

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

Студент гр. 9383

Лихашва А.Д.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Изучить режимы адресации, указать на ошибки в программе, объяснив их.

Основные теоретические положения.

Задание:

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 программы:

```
; 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 31,32,33,34,38,37,36,35
vec2 DB 50,60,-50,-60,70,80,-70,-80
matr DB -4,-3,7,8,-2,-1,5,6,-8,-7,3,4,-6,-5,1,2
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

```

Обработка результатов эксперимента.

1) `mov mem3,[bx]`

Ошибка: “Improper operand type”

Нельзя прямо передавать объекты с памяти в память. Если нужно передать данные из ячейки `[bx]` в ячейку, на которую ссылается переменная `mem3`, то это нужно сделать через регистр `AX`.

2) `mov cx,vec2[di]`

Ошибка: “Operand types must match”

Переменная `vec2` – массив, и каждая его ячейка имеет тип `DB`, то есть занимает ровно 1 байт. В то же время регистр `CX` занимает 2 байта. Место, которое занимают операнды должно быть одинаковым. Можно передать `vec2[di]` в `CH` или `CL`, но не в `CX`.

3) `mov cx,matr[bx][di]`

Ошибка: “Operand types must match”

То же самое, что и в прошлой ошибке. Ячейки двумерного массива имеют размерность 1 байт (DB), а регистр CX – 2 байта.

4) `mov ax,matr[bx*4][di]`

Ошибка: “Illegal register value”

Операцию умножение на число можно применять только к регистрам с префиксом E.

5) `mov ax,matr[bp+bx]`

Ошибка: “Multiple base registers”

Нельзя использовать более одного базового регистра. Размер элементов матрицы `matr` 1 байт, а `AX` – 2 байта .

6) `mov ax, matr[bp+di+si]`

Ошибка: “Multiple index registers”

Нельзя использовать более одного индексного регистра. Нельзя использовать более двух регистров. Размер элементов матрицы `matr` 1 байт, а `AX` – 2 байта .

Листинг трансляции программы:

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

; РЎС,РµРє РїСЃРѕРїСЃР°РјРјСЃ
0000          AStack SEGMENT STACK
0000 000C[          DW 12 DUP(?)
```

```

        ????
```

]

```

0018          AStack  ENDS

          ;P"P°PSPSC<Pμ PiCṪPsPiCṪP°PjPjC<

0000          DATA          SEGMENT

          ;P"PḔCṪPμPeC,PḔPIC<  PsPiPḔCÍP°PSPḔCȲ
PrP°PSPSC<

          C...

0000  0000          mem1      DW      0
0002  0000          mem2      DW      0
0004  0000          mem3      DW      0
0006  1F 20 21 22 26 25  vec1                                DB
31,32,33,34,38,37,36,35
          24 23
000E  32 3C CE C4 46 50  vec2                                DB  50,60,-50,-
60,70,80,-70,-80
          BA B0
0016  FC FD 07 08 FE FF  matr                                DB  -4,-3,7,8,-2,-
1,5,6,-8,-7,3,4,-6,-
          5,1,2
          05 06 F8 F9 03 04
          FA FB 01 02

0026          DATA          ENDS

          ; PḔPsPr PiCṪPsPiCṪP°PjPjC<

0000          CODE          SEGMENT

                                ASSUME  CS:CODE,  DS:DATA,
SS:AStack

```



```

;                                     PъPсCГPIPμPSPSP°CЦ
P°PГCБPμCГP°C†PëCЦ
001E 8A 07                                mov  al,[bx]
;mov  mem3,[bx]
;                                     P`P°P·PëCБPsPIP°PSPSP°CЦ
P°PГCБPμCГP°C†PëCЦ
0020 8A 47 03                            mov  al,[bx]+3
0023 8B 4F 03                            mov  cx,3[bx]
;                                     P∞PSPГPμPеCГPSP°CЦ
P°PГCБPμCГP°C†PëCЦ
0026 BF 0002                            mov  di,ind
0029 8A 85 000E R                        mov  al,vec2[di]
;mov  cx,vec2[di]
;                                     PђPГCБPμCГP°C†PëCЦ      CГ
P±P°P·PëCБPsPIP°PSPëPμ
Pј Pë PëPSPГPμPеCГPëCБPsPIP°PSPëPμPј
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]
;   PμP PђP'P•P PъPђ P P•P-P∞PђPђP'
PђP"P P•PŸPђ
P|P∞P∞ PŸ PJP$P•PŸPђPђ
PŸP•P`PђP•PќPŸPђP'
;   PμPμCБPμPsPіCБPμPГPμP»PμPSPëPμ
CГPμPiPјPμPSC
,P°
;   ----- PIP°CБPëP°PSC, 1

```



