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.

Студент гр. 9383 _____ Лапина А.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Исправить ошибки в программе на языке программирования Ассемблер и применить на практике знания о режимах адресации и формировании исполнительного адреса в языке программирования Ассемблер.

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.

Исходные данные.

Вариант 8

Таблица 1 — Исходные данные.

Название массива	Набор значений
vec1	28,27,26,25,21,22,23,24
vec2	20,30,-20,-30,40,50,-40,-50
matr	-8,-7,3,4,-6,-5,1,2,-4,-3,7,8,-2,-1,5,6

Исходный код.

Исходный код представлен в приложении А.

Обнаруженные ошибки.

1) «mov mem3,[bx]» - error A2052:
   Improper operand type
   нельзя передавать данные из памяти в память напрямую.
2) «mov cx,vec2[di]» - warning A4031:
   Operand types must match
   разные размеры (размер элемента vec2 – 1 байт, а cx – 2 байта)
3) «mov cx,matr[bx][di]» - warning A4031:
   Operand types must match
   разные размеры (matr — 1 байт, cx — 2 байта)
4) «mov ax,matr[bx*4][di]» - error A2055:
   Illegal register value
   нельзя умножать 16-битовые регистры.
5) «mov ax,matr[bp+bx]» - error A2046:
   Multiple base registers
   нельзя использовать несколько базовых регистров.
6) «mov ax,matr[bp+di+si]» - error A2047:
   Multiple index registers
   нельзя использовать несколько индексных регистров.

Протокол работы.

Таблица 2 – Результаты выполнения программы в пошаговом режиме.

Адрес команды	Символический код комманды	16-ричный код команды	Содержимое регистров и ячеек памяти	
			До выполнения	После выполнения
0000	PUSH DS	1E	(CS) = 1A0A (DS) = 19F5	(SP) = 0016 (DS) = 19F5

			(ES) = 19F5 (SS) = 1A05 (CX) = 00B0 (DX) = 0000 (SP) = 0018 (IP) = 0000 Stack: +0 0000	(IP) = 0001 Stack: +0 19F5
0001	SUB AX, AX	2BC0	(AX) = 0000 (IP) = 0001	(AX) = 0000 (IP) = 0003
0003	PUSH AX	50	(AX) = 0000 (SP) = 0016 (IP) = 0003 Stack: +0 19F5 Stack: +2 0000	(AX) = 0000 (SP) = 0014 (IP) = 0004 Stack: +0 0000 Stack: +2 19F5
0004	MOV AX, 1A07	B8071A	(AX) = 0000 (IP) = 0004	(AX) = 1A07 (IP) = 0007
0007	MOV DS, AX	8ED8	(DS) = 19F5 (AX) = 1A07 (IP) = 0007	(DS) = 1A07 (AX) = 1A07 (IP) = 0009
0009	MOV AX, 01F4	B8F401	(AX) = 1A07 (IP) = 0009	(AX) = 01F4 (IP) = 000C
000C	MOV CX, AX	8BC8	(CX) = 00B0 (AX) = 01F4 (IP) = 000C	(CX) = 01F4 (AX) = 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	C7060200CEFF	DS: 0002 = 00 DS: 0003 = 00 (IP) = 0012	DS: 0002 = CE DS: 0003 = FF (IP) = 0018
0018	MOV BX, 0006	BB0600	(BX) = CE24 (IP) = 0018	(BX) = 0006 (IP) = 001B
001B	MOV [0000], AX	A30000	(AX) = 01F4 DS: 0000 = 00	(AX) = 01F4 DS: 0000 = F4

