|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования  ***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н.Э. Баумана)*** |

**ФАКУЛЬТЕТ** ***ИУК «Информатика и управление»***

**КАФЕДРА** \_\_***ИУК4 «Программное обеспечение ЭВМ, информационные технологии»***

**ЛАБОРАТОРНАЯ РАБОТА №2**

**«Изучение методов адресации. Команды пересылки данных»**

**ДИСЦИПЛИНА: «Машинно-зависимые языки программирования»**

|  |  |  |
| --- | --- | --- |
| Выполнил: студент гр. ИУК4-32Б | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ( Карельский М.К. )  (Подпись) |
| Проверил: | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ( Амеличева К.А. )  (Подпись) |
| Дата сдачи (защиты):  Результаты сдачи (защиты): | | |
|  | - Балльная оценка:  - Оценка: | |

Калуга, 2021

**Цель:** практическое овладение навыками разработки программного кода на языке Ассемблер. Изучение команд передачи данных и использования различных способов адресации операндов. Практическое освоение основных функций отладчика TD.

**Вариант 15**

**Задание:**

*Часть 1*

1. Изучить методические указания и рекомендованную литературу.

2. Написать программу с помощью шаблона.

3. Задать начальные значения переменных A, B, C, D в сегменте данных в соответствии с вариантом.

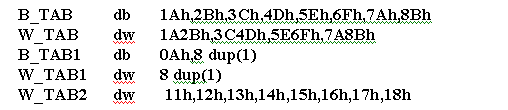
4. Проследить за работой в Турбоотладчике, заполнить таблицу для строк программы с 19 по 35.

5. Выполнить вариант задания.

*Часть 2*

1. Написать программу, используя различные виды режимов адресации, сделать исполняемый файл, и проследить за работой в Турбоотладчике.

2. Опишите в сегменте данных следующую информацию:

****

3. На основе работы программы в таблице зафиксировать значение операнда приемника на каждом шагу программы.

4. Выполнить задание варианта. На основе работы программы в таблице зафиксировать значение операнда приемника на каждом шагу программы.

4.1. Задать одномерный массив, состоящий из 10 элементов.

4.2. Заполнить массив числовыми константами. Размер элементов массива для четных номеров варианта - байт, для нечетных номеров варианта – слово.

4.3. Разместить элементы массива в регистры общего назначения, используя различные способы адресации, по следующей схеме:

• в АХ – элемент массива, номер которого соответствует первой цифре Вашего дня рождения,

• в ВХ – элемент массива, номер которого соответствует второй цифре Вашего дня рождения,

• в СХ – элемент массива, номер которого соответствует первой цифре Вашего месяца рождения,

• в DХ – элемент массива, номер которого соответствует второй цифре Вашего месяца рождения,

• в SI – элемент массива, номер которого соответствует первой цифре Вашего года рождения,

• в DI – элемент массива, номер которого соответствует второй цифре Вашего года рождения,

• в BP – элемент массива, номер которого соответствует третьей цифре Вашего года рождения,

• в SP – элемент массива, номер которого соответствует четвертой цифре Вашего года рождения.

4.4. Назначить переменной fio Вашу фамилию, имя, отчество. Определить физические адреса заглавных букв. Разместить в регистр AL среднюю букву ФИО (значение округлить до целого).

4.5. По адресу равному дню и месяцу Вашего рождения (например, 23 февраля – 2302) занести год Вашего рождения, представив его как шестнадцатеричное число.

4.6. Определить переменную const, присвоив ей день и месяц Вашего рождения в формате, описанном выше. Разместить это значение в регистре CX.

4.7. Поместить в переменную name уменьшительно-ласкательную форму Вашего имени. Определить адрес name.

4.8. В памяти сразу после name разместить символ, код которого в ASCII-кодах определить, как Ваш номер по порядку плюс 14. А затем через пробел дату Вашего рождения в формате 23021999.

