**Microprocessor and Assembly Programming Laboratory**

**B.Tech. III Semester**



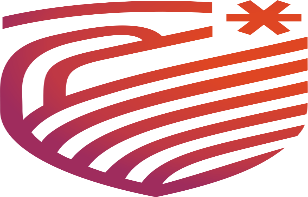
**Name : SUBHENDU MAJI**

**Roll Number : 18ETCS002121**

**Department : Computer Science and Engineering**

**Faculty of Engineering & Technology**

**Ramaiah University of Applied Sciences**



**Ramaiah University of Applied Sciences**

Private University Established in Karnataka State by Act No. 15 of 2013

|  |  |
| --- | --- |
| Faculty | Engineering & Technology |
| Programme | B. Tech. in Computer Science and Engineering |
| Year/Semester | 2018/3rd Semester |
| Name of the Laboratory | Microprocessor and Assembly Programming Laboratory |
| Laboratory Code |  |

**List of Experiments**

|  |  |
| --- | --- |
| 1. Data transfer operations |  |
| 1. Arithmetic operations |  |
| 1. Logical operations |  |
| 1. Controlling execution flow using conditional instructions |  |
| 1. String manipulation |  |
| 1. Searching an element in an array |  |
| 1. Sorting an array |  |
| 1. Interfacing |  |
| 1. Interfacing |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No.** |  |  |  |  | | **Lab Experiment** | **Viva**  **(6)** | **Results**  **(7)** | **Documentation**  **(7)** | **Total Marks**  **(20)** |
| 1 | Data transfer operations |  |  |  |  |
| 2 | Arithmetic operations |  |  |  |  |
| 3 | Logical operations |  |  |  |  |
| 4 | Controlling execution flow using conditional instructions |  |  |  |  |
| 5 | String manipulation |  |  |  |  |
| 6 | Searching an element in an array |  |  |  |  |
| 7 | Sorting an array |  |  |  |  |
| 8 | Interfacing |  |  |  |  |
| 9 | Interfacing |  |  |  |  |
| 10 | Lab Internal Test conducted along the lines of SEE and valued for 50 Marks and reduced for 20 Marks | | | |  |
|  | **Total Marks** | | | |  |

# Laboratory 1

Title of the Laboratory Exercise: Data transfer operations

1. Introduction and Purpose of Experiment

Students will be able to define data of different data types and perform data transfer operations on the data

1. Aim and Objectives

Aim

To develop assembly language program to perform data transfer operations on different data.

Objectives

At the end of this lab, the student will be able to

* + Define data of different data types
  + Perform data transfer operations
  + Create a simple assembly language program
  + Use GAS assembler
  + Understand GNU debugger

1. Experimental Procedure

1. Write algorithm to solve the given problem

2. Translate the algorithm to assembly language code

3. Run the assembly code in GNU assembler

4. Create a laboratory report documenting the work

1. Questions

1. Perform the following data transfer operations

|  |  |
| --- | --- |
| 1. 32 bit integer data to a | General Purpose register  Segment Register  Memory |
| 2. 16 bit integer data to a | General Purpose register  Segment Register  Memory |
| 3. 8 bit integer data to a | General Purpose register  Segment Register  Memory |
| 4. 32 bit integer data from a General purpose register to a  *(Repeat the same for 16 bit integer data and 8 bit integer data)* | General Purpose register  Segment Register  Memory |
| 5. 32 bit integer data from memory to a  *(Repeat the same for 16 bit integer data and 8 bit integer data)* | General Purpose register  Segment Register  Memory |
| 6. 32 bit integer data from memory to | Memory region |

1. Calculations/Computations/Algorithms

* Data transfer instructions move data from one place in the computer to another without changing the data.
* Typical transfers are between memory and processor registers, between processor registers and input and output registers, and among the processor registers themselves.
* Where there are two operands, the rightmost one is the destination. The leftmost one is the source.
* For example, movl %edx, %eax means moves the contents of the edx register into the eax register.

*ALGORITHM*

STEP 1: Start

STEP 2: declare data section

STEP 3: declare variable name, type and value

STEP 4: declare data section, mark it as \_start section

STEP 5: move 32bit value 1 into register eax

STEP 6: move 32bit value of register eax to register ebx

STEP 7: move 32bit value of register eax to memory (variable a)

