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 № 4
на тему:
“Безусловные переходы”

по дисциплине “Организация ЭВМ и Систем”

Вариант 1

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

Санкт-Петербург
2016

Цель

Ознакомиться с безусловными переходами на языке ассемблера intel 8086.

Задание

Реализовать две конструкции переходов: if-then-else и switch-case.

Текст программы

```
.Model small
.Stack 1000h
.Data
   greeting db "This programm demonstrate if\switch constructs work", 0dh, 0ah, "$"
   help db "firstly if statement  if( AH>=0 ) then {z = r^2 + s^2} else {z = (r + s)/(r-s)}",
0dh, 0ah
   db "second action is switch-case construct", 0dh, 0ah
   db " 1) z = r/s + s/r + 10", 0dh, 0ah
   db " 2) z = r*(s + 15)", 0dh, 0ah
   db " 3) z = r^2 - r*s + s^2", 0dh, 0ah
   db " 4) z = r^2 - r*s + s^2", 0dh, 0ah, "$"
   enterR db "Please input r from 1 to 90", 0dh, 0ah,"r = $"
   enters db "Please input s from 1 to 90", 0dh, 0ah,"s = $"
   enterAH db "Please input AH from -100 to 100", 0dh, 0ah,"AH = $"
   printZ db "z = $"
   printInt db "int: $"
   printMod db " mod: $"
   plusSym db " + $"
   firstAct db "first action if( AH >= 0 )", 0dh, 0ah,"$"
   secAct db "switch-case construct", 0dh, 0ah,"enter case from 1 to 4", 0dh, 0ah,"case =
$"
      errorMsg db "something went wrong", 0dh, 0ah,"$"
   question db "Press any key for retry, 0 to exit", 0dh, 0ah, "$"
   endl db 0ah, 0dh, "$"
   r dw 5
   s dw 3
   zI dw ?
   zM dw ?
   zM1 dw ?
   z dw ?
   negative dw ?
   temp dw ?
   buffer db 6      ;max num with 5 symbols
   blength db ?
   arrOfLabels dw 4 dup(?)
.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, help       ;help message
    mov ah, 09h
    int 21h

    lea dx, enterR     ;enter r num
    mov ah, 09h
    int 21h

    call input         ;input r
    mov r, ax
    mov es:[r], ax
    call endlp

    lea dx, enterS     ;enter s num
    mov ah, 09h
    int 21h

    call input         ;input s
    mov s, ax
    mov es:[s], ax
    call endlp

    lea dx, enterAH    ;enter AH num
    mov ah, 09h
    int 21h

    call input         ;input ax
    shl ax, 8
    call endlp

    push ax
    lea dx, firstAct   ;first action if( AH >= 0 )
    mov ah, 09h
    int 21h
    pop ax
;**************************************************
    test ah, 80h       ;if( AH>=0 ) then {z = r^2 + s^2} else {z = (r + s)/(r-s)}
    jnz elseState
```

```
    mov ax, r
    mov bx, ax
    mul bx
    mov z, ax
    mov ax, s
    mov bx, ax
    mul bx
    mov bx, z
    add ax, bx

    mov z, ax
    lea dx, printZ    ;z =
    mov ah, 09h
    int 21h
    mov ax, z
    call printAX
    call endlp

    jmp myEndIf

elseState:       ;{z = (r + s)/(r-s)}
    xor dx,dx
    mov ax, r
    mov cx, ax
    mov bx, s
    add ax, bx
    sub cx, bx
    div cx

    mov zI, ax
    mov zM, dx
    lea dx, printZ    ;z =
    mov ah, 09h
    int 21h
    lea dx, printInt  ;int:
    mov ah, 09h
    int 21h
    mov ax, zI
    call printAX
    lea dx, printMod  ; mod:
    mov ah, 09h
    int 21h
    mov ax, zM
    call printAX
       mov dl, "/"
    mov ah, 2h
    int 21h