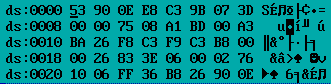
**Решение:**

Листинг первой программы:

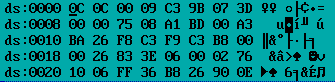
1. .model small
2. .stack 100h
3. .data
4. A db ?
5. B db ?
6. C db ?
7. D db ?
8. .code
9. Start:
10. mov ax, @data
11. mov ds, ax
12. mov A, 12h
13. mov B, 12
14. mov C, 05h
15. mov D, 9
16. mov al, A
17. mov ah, B
18. xchg al, ah
19. mov bx, 3E10h
20. mov cx, bx
21. push bx
22. push cx
23. push ax
24. lea si, C
25. mov ax, si
26. lea di, D
27. mov bx, di
28. pop ax
29. pop cx
30. pop bx
31. mov bx, ax
32. mov A, al
33. mov B, al
34. mov C, 0
35. mov ax, 4C00h
36. int 21h
37. end Start
38. END

|  |  |  |  |
| --- | --- | --- | --- |
| Номер строки | Команда Ассемблера | Машинный код | Состояние регистров и флагов |
| 19 | xchg al, ah | 86 C4 | AX = 12 0C, IP = 00 22 |
| 20 | mov bx, 3E10h | BB 10 3E | BX = 3E 10, IP = 00 25 |
| 21 | mov cx, bx | 8B CB | CX = 3E 10, IP = 00 27 |
| 22 | push bx | 53 | SP = 00 FE, IP = 00 28 |
| 23 | push cx | 51 | SP = 00 FC, IP = 00 29 |
| 24 | push ax | 50 | SP = 00 FA, IP = 00 2A |
| 25 | lea si, C | BE 02 00 | SI = 00 02, IP = 00 2D |
| 26 | mov ax, si | 8B C6 | AX = 00 02, IP = 00 2F |
| 27 | lea di, D | BF 03 00 | DI = 00 03, IP = 00 32 |
| 28 | mov bx, di | 8B DF | BX = 00 03, IP = 00 34 |
| 29 | pop ax | 58 | AX = 12 0C, SP = 00 FC, IP = 00 35 |
| 30 | pop cx | 59 | SP = 00 FE, IP = 00 36 |
| 31 | pop bx | 5B | BX = 3E 10, SP = 01 00, IP = 00 37 |
| 32 | mov bx, ax | 8B D8 | BX = 12 0C, IP = 00 39 |
| 33 | mov A, al | A2 00 00 | IP = 00 3C |
| 34 | mov B, al | A2 01 00 | IP = 00 3F |
| 35 | mov C, 0 | C6 06 02 00 00 | IP = 00 44 |

**Таблица 1.** Состояние первой программы в ходе ее выполнения



**Рисунок 1.** Dump первой программы до ее выполнения



**Рисунок 2.** Dump первой программы после ее выполнения

Листинг второй программы:

.model small

.stack 100h

.data

B\_TAB db 1Ah, 2Bh, 3Ch, 4Dh, 5Eh, 6Fh, 7Ah, 8Bh

W\_TAB dw 1A2Bh, 3C4Dh, 5E6Fh, 7A8Bh

B\_TAB1 db 0Ah, 8 dup(1)

W\_TAB1 dw 8 dup(1)

W\_TAB2 dw 11h, 12h, 13h, 14h, 15h, 16h, 17h, 18h

.code

Start:

mov ax, @data

mov ds, ax

mov al, -3

mov ax, 3

mov B\_TAB, -3

mov W\_TAB, -3

mov ax, 2A1Bh

mov bl, al

mov bh, al

sub ax, bx

sub ax, ax

mov ax, W\_TAB+3

mov ax, W\_TAB+5

mov al, byte ptr W\_TAB+6

mov al, B\_TAB

mov al, B\_TAB+2

mov ax, word ptr B\_TAB

mov es: W\_TAB2+4, ax

mov bx, offset B\_TAB

mov si, offset B\_TAB+1

mov di, offset B\_TAB+2

mov dl, [bx]

mov dl, [si]

mov ax, [di]

mov bp, bx

mov al, [bp]

mov al, ds:[bp]

mov al, es:[bx]

