Experiments No:13

Title: Find factorial of a given integer number.

Problem Statement: Write x86 ALP to find the factorial of a given integer number on a command line by using recursion. Explicit stack manipulation is expected in the code.

Objective:

- To understand assembly language programming instruction set
  To understand different assembler directives with example
- To apply instruction set for implementing X86/64 bit assembly language programs

Outcomes: On completion of this practical ,students will be able to

C218.1: Understand and apply various addressing modes and instruction set to implement assembly language programs

Hardware Requirement: NA

Software Requirement: OS: Ubuntu Assembler: NASM version 2.10.07 Linker: ld

Theory Contents : A recursive procedure is one that calls itself. There are two kind of recursion: direct and indirect. In direct recursion, the procedure calls itself and in indirect recursion, the first procedure calls a second procedure, which in turn calls the first procedure.

Recursion could be observed in numerous mathematical algorithms. For example, consider the case of calculating the factorial of a number. Factorial of a number is given by the equation −

Fact (n) = n * fact (n-1) for n > 0

For example: factorial of 5 is 1 x 2 x 3 x 4 x 5 = 5 x factorial of 4 and this can be a good example of showing a recursive procedure. Every recursive algorithm must have an ending condition, i.e., the recursive calling of the program should be stopped when a condition is fulfilled. In the case of factorial algorithm, the end condition is reached when n is 0.

Recursion occurs when a procedure calls itself. The following for example is a recursive procedure:

```
 Recursive
 proc
 callRecursive
 ret
Recursive endp
```

Of course the CPU will never execute the ret instruction at the end of this procedure. Upon entry into Recursive this procedure will immediately call itself again and control will never pass to the ret instruction. In this particular case run away recursion results in an infinite loop.

In many respects recursion is very similar to iteration (that is the repetitive execution of a loop). The following code also produces an infinite loop:

```
Recursive
proc      jmp
Recursive
ret
Recursive endp
```

There is however one major difference between these two implementations. The former version of Recursive pushes a return address onto the stack with each invocation of the subroutine. This does not happen in the example immediately above (since the jmp instruction does not affect the stack).

Like a looping structure recursion requires a termination condition in order to stop infinite recursion. Recursive could be rewritten with a termination condition as follows:

```
Recursive
proc      dec      ax
jzQuitRecurse
call       Recursive
QuitRecurse:
Ret
Recursiveendp
```

This modification to the routine causes Recursive to call itself the number of times appearing in the ax register. On each call Recursive decrements the ax register by one and calls itself again. Eventually Recursive decrements ax to zero and returns. Once this happens the CPU executes a string of ret instructions until control returns to the original call to Recursive.

So far however there hasn't been a real need for recursion. After all you could efficiently code this procedure as follows:

```
Recursive   proc
RepeatAgain:
dec              ax
jnzRepeatAgain
ret
Recursive endp
```

Both examples would repeat the body of the procedure the number of times passed in the ax register. As it turns out there are only a few recursive algorithms that you cannot implement in an iterative fashion. However many recursively implemented algorithms are more efficient than their iterative counterparts and most of the time the recursive form of the algorithm is much easier to understand.

Assignment Questions: -

1. State the difference between Iteration and Recursion?
2. By using which instruction recursion method pushes a return address onto the stack?
3. State the Difference between CALL and JMP instruction.

MPL Practical Oral Question Bank

Sr No	B L	Questions	Oral 1	Oral 2 (improvement)	Remark
1	1	What is the difference between Iteration and Recursion?			
2	1	What is the use for stack in Recursion?			
3	1	What is difference between CALL and JMP instruction.?			

Sign of Student