CPSC 240: Computer Organization and Assembly Language

Assignment 08, Fall Semester 2023

CWID:\_885857847\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Name: Kush Patel\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Download the "CPSC-240 Assignment08.docx" document.
2. Design the "macro.asm" program, input a value n (n=001 ~ 999) from the keyboard, calculate 1+2+3+…+n, and display the calculation result in the terminal emulator window. The corresponding C/C++ code is as follows:

#begin define print(string, numOfChar)

rax = 1;

rdi = 1;

rsi = &string;

rdx = numOfChar;

syscall;

#end

#begin define scan(buffer, numOfChar)

rax = 0;

rdi = 0;

rsi = &buffer;

rdx = numOfChar;

syscall;

#end

char buffer[4];

int n;

int sumN;

char msg1[26] = "Input a number (001~999): ";

char msg2[16] = "1 + 2 + 3 +...+ ";

char msg3[4] = " = ";

char ascii[10];

print(msg1, 26);

scan(buffer, 4);

n = atoi(buffer);

rsi = 0;

do {

sumN += rsi;

} while(rsi >= 0);

ascii = itoa(sumN);

print(msg2, 20);

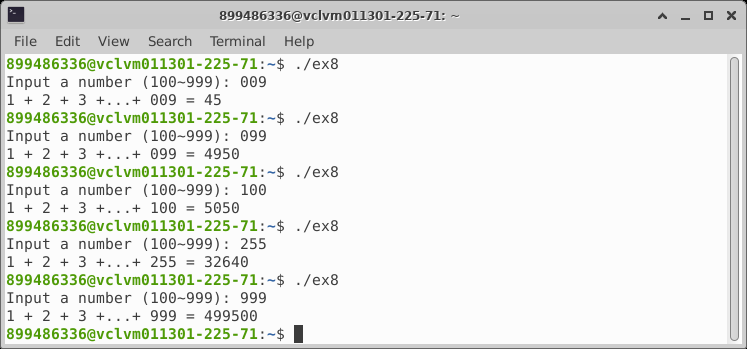
print(buffer, 3);

print(msg3, 3);

print(ascii, 7);

1. Run the "macro" file to display the calculation result in the Terminal Emulator window.
2. Insert source code (macro.asm) and simulation results (Terminal Emulator window) at the bottom of the document. Write an analysis to verify the simulation results.
3. Save the file in pdf format and submit the pdf file to Canvas before 23:59 pm on 11/12/2023.

Simulation Sample:



[Insert macro.asm source code here]

; ex8\_cinMacro.asm

; #begin define print(string, n)

; rax = 1;

; rdi = 1;

; rsi = &string;

; rdx = n;

; syscall;

; #end

; #begin define scan(buffer, n)

; rax = 0;

; rdi = 0;

; rsi = &buffer;

; rdx = n;

; syscall;

; #end

; char buffer[4];

; long n;

; short sumN;

; char msg1[] = "Input a number (100~140): ";

; char msg2[] = "1 + 2 + 3 +...+ N = ";

; char ascii[] = "0000\n";

;

; print(msg1, 26);

; scan(buffer, 4);

; n = atoi(buffer);

; rsi = 0;

; do {

; sumN += rsi;

; } while(rsi <= n);

; ascii = itoa(sumN);

; print(msg2, 20);

; print(ascii, 5);

%macro print 2

mov rax, 1 ;SYS\_write

mov rdi, 1 ;standard output device

mov rsi, %1 ;output string address

mov rdx, %2 ;number of character

syscall ;calling system services

%endmacro

%macro scan 2

mov rax, 0 ;SYS\_read

mov rdi, 0 ;standard input device

mov rsi, %1 ;input buffer address

mov rdx, %2 ;number of character

syscall ;calling system services

%endmacro

section .bss

         buffer resb 4

         n resd 1

         sumN resd 1

         ascii resb 10

section .data

         msg1 db "Input a number (001~999): "

         msg2 db "1 + 2 + 3 +...+ "

         msg3 db " = "

section .text

         global \_start

\_start:

         print msg1, 26 ;cout << msg1

         scan buffer, 4 ;cin >> buffer

         mov ax, 0 ;clear ax

         mov bx, 10 ;bx = 10

         mov rsi, 0 ;counter = 0

         next0:

         mov cl, byte[buffer+rsi]

         and cl, 0fh ;convert ascii to number

         add al, cl ;al = number

         adc ah, 0 ;ah = 0

         cmp rsi, 2 ;compare rsi with 2

         je skip0 ;if rsi=2 goto skip0

         mul bx ;dx:ax = ax \* bx

         skip0:

         inc rsi ;rsi++

         cmp rsi, 3 ;compare rsi with 3

         jl next0 ;if rsi<3 goto next0

         mov word[n], ax ;n = ax

         ; calculates 1+2+3+...+N

         mov ecx, 0 ;cx = 0

         next1:

         add dword[sumN], ecx ;sumN += cx

         inc ecx ;cx++

         cmp ecx, dword[n] ;compare cx with n

         jbe next1 ;if(cx<=100) goto next1

        ; converts sumN into ascii

        ; Part A - Successive division

        mov     eax, dword[sumN]            ;get integer

         mov    rcx, 0                  ;digitCount = 0

        mov     ebx, 10                 ;set for dividing by 10

divideLoop:

    mov     edx, 0

    div     ebx                     ;divide number by 10

    push    rdx                     ;push remainder

    inc     rcx                     ;increment digitCount

    cmp     eax, 0                  ;if (result > 0)

    jne     divideLoop              ;goto divideLoop

    ; Part B - Convert remainders and store

    mov     rbx, ascii              ;get addr of ascii

    mov     rdi, 0                  ;rdi = 0

popLoop:

    pop     rax                     ;pop intDigit

    add     al, "0"                 ;al = al + 0x30

    mov     byte [rbx+rdi], al          ;string[rdi] = al

    inc     rdi                     ;increment rdi

    loop    popLoop                 ;if (digitCount > 0) goto popLoop

    mov     byte [rbx+rdi], 10          ;string[rdi] = newline

    print   msg2, 16                ;cout << msg2

    print   buffer, 3               ;cout << buffer

    print   msg3, 3                 ;cout << " = "

    print   ascii, 7                ;cout << ascii

mov rax, 60 ;terminate program

mov rdi, 0 ;exit status

syscall ;calling system services

[Insert macro simulation result here]

A screenshot of a computer

Description automatically generated

[Insert macro simulation verification here]

A white background with black and white clouds

Description automatically generated with medium confidenceA white background with black and white clouds

Description automatically generated