			DS: 0001 = 00 (IP) = 001B	DS: 0001 = 01 (IP) = 001E
001E	MOV AL, [BX]	8A07	(AX) = 01F4 (BX) = 0006 DS: 0006 = 1C (IP) = 001E	(AX) = 011C (BX) = 0006 DS: 0006 = 1C (IP) = 0020
0020	MOV AL, [BX+03]	8A4703	(AX) = 011C (BX) = 0006 DS: 0009 = 19 (IP) = 0020	(AX) = 0119 (BX) = 0006 DS: 0009 = 19 (IP) = 0023
0023	MOV CX, [BX+03]	8B4F03	(CX) = 01F4 (BX) = 0006 DS: 0009 = 19 DS: 000A = 15 DS:000C = 17 (IP) = 0023	(CX) = 1519 (BX) = 0006 DS: 0009 = 19 DS: 000A = 15 DS:000C = 17 (IP) = 0026
0026	MOV DI, 0002	BF0200	(DI) = 0000 (IP) = 0026	(DI) = 0002 (IP) = 0029
0029	MOV AL, [000E+DI]	8A850E00	(AX) = 0119 (DI) = 0002 DS: 0010 = EC (IP) = 0029	(AX) = 01EC (DI) = 0002 DS: 0010 = EC (IP) = 002D
002D	MOV BX, 0003	BB0300	(BX) = 0006	(BX) = 0003
0030	MOV AL, [0016+BX+DI]	8A811600	(AX) = 01EC (BX) = 0003 (DI) = 0002 DS: 001B = FB	(AX) = 01FB (BX) = 0003 (DI) = 0002 DS: 001B = FB
0034	MOV AX, 1A07	B8071A	(AX) = 01FB	(AX) = 1A07
0037	MOV ES, AX	8EC0	(ES) = 19F5 (AX) = 1A07	(ES) = 1A07 (AX) = 1A07
0039	MOV AX, ES: [BX]	268B07	(AX) = 010A (ES) = 1A07 (BX) = 0003	(AX) = 00FF (ES) = 1A07 (BX) = 0003

			DS: 0003 = FF DS: 0004 = 00	DS: 0003 = FF DS: 0004 = 00
003C	MOV AX, 0000	B80000	(AX) = 00FF	(AX) = 0000
003F	MOV ES, AX	8EC0	(ES) = 1A07 (AX) = 0000	(ES) = 0000 (AX) = 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
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) = 1519 (ES) = 1A07 (BX) = 0003 DS: 0002 = CE DS: 0003 = FF	(CX) = FFCE (ES) = 1A07 (BX) = 0003 DS: 0002 = CE DS: 0003 = FF
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 = 1C	(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	DS: 0000 = F4 DS: 0001 = 01

			(SP) = 0014 Stack: +0 0000 Stack: +2 19F5 Stack: +4 0000	(SP) = 0012 Stack: +0 01F4 Stack: +2 0000 Stack: +4 19F5
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 (SP) = 0010 Stack: +0 FFCE Stack: +2 01F4	(DX) = 01F4 (BP) = 0010 (SP) = 0010 Stack: +0 FFCE Stack: +2 01F4
005D	RET 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

Исходный код программы в приложении А, листинг и исправленный код программы в приложении Б.

Выводы.

Были исправлены ошибки в программе на языке программирования Ассемблер и были применены на практике знания о режимах адресации и формировании исполнительного адреса в языке программирования Ассемблер.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ 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    28,27,26,25,21,22,23,24
vec2      DB    20,30,-20,-30,40,50,-40,-50
matr      DB    -8,-7,3,4,-6,-5,1,2,-4,-3,7,8,-2,-1,5,6
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
```

ПРИЛОЖЕНИЕ Б
ЛИСТИНГ И ИСПРАВЛЕННЫЙ КОД ПРОГРАММЫ

Название файла: lab2.LST

```
     #Microsoft (R) Macro Assembler Version 5.10                 10/11/20
20:47:4
                                                                      Page
1-1


                        ; Программа изучения режи
                        ов 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  1C 1B 1A 19 15 16 vec1      DB    28,27,26,25,21,22,23,24
            17 18
      000E  14 1E EC E2 28 32 vec2        DB     20,30,-20,-30,40,50,-
40,-50
```

```
                D8 CE
        0016    F8 F9 03 04 FA FB  MATR      DB    -8,-7,3,4,-6,-5,1,2,-4,-
3,7,8,-
                                   2,-1,5,6
                01 02 FC FD 07 08
                FE FF 05 06
        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
```

```
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
                        M 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 СЕГМЕН
                        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
```

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	lb2	

```
@VERSION . . . . . .    TEXT  510


     88 SOURCE  LINES
     88 TOTAL   LINES
     19 SYMBOLS

  47842 + 459418 BYTES SYMBOL SPACE FREE

      0 WARNING ERRORS
      0 SEVERE  ERRORS
```

Название файла: lab2_fix.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    28,27,26,25,21,22,23,24
vec2      DB    20,30,-20,-30,40,50,-40,-50
matr      DB    -8,-7,3,4,-6,-5,1,2,-4,-3,7,8,-2,-1,5,6
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
```