ЛАБОРАТОРНА РОБОТА №1

Робота з турбо - ассемблером

|  |
| --- |
|  |

***Мета:***ознайомлення з загальними принципами побудови програм мовою асемблера; робота з пакетом TASM; створення та налагодження файлів \*.EXE,\*.COM.

План:

1. Трансляція програми.
2. Компоновка програми.
3. Налагодження програми.

Короткі теоретичні відомості

Повний цикл створення програми мовою Асемблера складається з наступних етапів:

1. Вихідний текст програми

2. Створення об'єктного модуля

3. Створення завантажувального модуля

4.Налагодження програми

В більшості реалізацій асемблера не має інтегрованого середовища, тому для виконання операцій по набору коду програми , трансляції, редагуванню, і налагоджуванню використовують окремі модулі , що входять в спеціалізовані пакети асемблера. Найбільш широко використовують два пакета:

- "Макроассемблер" MASM фірми Microsoft;

- Turbo Assembler TASM фірми Borland.

Макроассемблер дозволяє задавати макровизначення (або макроси), тобто групи команд, які можна вставляти в програму в будь-якому місці, вказавши тільки ім'я групи.

Пакет Turbo Assembler цікавий тим, що має два режими роботи. Один з цих режимів називається MASM і підтримує основні можливості макроассемблера MASM. Другий режим IDEAL, надає більш зручний синтаксис для написання програм та ефективнішого використання пам'яті.

Трансляція програми

Процес створення виконуваного файлу відбувається в дві стадії:

.asm -> .obj -> .exe / .com

На першій стадії (.asm -> .obj) з ассемблерного файлу шляхом компіляції вихо-дить файл проміжного об'єктного коду, що має розширення .obj (при цьому можуть використовуватися додаткові inc-файли). Файл з розширенням .obj міс-тить оптимізований машинний код, за умови, що не зустрілися синтаксичні та семантичні помилки.

Наступна стадія (.obj -> .exe / .com) називається лінковкою або компонуванням і служить для заміщення символьних імен, що використовуються програмістом на реальні адреси.

Виконуваний файл може бути у двох форматах. COM і EXE.

Структура COM - файлу проста. У файлах даного типу, зазвичай не мають на-віть заголовка файлу, містяться тільки машинний код і дані програми. Розмір COM - файлу обмежений до 64 Кб, тобто розміром одного сегмента пам'яті.

Приклад 1.1. Стандартна структура СОМ-програми.

;Program Hello1

.Model Small

.Code

org 100h ;Задаємо стеком розміром 100h

begin: jmp start ;Переходимо на мітку початку програми

Hello db 'Hello!$' ;Резервуємо пам'ять під змінну Hello

start: lea dx, Hello

mov ah, 09h

int 21h

mov ah, 4ch

mov al, 00h

int 21h

end begin

Основні поняття про COM:

1. Обсяг COM-файлу обмежений до 64Кб.
2. COM-файл менший, ніж відповідний EXE-файл.
3. Програма, написана для виконання в COM-форматі не містить стека і сегмента даних і не вимагає ініціалізації регістра DS.
4. Програма, написана для виконання в COM-форматі, використовує директиву ORG 100H після директиви SEGMENT для виконання з адреси після префікса програмного сегмента.
5. Програма EXE2BIN перетворить EXE-файл у COM-файл, обумовлений указівкою типу COM у другому операнді.
6. Операційна система DOS визначає стек для COM-програми або наприкінці програми, якщо дозволяє розмір, або ж наприкінці пам'яті.

Приклад 1.2. Стандартна структура СОМ-програми.

.MODEL SMALL

.CODE

org 100h

begin: jmp start

msg db "Programm works!$"

start:

mov AH,09h

mov DX, offset msg ;Завантажуємо в DX зміщення адреси змінної msg

;відносно початку сегмента

int 21h

mov ax,4c00h

int 21h

END begin

Складатися EXE файли можуть з декількох сегментів, отже їх розмір необмежений 64 Кб. За структурою EXE файл складніший, крім коду програми у файлі також містяться: заголовок файлу, таблиця налаштування адрес, дані.

Заголовок EXE файлу - містить дані необхідні для завантаження програми.

Тіло програми - основна частина програми, що виконує будь-які корисні дії.

Кінець програми - завершальна частина програми, яка зберігає потрібні і видаляє непотрібні дані з ОЗУ, закриває всі відкриті даною програмою файли.

Перша і остання частини програми є обов'язковими для всіх EXE файлів, а друга частина зовсім не обов'язкова. Ви можете просто її пропустити.