```
#Microsoft      (R)      Macro      Assembler      Version      5.10
11/19/20 20:12:3
```

Symbols-1

Segments and Groups:

	N a m e	Length	Align
Combine Class			
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	LAB2
@VERSION	TEXT	510

88 Source Lines

88 Total Lines

19 Symbols

47828 + 459432 Bytes symbol space free

0 Warning Errors

0 Severe Errors

Выводы.

Получены навыки в области отладки программы на языке ассемблера и нахождения ошибок в готовой программе. Усвоены знания в области регистровой адресации.

ПРОТОКОЛ

Начальные значения регистров:

CS = 1A0A, DS=19F5, ES=19F5, SS=1A05

Адрес команд ы	Символический код команд	16-ричный код команд	Содержимое регистров и ячеек памяти	
			До выполнения	После выполнения
0000	PUSH DS	1E	DS= 19F5 SP=0018 STACK=+0 0000 IP=0000	DS= 19F5 SP=0016 STACK=+0 19F5 IP=0001
0001	SUB AX,AX	2BC0	AX=0000 IP=0001	AX=0000 IP=0003
0003	PUSH AX	50	AX=0000 SP=0016 STACK=+0 19F5 IP=0003	AX=0000 SP=0014 STACK=+0 0000 +2 19F5 IP=0004
0004	MOV AX,30C2	B8071A	AX=0000 IP=0004	AX=1A07 IP=0007
0007	MOV DS,AX	8ED8	AX=1A07 DS= 19F5 IP=0007	AX=1A07 DS=1A07 IP=0009
0009	MOV AX,01F4	B8F401	AX=1A07 IP=0009	AX=01F4 IP=000C
000C	MOV CX,AX	8BC8	AX=01F4 CX=00B0 IP=000C	AX=01F4 CX=01F4 IP=000E
000E	MOV BL,24	B324	BX=0000 IP=000E	BX=0024 IP=0010
0010	MOV BH,CE	B7CE	BX=0024 IP=0010	BX=CE24 IP=0012
0012	MOV [0002],FFCE	C7060200CE FF	IP=0012 DS[0002]=00 DS[0003]=00	IP=0018 DS[0002]=CE DS[0003]=FF
0018	MOV BX, 0006	BB0600	BX=CE24 IP=0018	BX=0006 IP=001B

001B	MOV [0000],AX	A30000	AX=01F4 IP=001B DS[0000]=00 DS[0001]=00	AX=01F4 IP=001E DS[0000]=F4 DS[0001]=01
001E	MOV AL,[BX]	8A07	AX=01F4 DS[BX]= DS[0006]=01 IP=001E	AX=0101 DS[BX]= DS[0006]=01 IP=0020
0020	MOV AL,[BX+03]	8A4703	AX=0101 DS[BX+03]= DS[0009]=04 IP=0020	AX=0122 DS[BX+03]= DS[0009]=04 IP=0023
0023	MOV CX,[BX+03]	8B4F03	CX=01F4 DS[BX+03]= DS[0009]=04 DS[000A]=08 IP=0023	CX=2622 DS[BX+03]= DS[0009]=04 DS[000A]=08 IP=0026
0026	MOV DI, 0002	BF0200	DI=0000 IP=0026	DI=0002 IP=0029
0029	MOV AL,[000E+DI]	8A850E00	AX=0122 DS[000E+DI]= DS[0010]=0A IP=0029	AX=01CE DS[000E+DI]= DS[0010]=0A IP=002D
002D	MOV BX,0003	BB0300	BX=0006 IP=002D	BX=0003 IP=0030
