

分支结构程序实验

实验代码:

```
DSEG SEGMENT
    LUT DB '0123456789ABCDEF$'
    STR_ENT DB 'Input a decimal number to get hex equivalent: ','$'
    STR_EQU DB ' -> ','$'
    STR_ERR_1 DB 'The string ','$'
    STR_ERR_2 DB ' you typed is illegal',0AH,'$'
    ORG 512
    BUF DB 255
    ORG 1024
    STR_OUT DB (?)
DSEG ENDS

SSEG SEGMENT
    DB 0
SSEG ENDS

CODE SEGMENT
    ASSUME cs:CODE, ds:DSEG, es:DSEG, ss:SSEG
START:
    mov ax, DSEG
    mov ds, ax
    mov es, ax
    mov ax, SSEG
    mov ss, ax
    xor ax, ax
    xor sp, sp
    xor bp, bp
MAIN:
    ; print welcome string
    lea dx, STR_ENT
    mov ah, 09H
    int 21H
    ; get user string input
    lea dx, BUF
    mov ah, 0AH
    int 21H
    ; Hold the input on console
    lea dx, STR_ENT
    mov ah, 09H
    int 21H
    lea dx, BUF
    call STR_ENDING
    add dx, 2
    mov ah, 09H
    int 21H
    ; convert the oct input to binary value
```

```

    lea dx, BUF
    call CONVERT
    ; convert binary to hex string, and STR_OUT
    lea dx, STR_OUT
    call FORMAT_HEX
    lea dx, STR_EQU
    mov ah, 09H
    int 21H
    lea dx, STR_OUT
    mov ah, 09H
    int 21H
EXIT:
    mov ah, 4CH
    mov al, 00H
    int 21H
STR_ENDING:
    mov bx, dx
    add bx, 1
    mov bl, [bx]
    xor bh, bh
    add bx, 2
    add bx, dx
    mov [bx], byte ptr '$'
    ret
CONVERT:
    mov bx, dx
    inc bx
    xor cx, cx
    mov cl, [bx]
    inc bx
    xor ax, ax
    xor dx, dx
    cmp cx, 0
    je CVT_FINAL
CVT_L1:
    xor dx, dx
    mov dl, [bx]
    sub dl, 30H
    jb EXIT_ERR
    cmp dl, 0AH
    jae EXIT_ERR
    push dx
    mov dx, 10
    mul dx
    cmp dx, 0
    ja EXIT_ERR
    pop dx
    add ax, dx
    inc bx
    loop CVT_L1
CVT_FINAL:
    ret
FORMAT_HEX:

```

```

mov cx, 4
lea di, STR_OUT
push ax
push dx
xor dx, dx
FH_L1:
push cx
mov cl, 4
rol ax, cl
mov bx, ax
and bx, 000FH
add dx, bx
jz FH_N1
lea si, LUT
add si, bx
xor dx, dx
inc dx
movsb
FH_N1:
pop cx
loop FH_L1

add dx, 0
jnz FH_N2
mov [di], byte ptr '0'
inc di
FH_N2:
mov [di], byte ptr '$'

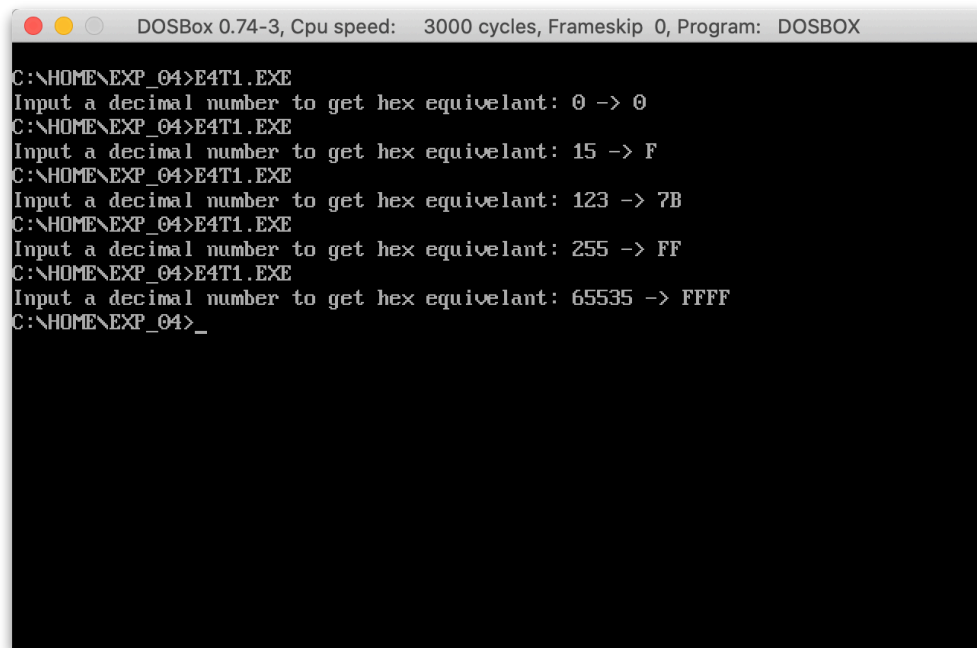
; mov cx, 3
; mov al, byte ptr '0'
; mov di, dx
; repz scasb
; mov dx, di

pop dx
pop ax
ret
EXIT_ERR:
mov ah, 09H
lea dx, STR_ERR_1
int 21H
lea dx, BUF
add dx, 2
int 21H
lea dx, STR_ERR_2
int 21H
jmp EXIT
CODE ENDS
END START

