

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ
по лабораторной работе №2
по дисциплине «Организация ЭВМ и систем»
Тема: Изучение режимов адресации и формирования исполнительного
адреса.

Студентка гр. 9383

Карпекина А.А.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Изучить режимы адресации в языке программирования Ассемблер и исправить ошибки в программе.

Текст задания.

Лабораторная работа 2 предназначена для изучения режимов адресации, использует готовую программу `lr2_comp.asm` на Ассемблере, которая в автоматическом режиме выполняться не должна, так как не имеет самостоятельного функционального назначения, а только тестирует режимы адресации. Поэтому ее выполнение должно производиться под управлением отладчика в пошаговом режиме.

В программу введен ряд ошибок, которые необходимо объяснить в отчете по работе, а соответствующие команды закомментировать для прохождения трансляции.

Необходимо составить протокол выполнения программы в пошаговом режиме отладчика по типу таблицы 1 предыдущей лабораторной работы и подписать его у преподавателя.

На защите студенты должны уметь объяснить результат выполнения каждой команды с учетом используемого вида адресации. Результаты, полученные с помощью отладчика, не являются объяснением, а только должны подтверждать ваши объяснения.

Исходный код.

```
; Программа изучения режимов адресации процессора IntelX86
EOL EQU '$'
ind EQU 2
n1 EQU 500
n2 EQU -50

; Стек программы
AStack SEGMENT STACK
    DW 12 DUP(?)
```

```

AStack ENDS

;Данные программы
DATA      SEGMENT ;
;Директивы описания данных
mem1      DW      0
mem2      DW      0
mem3      DW      0
vec1      DB      12,11,10,9,5,6,7,8
vec2      DB      -40,-50,40,50,-20,-30,20,30
matr      DB      5,6,7,8,-8,-7,-6,-5,1,2,3,4,-4,-3,-2,-1
DATA      ENDS

; Код программы
CODE      SEGMENT
          ASSUME CS:CODE, DS:DATA, SS:AStack

; Головная процедура
Main      PROC FAR
          push DS
          sub AX,AX
          push AX
          mov AX,DATA
          mov DS,AX

; ПРОВЕРКА РЕЖИМОВ АДРЕСАЦИИ НА УРОВНЕ СМЕЩЕНИЙ
; Регистровая адресация
          mov ax,n1
          mov cx,ax
          mov bl,EOL
          mov bh,n2

; Прямая адресация
          mov mem2,n2
          mov bx,OFFSET vec1
          mov mem1,ax

; Косвенная адресация

```

```

        mov  al,[bx]
        mov  mem3,[bx]
;   Базированная адресация
        mov  al,[bx]+3
        mov  cx,3[bx]
;   Индексная адресация
        mov  di,ind
        mov  al,vec2[di]
        mov  cx,vec2[di]
;   Адресация с базированием и индексированием
        mov  bx,3
        mov  al,matr[bx][di]
        mov  cx,matr[bx][di]
        mov  ax,matr[bx*4][di]

;   ПРОВЕРКА РЕЖИМОВ АДРЕСАЦИИ С УЧЕТОМ СЕГМЕНТОВ
;   Переопределение сегмента
;   ----- вариант 1
        mov  ax, SEG vec2
        mov  es, ax
        mov  ax, es:[bx]
        mov  ax, 0
;   ----- вариант 2
        mov  es, ax
        push ds
        pop  es
        mov  cx, es:[bx-1]
        xchg cx,ax
;   ----- вариант 3
        mov  di,ind
        mov  es:[bx+di],ax
;   ----- вариант 4
        mov  bp,sp
        mov  ax,matr[bp+bx]
        mov  ax,matr[bp+di+si]

```

```

; Использование сегмента стека
    push    mem1
    push    mem2
    mov     bp, sp
    mov     dx, [bp]+2
    ret     2
Main      ENDP
CODE      ENDS
END Main

```

Ошибки, обнаруженные в коде.

