Laboratory 5

Title of the Laboratory Exercise: Searching an element in an array

1. Introduction and Purpose of Experiment

   Students will be able to perform search operations in an array of integers or characters

2. Aim and Objectives

   Aim

   To develop assembly language program to perform search operations in an array

   Objectives

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

   – Identify instructions to be used in assembly language
   – Perform search operations in assembly language

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

   Develop an assembly language program to perform the following:

   1. Searching an element in an array of ‘n’ numbers
   2. Read a sentence with at least one special character and search for the special character and print it. E.g., consider the input {youremailid@msruas.ac.in }

      Output: @, .
   3. Develop an assembly language program to compute the parity of a hexadecimal number stored in the Register1. If Register1 has odd number of ones, update Register2 with 0x01. If Register1 has even number of ones, update Register2 with 0x00.

      Note: Register1 and Register2 can be any General Purpose Registers.

5. Calculations/Computations/Algorithms

```
.section .data
    array:
        .int 1,2,3,4,5,7
    search:
        .int 7
.section .text
.globl _start
_start:
movl $0,%ecx
loop:
    movl array( ,%ecx,4),%eax
    cmpl search,%eax
    jne loop1
loop1:
    addl $1,%ecx
    cmpl $6,%ecx
    jne loop
exit:
    jmp exit

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

Figure 1:represents code for 4.1

```
.section .data
    email:
        .asciz "backspacekaushal@gmail.com"
    search:
        .asciz "@"
.section .text
.globl _start
_start:
movl $0,%eax
loop:
    movb email( ,%eax,1),%bl
    addl $1,%eax
    cmpb search,%bl
    jne loop

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

Figure 2:represents code for 4.2

```
.section .data
.section .text
.globl _start
_start:

        movl $0b010110110,%esi
        movl $0,%ecx
        movl $0,%ebx
loop:
        movl %esi,%eax
        ANDl $1,%eax
        cmpl $1,%eax
        je inc_count
inc_count:
        ADDl $1,%ebx
        jmp shift
shift:
        SARl $1,%esi
        ADDl $1,%ecx
        cmpl $32,%ecx
        JNE loop
        jmp count
count:
        movl %ebx,%eax
        movl $2,%edi
        divl %edi
        cmpl $0,%eax
        JE even
        movl $1,%eax
        jmp exit
even:
        movl $0,%eax
exit:
        movl $1,%eax
        movl $0,%ebx
        int $0x80
```

Figure 3:represents the code for 4.3

6. Presentation of Results

```
Breakpoint 1, exit () at lab555.s:19
19              jmp exit
(gdb) info registers
eax            0x7      7
ecx            0x6      6
edx            0x0      0
ebx            0x0      0
esp            0xbffff050       0xbffff050
ebp            0x0      0x0
esi            0x0      0
edi            0x0      0
eip            0x8048090        0x8048090 <exit>
eflags         0x246    [ PF ZF IF ]
cs             0x73     115
ss             0x7b     123
ds             0x7b     123
es             0x7b     123
fs             0x0      0
gs             0x0      0
(gdb) print search
$1 = 7
(gdb)
```

Figure 4:represents output for 4.1

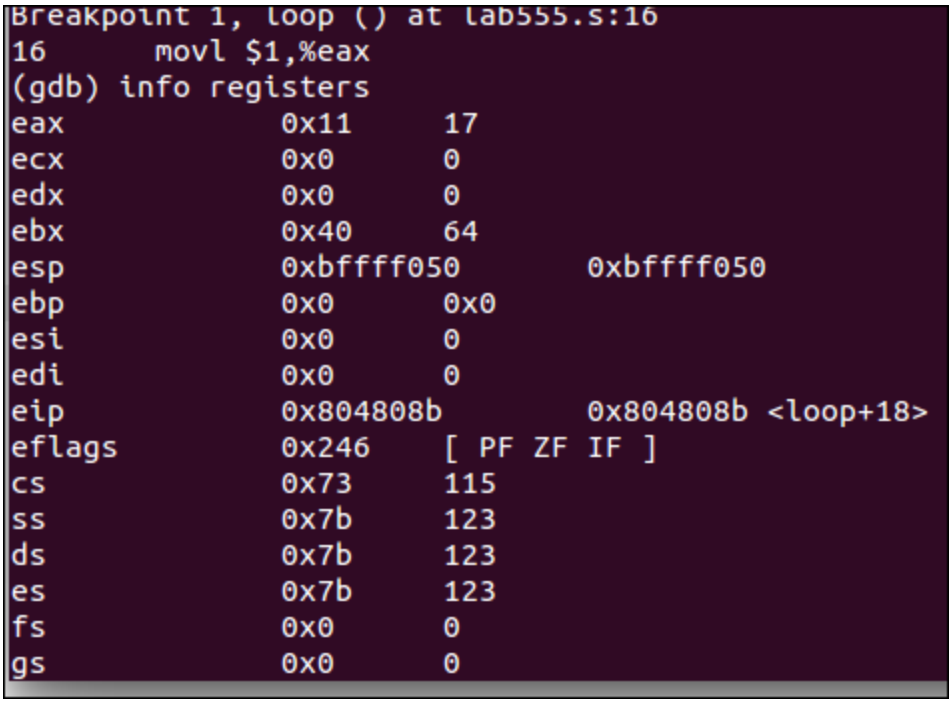


Figure 5:represents output for 4.2

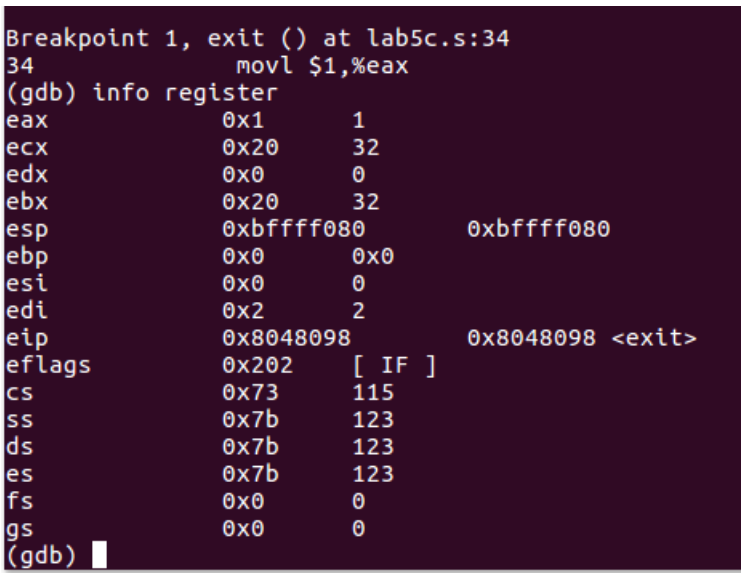


Figure 6: represents the output for 4.3

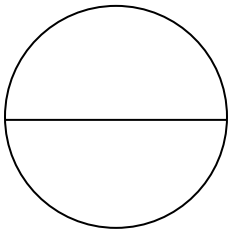
7. Analysis and Discussions

In this experiment, we have solved searching of an integer in an array and also searching a character in a array by the following logic:-

In this code we have used 2 loops, one for increment of the search element and one for the increment of the array value. The comparison command compare the element if found exits the loop or else continues the search.

8. Conclusions

hence, it can be concluded that we have successfully developed assembly language program to perform search operations in an array without any errors. IN PROGRAM 3, we have used “AND ” operator to compute the number of “ones” in hexadecimal number and if the number of ones are even the given register will store 0 else it will store 1(i.e odd).



Signature and date Marks