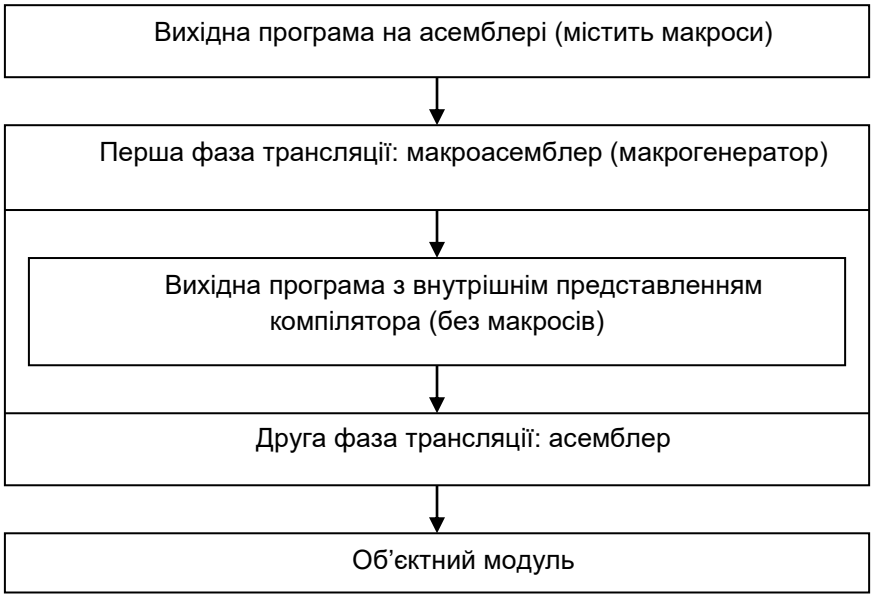
Лабораторна робота №8
Макрозасоби мови Асемблер

Мета роботи: Вивчення макросів, макрокоманд та макророзширень та їх застосування в асемблерних програмах.

Теоретичні відомості

Обробка програми на асемблері з використанням макрозасобів неявно здійснюється транслятором у дві фази:



Макроасемблер в загальній схемі трансляції програми TASM

Макрозасоби Асемблера IBM PC мають три складові:

1. **Макровизначення (макрос) –** набір команд, який містить опис якоїсь дії або алгоритму. *Макрос повинен знаходитися на початку програми, до визначення сегментів.*

Синтаксис *макровизначення*:

ім'я_макроса MACRO [список_формальних_аргументів]
;; тіло макровизначення
Endm

2. **Макрокоманда –** коротке посилання на макровизначення (виклик макроса):

ім'я_макроса MACRO [список_формальних_аргументів]

3**. Макророзширення** (макропідстановка, макровставка) – вставка замість макрокоманди макроса з заміною формальних параметрів на фактичні (якщо вони є).

Макровизначення може простим та вкладеним, тобто містити у собі інше

макровизначення. Рівень вкладання макровизначень може бути будь-яким, з одного макроса можна викликати інші макроси.

Існує три варіанти де повинні розташовуватися макровизначення:

1. *На початку тексту програми* до сегмента коду та даних. Цей варіант використовується тоді, коли визначені користувачем макрокоманди є актуальними в межах однієї програми.
2. *В окремому файлі*.

Такий варіант підходить при роботі з декількома програмами однієї проблемної області. Для того, щоб зробити доступними макровизначення у конкретній програмі, слід записати директиву **include ім'я_файл**у.

Наприклад:

```
.model      small
include     show.inc
;в це місце буде вставлено текст файлу show.inc
   ...
```

3. *В макробібліотеці.*

Універсальні макрокоманди, які часто використовуються в програмах користувача, (наприклад, фрагменти програмної затримки, призупинення програми до натискання клавіші, перетворення двійкових чисел у символьну форму) доцільно записати в макробібліотеку. Макробібліотека являє собою файл з текстами макровизначень, які записуються у цей файл, як у текст програми. Файл макробібліотеки може мати будь-яке ім'я і розширення, наприклад, MYMACRO.MAC. В програмі залишаються тільки макровиклики. Включати макрокоманди з цієї бібліотеки в програму також можна за допомогою директиви **include (**наприклад**,** include mymacro.mac **)**. Після цього у програмі можна використовувати будь-які макрокоманди з цієї макробібліотеки.