Строка «mov mem3,[bx]» (46) - error A2052: Improper operand type. Нельзя писать в память и читать из памяти одной командой.

Строки «mov cx, vec2[di]» и «cx,matr[bx][di]» (53,57) - warning A4031: Operand types must match. Несоответствие типов операнд. Размер элемента cx – 2 байта, а размер vec2[di] и matr[bx][di] – 1 байт.

Строка «ax,matr[bx*4][di]» (58) - error A2055: Illegal register value. Нельзя умножать 16-битовые регистры.

Строка «ax,matr[bp+bx]» (78) - error A2046: Multiple base registers. Множественно использование базовых регистров. Разрешен только 1.

Строка «ax,matr[bp+di+si]» (79) - error A2047: Multiple index registers. Множественное использование индексных регистров. Разрешен только 1.

Листинг успешной программы.

```

; Программа изучения режимов
; Чов адресации процессора I
ntelX86

= 0024                                EOL EQU '$'
= 0002                                ind EQU 2
= 01F4                                n1 EQU 500
=-0032                                n2 EQU -50

```

```

; Стек программы
0000          AStack SEGMENT  STACK
0000  000C[          DW 12 DUP(?)
      ???
      ]

0018          AStack  ENDS

;Данные программы
0000          DATA          SEGMENT ;
;Директивы описания данны
x
0000  0000          mem1      DW      0
0002  0000          mem2      DW      0
0004  0000          mem3      DW      0
0006  0C 0B 0A 09 05 06 vec1      DB      12,11,10,9,5,6,7,8
      07 08
000E  D8 CE 28 32 EC E2 vec2          DB
-40,-50,40,50,-20,-30,20,30
      14 1E
0016  05 06 07 08 F8 F9 matr          DB
5,6,7,8,-8,-7,-6,-5,1,2,3,4,-4,
      -3,-2,-1
      FA FB 01 02 03 04
      FC FD FE FF
0026          DATA          ENDS

; Код программы
0000          CODE          SEGMENT
      ASSUME CS:CODE, DS:DATA, SS:AStack

; Головная процедура
0000          Main          PROC  FAR
0000  1E          push  DS
0001  2B C0          sub   AX,AX
0003  50          push  AX

```

```

0004 B8 ---- R          mov    AX,DATA
0007 8E D8              mov    DS,AX

;   ПРОВЕРКА РЕЖИМОВ АДРЕСА
;   ЦИИ НА УРОВНЕ СМЕЩЕНИЙ
;   Регистровая адресация
0009 B8 01F4            mov    ax,n1
000C 8B C8              mov    cx,ax
000E B3 24              mov    bl,EOL
0010 B7 CE              mov    bh,n2

;   Прямая   адресация
0012 C7 06 0002 R FFCE  mov    mem2,n2
0018 BB 0006 R          mov    bx,OFFSET vec1
Microsoft      (R)      Macro      Assembler      Version      5.10
10/14/20 22:58:3

```

Page 1-2

```

001B A3 0000 R          mov    mem1,ax

;   Косвенная адресация
001E 8A 07              mov    al,[bx]

;           mov    mem3,[bx]

;   Базированная адресация
0020 8A 47 03            mov    al,[bx]+3
0023 8B 4F 03            mov    cx,3[bx]

;   Индексная адресация
0026 BF 0002            mov    di,ind
0029 8A 85 000E R        mov    al,vec2[di]

;           mov    cx,vec2[di]

;   Адресация с базирование
;   м и индексированием
002D BB 0003            mov    bx,3
0030 8A 81 0016 R        mov    al,matr[bx][di]

;           mov    cx,matr[bx][di]

