

КАФЕДРА ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ И СЕТЕЙ

Преподаватель

Старший преподаватель

Д.В.Куртяник

Отчёт

по лабораторной работе №3

по дисциплине Программирование на языках Ассемблера

на тему: «Разработка программы с использованием макрокоманд»

Работу выполнил

студент гр. 4141

В.С. Сыворотнев

**Цель лабораторной работы:** изучение методики логической структуризации ассемблерной программы с помощью процедур, рассмотрение способов вызова процедур, изучение функциональных возможностей макроопределений, расширение представления о возможностях синтаксиса ассемблера

## 18 Вариант

Задан массив чисел размером в байт. Найти максимальный элемент массива.

### 1. Использование процедур.

Т.к. задание идентичное, была использована математическая модель из прошлой лабораторной работы. Изначально был объявлен прототип процедуры findMax proto, куда были занесены аргументы. Затем была использована команда invoke, чтобы занести аргументы в стек. После чего была объявлена процедура с алгоритмом поиска максимального элемента массива findMax proc, необходимые ей аргументы и окончание процедуры findMax endp.

### Код программы

```
.8086
.MODEL small, stdcall
.stack 100h

.DATA
    array byte 0, 2, 9, 3, 1
    findMax PROTO pArray: PTR byte

.CODE
start:
    mov AX, @DATA
    mov DS, AX

    invoke findMax, offset array

    mov AX, 4Ch
    int 21h

findMax proc pArray:PTR byte
    mov CX, 5
    mov SI, offset array

    mov BX, 0
```

```

    @m1:
        lodsb
        cmp AL, BH

        jg max

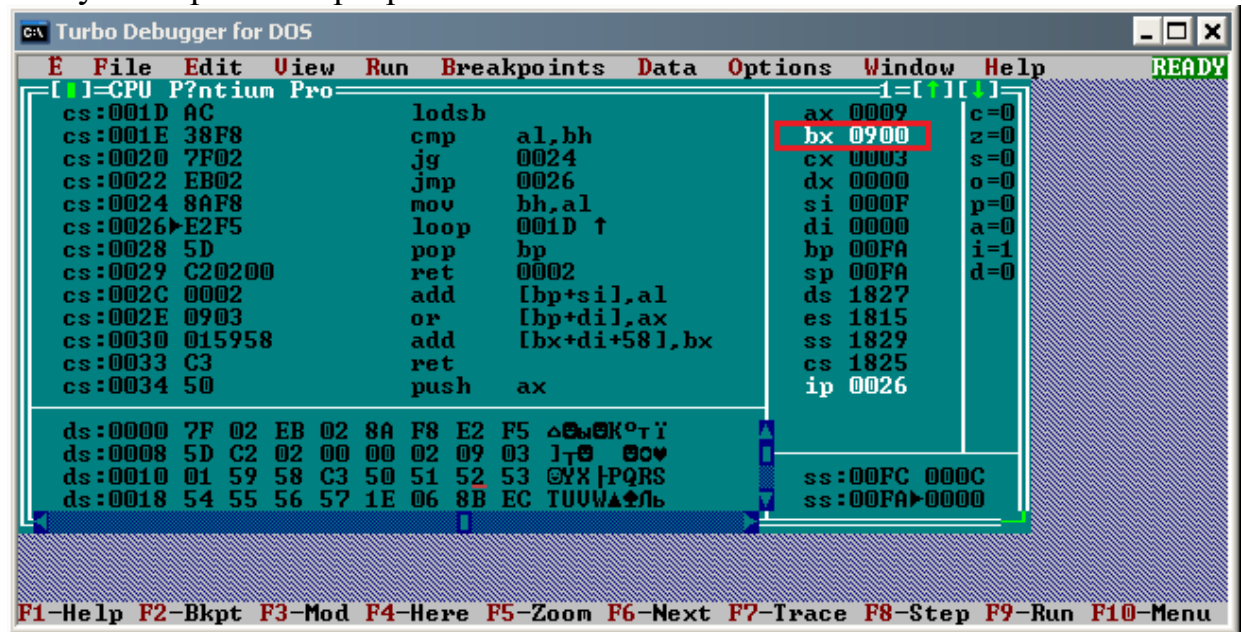
        jmp @@m2
max:
    mov BH, AL
@@m2:
    loop @m1

ret
findMax ENDP

end start

```

## Результат работы программы



## 2.Использование макросов

Было объявлено название макросов и их аргументы(dSeg macro, exit macro, findMax macro), затем в теле макросов было описано, что они делают, затем было прописано их окончание с помощью endm. Далее они были вызваны в коде программы с помощью имени макроса и при его аргумента, если он есть.