Недоліком двох останніх способів є той факт, що у вихідний текст програми включаються абсолютно усі макровизначення. Для усунення цього недоліку використовується директива **purge**, в якості операндів через кому слід перелічити імена макрокоманд, які не повинні включатися в тіло програми.

PURGE ім'я_макроса відміняє визначений раніше макрос (не підтримується WASM). Ця директива часто застосовується відразу після INCLUDE, програми, що включила в текст, файл з великою кількістю готових макроозначень.

Наприклад:

```
...
include     iomac.inc
```

```
purge       _outstr,_exit
        ...
```

В цьому прикладі у вихідний текст програми перед початком компіляції TASM замість рядка **include iomac.inc** вставить рядки з файлу **iomac.inc**, однак у ньому будуть відсутні макровизначення **_outstr** та **_exit**.

Для організації циклу у макровизначенні мітку треба оголосити за допомогою оператора **local**.

LOCAL мітка... перераховує мітки, які застосовуватимуться усередині макроозначення, щоб не виникало помилки «мітка вже визначена» при використанні макросу більше одного разу або якщо та ж мітка присутня в основному тексті програми (в WASM директива LOCAL дозволяє використовувати макрос з мітками кілька разів, але не дозволяє застосовувати мітку з тим же ім'ям в програмі). Операнд для LOCAL мітка або список міток, які використовуватимуться в макросі.

```
;   виведення на екран рядка з 10 зірочок, який може слугувати
;    розділювачем у фрагменті тексту в екранному кадрі
starts      macro
local       outpt
            mov  CX, 10       ;       лічильник циклу
            mov  AH,  02h     ;  функція виведення символу
            mov  DL, '*'      ;  символ *
outpt:      int     21h   ;  виклик DOS
            loop    outpt
```

Наприклад, написати програму цілочисельного знакового ділення для будь-яких 16-розрядних даних: Z=a/в. Операцію ділення та виведення рядка виконати у вигляді макроса.

; Макровизначення

```
Init    Macro
        mov ax, @data     ; ініціалізація сегменту даних
        mov ds, ax
endm

Vuraz       Macro     z, a, b;      z=a/b
      push      ax bx dx
      mov       ax, a
      mov       bx, b
```

```
        cwd
        idiv    bx
        mov     z, ax
        pop     dx bx ax
endm
```

Plus Macro ; Виведення на екран позитивного результату обчислення – переведення у десяткову систему і перетворення результату обчислення у ASCII-коди

```
        xor ax, ax
        mov ax, z
        mov bl, 1010b       ; основа 10-ї системи счислення
        div bl
        add ax, 30h         ; перетворення 1-го залишку у ASCII-код
        mov s, ax
        xor ax,ax
        mov ax, z
        div bl
        mul bl
        mov z1, ax
        mov ax,z
        sub ax, z1
        add ax, 30h
        mov s+1, ax
        mov s+2, '$'
        MessageDX s
        jmp a20
endm
```

Minus Macro ; Виведення на екран негативного результату обчислення – перетворення результату обчислення у ASCII-коди

```
        xor ax, ax
        xor bx, bx
        mov ax, z
        mov bx, m       ; знак ,,-"
        mov s,bx            ; рядок формування результату
        sub ax, 0001b  ; перетворення негативного результату
        xor ax, 0ffffh  ; у позитивне
        mov z2, ax
```

```
        mov ax, z2
        xor bx,bx
        mov bl, 1010b
        div bl
        add ax, 30h
        mov s+1, ax
        xor ax,ax
        mov ax, z2
        div bl
        mul bl
        mov z3, ax
        mov ax, z2
        sub ax, z3
        add ax, 30h
        mov s+2, ax
        mov s+3, '$'
        MessageDX s
endm

MessageDX       Macro       s
    ; Виведення рядка символів
    mov         ah, 9h
    mov dx, offset s
    int         21h
endm

Endpr   Macro       ; Успішне завершення програми
     mov ax, 4c00h
     int  21h
endm

.model small
.stack 100h
.data
    z     dw    ?
    z1    dw    ?
    z2    dw    ?
    z3    dw    ?
    a     dw    -278
```

```
        b     dw    22
        s     dw    5dup(?)
        m     dw    '-'
    .code
        begin  proc far
                    init
                    vuraz  z, a, b
                    cmp z, 0
                    jl a10
                    plus
              a10:
                    minus
              a20:
                    endpr
        begin  endp
        end begin
```

Індивідуальні завдання:

Створіть максимальну можливу кількість макросів для вашої задачі та вставте у програму їх виклики.