Приклад 1.3. : Стандартна структура ЕХЕ-програми

;Programm Hello1

.model small ;Встановлюємо модель пам'яті ближнього типу

.stack 100h ;Задаємо стек розміром 100h

.data ;Початок сегмента даних

Hello DB 'Hello!$' ;Резервуємо пам'ять під змінну Hello

.code ;Початок сегмента коду

start: ;Мітка початку програми

mov ax,@data ;Формування адреси сегмента даних

mov ds,ax

LEA DX,Hello ;Завантажуємо в DX адресу змінної HELLO

mov ah,09h ;Функція DOS виводу рядка

int 21h

mov ax,4c00h ;Функція DOS завершення програми

int 21h

end start ;Кінець програми

Приклад 1.4 : Альтернативна структура ЕХЕ-програми

;Program Hello3

;Задаємо сегмент стека з вирівнюванням на параграф (PARA, 16 байт)

OutStack SEGMENT PARA STACK 'STACK'

;Резервуємо пам'ять під стек

DB 64 DUP(?)

OutStack ENDS

;Задаємо сегмент даних

OutData SEGMENT PARA PUBLIC 'DATA'

Hello DB 'Hello!$'

OutData ENDS

;Задаємо сегмент коду

OutCode SEGMENT PARA PUBLIC 'CODE'

;Зв'язуємо сегменти з їх сегментними регістрами

ASSUME CS:OutCode, DS:OutData, SS:OutStack

Start:

mov ax, OutData ;Формування адреси сегмента даних

mov ds, ax

lea dx, Hello

mov ah,09h

int 21h

mov ah, 4ch

mov al, 00h

int 21h

OutCode ENDS

END Start

Різниця між COM і EXE в тому, що:

1. У EXE - файлах присутня заголовна частина з деяким набором службових таблиць для ОС (розміром 512 байт або більше ), а COM- файли заголовка не мають і являють собою звичайний бінарник призначений для роботи в межах одного сегмента пам'яті DOS.
2. Стандартний розмір COM- файлу не може бути більше 64 Кб , а EXE - файл може мати будь-який розмір.

Приклад 1.5 : ЕХЕ-програми без сегментів стеку і даних

.MODEL SMALL ;Модель пам'яті

CODE SEGMENT ;Початок сегменту коду

ASSUME CS:CODE ;Зв’язування секції коду з сегментним регістром

START: ;Мітка початку програми

mov ah,2 ;Функція DOS виводу символу

;в ре гістрі DX

mov dl,'#'

int 21h

mov ah,4ch ;Функція DOS завершення програми

mov al,0

int 21h

CODE ENDS ;Кінець сегменту коду

END START ;Кінець програми

Також формується *об'єктний модуль,* що містить у собі представлення вихідної програми і машинних кодах Для одержання об'єктного модуля вихідний файл необхідно відтранслювати за допомогою програми tasm.exe з пакета TASM.

Формат командного рядка для запуску Tasm.exe наступний:

**Tasm [опції] імя\_файла [, імя\_обєктного\_файла]**

**[, імя\_файла\_лістингу] [, імя\_файла\_посилань]**

Обов’язковим аргументом є тільки **імя\_файлу**, а інші параметри, що записані в квадратних дужках, можуть бути відсутніми. Файл, який вказаний як обов’язковий параметр, повинен знаходитися на диску і мати розширення **.asm**. Якщо не задавати необов’язкові параметри, то вище вказані файли не будуть створенні. Якщо ж необов’язкові параметри задаються, то необхідно враховувати наступне:

Якщо імена об’єктного файлу, файлу лістингу і файлу посилань повинні співпадати з ім’ям вихідного файлу, то замість імен цих файлів необхідно поставити коми:

tasm.exe ***імя\_файла , , ,***

В результаті будуть створені файли ***імя\_файла.obj, імя\_файла.lst, імя\_файла.crt.***

Якщо необхідно вибірково створити файли, то замість непотрібних файлів необхідно поставити параметр ***nul:***

tasm.exe ***імя\_файла , , ,nul .***

В результаті будуть створені файли ***імя\_файла.obj, імя\_файла.lst.***

Необов’язковий аргумент ***опції*** дозволяє задавати режими роботи транслятора, які наведено в таблиці 1.

