Laboratory 2

Title of the Laboratory Exercise: Arithmetic Operations

1. Introduction and Purpose of Experiment

   Students will be able to perform all arithmetic operations and shift operations using assembly instructions

2. Aim and Objectives

   Aim

   To develop assembly language program to perform all arithmetic operations.

   Objectives

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

   - Identify the appropriate assembly language instruction for the given arithmetic operations
   - Perform all arithmetic operations using assembly language instructions
   - Understand different data types and memory used
   - Get familiar with assembly language program by developing simple programs

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. Questions

1. Consider the following source code fragment

   Int a,b,c,d;

   *a= (b + c)-d + (b*c) / d;*

- Assume that *b, c, d* are in registers. Develop an assembly language program to perform this assignment statements.
- Assume that *b is* in registers and *c*, *d* in memory. Develop an assembly language program to perform this assignment statements.

Value of b= 7654321
Value of c= 3110000
Value of d=2344

2. Consider the following source code fragment

Int a,b,c,d;

*A = (b*c) / d;*

Perform multiplication and division by shift operations

5. Calculations/Computations/Algorithms

Q1:-

Part1:-

Step1:- define “a”, “b”, “c”, “d” in section data

Step2:- start

Step3:-move b to ebx general purpose register.

Step4:- move c to eax general purpose register.

Step5:- use command “mull %ebx ” to multiply value of ebx and eax general register and value will be stored in eax.

Step6:- move d to ecx general purpose register.

Step7:- :- use command “divl %ecx ” to divide the value of eax by ebx general register and quotient will be stored in eax.

Step8:- move value of eax to esi general purpose register.

Step9:- move c to eax general purpose register.

Step10:- use command “addl %ebx,%eax ” to add value of ebx and eax general register and value will be stored in eax.

Step11:-similarly do subtraction by “subl %ecx,%eax ” and value will be stored in eax.

Step12:- finally use command “addl %esi,%eax ” to add value of esx and eax general register and value will be stored in eax

Step13:- finally move the value of eax to a.

Step14:- stop

Part2:-

Just change the values of a,b,c,d integers to

Value of b= 7654321
Value of c= 3110000
Value of d=2344

in above algorithm and rest of the algorithm remains same.

Q2:-

Ans:-

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

Step2:- start

Step3:-move b to eax general purpose register.

Step4:- use command “sall $3 %ebx ” to multiply values by 8.

Step5:- move eax to esi general purpose register.

Step6:- use command “sarl $4 %ebx ” to multiply values by 16.

Step7:- :- store the value in a.

Step8:- stop

6. Presentation of Results

Question 1:-

Part1:-

Program 1:-

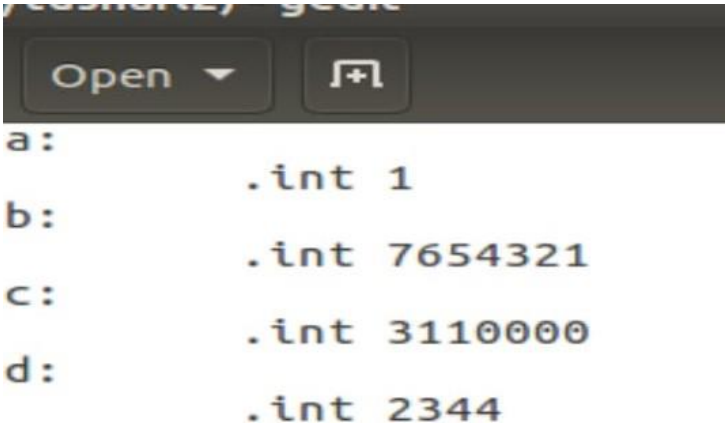
*a= (b + c)-d + (b*c) / d;*

```
.section .data
        b:
                .int 6
        c:
                .int 5
        d:
                .int 10
        a:
                .int

.section .text
.globl _start
_start:

movl b,%ebx
movl c,%eax
mull %ebx

movl d,%ecx
divl %ecx
movl %eax,%esi
movl c,%eax

addl %ebx,%eax
subl %ecx,%eax

addl %esi,%eax
movl %eax,a
```

```
movl $1,%eax
movl $0,%ebx
int $0*80
```

Part 2:-

We just need to change the variable part for part 2 question:-



Question2:-

Program 2:-

*A = (b*c) / d;*

```
.section .data
    b:
        .int 30
    a:
        .int 1

.section .text
.globl _start
_start:

movl b,%eax
sall $3,%eax

movl %eax,%esi
sarl $4,%esi
movl %esi,a

movl $1,%eax
movl $0,%ebx
int $0*80
```

Terminal results:-

Question 1:-

Part1:-

```
Breakpoint 1, _start () at lab2.s:28
28      movl %eax,a
(gdb) info registers
eax            0x4      4
ecx            0xa      10
edx            0x0      0
ebx            0x6      6
esp            0xbffff050       0xbffff050
ebp            0x0      0x0
esi            0x3      3
edi            0x0      0
eip            0x8048096        0x8048096 <_start+34>
eflags         0x202    [ IF ]
cs             0x73     115
ss             0x7b     123
ds             0x7b     123
es             0x7b     123
fs             0x0      0
gs             0x0      0
```

```
Breakpoint 2, _start () at lab2.s:30
30      movl $1,%eax
(gdb) print a
$1 = 4
```

Part 2:-

```
(gdb) info registers
eax            0x10092144       269033796
ecx            0x80490ae        134516910
edx            0x20     32
ebx            0x80490a2        134516898
esp            0xbffff080       0xbffff080
ebp            0x0      0x0
esi            0x0      0
edi            0x0      0
eip            0x8048091        0x8048091 <_start
eflags         0x206    [ PF IF ]
cs             0x73     115
ss             0x7b     123
ds             0x7b     123
es             0x7b     123
fs             0x0      0
gs             0x0      0
(gdb) print a
$3 = 1
(gdb) c
Continuing.
```

Question2:-

```
Breakpoint 3, _start () at lab3.s:18
18      movl $1,%eax
(gdb) print a
$2 = 15
(gdb) info registers
eax            0xf0     240
ecx            0x0      0
edx            0x0      0
ebx            0x0      0
esp            0xbffff050       0xbffff050
ebp            0x0      0x0
esi            0xf      15
edi            0x0      0
eip            0x8048087        0x8048087 <_start+19>
eflags         0x206    [ PF IF ]
cs             0x73     115
ss             0x7b     123
ds             0x7b     123
es             0x7b     123
fs             0x0      0
gs             0x0      0
(gdb) print a
$3 = 15
(gdb)
```

7. Analysis and Discussions

   In question number 1 part 2: there will be overflow of data due to extra bytes of integers declared in section data.

8. Conclusions :-

   Arithmetic operators in assembly language have been studied both practically and theoretically.

   The basic athematic calculation can be performed easily with the help of athematic operators.

Signature and date

Marks