Microprocessor and Assembly Programming Laboratory

B.Tech. III Semester

Name : KAUSHAL VASHISTH

Roll Number :18ETCS002147

Department : Computer Science and Engineering

Faculty of Engineering & Technology

Ramaiah University of Applied Sciences

Name: KAUSHAI

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	
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	

Name: KAUSHAL VASHISTH Registration Number: 18ETCS002147

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				

Name: KAUSHAL VASHISTH Registration Number: 18ETCS002147

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

2. 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

3. 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

4. Questiobns

   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

5. Calculations/Computations/Algorithms:-

Step1:- define “a” and “b” in section data

Step2:- start

Step3:-move 32 bit integer data to eax general purpose register.

Step4:- move data (32bit) from eax register to ebx register.

Step5:- move data (32bit) from “eax” register to memory (a).

Step6:- move memory data (32bit) of (b) to general register “ecx”.

Step7:- move 32 bit integer data to memory(b).

Step8:- move 32 bit data from memory(a) to memory (b).

Step9:- move 32 bit integer data to segment register(cs)

Step10:- Repeat the same procedure for 16bit and 8bit data.

Step11:- stop

```
.section .data
a:
    .int 46
b:
    .int 10
.section .text
.globl _start
_start:
    movl $19,%eax
    movl %eax,%ebx
    movl %eax,a
    movl b,%ecx
    movl $3,b
    movl a,b

    movl $67,%cs

    movw $20,%ax
    movw %ax,%bx
    movw %ax,a
    movw b,%cx
    movw $4,b
    movw a,b

    movb $21,%ah
    movb %ah,%bh
    movb %ah,a
    movb b,%ch
    movb $4,b
    movb a,b
    movl $1,%eax
    movl $0,%ebx
```

continued.....

```
    movl $1,%eax
    movl $0,%ebx
    int $0x80
```

6. Presentation of Results

Error for memory to memory is found in lines 14,23,30 and also error for integer data to segment register is found in line 16 of the code.

```
exam@msruas-cse-vbox-ubt:~/kaushalv$ as -gstabs lab1.s -o lab1.o
lab1.s: Assembler messages:
lab1.s:14: Error: too many memory references for `mov'
lab1.s:16: Error: operand type mismatch for `mov'
lab1.s:23: Error: too many memory references for `mov'
lab1.s:30: Error: too many memory references for `mov'
```

Results after break point of line 9:-

```
Starting program: /home/exam/kaushalv/lab1

Breakpoint 1, _start () at lab1.s:9
9               movl $19,%eax
(gdb) info register
eax            0x0      0
ecx            0x0      0
edx            0x0      0
ebx            0x0      0
esp            0xbffff050       0xbffff050
ebp            0x0      0x0
esi            0x0      0
edi            0x0      0
eip            0x8048074        0x8048074 <_start>
eflags         0x202    [ IF ]
cs             0x73     115
ss             0x7b     123
ds             0x7b     123
es             0x7b     123
fs             0x0      0
gs             0x0      0
(gdb) c
Continuing.
```

Results after break point of line 15:-

```
Breakpoint 2, _start () at lab1.s:15
15              movw $20,%ax
(gdb) info register
eax            0x13     19
ecx            0xa      10
edx            0x0      0
ebx            0x13     19
esp            0xbffff050       0xbffff050
ebp            0x0      0x0
esi            0x0      0
edi            0x0      0
eip            0x8048090        0x8048090 <_start+28>
eflags         0x202    [ IF ]
cs             0x73     115
ss             0x7b     123
ds             0x7b     123
es             0x7b     123
fs             0x0      0
gs             0x0      0
(gdb) print a
$1 = 19
(gdb) print b
$2 = 3
(gdb)
```

7. Analysis and Discussions
    - Error will be shown for memory to memory data transfer, also there will be error for transferring integer data to segment registers and rest of program will be executed only if these errors are removed.

8. Conclusions

    It can be concluded that we can transfer data from:-

    1. Integer data to general purpose register (respectively to their bits, example:- 32 bit integer data to 32 bit general purpose register (eax)).
    2. It is not possible to transfer data from memory to memory.
    3. Segment registers are just used to read data. It is not possible to write data in segment registers.

Signature and date Marks