```

```

;      mov  ax,matr[bx*4][di]

;   ПРОВЕРКА РЕЖИМОВ АДРЕСА
;   ЦИИ С УЧЕТОМ СЕГМЕНТОВ
;   Переопределение сегментов
a
;   ----- вариант 1
0034  B8 ---- R      mov  ax, SEG vec2
0037  8E C0          mov  es, ax
0039  26: 8B 07      mov  ax, es:[bx]
003C  B8 0000        mov  ax, 0
;   ----- вариант 2
003F  8E C0          mov  es, ax
0041  1E            push ds
0042  07            pop  es
0043  26: 8B 4F FF   mov  cx, es:[bx-1]
0047  91            xchg cx,ax
;   ----- вариант 3
0048  BF 0002        mov  di,ind
004B  26: 89 01      mov  es:[bx+di],ax
;   ----- вариант 4
004E  8B EC          mov  bp,sp
;      mov  ax,matr[bp+bx]
;      mov  ax,matr[bp+di+si]
;   Использование сегмента
;   стека
0050  FF 36 0000 R   push  mem1
0054  FF 36 0002 R   push  mem2
0058  8B EC          mov  bp,sp
005A  8B 56 02       mov  dx, [bp]+2
005D  CA 0002        ret   2
0060                      Main      ENDP
0060                      CODE      ENDS
END Main

```


Symbols-1

Segments and Groups:

| Combine Class | N a m e | Length | Align |
|---------------|------------------|--------|------------|
| | ASTACK | 0018 | PARA STACK |
| | CODE | 0060 | PARA NONE |
| | DATA | 0026 | PARA NONE |

Symbols:

| | N a m e | Type | Value | Attr |
|--------|----------------|--------|-------|-------------|
| | EOL | NUMBER | 0024 | |
| | IND | NUMBER | 0002 | |
| | MAIN | F PROC | 0000 | CODE Length |
| = 0060 | | | | |
| | MATR | L BYTE | 0016 | DATA |
| | MEM1 | L WORD | 0000 | DATA |
| | MEM2 | L WORD | 0002 | DATA |
| | MEM3 | L WORD | 0004 | DATA |
| | N1 | NUMBER | 01F4 | |
| | N2 | NUMBER | -0032 | |
| | VEC1 | L BYTE | 0006 | DATA |
| | VEC2 | L BYTE | 000E | DATA |

```
@CPU . . . . . TEXT 0101h
@FILENAME . . . . . TEXT QWE
@VERSION . . . . . TEXT 510
```

88 Source Lines

88 Total Lines

19 Symbols

47826 + 459431 Bytes symbol space free

0 Warning Errors

0 Severe Errors

Протокол работы на компьютере.

| | | | | | |
|---------|---------|---------|---------|---------------|-------------------------|
| AX 0000 | SI 0000 | CS 1A0A | IP 0000 | Stack +0 0000 | Flags 7202 |
| BX 0000 | DI 0000 | DS 19F5 | | +2 0000 | |
| CX 00B0 | BP 0000 | ES 19F5 | HS 19F5 | +4 0000 | OF DF IF SF ZF AF PF CF |
| DX 0000 | SP 0018 | SS 1A05 | FS 19F5 | +6 0000 | 0 0 1 0 0 0 0 0 |

| | | | | | | | | | | | |
|-------|--|--|--|---------------------------------|--|--|--|--|--|--|--|
| CMD > | | | | 1 0 1 2 3 4 5 6 7 | | | | | | | |
| | | | | DS:0000 CD 20 FF 9F 00 EA F0 FE | | | | | | | |
| | | | | DS:0008 AD DE 1B 05 C5 06 00 00 | | | | | | | |
| | | | | DS:0010 18 01 10 01 18 01 92 01 | | | | | | | |
| | | | | DS:0018 01 01 01 00 02 FF FF FF | | | | | | | |
| | | | | DS:0020 FF FF FF FF FF FF FF FF | | | | | | | |
| | | | | DS:0028 FF FF FF FF EB 19 C0 11 | | | | | | | |
| | | | | DS:0030 A2 01 14 00 18 00 F5 19 | | | | | | | |
| | | | | DS:0038 FF FF FF FF 00 00 00 00 | | | | | | | |
| | | | | DS:0040 05 00 00 00 00 00 00 00 | | | | | | | |
| | | | | DS:0048 00 00 00 00 00 00 00 00 | | | | | | | |

