Министерство науки и образования РФ
Федеральное государственное автономное образовательное
учреждение высшего профессионального образования
«Санкт-Петербургский государственный электротехнический
университет «ЛЭТИ» им. В. И. Ульянова (Ленина)»
(СПбГЭТУ «ЛЭТИ»)
Факультет компьютерных технологий и информатики

Кафедра вычислительной техники

Отчёт
по лабораторной работе № 5
на тему:
“Вызов подпрограмм и стековые операции”

по дисциплине “Организация ЭВМ и Систем”

Вариант 3

Выполнил студент гр. 4306: Табаков А.В.
Принял: Манирагена Валенс

Санкт-Петербург
2016

Цель

Ознакомиться с вызовом подпрограмм и стековыми операциями на языке ассемблера intel 8086.

Задание

Реализовать вызов подпрограммы для ввода массива. Передать длину массива и начальный адрес через стек.

Текст программы

```
 .Model small
.Stack 1000h
.Data
    greeting db "This programm demonstrate procedures and stack operations", 0dh, 0ah,
"$"
    textSizeArr db "Please input size of array from 1 to 9", 0dh, 0ah,"array size = $"
    textArr db "ARR[$"
    printAver db "Average = int:$"
    printMod db " mod:$"
    errorMsg db "something went wrong", 0dh, 0ah,"$"
    question db "Press any key for retry, 0 to exit", 0dh, 0ah, "$"
    endl db 0ah, 0dh, "$"
    arrSize dw ?
    negative dw ?
    buffer db 6       ;max num with 5 symbols
    blength db ?
.Code

start:
    mov ax, @data
    mov ds, ax
    mov ax, 4000
    mov es, ax
    call setDisp

    lea dx, greeting   ;greeting message
    mov ah, 09h
    int 21h

    lea dx, textSizeArr   ;array size =
    mov ah, 09h
    int 21h
    call input
    mov arrSize, ax

    push ax
```

```
    push es

    call inputArr

    mov cx, arrSize
    xor di,di
    xor ax,ax
SumAver:
    mov bl, es:[di]
    xor bh,bh
    add ax, bx
    inc di
    loop SumAver

    mov bx, arrSize
    xor dx, dx
    div bx

    push dx
    push ax
    lea dx, printAver   ;Average = int:
    mov ah, 09h
    int 21h
    pop ax
    call printAX

    lea dx, printMod   ; mod:
    mov ah, 09h
    int 21h
    pop ax
    call printAX
    mov dl, "/"
    mov ah, 2h
    int 21h
    mov ax, arrSize
    call printAX
    call endlp
    lea dx, question     ;Enter any key for retry, 0 to exit
    mov ah, 09h
    int 21h
    mov ah, 01h
    int 21h
    cmp al, '0'
    jz stopLab
    jmp start

stopLab:
    call quit
```

```
proc setDisp
    xor dx,dx     ;cursor's position
    mov ah,02h    ;set at (0,0)
    int 10h
    mov bl,00001010b  ;colors green on black
    mov cx,30*80      ;count of simbols on display
    mov ax,0920h      ;printing 30*80 spaces
    int 10h
    ret
endp

proc quit
    mov ax, 4c00h ; exit to operating system.
    int 21h
endp

proc endlp        ;press enter
    push dx
    push ax
    lea dx, endl
    mov ah, 09h
    int 21h
    pop ax
    pop dx
    ret
endp

proc input
    push di
    push si
    push cx
    lea dx, buffer                ;buffer's address
    mov ah,0ah          ;write in buffer
    int 21h

;from string to bin

    mov di, 2         ;start of buffer
    xor ax,ax         ;clear ax
    mov cl, blength
    xor ch,ch
    xor bx,bx
    add cx, 2
    mov si,cx         ;buffer's length
    mov cl,10         ;multiplier
    mov negative, 0
    mov bl, byte ptr buffer[di]
```

```
    cmp bl, '-'
    jnz toHex
    mov negative, 1
    inc di

toHex:
    mov bl,byte ptr buffer[di]
    sub bl,'0'          ;num = num's code - 30h
    jb  badInp          ;if symbol not a num
    cmp bl,9            ;same
    ja  badInp          ;try input again
    mul cx              ;multiply on 10
    add ax,bx           ;+new num to ax
    inc di                  ;next symbol
    cmp di,si           ;if di<blength + 1
    jb toHex

    mov bx, negative
    cmp bx, 1
    jnz endInp
    neg ax
nM:
    jmp endInp


badInp:
    jmp start

endInp:
    pop cx
    pop si
    pop di
    ret
endp

proc printAX
    push cx
    push bx
    mov bx,0ah        ;divider
    xor cx,cx       ;clear count

divloop:
    xor dx,dx       ;clear dx
    div bx          ;divide on 10
    add dx,'0'      ;make a symbol from num
    push dx             ;save dx
    inc cx
    test ax,ax      ;if ax!=0
```

```
    jnz divloop          ;continue to divide

restore:
    pop ax                      ;read from stack
    mov dx, ax
    mov ah,2          ;print symbol from dl
    int 21h           ;
    loop restore
    pop bx
    pop cx
    ret
endp

proc inputArr
    pop ax
    pop es
    mov es:[11], ax
    pop ax

    mov cx, ax
    xor di, di
    xor si, si

inp:
    call endlp
    lea dx, textArr    ;DSARR[
    mov ah, 09h
    int 21h
    mov ax, si
    call printAX           ;i for DSARR[i]
    mov dl, ']'
    mov ah, 02h
    int 21h
    mov dl, '='
    mov ah, 02h
    int 21h
    call input                ;write in DSARR[i] arr num
    mov es:[di], ax
    inc di
    inc si
    loop  inp
    call endlp
    push es:[11]
    ret
endp

end start ; set entry point and stop the assembler.
```

Трассировка команд переходов				
Адрес	Мнемокод	Двоичный код	Изменения данных	Комментарий
0021	push ax	Байт 1: 01010000 0101 – операция со стеком 0 – втолкнуть в стек 000 – код регистра ax	ss:[1000] = arrSize	Записали в стек длину массива
0022	push es	Байт 1: 00000110 0000 – операция со стеком 011 – код es 0 – вытолкнуть из стека	ss:[00FE] = 4000	Записали в стек адрес начала es
0023	call inputArr	Байт 1: 11101000 – операция вызова процедуры (rel 16, call near, relative, displacement relative to next instruction) Байт 2-3: смещение		Вызов процедуры inputArr
….				
011D	pop ax	Байт 1: 01011000 0101 – операция со стеком 1 – вытолкнуть из стека 000 – код регистра ax	ax = arrSize	Записали в ax переданную длину массива
011E	pop es	Байт 1: 00000111 0000 – операция со стеком 011 – код es 1 – вытолкнуть из стека	es = 4000	Записали в es переданный адрес начала
…				
0157	ret	Байт 1: 11000011 – операция возврата	ip = 0026	
-	ret 12	Байт 1: 11000010 – операция возврата с коррекцией стека Байт 2-3: смещение в стеке		

Вывод

Я ознакомился с вызовом подпрограмм и стековыми операциями на языке ассемблера intel 8086.