Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №1

за 4 семестр

По дисциплине: «КСиС»

Тема: «Организация и исполнение программы на Ассемблере. Назначение и использование регистров МП. Изучение этапов создания программ на языке ассемблера. Изучение и работа с основной частью файла листинга ассемблерной

программы. Отладка программы с помощью программы Turbo Debugger»

Выполнил:

Студент 2 курса

Группы ПО-4(1)

Иваненко И. Л.

Проверил:

Савицкий Ю. В.

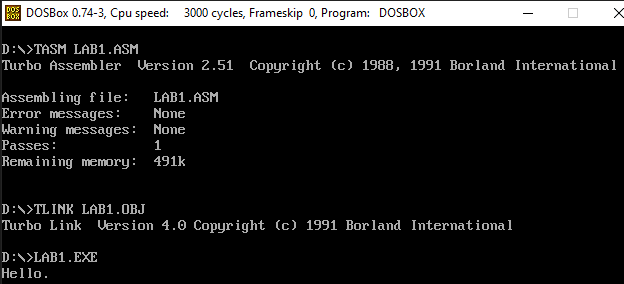
2021

Цель работы: “Знакомство со структурой ассемблерной программы,  процессом ее

получения и основными возможностями программ TASM, TLINK.”

Часть 1

1) Пользуясь правилами оформления ассемблерных программ, набрать программу Hello в файл LAB1.ASM.  Получить выполняемый файл и запустить его. Каков результат?



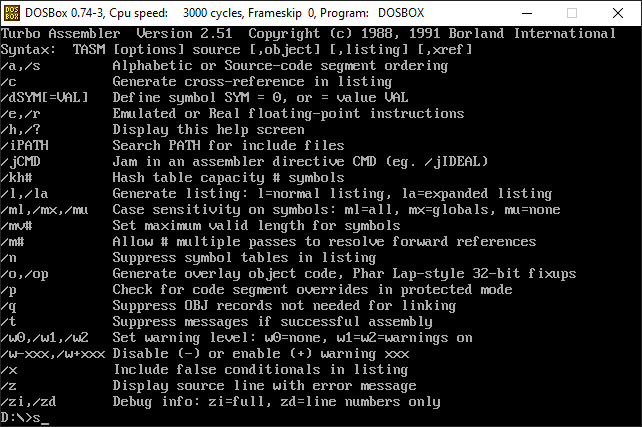
2) Определить, какие файлы создались при получении выполняемого файла. Пользуясь редактором, определить внутренний формат каждого из них.



LAB.EXE - бинарный

LAB.OBJ - бинарный

LAB.MAP - текстовый

3) Получить список ключей TASM, задав пустую командную строку

4) Получить файл aaa.obj из файла LAB1.asm. Обратить внимание на сообщение tasm. Что изменилось? Выполнить трансляцию еще раз с ключом /t.

На что он влияет

Получаем файл aaa.obj из файла lab1.asm:



Повторяем трансляцию с ключом /t:



Ключ /t позволяет скрыть сообщение об успешной трансляции.

5) Получить из файла aaa.obj файл bbb.exe без генерации карты загрузки.



Ключ /x позволяет генерировать исполняемый файл без карты загрузки.

Часть 2

4.1. \* Изучить предложенный материал по структуре и назначению полей

       файла листинга ассемблерной программы.

     \* Изучить структуру машинной команды

4.2. Пользуясь информацией лаб.раб.1, написать программу Lab2.asm, работающую по следующему алгоритму:

     \* вывести на экран первые N символов фамилии студента (использовать

       функцию 40h DOS INT 21h) – число N задается преподавателем;

     \* вывести строку полностью в цикле N-1 раз на экран.

**Привести программу в отчете**

**N = 5**

Код программы:

.MODEL SMALL

.STACK 100h

.DATA

surname DB 'Ivanenko',13,10,'$'

.CODE

START:

mov ax, @Data ; пересылаем адрес сегмента данных в регистр AX

mov ds, ax ; установка регистра dx на сегмент данных

mov ah, 40h ; функция вывода строки

mov bx, 1 ; информация будет отображена на экране

mov dx, OFFSET surname ; записать адрес строки в регистр DX

mov cx, 5 ; количество букв фамилии

int 21h ; вывод

mov ah, 02h ; перейти на новую строку

mov dl, 10 ; вернуть каретку в начало строки

int 21h

mov ah, 9h ; вывести строку

mov cx, 4 ; количество повторений цикла N-1

BEGIN: ;