## Код программы

```
.8086
.MODEL small, stdcall
.stack 100h

.DATA
    array byte 0, 2, 9, 3, 1

dSeg macro
    mov AX, @DATA
    mov DS, AX
ENDM

exit macro
    mov AX, 4Ch
    int 21h
ENDM

findMax macro array
    mov CX, 5
    mov SI, offset array

    mov BX, 0

    @m1:
        lodsb
        cmp AL, BH

        jg max

        jmp @@m2
    max:
        mov BH, AL
    @@m2:
        loop @m1

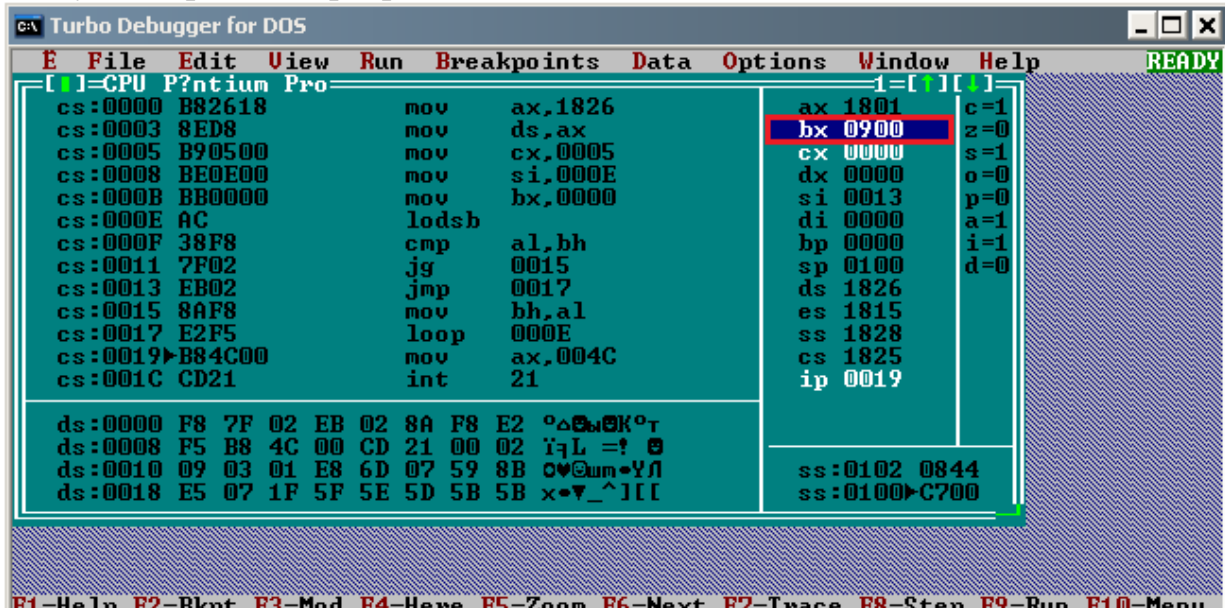
ENDM

.CODE
start:

dSeg
findMax offset array
exit

end start
```

## Результат работы программы



The screenshot shows the Turbo Debugger for DOS interface. The main window displays assembly code for a program running on a Pentium processor. The code is as follows:

```
cs:0000 B82618 mov ax,1826
cs:0003 8ED8 mov ds,ax
cs:0005 B90500 mov cx,0005
cs:0008 BE0E00 mov si,000E
cs:000B BB0000 mov bx,0000
cs:000E AC lodsb
cs:000F 38F8 cmp al,bh
cs:0011 7F02 jg 0015
cs:0013 EB02 jmp 0017
cs:0015 8AF8 mov bh,al
cs:0017 E2F5 loop 000E
cs:0019 B84C00 mov ax,004C
cs:001C CD21 int 21
```

On the right side, the register values are displayed:

```
ax 1801 c=1
bx 0900 z=0
cx 0000 s=1
dx 0000 o=0
si 0013 p=0
di 0000 a=1
bp 0000 i=1
sp 0100 d=0
ds 1826
es 1815
ss 1828
cs 1825
ip 0019
```

At the bottom, the status bar shows the following keyboard shortcuts:

```
F1-Help F2-Bkpt F3-Mod F4-Here F5-Zoom F6-Next F7-Trace F8-Step F9-Run F10-Menu
```

Так как результат работы программ идентичен, обе программы работают корректно.

**Вывод:** я изучил методики логической структуризации ассемблерной программы с помощью процедур, рассмотрел способы вызова процедур, изучил функциональные возможности макроопределений, расширил представления о возможностях синтаксиса ассемблера.