****

**Green University of Bangladesh**

**Department of Computer Science and Engineering (CSE)**

**Faculty of Sciences and Engineering**

**Semester: (Fall: Year 2023), B.Sc. in CSE (Day)**

**Lab Report No:** 04

**Course Title:** Microprocessor & Microcontroller Lab

**Course Code:** CSE 304 **Section:** 213D2

**Lab Experiment Name:** Implementation of loop using assembly language

**Student Details**

|  |  |  |
| --- | --- | --- |
| **Name** | | **ID** |
| **1.** | Md Masud Rana | 213902101 |

**Lab Date :** 19-10-2023

**Submission Date :** 26-10-2023

## **Course Teacher’s Name :** Sudip Ghoshal

|  |
| --- |
| **Lab Report Status**  **Marks: ………………………………… Signature:.....................**  **Comments:.............................................. Date:..............................** |

**1. TITLE OF THE LAB REPORT EXPERIMENT**

Implementation of loop using assembly language.

**2. OBJECTIVES/AIM**

* To understand basic loop
* To simplify repetitive tasks and reduce code redundancy.
* To enable the execution of specific operations for a specified number of iterations.
* To efficiently repeat a block of instructions.

**3. PROCEDURE**

**Problem-1: Sum of 1+2+3+4+…..+99 natural number series**

Step 1: Set the data segment.

Step 2: Initialize AX to 0 and set the initial value of i to 1.

Step 3: Start the loop 'doSum.'

Step 4: Add the value of i to AX and increment i by 2.

Step 5: Check if i is less than or equal to 99. If true, continue the loop.

Step 6: If i is greater than 99, exit the program.

Step 7: store ax result to ‘sum’ variable

**Problem-2: Find factorial of a given number**

Step 1: Set the data segment and stack.

Step 2: Take input from the user using DOS interrupt 21h.

Step 3: Subtract 48 from the ASCII value to convert it to a numerical digit.

Step 4: Set up the counter CX with the input value and initialize AX to 1.

Step 5: Begin the loop 'doFact' to calculate the factorial iteratively.

Step 6: Store the computed factorial result in the variable 'fact' defined in the data segment.

**Problem-3: Check a number Odd or Even & if input N then exit the program**

Step 1: Set the data segment and stack.

Step 2: Display the message to prompt the user for input.

Step 3: Take a number from the user using DOS interrupt 21h and store the input in register AL.

Step 4: Store the input value into BL for checking if it is 'N' to exit the program.

Step 5: Divide the input by 2 and store the remainder in AH to check if the number is even or odd.

Step 6: Compare the input with 'N' and exit the program if it matches.

Step 7: If the input is even, display the message 'Even'; otherwise, display 'Odd' and repeat the process.

**4. IMPLEMENTATION**

**Problem-1: Sum of 1+3+5+…..+99 odd natural number series [using loop]**

|  |  |
| --- | --- |
| ;summation of 1+3+5+7..+99 odd number  org 100h  .model small  .stack 100h  .data  i dw 1  sum dw ?  .code  main proc  ;set data segment  mov ax, @data  mov ds, ax  mov ax, 0 | doSum:  add ax, i  inc i ;increament i by 2 times  inc i    ;condition check to stop loop or continue  cmp i, 99  JLE doSum  JG exit    exit:  ;before exit program, set ax to sum  mov sum, ax  mov ah, 4ch  int 21h    main endp  end main |

**Problem-2: Find factorial of a given number**

|  |  |
| --- | --- |
| ;factorial of a number  org 100h  .model small  .stack 100h  .data  msg dw 'Enter a number: $'  fact dw ? ;sum may big so taken 16bit    .code  main proc    mov ax, @data  mov ds, ax    mov ah, 9  lea dx, msg  int 21h | mov ah, 1  int 21h  sub al, 48    mov cx, 0  mov cl, al ;i have used cl as counter    ;since need to multiply like 1x2x3x4x.....  mov ax, 1    doFact:  mul cx ;ax= ax\*cx  loop doFact    mov fact, ax ;transfer b to sum variable    main endp  end main |