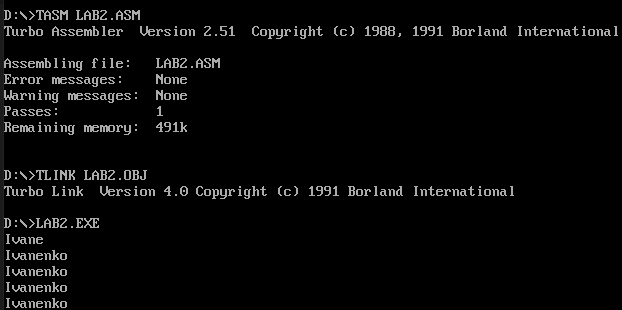
mov dx, OFFSET surname ; вывод фамилии

int 21h ;

loop BEGIN ;

END START

Результат выполнения:



4.3. Получить файл листинга и внимательно ознакомиться с его  форматом  и

     содержимым.



4.4. В любой инструкции с двумя операндами удалить один операнд и проас-

     семблировать программу с получением файла листинга.  Какие  выходные

     файлы создаются в этом случае? Что добавляется в листинге?

**Привести результаты наблюдений в отчете и объяснить**

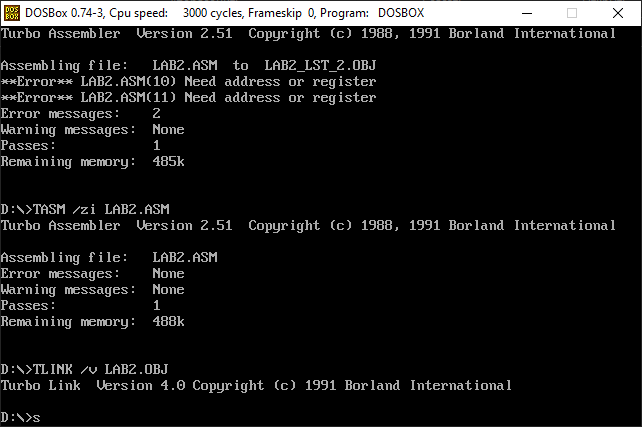
В таком случае создается только файл листинга, без объектного файла. Также, в файл листинга добавляются сообщения об ошибках:

\*\*Error\*\* LAB2.ASM(10) Need address or register

\*\*Error\*\* LAB2.ASM(11) Need address or register

4.5. Восстановить исходную программу и создать загрузочный модуль Lab2.exe

     (при этом использовать команду d:\tasm\tasm /zi lab2.asm на этапе трансляции и команду lab2.obj d:\tasm\tlink /v lab2.obj на этапе компоновки. **Привести в отчете назначение ключей /zi и /v**).

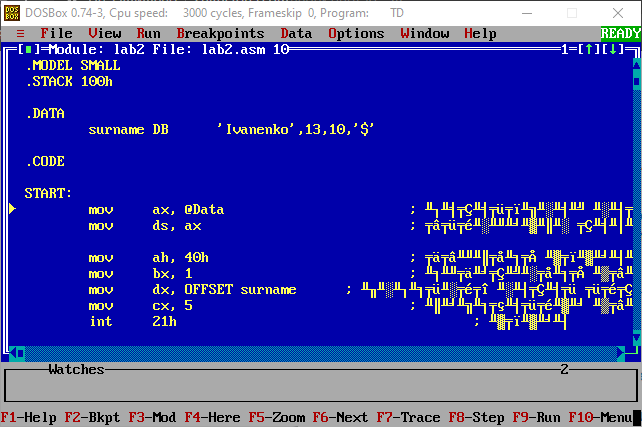


Ключ /zi включает в объектный файл информации, используемой отладчиком

Ключ /v передает в исполняемый файл информацию, позволяющую отладчику выодить текст исходной программы.

4.6. Загрузить программу в отладчик (td d:\asm\lab2). Какими способами

     это можно сделать? **Отразить способы в отчете.**

****

1 способ – td <имя исполняемого файла>

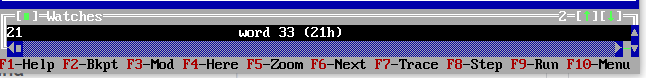
2 способ – запустить turbo debugger, File -> Open -> <имя исполняемого файла>

4.6. Выполнить программу по шагам, нажимая клавишу F7, до конца. Проcмотреть

содержимое экрана (Alt+F5). После какой операции ассемблера на экране появляются строки?

