

AIDS MICROPROCESSOR LAB S21 BATCH (2023-24)

Experiment 1 Title: Assembly language programming for MOV instruction in various addressing modes using software tool TASM 1.4

Name of student: Meet Raut Class Roll Number: 2201084

Date of Performance: 11/03/2024

Batch: S2-1 Timing: 3:00-5:00 Date of Submission: 05/02/2024

Assembly language code

```
data segment                      # Data segment

n1 db 15h                        # variable n1 define 8 bit number
n2 dw 1367h                      # variable n2 define 16 bit number
n3 dw 0                          #Initialize n3 to 0
n4 dw 0                          #unused variable

arr1 db 00h,12h,23h,34h,
45h,56h,67h,78h,89h,92h                #Array arr1

arr2 db 20h,21h,22h,23h,
24h,25h,26h,27h,28h,29h                #Array arr2

arr3 db 10 DUP(0)                      #Array arr3 with 10 elements initialized to 0

data ends                        # Assemble directives

code segment

assume cs:code, ds:data                # Assemble directives defining cs and ds
start:                                # start of code segment

mov ax,data                        # moving data to ax register- initialization process
mov ds,ax                          # moving ax to dx register -- initialization process

;IMMEDIATE ADDRESING MODE
mov al,34h                          # Load immediate value 34h to al register
mov cx,1257h                        # Load immediate value 1257h to cx register
```

;REGISTER ADDRESSING MODE

mov ah,al *# Copy the value from al to ah register*
mov dx,cx *# Copy the value from cx to dx register*

;DIRECT ADDRESSING MODE

mov al,n1 *#Load the value of variable n1 into al register*
mov bx,n2 *#Load the value of variable n2 into bx register*
mov n3,al *#Copy the value of al register to variable n3*

mov bx,OFFSET arr1 *#Load the offset address arr1 into bx register*
mov si,OFFSET arr2 *#Load the offset address arr2 into si register*
mov di,OFFSET arr3 *#Load the offset address arr3 into di register*

;INDIRECT ADDRESSING MODE

mov cl,[bx] *#Load the value at the address stored in bx into cl register*
mov ch,[si] *#Load the value at the address stored in si into ch register*
mov [di],ch *#Store the value of ch in the address stored in di*

;BASE ADDRESSING MODE

mov dl,3[bx] *#Load the value at the address (bx + 3) into dl register*

;INDEXED ADDRESSING MODE

mov 7[di],dl *#Store the value of dl at the address (di + 7)*
mov ah,4ch *#Set ah register for exit code*

int 21h *# breakpoint interrupt*

code ends *# Assembler directives to end code*
end start *# Assembler directives to end start*

Result:

```

CPU 80486
cs:0000 BBAD48    mov     ax,4BAD
cs:0003 8ED8      mov     ds,ax
cs:0005 B034      mov     al,34
cs:0007 B97512    mov     cx,1275
cs:000A 8AE0      mov     ah,al
               mov     dx,cx
               mov     al,[0000]
               mov     bx,[0001]
               mov     [0003],al
               mov     bx,0006
               mov     si,0010
               mov     di,001A
               mov     cl,[bx]
ax 3415  c=0
bx 0006  z=0
cx 1275  s=0
dx 1275  o=0
si 0000  p=0
di 0000  a=0
bp 0000  i=1
sp 0000  d=0
ds 4BAD
es 489D
ss 48AC
cs 48B0
ip 001B

F 9F 00 EA FF FF = f 0
0 01 C5 15 AA 01  i 00+8-0
9 02 20 10 92 01
1 00 02 FF FF FF

Dump
es:0000 CD 20 FF 9F 00 EA FF FF = f 0
es:0008 AD DE E0 01 C5 15 AA 01  i 00+8-0
es:0010 C5 15 89 02 20 10 92 01  +Se0 0A0
es:0018 01 03 01 00 02 FF FF FF 000 0

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

```

```

CPU 80486
cs:0021 8A0F      mov     cl,[bx]
cs:0023 8A2C      mov     ch,[si]
cs:0025 8B2D      mov     [di],ch
cs:0027 8A5703    mov     dl,[bx+03]
cs:002A 8B5507    mov     [di+07],dl
               mov     ah,4C
               int     21
               add     [bx+si],al
               add     [bx+si],al
               add     [bx+si],al
               add     [bx+si],al
               add     [bx+si],al
               add     [bx+si],al
ax 3415  c=0
bx 0006  z=0
cx 1275  s=0
dx 1275  o=0
si 0010  p=0
di 001A  a=0
bp 0000  i=1
sp 0000  d=0
ds 4BAD
es 489D
ss 48AC
cs 48B0
ip 0021

F 9F 00 EA FF FF = f 0
0 01 C5 15 AA 01  i 00+8-0
9 02 20 10 92 01
1 00 02 FF FF FF

Dump
es:0000 CD 20 FF 9F 00 EA FF FF = f 0
es:0008 AD DE E0 01 C5 15 AA 01  i 00+8-0
es:0010 C5 15 89 02 20 10 92 01  +Se0 0A0
es:0018 01 03 01 00 02 FF FF FF 000 0

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

