



**МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»
(РУТ (МИИТ))**

**Институт транспортной техники и систем управления
Кафедра «Управление и защита информации»**

**Отчет по лабораторной работе №1
«Регистры, данные и команды пересылки данных»
по дисциплине
«Машинно-ориентированные языки программирования»**

Выполнил: студент ТКИ-341 Козлов А. Д.

Проверили: Доцент “УиЗИ” Логинова Л.Н.

Зав. лаборатории “УиЗИ” Антонов Д. А.

Москва 2024 г.

Цель работы: изучение регистров общего назначения (РОН) и команд пересылки данных.

Постановка задачи:

1. Занести число из столбца «Число 1» таблицы 1 и строки, соответствующей заданному варианту, в любой РОН;
2. Занести число из столбца «Число 2» таблицы 1 и строки, соответствующей заданному варианту, в незанятый РОН;
3. Занести число из столбца «Число 3» таблицы 1 и строки, соответствующей заданному варианту, в незанятый РОН;
4. Занести число из столбца «Число 4» таблицы 1 и строки, соответствующей заданному варианту, в незанятый РОН;
5. Занести число из столбца «Число 5» таблицы 1 и строки, соответствующей заданному варианту, в незанятый РОН;
6. Занести число из столбца «Число 6» таблицы 1 и строки, соответствующей заданному варианту, в незанятый РОН;
7. Обменять числа, хранящиеся в РОН после выполнения пунктов 1 и 2, между собой, 4-мя разными способами, не потеряв информацию в занятых РОН;
8. Обменять числа, хранящиеся в РОН после выполнения пунктов 3 и 4, между собой, 4-мя разными способами, не потеряв информацию в занятых РОН;
9. Обменять числа, хранящиеся в РОН после выполнения пунктов 5 и 6, между собой, 4-мя разными способами, не потеряв информацию в занятых РОН;
10. Переслать числа, оказавшиеся в РОН после выполнения пункта с номером, взятым из столбца «Пункт 1» таблицы 1 и строки, соответствующей заданному варианту, в любые 16-разрядные РОН без потери знака;
11. Переслать числа, оказавшиеся в РОН после выполнения пункта с номером, взятым из столбца «Пункт 2» таблицы 1 и строки, соответствующей заданному варианту, в любые 32-разрядные РОН расширенные нулем;

Примечание: Регистр ESP и EBP использовать НЕЛЬЗЯ !!!

Номер варианта	Число 1	Число 2	Число 3	Число 4	Число 5	Число 6	Пункт 1	Пункт 2
25	2 516 094 474	1 472 212 419	51 469	16 564	145	86	8	8

Код программы:

```
#include <iomanip>

int main()
{
    int32_t temp_data;
    int32_t num1 = 2516094474; // 0x95F88E0A
    int32_t num2 = 1472212419; // 0x57C02DC3
    int16_t num3 = 51469; // 0xC90D
    int16_t num4 = 16564; // 0x40B4
    int8_t num5 = 145; // 0x91
    int8_t num6 = 86; // 0x56

    __asm {
        PUSHAD

        //1-6
        MOV EAX, num1
        MOV EBX, num2
        MOVZX SI, num3
        MOVZX DI, num4
        MOVZX CL, num5
        MOV CH, num6

        // EDX free

        //7.1
        MOV EDX, EAX
        MOV EAX, EBX
        MOV EBX, EDX
        //7.2
        PUSH EAX
        MOV EAX, EBX
        POP EBX
        //7.3
        XCHG EAX, EBX
        //7.4
        LEA EDX, temp_data
        MOV [EDX], EAX
        MOV EAX, EBX
        MOV EBX, [EDX]

        //8.1
        MOVZX DX, SI
        MOV SI, DI
        MOV DI, DX
        //8.2
        PUSH SI
        MOV SI, DI
        POP DI
        //8.3
        XCHG SI, DI
        //8.4
        LEA EDX, temp_data
        MOV [EDX], SI
        MOV SI, DI
        MOV DI, [EDX]

        //9.1
        MOV DL, CH
        MOV CH, CL
        MOV CL, DL
        //9.2 *
        BSWAP ECX
    }
```

```

        PUSH ECX
        POP CX
        POP CX
        //9.3
        XCHG CH, CL
        //9.4
        LEA EDX, temp_data
        MOV [EDX], CH
        MOV CH, CL
        MOV CL, [EDX]

        //10
        PUSHAD
        MOVSX AX, SI
        MOVSX BX, DI
        POPAD

        //11
        PUSHAD
        MOVZX EAX, SI
        MOVZX EBX, DI
        POPAD

        POPAD
    }
}

```

Ход выполнения:

1. Пункт 1-6:

```
Регистры
EAX = 95F88E0A EBX = 57C02DC3
ECX = 00005691 EDX = 00EE8428
ESI = 0000C90D EDI = 000040B4
EIP = 006E2054 ESP = 00CFFD64
EBP = 00CFFEA8 EFL = 00000286
```

2. Пункт 7:

2.1 Перемещение с помощью другого РОН:

```
Регистры
EAX = 57C02DC3 EBX = 95F88E0A
ECX = 00005691 EDX = 95F88E0A
ESI = 0000C90D EDI = 000040B4
EIP = 006E205A ESP = 00CFFD64
EBP = 00CFFEA8 EFL = 00000286
```

2.2 Перемещение через стек:

```
Регистры
EAX = 95F88E0A EBX = 57C02DC3
ECX = 00005691 EDX = 95F88E0A
ESI = 0000C90D EDI = 000040B4
EIP = 006E205E ESP = 00CFFD64
EBP = 00CFFEA8 EFL = 00000286
```