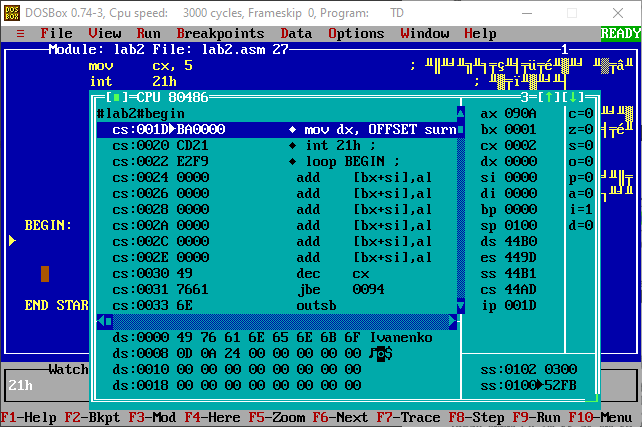
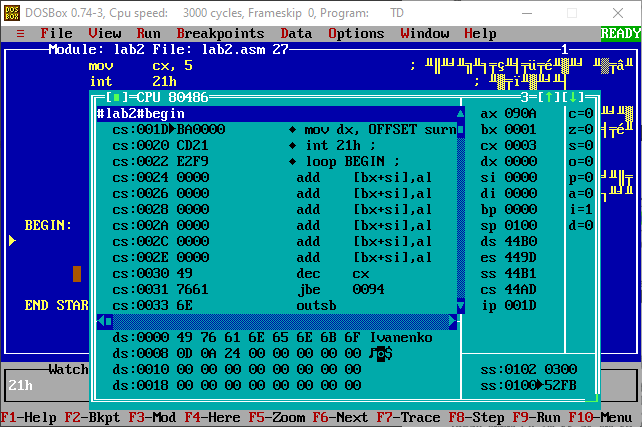
Строки появляются на экране после операции int 21h

4.7. Вывести в окне Watch (View->Watch) содержимое выходного буфера.



4.8. Выполнить  программу (Goto  cursor  -  клавиша F4) до места начала цикла.

4.9. Выполнить 2 прохода цикла по F7, контролируя значения регистров. Для этого необходимо открыть окно CPU (View->CPU). Какие регистры изменяются в цикле?



**Привести результаты выполнения данного фрагмента программы в отчете в виде таблицы:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № шага | IP | AX | BX | CX | DX |
| 1 | 001D | 090A | 0001 | 0003 | 0000 |
| 2 | 001D | 090A | 0001 | 0002 | 0000 |
| ... |  |  |  |  |  |

**Объяснить в отчете изменения регистров**

При выполнении данного задания в программе сх было присвоено значение 3.

На каждом проходе это значение уменьшалось на 1

4.10. Остальные проходы цикла выполнить по F8. В чем разница?

**Привести объяснения в отчете.**

F7 – trace – отладчик выполнит код текущей строчки кода и остановится на слудющей. Если текущая строка вызывает какую-либо процедуру или блок, то следующей строкой будет первая строка кода этого блока.

F8 – step – аналогично команде trace, но вход в тело блока не происходит.

4.11. Вывести содержимое выходного буфера в окне Watches.



4.13. Вывести содержимое регистра CX буфера в окне Watches в шестнадцатеричном

      формате (Options->Display options->Hex).



4.13. Вывести содержимое регистра CX буфера в окне Watches в десятичном

      формате (Options->Display options->Decimal).



4.14. Изменить содержимое регистра CX на величину, меньшую на 2 единицы (Data->Evaluate/Modify->Expression=cx, New Value=cx-2). Выполнить пошагово (F7) программу до конца. Что изменилось? **Объяснить в отчете.**



Цикл выполнился на 2 раза меньше, т.к. СХ содержит количество итераций цикла

4.15. Выполнить команду Animate из меню Run с задержкой 1000 мс.

Команда выполняется с задержкой между выполнением инструкций кажой строке в 1000мс.

4.16. Сбросить программу по Ctrl-F2. Выполнить первые 2 инструкции. Отменить

      их действие по Alt-F4. **Привести объяснения о результате в отчете.**

С помощью команды Alt-F4 можно отменять ранее выполненные инструкции

4.17. По шагам пройти до первой после  int  21h  инструкции.  Попытаться  по

      Alt-F4 все отменить. Что получилось? **Привести объяснения о результате в отчете.**

Ничего не произошло, т.к. int 21h является инструкцией прерывания.

4.18 Сбросить программу по Ctrl-F2.  По F7 выполнить первые  6  инструкций.

      Открыть окно Execution History. Отменить последние 3 шага.

Вывод: в ходн выполнения данной работы ознакомился со структурой ассемеблерной программы, с процессом получения и запуска исполняемого файла и с отладкой программы.