mov ax, cs:[bx]

mov ax, [bx]+2

mov ax, [bx]+4

mov ax, [bx+2]

mov ax, [4+bx]

mov ax, 2+[bx]

mov ax, 4+[bx]

mov al, [bx]+2

mov bp, bx

mov ax, [bp+2]

mov ax, ds:[bp]+2

mov ax, ss:[bx+2]

mov si, 2

mov ah, B\_TAB[si]

mov al, [B\_TAB+si]

mov bh, [si+B\_TAB]

mov bl, [si]+B\_TAB

mov bx, es:W\_TAB2[si]

mov di, 4

mov bl, byte ptr es:W\_TAB2[di]

mov bl, B\_TAB[si]

mov bx, offset B\_TAB

mov al, 3[bx][si]

mov ah, [bx+3][si]

mov al, [bx][si+2]

mov ah, [bx+si+2]

mov bp, bx

mov ah, 3[bp][si]

mov ax, ds:3[bp][si]

mov ax, word ptr ds:2[bp][si]

mov ax, 4c00h

int 21h

end start

|  |  |  |
| --- | --- | --- |
| Оператор | Операнд-приемник | |
| До выполнения | После выполнения |
| mov al, -3 | al: CB | al: FD |
| mov ax, 3 | ax: 0B FD | ax: 00 03 |
| mov B\_TAB, -3 | B\_TAB: 1A | B\_TAB: FD |
| mov W\_TAB, -3 | W\_TAB: 2B 1A | W\_TAB: FD FF |
| mov ax, 2A1Bh | ax: 00 03 | ax: 2A 1B |
| mov bl, al | bl: 00 | bl: 1B |
| mov bh, al | bh: 00 | bh: 1B |
| sub ax, bx | ax: 2A 1B | ax: 0F 00 |
| sub ax, ax | ax: 0F 00 | ax: 00 00 |
| mov ax, W\_TAB+3 | ax: 00 00 | ax: 6F 3C |
| mov ax, W\_TAB+5 | ax: 6F 3C | ax: 8B 5E |
| mov al, byte ptr W\_TAB+6 | al: 5E | al: 8B |
| mov al, B\_TAB | al: 8B | al: FD |
| mov al, B\_TAB+2 | al: FD | al: 3C |
| mov ax, word ptr B\_TAB | ax: 8B 3C | ax: 2B FD |
| mov es: W\_TAB2+4, ax | es: W\_TAB2+4: 0A 72 | es: W\_TAB2+4: FD 2B |
| mov bx, offset B\_TAB | bx: 1B 1B | bx: 00 00 |
| mov si, offset B\_TAB+1 | si: 00 00 | si: 00 01 |
| mov di, offset B\_TAB+2 | di: 00 00 | di: 00 02 |
| mov dl, [bx] | dl: 00 | dl: FD |
| mov dl, [si] | dl: FD | dl: 2B |
| mov ax, [di] | ax: 2B FD | ax: 4D 3C |
| mov bp, bx | bp: 00 00 | bp: 00 00 |
| mov al, [bp] | al: 3C | al: 5F |
| mov al, ds:[bp] | al: 5F | al: FD |
| mov al, es:[bx] | al: FD | al: CD |
| mov ax, cs:[bx] | ax: 4D CD | ax: 6C B8 |
| mov ax, [bx]+2 | ax: 6C B8 | ax: 4D 3C |
| mov ax, [bx]+4 | ax: 4D 3C | ax: 6F 5E |
| mov ax, [bx+2] | ax: 6F 5E | ax: 4D 3C |
| mov ax, [4+bx] | ax: 4D 3C | ax: 6F 5E |
| mov ax, 2+[bx] | ax: 6F 5E | ax: 4D 3C |
| mov ax, 4+[bx] | ax: 4D 3C | ax: 6F 5E |
| mov al, [bx]+2 | al: 5E | al: 3C |
| mov bp, bx | bp: 00 00 | bp: 00 00 |
| mov ax, [bp+2] | ax: 6F 3C | ax: 5B 5D |
| mov ax, ds:[bp]+2 | ax: 5B 5D | ax: 4D 3C |
| mov ax, ss:[bx+2] | ax: 4D 3C | ax: 5B 5D |
| mov si, 2 | si: 00 01 | si: 00 02 |
| mov ah, B\_TAB[si] | ah: 5B | ah: 3C |
| mov al, [B\_TAB+si] | al: 5D | al: 3C |
| mov bh, [si+B\_TAB] | bh: 00 | bh: 3C |
| mov bl, [si]+B\_TAB | bl: 00 | bl: 3C |
| mov bx, es:W\_TAB2[si] | bx: 3C 3C | bx: FD FF |
| mov di, 4 | di: 00 02 | di: 00 04 |
| mov bl, byte ptr es:W\_TAB2[di] | bl: FF | bl: FD |
| mov bl, B\_TAB[si] | bl: FD | bl: 3C |
| mov bx, offset B\_TAB | bx: FD 3C | bx: 00 00 |
| mov al, 3[bx][si] | al: 3C | al: 6F |
| mov ah, [bx+3][si] | ah: 3C | ah: 6F |
| mov al, [bx][si+2] | al: 6F | al: 5E |
| mov ah, [bx+si+2] | ah: 6F | ah: 5E |
| mov bp, bx | bp: 00 00 | bp: 00 00 |
| mov ah, 3[bp][si] | ah: 5E | ah: 5A |
| mov ax, ds:3[bp][si] | ax: 5A 5E | ax: 7A 6F |
| mov ax, word ptr ds:2[bp][si] | ax: 7A 6F | ax: 6F 5E |

