Lab: 08

Department of Computer Science

Iqra University Islamabad

Computer Organization and Assembly Language

Maqsood Ahmed

ID: 38186

4.1 Data Transfer Instructions

MOV and MOVZX Instructions

Assembly Code: `moves.asm`

```
TITLE Data Transfer Examples       (File: moves.asm)
.686
.MODEL flat, stdcall
.STACK
INCLUDE Irvine32.inc

.data
var1  WORD 1000h
var2  WORD 2000h

.code
main PROC
    ; Demonstrating MOV and MOVZX
    mov    ax, 0A69Bh
    movzx  bx, al
    movzx  ecx, ah
    movzx  edx, ax

    ; Demonstrating MOVSX
    movsx  bx, al
    movsx  ecx, ah
    movsx  edx, ax

    ; Demonstrating XCHG
    xchg ax, var1
    xchg ax, var2
    xchg ax, var1

    exit
main ENDP
END main
```

Execution Steps and Register Values:

1. **MOV and MOVZX Instructions:**

```
mov    ax, 0A69Bh
```

- **AX = 0A69Bh**
- **AL = 9Bh**
- **AH = 0Ah**

```
movzx  bx, al
```

- `BX` = `009Bh` **(Zero-extension of `AL`)**

```
movzx  ecx, ah
```

- `ECX` = `0000000Ah` **(Zero-extension of `AH`)**

```
movzx  edx, ax
```

- `EDX` = `0000A69Bh` **(Zero-extension of `AX`)**

Predicted Values:

Instruction	Register	Value (hex)
`mov ax, 0A69Bh`	`AX`	`0A69Bh`
	`AL`	`9Bh`
	`AH`	`0Ah`
`movzx bx, al`	`BX`	`009Bh`
`movzx ecx, ah`	`ECX`	`0000000Ah`
`movzx edx, ax`	`EDX`	`0000A69Bh`

2. MOVSX Instructions:

```
movsx  bx, al
```

- `BX` = `FF9Bh` **(Sign-extension of `AL`)**

```
movsx  ecx, ah
```

- `ECX` = `FFFFFFFAh` **(Sign-extension of `AH`)**

```
movsx  edx, ax
```

- `EDX` = `FFFFA69Bh` **(Sign-extension of `AX`)**

Predicted Values:

Instruction	Register	Value (hex)
`movsx bx, al`	BX	FF9Bh
`movsx ecx, ah`	ECX	FFFFFFFAh
`movsx edx, ax`	EDX	FFFFA69Bh

3. **XCHG Instructions:**

`xchg ax, var1`

- AX = 1000h
- var1 = 0A69Bh

`xchg ax, var2`

- AX = 2000h
- var2 = 1000h

`xchg ax, var1`

- AX = 0A69Bh
- var1 = 2000h

Predicted Values:

Instruction	Register/Memory	Value (hex)
xchg ax, var1	AX	1000h
	var1	0A69Bh
xchg ax, var2	AX	2000h
	var2	1000h
xchg ax, var1	AX	0A69Bh
	var1	2000h

4.2 Addition and Subtraction

Assembly Code: `SimpleArith.asm`

```
assembly
TITLE  Simple Arithmetic         (SimpleArith.asm)
.686
```

```
.MODEL flat, stdcall
.STACK
INCLUDE Irvine32.inc

.data ; No data

.code
main PROC
    ; ADD
    mov eax, 91ab0748h
    mov ebx, 3f54f8f2h
    add eax, ebx

    ; SUB
    mov eax, 91ab0748h
    sub eax, ebx

    ; NEG
    mov eax, 91ab0748h
    neg eax

    ; INC
    clc  ; clear carry flag to show that it is not affected
    mov eax, 7fffffffh
    inc eax

    ; DEC
    mov eax, 0
    dec eax

    exit
main ENDP
END main
```

Execution Steps and Predicted Values:

1. **ADD Instruction:**

```
assembly
mov eax, 91ab0748h
mov ebx, 3f54f8f2h
add eax, ebx
```

- **EAX = `91ab0748h + 3f54f8f2h` = `D1000000h`**
- **Flags: `CF=1`, `OF=0`, `SF=1`, `ZF=0`, `PF=1`**

2. **SUB Instruction:**

```
mov eax, 91ab0748h
```

```
sub eax, ebx
```

- `EAX` = `91ab0748h - 3f54f8f2h` = `523A0E56h`
- **Flags:** `CF=0`, `OF=0`, `SF=0`, `ZF=0`, `PF=0`

3. **NEG Instruction:**

```
mov eax, 91ab0748h
neg eax
```

- `EAX` = `-91ab0748h` = `6E54F8B8h`
- **Flags:** `CF=1`, `OF=0`, `SF=1`, `ZF=0`, `PF=0`

4. **INC Instruction:**

```
mov eax, 7fffffffh
inc eax
```

- `EAX` = `7FFFFFFFh + 1` = `80000000h`
- **Flags:** `CF=0`, `OF=1`, `SF=1`, `ZF=0`, `PF=0`

5. **DEC Instruction:**

```
assembly
mov eax, 0
dec eax
```

- `EAX` = `0 - 1` = `FFFFFFFFh`
- **Flags:** `CF=0`, `OF=0`, `SF=1`, `ZF=0`, `PF=0`

Predicted Values:

Instruction Register Value (hex) CF OF SF ZF PF Top of Form

ADD	EAX	D1000000h	1	0	1	0	1
SUB	EAX	523A0E56h	0	0	0	0	0
NEG	EAX	6E54F8B8h	1	0	1	0	0
INC	EAX	80000000h	0	1	1	0	0
DEC	EAX	FFFFFFFFh	0	0	1	0	0