| | | |
|-------------|------|---------|
| 0000 1E | PUSH | DS |
| 0001 2BC0 | SUB | AX,AX |
| 0003 50 | PUSH | AX |
| 0004 B8071A | MOV | AX,1A07 |
| 0007 8ED8 | MOV | DS,AX |
| 0009 B8F401 | MOV | AX,01F4 |
| 000C 8BC8 | MOV | CX,AX |
| 000E B324 | MOV | BL,24 |

| | | | | | | | | | | | | | | | | | |
|---------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-------------------|
| 2 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F | |
| DS:0000 | CD | 20 | FF | 9F | 00 | EA | F0 | FE | AD | DE | 1B | 05 | C5 | 06 | 00 | 00 | = f.Ω=■ ï ..†... |
| DS:0010 | 18 | 01 | 10 | 01 | 18 | 01 | 92 | 01 | 01 | 01 | 00 | 02 | FF | FF | FF | FF |f. |
| DS:0020 | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | FF | EB | 19 | C0 | 11 | δ.L. |
| DS:0030 | A2 | 01 | 14 | 00 | 18 | 00 | F5 | 19 | FF | FF | FF | FF | 00 | 00 | 00 | 00 | ó.....J. |
| DS:0040 | 05 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |

Рисунок 1 – Начальные значения регистров и ячеек памяти.

Таблица 1 – Результаты пошагового выполнения lab_2.asm.

| Адрес команды | Символический код команды | 16-ричный код команды | Содержимое регистров и ячеек памяти | |
|------------------|------------------------------|--------------------------|--|--|
| | | | До выполнения | После выполнения |
| 0000 | PUSH DS | 1E | (SP) = 0018 (DS) = 19F5 Stack: +0 0000 | (SP) = 0016 (DS) = 19F5 Stack: +0 19F5 |
| 0001 | SUB AX,AX | 2BC0 | (AX) = 0000 | (AX) = 0000 |
| 0003 | PUSH AX | 50 | (AX) = 0000 (SP) = 0016 Stack: +0 19F5 Stack: +2 0000 | (AX) = 0000 (SP) = 0014 Stack: +0 0000 Stack: +2 19F5 |
| 0004 | MOV AX, 1A07 | BB071A | (AX) = 0000 | (AX) = 1A07 |
| 0007 | MOV DS,AX | 8ED8 | (DS) = 19F5 (AX) = 1A07 | (DS) = 1A07 (AX) = 1A07 |
| 0009 | MOV AX, 01F4 | B8F401 | (AX) = 1A07 | (AX) = 01F4 |
| 000C | MOV CX, AX | 8BC8 | (CX) = 00B0 (AX) = 01F4 | (CX) = 01F4 (AX) = 01F4 |
| 000E | MOV BL, 24 | B324 | (BX) = 0000 | (BX) = 0024 |
| 0010 | MOV BH, CE | B7CE | (BX) = 0024 | (BX) = CE24 |
| 0012 | MOV [0002], FFCE | C7060200CEFF | DS: 0002 = 00 DS: 0003 = 00 | DS: 0002 = CE DS: 0003 = FF |
| 0018 | MOV BX, 0006 | BB0600 | (BX) = CE24 | (BX) = 0006 |
| 001B | MOV [0000], AX | A30000 | (AX) = 01F4 DS: 0000 = 00 DS: 0001 = 00 | (AX) = 01F4 DS: 0000 = F4 DS: 0001 = 01 |
| 001E | MOV AL, [BX] | 8A07 | (AX) = 01F4 (BX) = 0006 | (AX) = 010C (BX) = 0006 |

Продолжение 1.

