и выполнили программу HELLO.ASM,

давайте вернемся назад и рассмотрим подробно, что происходит с

момента ввода текста программы до ее выполнения.

Когда вы ассемблируете файл HELLO.ASM, Турбо Ассемблер прев-

ращает текст инструкций в этом файле в их двоичный эквивалент в

объектном файле HELLO.OBJ. Этот файл является промежуточным фай-

лом (промежуточным звеном в процессе перехода от текстового к вы-

полняемому файлу). Файл HELLO.OBJ содержит всю информацию, необ-

ходимую для создания выполняемого кода из инструкций, содержащих-

ся в файле HELLO.ASM, но она записана в виде, который позволяет

комбинировать ее с другими объектными файлами для создания одной

программы.

При компоновке файла HELLO.OBJ TLINK преобразует его в вы-

полняемый файл HELLO.EXE, который вы запускаете, введя hello в

ответ на подсказку DOS.

Теперь введите:

dir hello.\*

При этом будет выведен список файлов HELLO на диске. Это будут

файлы HELLO.ASM, HELLO.OBJ, HELLO.EXE и HELLO.MAP.

**Модификация вашей первой программы на Турбо Ассемблере**

Теперь снова войдем в редактор и модифицируем программу та-

ким образом, чтобы она могла воспринимать какие-то данные из

внешней среды (этой "внешней средой" будете вы, а вводимые данные

будут набираться на клавиатуре). Измените программу следующим об-

разом:

.MODEL SMALL

.STACK 100h

.DATA

TimePrompt DB 'Это время после полудня? (ДА/НЕТ) - [Y/N]$'

GoodMorningMessage LABEL BYTE

DB 13,10,'Доброе утро!',13,10,'$'

GoodAfternoonMessage LABEL BYTE

DB 13,10,'Здравствуйте!',13.10,'$'

.CODE

mov ax,@Data

mov ds,ax ; установить регистр DS таким

; образом, чтобы он указывал

; на сегмент данных

mov dx,OFFSET TimePrompt ; ссылка на сообщение-запрос

mov ah,9 ; функция DOS вывода строки

int 21h ; получить ответ из одного

; символа

cmp al,'Y' ; указано время после полудня

; (прописная буква Y)

jz IsAfternoon ; да, время указано после

; полудня

cmp al,'y' ; указано время после полудня

; (строчная буква y)

jnz IsMorning ; нет, время указано до

; полудня

IsAfternoon:

mov dx,OFFSET GoodAfternoonMessage ; указывает на

; приветствие "Здравствуйте"

jmp DisplayGreeting

IsMorning:

mov dx,OFFSET GoodMorningMessage ; указывает на

; приветствие "Доброе утро"

DisplayGreeting:

mov ah,9 ; функция DOS вывода сообщения

int 21h ; вывести соответствующее

; сообщение

mov ah,4ch ; функция DOS завершения

; программы

int 21h ; завершить программу

END

Таким образом вы добавили в программу два очень важных новых

средства: возможность ввода и принятие решений. Эта программа

запрашивает у вас, является ли вводимое время временем после по-

лудня, воспринимая ответ (один символ) с клавиатуры. Если таким

ответом будет буква Y в верхнем или нижнем регистре (что означает

ответ ДА), то программа выводит сообщение "Здравствуйте!", в про-

тивном случае выводится сообщение "Доброе утро!". В данной прог-

рамме имеются все основные элементы полезной программы: ввод из

информации внешней среды, обработка данных и принятие решения.

Сохраните эту модифицированную программу на диске. После

этого заново ассемблируйте и скомпонуйте программу, как в преды-

дущем примере. Запустите ее снова, введя hello в ответ на подс-

казку DOS. Выведется сообщение:

Это время после полудня? (ДА/НЕТ) - [Y/N]

Курсор будет мерцать у последнего символа в ожидании ввода

ответа. Нажмите Y. Программа ответит:

Здравствуйте!