2.3 Перемещение с использованием XCHG:

```
Регистры
EAX = 57C02DC3 EBX = 95F88E0A
ECX = 00005691 EDX = 95F88E0A
ESI = 0000C90D EDI = 000040B4
EIP = 006E205F ESP = 00CFFD64
EBP = 00CFFEA8 EFL = 00000286
```

2.4 Перемещение через ячейку памяти в куче(LEA):

```
Регистры
EAX = 95F88E0A EBX = 57C02DC3
ECX = 00005691 EDX = 00CFFE9C
ESI = 0000C90D EDI = 000040B4
EIP = 006E2068 ESP = 00CFFD64
EBP = 00CFFEA8 EFL = 00000286
```

3. Пункт 8:

3.1 Перемещение с помощью другого РОН:

```

Регистры
EAX = 95F88E0A EBX = 57C02DC3
ECX = 00005691 EDX = 0000C90D
ESI = 000040B4 EDI = 0000C90D
EIP = 006E2071 ESP = 00CFFD64
EBP = 00CFFEA8 EFL = 00000286

```

3.2 Перемещение через стек:

```

Регистры
EAX = 95F88E0A EBX = 57C02DC3
ECX = 00005691 EDX = 0000C90D
ESI = 0000C90D EDI = 000040B4
EIP = 006E2078 ESP = 00CFFD64
EBP = 00CFFEA8 EFL = 00000286

```

3.3 Перемещение с использованием XCHG:

```

Регистры
EAX = 95F88E0A EBX = 57C02DC3
ECX = 00005691 EDX = 0000C90D
ESI = 000040B4 EDI = 0000C90D
EIP = 006E207B ESP = 00CFFD64
EBP = 00CFFEA8 EFL = 00000286

```

3.4 Перемещение через ячейку памяти в куче(LEA):

```

Регистры
EAX = 95F88E0A EBX = 57C02DC3
ECX = 00005691 EDX = 00CFFE9C
ESI = 0000C90D EDI = 000040B4
EIP = 006E2087 ESP = 00CFFD64
EBP = 00CFFEA8 EFL = 00000286

```

4. Пункт 9:

4.1 Перемещение с помощью другого РОН:

```

Регистры
EAX = 95F88E0A EBX = 57C02DC3
ECX = 00009156 EDX = 00CFFE56
ESI = 0000C90D EDI = 000040B4
EIP = 006E208D ESP = 00CFFD64
EBP = 00CFFEA8 EFL = 00000286

```

4.2 Перемещение через BSWAP и стек:

```

Регистры
EAX = 95F88E0A EBX = 57C02DC3
ECX = 56915691 EDX = 00CFFE56
ESI = 0000C90D EDI = 000040B4
EIP = 006E2094 ESP = 00CFFD64
EBP = 00CFFEA8 EFL = 00000286

```

4.3 Перемещение с использованием XCHG:

```
Регистры
EAX = 95F88E0A EBX = 57C02DC3
ECX = 56919156 EDX = 00CFFE56
ESI = 0000C90D EDI = 000040B4
EIP = 006E2096 ESP = 00CFFD64
EBP = 00CFFEA8 EFL = 00000286
```

4.4 Перемещение через ячейку памяти (LEA):

```
Регистры
EAX = 95F88E0A EBX = 57C02DC3
ECX = 56915691 EDX = 00CFFE9C
ESI = 0000C90D EDI = 000040B4
EIP = 006E209F ESP = 00CFFD64
EBP = 00CFFEA8 EFL = 00000286
```

5. Пункт 10:

```
Регистры
EAX = FFFFC90D EBX = 000040B4
ECX = 56915691 EDX = 003EF854
ESI = 0000C90D EDI = 000040B4
EIP = 006E20A6 ESP = 003EF6FC
EBP = 003EF860 EFL = 00000286
```

6. Пункт 11:

```
Регистры
EAX = 0000C90D EBX = 000040B4
ECX = 56915691 EDX = 003EF854
ESI = 0000C90D EDI = 000040B4
EIP = 006E20AE ESP = 003EF6FC
EBP = 003EF860 EFL = 00000286
```

Вывод:

1. В ходе выполнения лабораторной работы были изучены функции и особенности использования регистров общего назначения (РОН).
2. Было продемонстрировано, как с помощью команд MOV, PUSH, POP, XCHG, BSWAP и LEA можно эффективно организовывать обмен данными между регистрами, стеком и памятью.
3. Также выяснено, что при работе с регистрами важно учитывать их размерность (8-bit, 16-bit, 32-bit) и как сам регистр делится на младшие и старшие части (например, EAX, AX, AH, AL).
4. Были изучены команды знакового (MOVSX) и беззнакового (MOVZX) расширения, которые позволяют корректно пересылать данные между регистрами разной разрядности.
5. Было уделено особое внимание при работе с командами PUSH и POP, так как ошибки при их использовании могут привести к сбоям в работе всей программы.

Заключение

В результате работы были освоены базовые команды пересылки данных и работы с памятью языка ассемблера.

Полученные знания дали представление об организации взаимодействия процессора с памятью и регистрами, как обрабатываются данных разной разрядности. Все это важно для понимания принципов работы вычислительных систем на низком уровне.

Данная лабораторная работа заложила фундамент для дальнейшего изучения языка ассемблера, где базовые команды и конструкции станут основой для реализации более сложных алгоритмов и программ.