| | | | | |
|------|----------------------|----------|--|--|
| 0020 | MOV AL, [BX+03] | 8A4703 | (AX) = 010C (BX) = 0006 | (AX) = 0109 (BX) = 0006 |
| 0023 | MOV CX, [BX+03] | 8B4F03 | (CX) = 01F4 (BX) = 0006 | (CX) = 0509 (BX) = 0006 |
| 0026 | MOV DI, 0002 | BF0200 | (DI) = 0000 | (DI) = 0002 |
| 0029 | MOV AL, [000E+DI] | 8A850E00 | (AX) = 0109 (DI) = 0002 | (AX) = 0128 (DI) = 0002 |
| 002D | MOV BX, 0003 | BB0300 | (BX) = 0006 | (BX) = 0003 |
| 0030 | MOV AL, [0016+BX+DI] | 8A811600 | (AX) = 0128 (BX) = 0003 (DI) = 0002 | (AX) = 01F9 (BX) = 0003 (DI) = 0002 |
| 0034 | MOV AX, 1A07 | B8071A | (AX) = 01F9 | (AX) = 1A07 |
| 0037 | MOV ES, AX | 8EC0 | (ES) = 19F5 (AX) = 1A07 | (ES) = 1A07 (AX) = 1A07 |
| 0039 | MOV AX, ES:[BX] | 268B07 | (AX) = 1A07 (ES) = 1A07 (BX) = 0003 | (AX) = 00FF (ES) = 1A07 (BX) = 0003 |
| 003C | MOV AX, 0000 | B80000 | (AX) = 00FF | (AX) = 0000 |
| 003F | MOV ES, AX | 8EC0 | (AX) = 0000 (ES) = 1A07 | (AX) = 0000 (ES) = 0000 |
| 0041 | PUSH DS | 1E | (DS) = 1A07 (SP) = 0014 Stack: +0 0000 Stack: +2 19F5 Stack: +4 0000 | (DS) = 1A07 (SP) = 0012 Stack: +0 1A07 Stack: +2 0000 Stack: +4 19F5 |

Продолжение 1.

| | | | | |
|------|-----------------------|----------|---|---|
| 0042 | POP ES | 07 | (ES) = 0000 (SP) = 0012 Stack: +0 1A07 Stack: +2 0000 Stack: +4 19F5 | (ES) = 1A07 (SP) = 0014 Stack: +0 0000 Stack: +2 19F5 Stack: +4 0000 |
| 0043 | MOV CX, ES:[BX-01] | 268B4FFF | (CX) = 0509 (ES) = 1A07 (BX) = 0003 | (CX) = FFCE (ES) = 1A07 (BX) = 0003 |
| 0047 | XCHG AX, CX | 91 | (AX) = 0000 (CX) = FFCE | (AX) = FFCE (CX) = 0000 |
| 0048 | MOV DI, 0002 | BF0200 | (DI) = 0002 | (DI) = 0002 |
| 004B | MOV ES:[BX+DI], AX | 268901 | (ES) = 1A07 (BX) = 0003 (DI) = 0002 (AX) = FFCE DS:0005 = 00 DS: 0006 = 08 | (ES) = 1A07 (BX) = 0003 (DI) = 0002 (AX) = FFCE DS: 0005 = CE DS: 0006 = FF |
| 004E | MOV BP, SP | 8BEC | (BP) = 0000 (SP) = 0014 | (BP) = 0014 (SP) = 0014 |
| 0050 | PUSH [0000] | FF360000 | DS:0000 = F4 DS:0001 = 01 (SP) = 0014 Stack: +0 0000 Stack: +2 19F5 Stack: +4 0000 | DS:0000 = F4 DS:0001 = 01 (SP) = 0012 Stack: +0 01F4 Stack: +2 0000 Stack: +4 19F5 |

Продолжение 1.