1. В сегменті даних створити двовимірний масив, який складається з 25 натуральних чисел, розміщених у п'яти рядках та п'яти стовпцях. Вивести елементи на екран. Подвоїти значення елементів масивів, для яких номер рядка дорівнює номеру стовпця. Вивести елементи на екран.
2. В сегменті даних створити двовимірний масив, який складається з 20 натуральних чисел, розміщених у чотирьох рядках та п'яти стовпцях (масив відобразити на екрані). Зменшити втричі значення останніх елементів першого та останнього рядків. Вивести елементи на екран.
3. В сегменті даних створити двовимірний масив, який складається з 15 натуральних чисел, розміщених у трьох рядках та п'яти стовпцях (масив відобразити на екрані). Піднести до другого ступеня останній елемент кожного стовпця. Результат вивести на екран.
4. В сегменті даних створити двовимірний масив, який складається з 25 натуральних чисел, розміщених у п'яти рядках та п'яти стовпцях (введений масив вивести на екран). Знайти найбільший елемент масиву та вивести на екран.
5. В сегменті даних створити двовимірний масив, який складається з 20 натуральних чисел, розміщених у чотирьох рядках та п'яти стовпцях (масив відобразити на екрані). Знайти найменший елемент масиву та вивести його на екран.
6. В сегменті даних створити двовимірний масив, який складається з 15 натуральних чисел, розміщених у трьох рядках та п'яти стовпцях (масив відобразити на екрані). Обчислити середнє значення у кожному рядку (вивести

значення на екран) та додати це значення до відповідного елемента у рядку. Результат вивести на екран.

7. В сегменті даних створити двовимірний масив, який складається з 25 натуральних чисел, розміщених у п’яти рядках та п’яти стовпцях. Вивести елементи на екран. Кожний п’ятий елемент масиву помножити на 10 та відняти 2. Результат вивести на екран.
8. В сегменті даних створити двовимірний масив, який складається з 20 натуральних чисел, розміщених у чотирьох рядках та п’яти стовпцях (масив відобразити на екрані). Кожний четвертий елемент поділити на два та додати до нього 4. Результат вивести на екран.
9. В сегменті даних створити двовимірний масив, який складається з 24 натуральних чисел, розміщених у чотирьох рядках та шести стовпцях (введений масив вивести на екран). До кожного третього елемента додати 12 та помножити на 3. Результат вивести на екран.
10. В сегменті даних створити двовимірний масив, який складається з 18 натуральних чисел, розміщених у трьох рядках та шести стовпцях (введений масив вивести на екран). Кожний третій елемент помножити на 3 та відняти 7. Результат вивести на екран.
11. В сегменті даних створити лінійний масив, який містить 12 символьних величин. Вивести їх у зворотному порядку, розмістивши по три в рядок через чотири пропуски. Результат вивести на екран.
12. В сегменті даних створити масив, який містить 14 чисел. Створити інший масив, в якому спочатку розміщені всі від’ємні елементи першого масиву, а потім всі додатні елементи масиву цього ж масиву.
13. В сегменті даних створити масив А, який містить 12 елементів, створити масив В з такою ж кількістю чисел. Утворити масив С , який є поєднанням перших двох масивів, помножений на 3. Вивести масив С, упорядкувавши за спаданням.
14. В сегменті даних створити лінійний масив А, який містить парну кількість елементів. Створити масив В, елементи якого буде утворено за схемою: перший є сумою перших двох елементів масиву А, другий – сумою – третього і четвертого і так до кінця.
15. В сегменті даних створити квадратну матрицю 5*5. Діагональні елементи помножити на номер рядка, в якому він знаходиться.
16. В сегменті даних створити квадратну матрицю 5*5. Діагональні елементи піднести до другого ступеня.
17. В сегменті даних створити квадратну матрицю 5*5. До кожного діагонального елемента додати 10.
18. В сегменті даних створити двовимірний масив 7*3. Кожний другий елемент збільшити на 7.
19. В сегменті даних створити двовимірний масив 6*3. Для кожного третього елементу треба змінити знак на протилежний та додати 3.
20. В сегменті даних створити двовимірний масив 5*3, кожний наступний елемент якого є сумою попереднього та 10. Від діагональних елементів відняти два.

Контрольні питання

1. Прокоментуйте лістинг вашої програми.
2. Способи розташування макросів.