```


(i) IMMEDIATE ADDRESSING MODE:

File Edit View Run Breakpoints Data Options Window Help READY

CPU 80486

Address	Instruction	Register	Value	Flag
cs:0000	mov ax, 48AD	ax	4834	c=0
cs:0003	mov ds, ax	bx	0000	z=0
cs:0005	mov al, 34	cx	1275	s=0
cs:0007	mov cx, 1275	dx	0000	o=0
cs:000A	mov ah, al	si	0000	p=0
	mov dx, cx	di	0000	a=0
	mov al, [0000]	bp	0000	i=1
	mov bx, [0001]	sp	0000	d=0
	mov [0003], al	ds	48AD	
	mov bx, 0006	es	489D	
	mov si, 0010	ss	48AC	
	mov di, 001A	cs	48B0	
	mov cl, [bx]	ip	000A	

Regs=3=

ax 4834 c=0
 bx 0000 z=0
 cx 1275 s=0
 dx 0000 o=0
 si 0000 p=0
 di 0000 a=0
 bp 0000 i=1
 sp 0000 d=0
 ds 48AD
 es 489D
 ss 48AC
 cs 48B0
 ip 000A

Dump

es:0000 CD 20 FF 9F 00 EA FF FF = f R
 es:0008 AD DE E0 01 C5 15 AA 01 = f R
 es:0010 C5 15 89 02 20 10 92 01 = f R
 es:0018 01 03 01 00 02 FF FF FF = f R

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

IMMEDIATE AM

(ii) REGISTER ADDRESSING MODE:

File Edit View Run Breakpoints Data Options Window Help READY

CPU 80486

Address	Instruction	Register	Value	Flag
cs:0000	mov ax, 48AD	ax	3434	c=0
cs:0003	mov ds, ax	bx	0000	z=0
cs:0005	mov al, 34	cx	1275	s=0
cs:0007	mov cx, 1275	dx	1275	o=0
cs:000A	mov ah, al	si	0000	p=0
	mov dx, cx	di	0000	a=0
	mov al, [0000]	bp	0000	i=1
	mov bx, [0001]	sp	0000	d=0
	mov [0003], al	ds	48AD	
	mov bx, 0006	es	489D	
	mov si, 0010	ss	48AC	
	mov di, 001A	cs	48B0	
	mov cl, [bx]	ip	000E	

Regs=3=

ax 3434 c=0
 bx 0000 z=0
 cx 1275 s=0
 dx 1275 o=0
 si 0000 p=0
 di 0000 a=0
 bp 0000 i=1
 sp 0000 d=0
 ds 48AD
 es 489D
 ss 48AC
 cs 48B0
 ip 000E

Dump

es:0000 CD 20 FF 9F 00 EA FF FF = f R
 es:0008 AD DE E0 01 C5 15 AA 01 = f R
 es:0010 C5 15 89 02 20 10 92 01 = f R
 es:0018 01 03 01 00 02 FF FF FF = f R

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

REGISTER AM

(iii) DIRECT ADDRESSING MODE:

```
File Edit View Run Breakpoints Data Options Window Help READY
CPU 80486
cs:0000 B8AD48 mov ax,48AD ax 3415 c=0
cs:0003 8ED8 mov ds,ax bx 1367 z=0
cs:0005 B034 mov al,34 cx 1275 s=0
cs:0007 B97512 mov cx,1275 dx 1275 o=0
cs:000A 8AE0 mov ah,al si 0000 p=0
mov dx,cx di 0000 a=0
mov al,[0000] bp 0000 i=1
mov bx,[0001] sp 0000 d=0
mov [0003],al ds 48AD
mov bx,0006 es 489D
mov si,0010 ss 48AC
mov di,001A cs 48B0
mov cl,[bx] ip 0015

F 9F 00 EA FF FF = f 0
0 01 C5 15 AA 01 1 00 02 FF FF FF

Dump
es:0000 CD 20 FF 9F 00 EA FF FF = f 0
es:0008 AD DE E0 01 C5 15 AA 01 1 00 02 FF FF FF
es:0010 C5 15 89 02 20 10 92 01 1 00 02 FF FF FF
es:0018 01 03 01 00 02 FF FF FF 1 00 02 FF FF FF

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

CONCLUSION: LO 1, LO 3 mapped.

-----*****-----