| | | | | |
|------|-----------------|----------|---|---|
| 0054 | PUSH [0002] | FF360200 | DS:0002 = CE DS:0003 = FF (SP) = 0012 Stack: +0 01F4 Stack: +2 0000 Stack: +4 19F5 Stack: +6 0000 | DS:0002 = CE DS:0003 = FF (SP) = 0010 Stack: +0 FFCE Stack: +2 01F4 Stack: +4 0000 Stack: +6 19F5 |
| 0058 | MOV BP, SP | 8BEC | (BP) = 0014 (SP) = 0010 | (BP) = 0010 (SP) = 0010 |
| 005A | MOV DX, [BP+02] | 8B5602 | (DX) = 0000 (BP) = 0010 | (DX) = 01F4 (BP) = 0010 |
| 005D | FAR 0002 | CA0200 | (SP) = 0010 (CS) = 1A0A Stack: +0 FFCE Stack: +2 01F4 Stack: +4 0000 Stack: +6 19F5 | (SP) = 0016 (CS) = 01F4 Stack: +0 19F5 Stack: +2 0000 Stack: +4 0000 Stack: +6 0000 |

Выводы.

Получены навыки в области отладки программы на языке Ассемблер, усвоены знания в области регистровой адресации. Были найдены ошибки в программе.

ПРИЛОЖЕНИЕ А

ИСПРАВЛЕННЫЙ КОД ПРОГРАММЫ

Название файла: lab_2.asm

```
; Программа изучения режимов адресации процессора IntelX86
EOL EQU '$'
ind EQU 2
n1 EQU 500
n2 EQU -50

; Стек программы
AStack SEGMENT STACK
    DW 12 DUP(?)
AStack ENDS

;Данные программы
DATA SEGMENT ;
;Директивы описания данных
mem1 DW 0
mem2 DW 0
mem3 DW 0
vec1 DB 12,11,10,9,5,6,7,8
vec2 DB -40,-50,40,50,-20,-30,20,30
matr DB 5,6,7,8,-8,-7,-6,-5,1,2,3,4,-4,-3,-2,-1
DATA ENDS

; Код программы
CODE SEGMENT
    ASSUME CS:CODE, DS:DATA, SS:AStack

; Головная процедура
Main PROC FAR
    push DS
    sub AX,AX
```

```

push AX
mov AX,DATA
mov DS,AX

; ПРОВЕРКА РЕЖИМОВ АДРЕСАЦИИ НА УРОВНЕ СМЕЩЕНИЙ
; Регистровая адресация
    mov ax,n1
    mov cx,ax
    mov bl,EOL
    mov bh,n2
; Прямая адресация
    mov mem2,n2
    mov bx,OFFSET vec1
    mov mem1,ax
; Косвенная адресация
    mov al,[bx]
;    mov mem3,[bx]
; Базированная адресация
    mov al,[bx]+3
    mov cx,3[bx]
; Индексная адресация
    mov di,ind
    mov al,vec2[di]
;    mov cx,vec2[di]
; Адресация с базированием и индексированием
    mov bx,3
    mov al,matr[bx][di]
;    mov cx,matr[bx][di]
;    mov ax,matr[bx*4][di]

; ПРОВЕРКА РЕЖИМОВ АДРЕСАЦИИ С УЧЕТОМ СЕГМЕНТОВ
; Переопределение сегмента
; ----- вариант 1
    mov ax, SEG vec2

```



```

        mov es, ax
        mov ax, es:[bx]
        mov ax, 0
; ----- вариант 2
        mov es, ax
        push ds
        pop es
        mov cx, es:[bx-1]
        xchg cx, ax
; ----- вариант 3
        mov di, ind
        mov es:[bx+di], ax
; ----- вариант 4
        mov bp, sp
;   mov ax, matr[bp+bx]
;   mov ax, matr[bp+di+si]
; Использование сегмента стека
        push mem1
        push mem2
        mov bp, sp
        mov dx, [bp]+2
        ret 2
Main    ENDP
CODE    ENDS
END Main

```