STEP 8: move 32bit value stored in memory to the register ecx

STEP 9: move 32bit value 2 to memory

STEP 10: repeat same steps for 16bit data and 8bit data

STEP 11: Stop

1. Presentation of Results



Figure 1 Make file

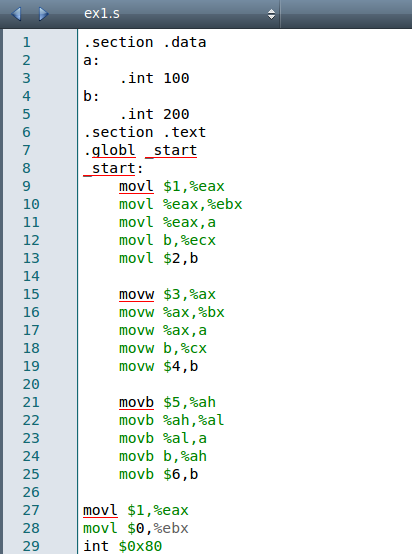


Figure 2 ASM code

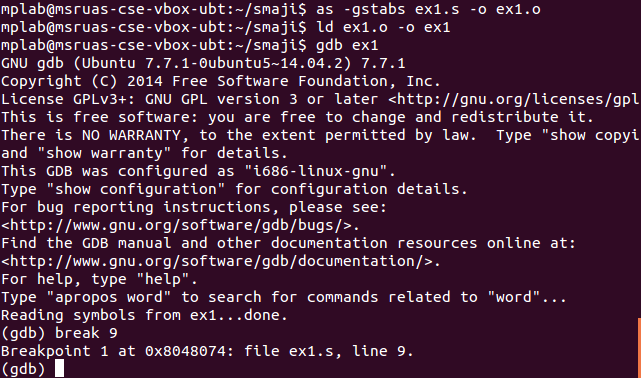


Figure 3 compile and link

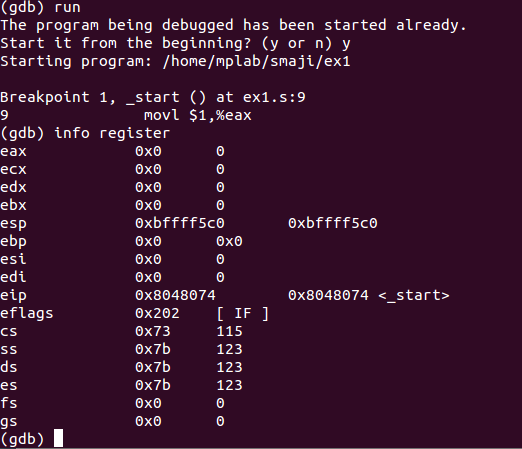


Figure 4 running and register status

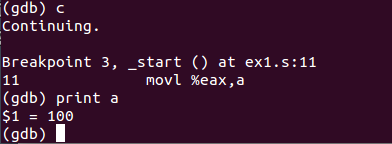


Figure 5 value status

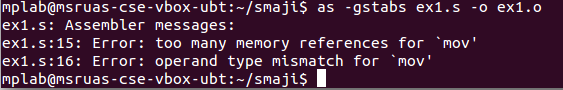


Figure 6 Error Screenshot for trying to move data from memory to memory

1. Analysis and Discussions

The mov instruction copies the data item referred to by its first operand (i.e. register contents, memory contents, or a constant value) into the location referred to by its second operand (i.e. a register or memory). While register-to-register moves are possible, direct memory-to-memory moves are not. The assembler gives error if we try to move data direct from memory-to-memory (Fig. 6). In cases where memory transfers are desired, the source memory contents must first be loaded into a register, then can be stored to the destination memory address.

We cannot move data into segment registers. As we can see in the following code, assembler gives error while linking.

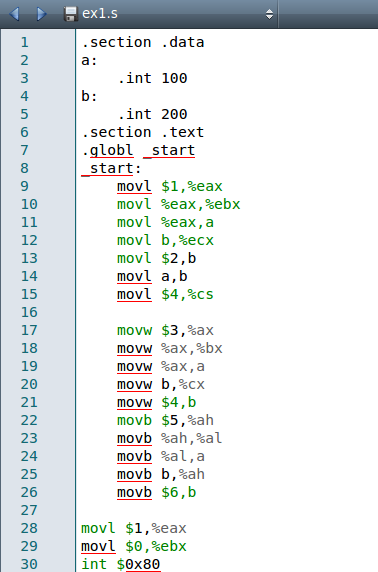


Figure 7 Code snippet

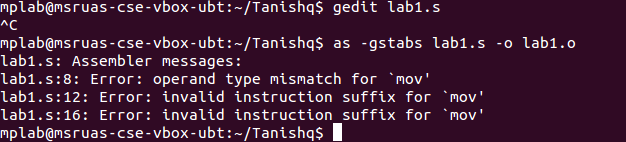


Figure 8 Error screenshot

1. Conclusions

Data can be moved from and between registers and memory using mov , which can be used with b, w, l i.e. 8bit, 16bit, and 32bit data movement.

We successfully executed the doable parts of the question of moving data from registers, memory and segments of the registers.

Signature and date Marks