```

```
    mov ax, r
    mov bx, s
    sub ax, s
    call printAX
    call endlp

myEndIf:
;*****************************************
    lea dx, secAct    ;switch-case construct
    mov ah, 09h
    int 21h
    call input
    call endlp

    cmp al, 5
    jc SwitchLabel
    lea dx, errorMsg     ;something went wrong
    mov ah, 09h
    int 21h
    jmp endMySwitch

SwitchLabel:
    cmp al, 1
    jz first
    jmp forJump

first:        ;z = r/s + s/r + 10
    mov ax, es:[r]
    mov bx, es:[s]
    xor dx,dx
    div bx
    mov es:[zM], dx
    mov es:[zI], ax

    mov ax, bx
    mov bx, es:[r]
    xor dx, dx
    div bx
    mov es:[zM1], dx

    mov bx, es:[zI]
    mov cx, es:[zM]
    mov dx, es:[zM1]
    add ax, bx
    ;add dx, cx
    cmp dx, 0Ah
    jc checkDiv
    inc ax
```

```
    sub dx, 0Ah
    mov es:[zM1], dx

checkDiv:
    cmp cx, 0Ah
    jc divOk
    inc ax
    sub cx, 0Ah
    mov es:[zM], cx
    jmp divOk

forJump:
    cmp al, 2
    jz sec
    cmp al, 3
    jz third
    cmp al, 4
    jz fourth

divOk:
    add ax, 10d
    mov es:[zI], ax
    ;mov es:[zM], dx
    lea dx, printZ    ;z =
    mov ah, 09h
    int 21h
    lea dx, printInt  ;int:
    mov ah, 09h
    int 21h
    mov ax, es:[zI]
    call printAX

    lea dx, printMod  ; mod:
    mov ah, 09h
    int 21h

    mov ax, es:[zM]
    call printAX
       mov dl, "/"
    mov ah, 2h
    int 21h
    mov ax, es:[s]
    call printAX

       lea dx, plusSym
    mov ah, 9h
    int 21h
```

```
    mov ax, es:[zM1]
    call printAX
       mov dl, "/"
    mov ah, 2h
    int 21h
    mov ax, es:[r]
    call printAX
    call endlp

    jmp endMySwitch

sec:            ;z = r*(s + 15)
    mov ax, es:[s]
    add ax, 15d
    mov bx, es:[r]
    mul bx
    jmp endSwitch

third:          ;z = r^2 - r*s + s^2
fourth:
    mov ax, es:[r]
    mov bx, ax
    mul bx
    mov cx, ax
    mov ax, es:[r]
    mov bx, es:[s]
    mul bx
    sub cx, ax
    mov ax, es:[s]
    mov bx, ax
    mul bx
    add ax, cx

endSwitch:
    mov es:[z], ax
    lea dx, printZ    ;z =
    mov ah, 09h
    int 21h
    mov ax, es:[z]
    call printAX
    call endlp

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

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

Трассировка команд переходов				
Адрес	Мнемокод	Двоичный код	Изменения данных	Комментарий
0092	call printAX	Байт 1: 11101000 – операция вызова процедуры (rel 16, call near, relative, displacement relative to next instruction) Байт 2-3: смещение		Вызов процедуры printAX
0095	call endlp	Байт 1: 11101000 – операция вызова процедуры (rel 16, call near, relative, displacement relative to next instruction) Байт 2-3: смещение		Вызов процедуры endlp
0098	jmp myEndIf	Байт 1: 11101011 – операция перехода (rel 8, jump short, relative, displacement relative to next instruction) Байт 2: смещение		Переход на метку myEndIf
….				
00EF	jmp endMySwitch	Байт 1: 11101001 – операция перехода (rel 16, jump near, relative, displacement relative to next instruction) Байт 2-3: смещение		Переход на метку myMySwitch

Вывод

Я ознакомился с безусловными переходами на языке ассемблера intel 8086.