0030	MOV AL,[0016 + BX + DI]	8A811600	AX=01CE DS[0016+BX+DI]= DS[001B]=FD IP=0030	AX=01FF DS[0016+BX+DI]= DS[001B]=FD IP=0034
0034	MOV AX,30C2	B8C230	AX=01FF IP=0034	AX=1A07 IP=0037
0037	MOV ES,AX	8EC0	ES=19F5 AX=1A07 IP=0037	ES=1A07 AX=1A07 IP=0039
0039	MOV AX,ES:[BX]	268B07	AX=1A07 ES=1A07 ES[BX]=ES[0003]=FF	AX=00FF ES=1A07 ES[BX]=ES[0003]=FF

			ES[0004]=00 IP=0039	ES[0004]=00 IP=003C
003C	MOV AX,0000	B80000	AX=00FF IP=003C	AX=0000 IP=003F
003F	MOV ES,AX	8EC0	ES=1A07 AX=0000 IP=003F	ES=0000 AX=0000 IP=0041
0041	PUSH DS	1E	DS=1A07 SP=0014 STACK=+0 0000 +2 19F5 IP=0041	DS=30C2 SP=0012 STACK=+0 30C2 +2 0000 +4 30B0 IP=0042
0042	POP ES	07	SP=0012 ES=0000 STACK=+0 1A07 +2 0000 +4 19F5 IP=0042	SP=0014 ES=1A07 STACK=+0 0000 +2 19F5 IP=0043
0043	MOV CX,ES:[BX-01]	268B4FFF	CX=2622 ES=1A07 ES[BX-01]= ES[0002]=CE ES[0003]=FF IP=0043	CX=FFCE ES=1A07 ES[BX-01] =ES[0002]=CE ES[0003]=FF IP=0047
0047	XCHG AX,CX	91	AX = 0000 CX = FFCE IP=0047	AX=FFCE CX=0000 IP=0048
0048	MOV DI,0002	BF0200	DI=0002 IP=0048	DI=0002 IP=004B
004B	MOV ES:[BX+DI],AX	268901	ES=1A07 ES[BX+DI] = [0005]= 00 ES[0006] = 01 AX=FFCE IP=004B	ES=1A07 ES[0005] = CE ES[0006] = FF IP=004E
004E	MOV BP,SP	8BEC	BP=0000	BP=0014

			SP=0014 IP=004E	SP=0014 IP=0050
0050	PUSH [0000]	FF360000	DS[0000] = F4 DS[0001] = 01 SP = 0014 STACK = +0 0000 +2 19F5 IP=0050	DS[0000] = F4 DS[0001] = 01 SP = 0012 STACK= +0 01F4 +2 0000 +4 19F5 IP=0054
0054	PUSH [0002]	FF360200	DS[0002] = CE DS[0003] = FF SP = 0012 STACK=+0 01F4 +2 0000 +4 19F5 IP=0054	DS[0002] = CE DS[0003] = FF SP = 0010 STACK=+0 FFCE +2 01F4 +4 0000 +6 19F5 IP=0058
0058	MOV BP,SP	8BEC	SP=0010 BP=0014 IP=0058	SP=0010 BP=0010 IP=005A
005A	MOV DX,[BP+02]	8B5602	DX=0000 SS[BP+02] = SS[0012]=F4 SS[0013]=01 IP=005A	DX=01F4 SS[BP+02] = SS[0012]=F4 SS[0013] = 01 IP=005D
005D	RET FAR 0002	CA0200	CS=1A0A SP=0010 IP=005D STACK=+0 FFCE +2 01F4 +4 0000 +6 19F5	CS=01F4 SP=0016 IP=FFCE STACK=+0 19F5

ПРИЛОЖЕНИЕ А

ИСПРАВЛЕННЫЙ КОД ПРОГРАММЫ

Название файла: lab2.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 31,32,33,34,38,37,36,35
vec2 DB 50,60,-50,-60,70,80,-70,-80
matr DB -4,-3,7,8,-2,-1,5,6,-8,-7,3,4,-6,-5,1,2
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
```