**Таблица 2.** Состояния регистров в ходе выполнения второй программы

Листинг третьей программы:

.model small

.stack 100h

.data

\_array dw 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

fio db 'Karelsky Mikhail Konstantinovich', '$'

fPtr dw fio

iPtr dw fio[9]

oPTR dw fio[17]

\_const dw 1103h

\_name db 'Mishka', '$'

symbol db 24

birthDay db ' 11032002', '$'

namePtr dw \_name

.code

Start:

mov ax, @data

mov ds, ax

mov ax, 0000

mov bx, offset \_array

mov ax, [bx] + 2

mov cx, \_array + 2

mov bx, cx

mov cx, \_array

mov dx, \_array[3\*2]

mov di, 4

mov si, \_array[di]

mov di, \_array[0]

mov bp, \_array

mov sp, \_array[4]

mov al, fio[15]

mov ds:1103h, 2002h

mov si, offset \_const

mov cx, [si]

mov ax, 4c00h

int 21h

end Start

END

|  |  |  |
| --- | --- | --- |
| Оператор | Операнд-приемник | |
| До выполнения | После выполнения |
| mov bx, offset \_array | bx: 00 00 | bx: 00 00 |
| mov ax, [bx] + 2 | ax: 00 00 | ax: 00 01 |
| mov cx, \_array + 2 | cx: 00 00 | cx: 00 01 |
| mov bx, cx | bx: 00 00 | bx: 00 01 |
| mov cx, \_array | cx: 00 01 | cx: 00 00 |
| mov dx, \_array[3\*2] | dx: 00 00 | dx: 00 03 |
| mov di, 4 | di: 00 00 | di: 00 04 |
| mov si, \_array[di] | si: 00 00 | si: 00 02 |
| mov di, \_array[0] | di: 00 04 | di: 00 00 |
| mov bp, \_array | bp: 00 00 | bp: 00 00 |
| mov sp, \_array[4] | sp: 01 00 | sp: 00 02 |
| mov al, fio[15] | al: 01 | al: 6C |
| mov ds:1103h, 2002h | ds:1103: 30 58 | ds:1103: 02 20 |
| mov si, offset \_const | si: 00 02 | si: 00 3B |
| mov cx, [si] | cx: 00 00 | cx: 11 03 |

**Таблица 3.** Состояния регистров в ходе выполнения третьей программы

**Вывод:** в ходе выполнения лабораторной работы были изучены команды передачи данных и различные способы адресации операндов.