Компоновка програми

Головна мета створення завантажувального модуля (компонування програми) - перетворити код і дані в об'єктних файлах у їх **переміщуване виконуване відображення.** Щоб зрозуміти, у чому тут суть, потрібно розібратися, навіщо розділяють процес створення модуля, що виконується, на два кроки — трансляцію і компонування. Це зроблено для того, щоб можна було поєднувати разом кілька модулів (написаних на одній чи декількох мовах). Формат об'єктного файлу дозволяє, при визначених умовах об'єднати окремо відтрансльовані результати модулів в один модуль. При цьому у функції компоновщика входить дозвіл зовнішніх посилань у цих модулях. Результатом роботи компоновщика є створення файлу з розширенням .ехе. Після цього операційна система може завантажити такий файл у пам'ять і виконати його.

Повний формат командного рядка для запуску компоновщика:

**TLINK [опції] список\_об’єктних\_файлів [, ім'я\_завантажувального модуля] [, імя\_файлу\_карти] [,ім'я\_файлу\_бібліотеки] [, імя\_файлу\_визначень] [,імя\_ресурсного\_файлу].**

1. опції - необов’язкові параметри, що керують роботою компоновщика. Кожній опції повинен передувати один з наступних символів: «-» чи «/»;
2. список\_об’єктних\_файлів - обов'язковий параметр, що містить список компонованих файлів з розширенням .obj. Файли повинні бути розділені пропусками чи знаком <+>, наприклад:
3. t1іnk /v prog + mdf + fdr
4. При необхідності вказують шлях до файлів;
5. імя\_завантажувального\_модуля - необов’язковий параметр, що означає ім'я завантажувального модуля. Якщо воно не зазначено, то ім'я завантажувального модуля буде співпадати з першим зі списку імен об'єктних файлів;
6. ім'я\_файлу\_карти — необов'язковий параметр, наявність якого вказує компоновщику створити спеціальний файл із картою завантаження. У ній перераховуються імена, адреси завантаження і розміри всіх сегментів, що входять у програму;
7. імя\_файлу\_бібліотеки— необов'язковий параметр, що який вказує шлях до файлу бібліотеки (.1іb). Цей файл створюється й обслуговується спеціальною утилітою tlіb.exe . Утиліта дозволяє об'єднати часто використовувані підпрограми у виді об'єктних модулів і один файл.
8. ім'я\_файлу\_визначень - необов'язковий параметр, який вказує шлях до файлу визначень (.def). Цей файл використовується при компонуванні Windows-додатків;
9. імя\_ресурсного\_файла — необов'язковий параметр, що вказує шлях до файлу з ресурсами Wіndows-додатків (.res). Цей файл використовується при компонуванні Wіndows-додатків.

Розглянутий формат командного рядка використовується і для 32-розрядного варіанта компоновщика t1ink32.exe.

Опції компоновщика (редактора зв'язків) TLINK

|  |  |
| --- | --- |
| /x | Не створювати файл карти (map) |
| /m | Створити файл карти |
| /s | Те ж, що /m, але додатково у файл карти включається інформація про сегменти (адреса, довжина в байтах, клас, ім'я сегмента і т.д.) |
| /l | Створити розділ у файлі карти з номерами рядків |
| /n | Ігнорувати бібліотеки, що вказуються іншими компіляторами |
| /c | Розрізняти рядкові і прописні букви в ідентифікаторах (у тому числі і зовнішніх) |
| /v | Включити відлагоджувальну інформацію у виконуваний файл |
| /3 | Підтримка 32-бітного коду |
| /d | Попереджати про дублювання символів у компонуємих бібліотеках |
| /t | Створити файл типу .com (за замовчуванням .exe) |

Обов'язковим етапом процесу розробки є *налагодження.*

На етапі налагодження, використовуючи опис алгоритму, виконується контроль правильності функціонування як окремих ділянок коду, так і всієї програми в цілому.

Специфіка програм на ассемблері полягає в тому, що вони інтенсивно працюють з апаратними ресурсами комп'ютера. Ця обставина змушує програміста постійно відслідковувати вміст регістрів і областей пам'яті. Для локалізації логічних помилок у програмах використовують спеціальний тип програмного забезпечення — програмні відлагоджувальники*.* Відлагоджувальники бувають двох типів:

1. **інтегровані** - відлагоджувальник реалізований у інтегрованому середовищі мов програмування типу Turbo Pascal, Quick С и т. д.;
2. **автономні** *—* відлагоджувальник - окрема програма. Через те, що ассемблер не має свого інтегрованого середовища, для налагодження написаних на ньому програм використовують автономні відлагоджувальники.