Таким образом HELLO.ASM стала теперь интерактивной програм-

мой с принятием решений.

В ходе ассемблирования вы, конечно, получите различные сооб-

щения об ошибках из-за неправильного набора программы и ошибках в

синтаксисе. Турбо Ассемблер перехватывает такие ошибки, сообщая о

них. Выводимые сообщения об ошибках разбиваются на две категории:

предупреждения и ошибки. Если Турбо Ассемблер обнаруживает что-то

подозрительное, но необязательно неверное, он выводит предупреж-

дающее сообщение. Иногда предупреждающие сообщения можно игнори-

ровать, но всегда лучше их проверить и убедиться в том, что вы

понимаете суть проблемы. При обнаружении чего-либо явно непра-

вильного в вашей программе, что делает невозможным завершение ас-

семблирования и формирование объектного файла, Турбо Ассемблер

выводит сообщение об ошибке.

Другими словами, предупреждающие сообщения не свидетельству-

ют о критических ошибках, в то время как ошибки, о которых гово-

риться в сообщениях об ошибках, должны быть исправлены перед за-

пуском программы. В Приложении E содержится перечень сообщений об

ошибках и предупреждающих сообщений.

Как и любой язык программирования, Турбо Ассемблер не может

распознавать ошибки в логике программы. Турбо Ассемблер только

сообщает вам, может ли ваш ассемблируемый код быть выполнен в том

виде, как он введен, но он не может сделать вывод о том, будет ли

программа работать так, как вы этого хотите. Об этом можете су-

дить только вы сами.

Чтобы получить распечатку программы (вывести ее на устройс-

тво печати), обратитесь к руководству по редактору текстов. Ис-

ходные файлы Турбо Ассемблера представляют собой обычные тексто-

вые файлы в коде ASCII (американский стандартный код обмена

информацией), поэтому вы можете также напечатать исходный текст

программы на Ассемблере с помощью команды PRINT, введя ее в ответ

на подсказку операционной системы DOS.

**Вывод информации на устройство печати**

Устройство печати (принтер) - это очень полезное устройство.

Вам может не только потребоваться распечатать текст программы, но

и передать на принтер выводимую информацию. Следующая версия

программы выводит информацию вместо экрана на принтер:

.MODEL SMALL

.STACK 100h

.DATA

Message DB 'Привет!',13,10,'$'

Message\_Length EQO $ - Message

.CODE

mov ax,@Data

mov ds,ax ; установить регистр DS таким

; образом, чтобы он указывал

mov ah,40h ; функция DOS вывода строки

; на устройство

mov bx,4 ; описатель принтера

mov cx,Message\_Length ; число печатаемых символов

mov dx,OFFSET Message ; ссылка на "Привет!"

int 21h ; вывести "Привет!" принтер

mov ah,4ch ; функция DOS завершения

; программы

int 21h ; завершить программу

END

В данной версии программы функция DOS вывода строки на экран

заменена на функцию DOS, которая передает информацию на выбранное

устройство или в файл, в данном случае - на принтер. Введите и

запустите программу. Посмотрите, как она напечатает на принтере

слово "Привет!". (Перед запуском программы не забудьте ее сохра-

нить при завершении работы в редакторе. При этом программа будет

сохранена в файле HELLO.ASM, а предыдущая версия программы будет

потеряна.)

Вы можете модифицировать эту программу таким образом, чтобы

она снова посылала выходные данные на экран, а не на устройство

печати, заменив просто строку:

mov bx,4 ; описатель принтера

на строку:

mov bx,1 ; описатель стандартного вывода

Сделайте такое изменение, а затем снова выполните перекомпи-

ляцию и перекомпоновку программы перед ее запуском. Запустив

программу, вы увидите, что сообщение появится на экране, а пос-

ледним символом будет графический символ перевода формата (кружок

с крестиком внизу - "зеркальце Венеры"). Этот символ программа

передает на принтер, чтобы вынудить его после вывода сообщения

выполнить перевод страницы, поскольку на экране страниц нет, он

ничего не знает о переводе формата и просто выводит на экран сим-

вол из набора символов компьютера РС.