**Problem-3: Check a number Odd or Even & if input N then exit the program**

|  |  |
| --- | --- |
| ;conditional loop  org 100h  .model small  .stack 100h  .data  msg1 db 10, 13,'Enter a number: $'  msg2 db 10, 13, 'Even $'  msg3 db 10, 13, 'Odd $'  .code  main proc  mov ax, @data  mov ds, ax    again:  mov ah, 9  lea dx, msg1  int 21h    mov ah, 1  int 21h ;take a number from user  ;store user input into bl to check user input is N or not  mov bl, al    mov ah, 0  mov dl, 2  div dl | ;when input is N then exit the program  cmp bl, 'N'  JE exit    ;if ah equal 0 goto even level  cmp ah, 0  JE even  ;else if not equal, goto odd level  JNE odd    even:  mov ah, 9  lea dx, msg2  int 21h  jmp again    odd:  mov ah, 9  lea dx, msg3  int 21h  jmp again    exit:  mov ah, 4ch  int 21h    main endp  end main |

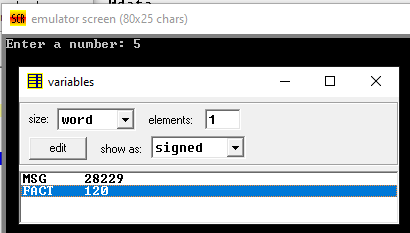
**5. TEST RESULT / OUTPUT**

**Summation of 99th natural odd number series: **

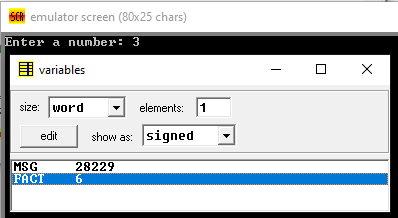
*Figure-1: Summation of 99th natural number using loop*

- The program computes the sum of the natural numbers from 1 to 99.

- The final value of the sum is stored in the variable 'sum' declared in the data segment.

**Find factorial of a given number**

*Figure-2: Factorial of a number and showed into variable*



*Figure-3: Factorial of a number and showed into variable*

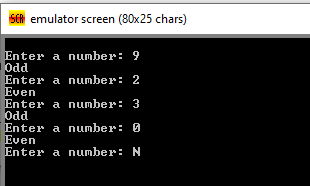
- Running the program to calculate the factorial of the number 5

- It shows the factorial of 5 as 120 showed in figure-2

- Second text case I have given 3 to find factorial and it return 6 as answer showed in figure-3

- The final factorial value is stored in the variable 'fact' declared in the data segment.

**Check a number Odd or Even & if input N then exit the program**

****

*Figure-4:* Check a number Odd or Even & if input N then exit the program

Test Cases:

- Test 1: Inputed an odd number 9 to the program.

- Test 2: Inputed an even number 2 to the program.

- Test 3: Inputed a character N to the program.

Output:

For Test 1: The program displays the message 'Odd'

For Test 2: The program displays the message 'Even' upon inputting the number 2, which is an even number.

For Test 3: The program exits gracefully upon inputting the character 'N', as per the exit condition.

**6. ANALYSIS AND DISCUSSION**

* The first problem, involving the summation of a natural odd number series, effectively utilizes a loop to compute the sum from 1 to 99. The algorithm demonstrates an efficient approach to handle iterative processes, showcasing the significance of proper initialization and control flow within a loop structure.
* In the second problem, calculating the factorial of a given number, the program uses iterative multiplication to find the factorial. This approach demonstrates the effective use of the 'MUL' instruction within a loop construct. It effectively manages the multiplication of consecutive numbers to determine the factorial.
* The third problem, determining whether a given number is even or odd, showcases the use of conditional branching in the context of user input. The program efficiently handles different types of input, displaying the appropriate message based on the parity of the input number. It effectively integrates interrupt services and conditional statements to achieve the desired functionality.

**7. SUMMARY**

The provided programs successfully address the respective problems using loop constructs, iterative multiplication, and conditional branching. They demonstrate effective utilization of the emu8086 microprocessor's capabilities in handling repetitive tasks, iterative calculations, and conditional operations. The programs showcase the significance of proper control flow, variable management, and efficient use of assembly language instructions. Despite the inherent complexity of low-level programming, the solutions effectively demonstrate the logical flow and control required to tackle various computational challenges.