```

这行代码在编译器中无法正确编译

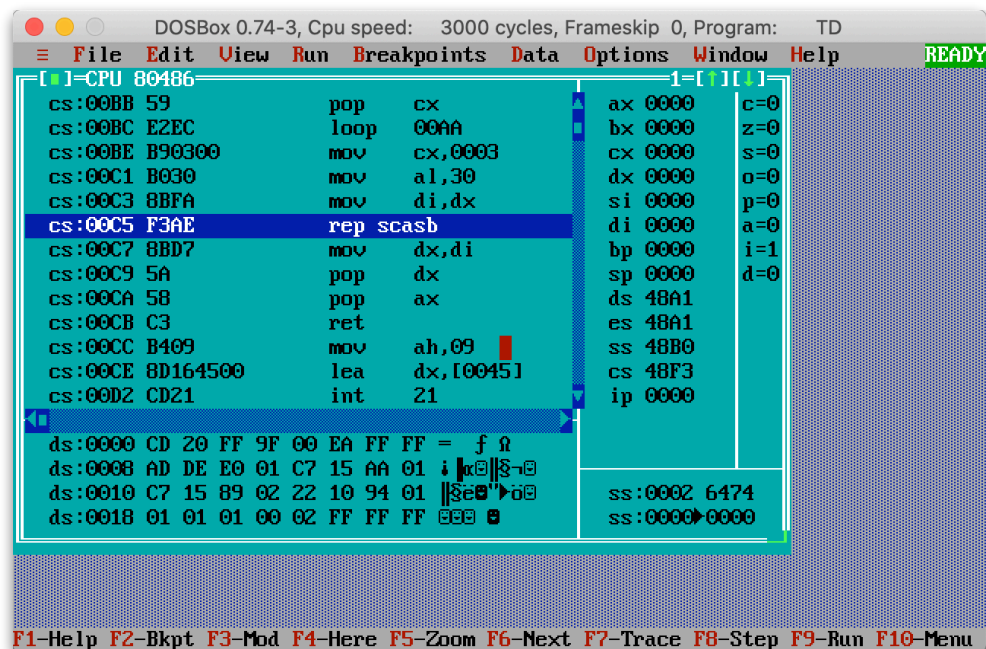
实验结果:



```
C:\HOME\EXP_04>E4T1.EXE
Input a decimal number to get hex equivalent: 0 -> 0
C:\HOME\EXP_04>E4T1.EXE
Input a decimal number to get hex equivalent: 15 -> F
C:\HOME\EXP_04>E4T1.EXE
Input a decimal number to get hex equivalent: 123 -> 7B
C:\HOME\EXP_04>E4T1.EXE
Input a decimal number to get hex equivalent: 255 -> FF
C:\HOME\EXP_04>E4T1.EXE
Input a decimal number to get hex equivalent: 65535 -> FFFF
C:\HOME\EXP_04>_
```

FIG 1.1

以上代码在不同输入值的运行结果（均正确，且不显示多余的0）



```
File Edit View Run Breakpoints Data Options Window Help
[.] CPU 80486
cs:00BB 59      pop     cx
cs:00BC E2EC    loop    00AA
cs:00BE B90300  mov     cx,0003
cs:00C1 B030    mov     al,30
cs:00C3 8BFA    mov     di,dx
cs:00C5 F3AE    rep     scasb
cs:00C7 8BD7    mov     dx,di
cs:00C9 5A      pop     dx
cs:00CA 58      pop     ax
cs:00CB C3      ret
cs:00CC B409    mov     ah,09
cs:00CE 8D164500 lea     dx,[0045]
cs:00D2 CD21    int     21

ax 0000  c=0
bx 0000  z=0
cx 0000  s=0
dx 0000  o=0
si 0000  p=0
di 0000  a=0
bp 0000  i=1
sp 0000  d=0
ds 48A1
es 48A1
ss 48B0
cs 48F3
ip 0000

ds:0000 CD 20 FF 9F 00 EA FF FF = f 0
ds:0008 AD DE E0 01 C7 15 AA 01 i 0 0 0 0 0 0 0 0
ds:0010 C7 15 89 02 22 10 94 01 || 0 0 0 0 0 0 0 0
ds:0018 01 01 01 00 02 FF FF FF 0 0 0 0 0 0 0 0

ss:0002 6474
ss:0000 0000

F1-Help F2-Bkpt F3-Mod F4-Here F5-Zoom F6-Next F7-Trace F8-Step F9-Run F10-Menu
```

FIG 1.2

“实验代码”中红色标注的代码在编译器中无法正常编译，REPZ 被编译为 REP

我因此采用了另外一种消除多余 '0' 的方法（代码中蓝色部分），并屏蔽了编译出错的代码