**Завдання до лабораторної роботи №1**

**Перший рівень**

**Завдання 1.** Ввести текст наступної програми. Створити файл HELLO.EXE, виконавши асемблювання та компоновку файлу HELLO.ASM. Ключі вибираються самостійно згідно отриманих теоретичних відомостей. Внести зміни в програму (замість слова HELLO! Набрати будь-яке інше повідомлення). Створити командний файл з розширенням .BAT, наприклад, TEST.BAT . Для того, щоб файл працював для будь-якої програми в ньому замість імені файлу програми наберіть символи %1. Тоді вміст командного файлу матиме вигляд:

TASM /L %1  
TLINK %1  
%1.EXE

Код програми:

; Programm Hello1 ( Спрощене оформлення програми )  
.model small        ; Модель пам’яті ближнього типу  
.stack 100h           ;Визначає стек розміром 100h  
.data                 ;Початок сегмента даних  
Hello DB 'Hello!$'   ;Зарезервували пам’ять для змінної HELLO  
.code                 ;Початок сегмента коду  
start:  
mov ax,@DATA          ;Формування адреси сегмента  
mov ds,ax              ;даних  
LEA DX,Hello           ;встановити в DX адресу змінної HELLO  
MOV ah,09h             ;Функція DOS виводу рядка  
INT 21h                 ;Виведення рядка на екран  
MOV ax,4C00h             ;Функція DOS завершення  
INT 21h                    ;програми  
END start                 ;кінець програми



**Завдання 2.**

За допомогою відлагоджувальника Turbo Debugger (TD), виконати трасування скомпільованої Вами програми із **Завдання 1**, виконати перегляд і зміну стану апаратних ресурсів мікропроцесора під час покомандного (покрокового) виконання програми. Проаналізувати програму за допомогою відлагоджувальника, та дати відповіді на питання:

* З якої фізичної адреси програма завантажена в пам’ять комп’ютера і чому?
* Який вміст регістрів загального призначення (AX, BX, CX, DX, SI, DI і ВР).
* Який вміст після завантаження програми в пам'ять мають регістри DS і ES. Чому ? Який вміст регістра CS. Чому ?

**Другий рівень**

**Завдання 1.** Ввести текст наступної програми. Створити файл HELLO.COM, виконавши асемблювання та компоновку файлу HELLO.ASM. Ключі вибираються самостійно згідно отриманих теоретичних відомостей. Внести зміни в програму (замість слова HELLO! Набрати будь-яке інше повідомлення). Створити командний файл з розширенням .BAT, наприклад, TEST.BAT.

Код програми:

; Programm Hello3   
.MODEL SMALL  
.CODE  
org 100h  
begin: jmp start  
Hello DB 'Hello!$'  
start: LEA DX,Hello  
MOV AH,09h  
INT 21h  
MOV AH,4Ch  
MOV AL,00h  
INT 21h  
END begin

**Завдання 2.** Ввести текст наступної програми. Створити файл HELLO.EXE, виконавши асемблювання та компоновку файлу HELLO.ASM. Ключі вибираються самостійно згідно отриманих теоретичних відомостей. Внести зміни в програму (замість слова HELLO! Набрати будь-яке інше повідомлення). Створити командний файл з розширенням .BAT, наприклад, TEST.BAT .

Код програми:

;X =(A\*2+B\*C)/(D-3)

;prog1

.MODEL SMALL

.STACK 100H

.DATA

;Резервуємо память для змінних

;A,B,C,D,X

A DB ?

B DB ?

C DB ?

D DB ?

X DW ?

.CODE

start:

MOV AX,@DATA

MOV DS,AX

MOV A,3

MOV B,4

MOV C,2

MOV D,5

MOV AL,2

MUL A

MOV CX,AX

MOV AL,B

MUL C

ADD AX,CX

MOV CL,D

SUB CL,3

DIV CL

MOV X,AX

MOV AH,4CH

INT 21H

END start

**Завдання 3.** Внести зміни у програму з **Завдання 2.** з метою оформлення програми у вигляді \*.COM файлу. Здійснити асемблювання та компоновку файлу HELLO.ASM.

**Завдання 4.** За допомогою відлагоджувальника Turbo Debugger (TD), виконати трасування скомпільованих Вами програм із **Завдання1**, **Завдання2** та **Завдання3**, виконати перегляд і зміну стану апаратних ресурсів мікропроцесора під час покомандного (покрокового) виконання програми. Проаналізувати програму за допомогою відлагоджувальника, та дати відповіді на питання:

* З якої фізичної адреси програма завантажена в пам’ять комп’ютера і чому?
* Який вміст регістрів загального призначення (AX, BX, CX, DX, SI, DI і ВР) . Чому ?
* Який вміст після завантаження програми в пам'ять мають регістри DS і ES. Чому ?
* Який вміст регістра CS. Чому ?

