Name: SATYAJIT GHANA Registration Number: 17ETCS002159

Laboratory 7

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. Searching a character from a string
   3. 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: @, .

5. Calculations/Computations/Algorithms

```
# Searching
.section .data
array:
    .int 1, 2, 3, 4, 5

.section .bss

.section .text

.globl _start

# function for system exit code
_ret:
    movq    $60, %rax               # sys_exit
    movq    $0, %rdi                # exit code
    syscall

# driver function
_start:

    movl $2, %edx       # element to search for = 2
    movl $0, %ebx       # i = 0
    movl $5, %eax

search_loop:
    cmp array( , %ebx, 4), %edx
    je found
    add $1, %ebx
    cmp %ebx, %eax    # check if i == 5
    jne search_loop
    jmp not_found

found:
    movl %ebx, %eax
    jmp _end

not_found:
    movl $-1, %eax
    jmp _end

_end:
    syscall
    call _ret           # exit

```

```
# Search in a String
.section .data
string:
    .ascii "sg159.cs.et17@msruas.ac.in"

key:
    .ascii "@"

.section .bss

.section .text

.globl _start

# function for system exit code
_ret:
    movq    $60, %rax               # sys_exit
    movq    $0, %rdi                # exit code
    syscall

# driver function
_start:

    movl $26, %ecx
    leal string, %edi   # load the effective address
    leal key, %esi
    lodsb
    repne scasb
    jne not_found
    movl $1, %edx
    jmp _end

not_found:
    movl $0, %edx

_end:
    syscall
    call _ret           # exit

```

6. Presentation of Results

```
(gdb) print array@5
$1 = {1, 2, 3, 4, 5}
(gdb) info register eax
eax            0x1      1
(gdb)
```

Figure 0-1 Searching for element in an array

```
(gdb) info register edx
edx            0x1      1
(gdb)
```

Figure 0-2 Searching for character in a string

7. Analysis and Discussions

Code	`lods`
Example	`lodsb`
Explanation	Performs: Loads String Description: Loads a byte, word, or doubleword from the source operand into the AL, AX, or EAX register, respectively. The source operand is a memory location, the address of which is read from the DS:EDI or the DS:SI registers (depending on the address-size attribute of the instruction, 32 or 16, respectively). The DS segment may be overridden with a segment override prefix. At the assembly-code level, two forms of this instruction are allowed: the "explicit-operands" form and the "no-operands" form. The explicit-operands form (specified with the LODS mnemonic) allows the source operand to be specified explicitly. Here, the source operand should be a symbol that indicates the size and location of the source value. The destination operand is then automatically selected to match the size of the source operand (the AL register for byte operands, AX for word operands, and EAX for doubleword operands). This explicit-operands form is provided to allow documentation; however, note that the documentation provided by this form can be misleading. That is, the source operand symbol must specify the correct type (size) of the operand

	(byte, word, or doubleword), but it does not have to specify the correct location. The location is always specified by the DS:(E)SI registers, which must be loaded correctly before the load string instruction is executed.

Code	`scas`
Example	`scasb`
Explanation	Performs: Scans String Description: Compares the byte, word, or double word specified with the memory operand with the value in the AL, AX, or EAX register, and sets the status flags in the EFLAGS register according to the results. The memory operand address is read from either the ES:EDI or the ES:DI registers (depending on the address-size attribute of the instruction, 32 or 16, respectively). The ES segment cannot be overridden with a segment override prefix.

Code	`lea`
Example	`leal`
Explanation	Performs: Load Effective Address Description: Computes the effective address of the second operand (the source operand) and stores it in the first operand (destination operand). The source operand is a memory address (offset part) specified with one of the processors addressing modes; the destination operand is a general-purpose register. The address-size and operand-size attributes affect the action performed by this instruction, as shown in the following table. The operand-size attribute of the instruction is determined by the chosen register; the address-size attribute is determined by the attribute of the code segment.

8. Conclusions

Specific character can be searched in a string using the scas instruction which stands for string scan.

Elements from an array can be searched using a simple loop and incrementing the current element counter, once found the flag register is updated and the loop is broken.

9. Comments

1. Limitations of Experiments

The string search algorithm is only application for searching for a character in a string, it cannot be used to search for matching a substring in a string.

2. Limitations of Results

When comparing the character of strings, the flag register cannot be checked if using the rep instruction, hence the results are limited to only checking for equality of characters.

3. Learning happened

Ways to search for an element in an array was learnt along with searching for a character in a string.

4. Recommendations

When testing the program for searching a character in a string, make sure that the search key is of length 1.

Signature and date Marks