**Ваша вторая программа на Турбо Ассемблере**

Теперь вы готовы к тому, чтобы ввести и запустить программу,

которая действительно что-то делает. Вернитесь в текстовый редак-

тор и введите следующую программу REVERSE.ASM:

.MODEL SMALL

.STACK 100h

.DATA

MAXIMUM\_STRING\_LENGTH EQU 1000

StringToReverse DB MAXIMUM\_STRING\_LENGTH DUP (?)

ReverseString DB MAXIMUM\_STRING\_LENGTH DUP (?)

.CODE

mov ax,@Data

mov ds,ax ; установить регистр DS таким

; образом, чтобы он указывал

mov ah,3fh ; функция DOS чтения ввода

mov bx,0 ; описатель стандартного ввода

mov cx,MAXIMUM\_STRING\_LENGTH ; считать до максималь-

; ного числа символов

mov dx,OFFSET StringToReverse ; сохранить строку

int 21h ; получить строку

and ax,ax ; были считаны символы?

jz Done ; нет, конец

mov cx,ax ; поместить длину строки в

; регистр СХ, который можно

; использовать, как счетчик

push cx ; сохранить в стеке длину

; строки

mov bx,OFFSET StringToReverse

mov si,OFFSET ReverseString

add si,cx

dec si ; указывает на конец буфера

; строки

ReverseLoop:

mov al,[bx] ; получить следующий символ

mov [si],al ; сохранить символы в

; обратном порядке

inc bx ; указатель на следующий

; символ

dec si ; указатель на предыдущую

; ячейку buffer

loop ReverseLoop ; переместить следующий

; символ, если он имеется

pop cx ; извлечь длину строки

mov ax,40h ; функция записи DOS

mov bx,1 ; описатель стандартного

; вывода

mov dx,OFFSET ReverceString ; напечатать строку

Done:

mov ah,4ch ; функция DOS завершения

; программы

int 21h ; завершить программу

END

Скоро вы увидите, что сможет делать эта программа. Для нача-

ла не забудьте ее сохранить (под именем REVERSE.ASM).

**Запуск программы REVERSE.ASM**

Для запуска программы REVERSE.ASM вы должны сначала ассем-

блировать ее:

TASM reverse

а затем ввести:

TLINK reverse

для создания выполняемого файла. Запустите программу, введя в от-

вет на подсказку DOS слово reverse. В случае вывода при ассембли-

ровании сообщений об ошибках, внимательно проверьте исходный код,

сравните его с приведенным текстом, а затем попытайтесь повторить

процесс.

После запуска программы на экране останется мерцающий кур-

сор. Очевидно, программа ожидает, что вы что-нибудь введете. Поп-

робуйте ввести:

ABCDEFG

а затем нажмите клавишу ENTER. Программа выведет на экран:

GFEDCBA

и завершит работу.

Теперь ясно, что делает программа REVERSE.ASM: она изменяет

порядок символов во введенной строке на обратный. Быстрая работа

со строками и символами - эта одна из областей, где язык Ассем-

блера превосходно демонстрирует свои качества. Вы увидите это в

следующих нескольких главах.

Вас можно поздравить! Вы только что ввели, ассемблировали и

скомпоновали несколько программ на Ассемблере и, таким образом,

ознакомились в действии с основами программирования на Ассембле-

ре: вводом, обработкой данных и выводом.

Если вы не хотите создавать объектный файл, но хотите полу-

чить файл листинга, или если вы хотите получить файл перекрестных

ссылок, но не хотите создавать файл листинга или объектный файл,

задайте в качестве имени файла пустое (нулевое) устройство

(NULL). Например, команда:

TASM FILE1,,NUL,

ассемблирует файл FILE1.ASM в объектный файл FILE1.OBJ, не созда-

вая файла листинга, и создает файл перекрестных ссылок FILE1.XRF.