Виконати покроково дві перші команди (F8) та дайте відповіді на питання:

* + Чому не змінився стан регістра прапорів ?
  + Скільки пам’яті виділено під сегмент даних ?
  + Чим пояснюється вміст сегментного регістра стека ?
  + Що означають команди нижньої половини вікна команд add[bx+si],al ?

**Третій рівень**

**Завдання 1.** Ввести текст наступної програми. Створити файл HELLO.COM, виконавши асемблювання та компоновку файлу HELLO.ASM. Ключі вибираються самостійно згідно отриманих теоретичних відомостей. Внести зміни в програму (замість слова HELLO! Набрати будь-яке інше повідомлення). Створити командний файл з розширенням .BAT, наприклад, TEST.BAT.

Код програми:

; Programm Hello3   
.MODEL SMALL  
.CODE  
org 100h  
begin: jmp start  
Hello DB 'Hello!$'  
start: LEA DX,Hello  
MOV AH,09h  
INT 21h  
MOV AH,4Ch  
MOV AL,00h  
INT 21h  
END begin

**Завдання 2.** Внести зміни у програму з **Завдання 1.** з метою оформлення програми у вигляді \*.EXE файлу. Здійснити асемблювання та компоновку файлу HELLO.ASM.

**Завдання 3.** Ввести текст наступної програми. Створити файл HELLO.EXE, виконавши асемблювання та компоновку файлу HELLO.ASM. Ключі вибираються самостійно згідно отриманих теоретичних відомостей. Створити командний файл з розширенням .BAT, наприклад, TEST.BAT .

Код програми:

IDEAL

MODEL small

STACK 256

DATASEG

r dw 34h

CODESEG

Start: mov ax, @data

mov ds, ax

mov ax, 65h

mov [r], ax

mov si, [r]

mov bx, 70h

mov [r], bx

mov di, [r]

mov cx, 40h

mov [r], cx

mov bp, [r]

Exit: mov al, 04ch

int 21h

END Start

**Завдання 4.** Внести зміни у програму з **Завдання 3.** з метою оформлення програми у вигляді \*.COM файлу. Здійснити асемблювання та компоновку файлу HELLO.ASM.

**Завдання 5.** За допомогою відлагоджувальника Turbo Debugger (TD), виконати трасування скомпільованої Вами програми із **Завдань1-4** виконати перегляд і зміну стану апаратних ресурсів мікропроцесора під час покомандного (покрокового) виконання програми. Проаналізувати програму за допомогою відлагоджувальника, та дати відповіді на питання:

З якої фізичної адреси програма завантажена в пам’ять комп’ютера і чому?

Який вміст регістрів загального призначення (AX, BX, CX, DX, SI, DI і ВР). Який вміст після завантаження програми в пам'ять мають регістри DS і ES. Чому ? Який вміст регістра CS. Чому ?

Виконати покроково дві перші команди (F8) та дайте відповіді на питання:

* + Чому не змінився стан регістра прапорів ?
  + Скільки пам’яті виділено під сегмент даних ?
  + Чим пояснюється вміст сегментного регістра стека ?
  + Що означають команди нижньої половини вікна команд add[bx+si],al ?

Відредагувати програму, замінивши рядок в тексті програми ***data segment*** на ***data segment byte.*** Створити об’єктний модуль (\*.obj), файл лістингу (\*.lst), та завантажувальний модуль (\*.exe).

Проаналізувати програму за допомогою відлагоджувальника, та дати відповіді на питання : чому програма займає менше пам’яті?

Проаналізувати програму цю програму за допомогою відлагоджувальника.

**Контрольні питання**

1. Поясніть, яким чином визначається стек для а) програми типу .ЕXE. б) програми типу .СОМ.
2. Охарактеризуйте групу сегментних регістрів.
3. Охарактеризуйте групу регістрів загального призначення.
4. Охарактеризуйте групу регістрів-покажчиків SP і BP.
5. Охарактеризуйте групу індексних регістрів.
6. Навіщо потрібен регістр ІР? Поясніть призначення прапорів регістру FR.
7. Які функції виконує машинна команда?
8. У чому відмінність програм типу .СОМ від програм типу .ЕХЕ?