CSE 331L: Microprocessor Interfacing & Embedded System Lab

Faculty: DR. DIHAN MD. NURUDDIN HASAN

Instructor: Moin Shahriyar

EEE 332/ CSE 331

Lab 3

Topic: Loops, Jump, Interrupt (I/O)

Topics to be covered in class today:

- Conditional Jumps/Unconditional Jumps
- Procedures
- Instructions:
    - CMP = Compare
    - AND/OR = Logic AND/OR operation
    - JZ = Jump if Zero
    - JNZ = Jump if not Zero
    - JMP =(Unconditional) Jump
    - INT = Interrupt

Instruction	**Operands**	**Description**
CMP	REG, memory memory, REG REG, REG memory, immediate REG, immediate	Compare. Algorithm: operand1 - operand2 Result is not stored anywhere, flags are set (OF, SF, ZF, AF, PF, CF) according to result. Example: MOV AL, 5 MOV BL, 5 CMP AL, BL; (AL = 5, ZF = 1 so equal!) RET

JZ	Label	Short Jump if Zero (equal). Set by CMP, SUB, ADD, TEST, AND, OR, XOR instructions. Algorithm: if ZF = 1 then jump (ZF=Zero Flag. So, ZF=1 means it is 0) Example: include 'emu8086.inc' ORG 100h MOV AL, 5 CMP AL, 5 JZ label1 PRINT 'AL is not equal to 5.' JMP exit label1: PRINT 'AL is equal to 5.' exit: RET

JNZ	Label	Short Jump if NOT Zero (equal). Set by CMP, SUB, ADD, TEST, AND, OR, XOR instructions.
		
		Algorithm:
		
		if ZF = 0 then jump (ZF=Zero Flag. So, ZF=0 means it is 1[NOT ZERO])
		
		Example:
		
		   include 'emu8086.inc'
		
		   ORG 100h
		   MOV AL, 5
		   CMP AL, 5
		   JNZ label1
		   PRINT 'AL is equal to 5.'
		   JMP exit
		label1:
		   PRINT 'AL is not equal to 5.'
		exit:
		   RET

JMP	Label	Unconditional Jump. Transfers control to another part of the program. 4-byte address may be entered in this form: 1234h:5678h, first value is a segment second value is an offset. Algorithm: always jump Example: include 'emu8086.inc' ORG 100h MOV AL, 5 JMP label1 ; jump over 2 lines! PRINT 'Not Jumped!' MOV AL, 0 label1: PRINT 'Got Here!' RET
INT	Label	Interrupt, used to take input or to show output. Algorithm: Halt the program to fulfill the interrupt depending on “ah” register value. Example: org 100h mov ah,1 int 21h ret

Single Input	ah=1 int 21h (al=input)
Single Output	ah=2 int 21h (print dl as ascii)
Single Message/String Print:	ah=9 dx->offset “string name” int 21h

Task 1

Concept of JUMP:

Copy, compile and run the following code:

```
org 100h

jmp adder


printer:
mov ah,2
mov dl,al
add dl,'0'
int 21h
jmp finish


adder:
mov al,2
mov bl,2
add al,bl
jmp printer

finish:

ret
```

Task 2

Concept of ARRAY:

Copy, compile and run the following code:

```
org 100h

lea si,arr
mov cx,5

search_loop:

mov al,[si]
cmp al,key
JZ found
inc si
LOOP search_loop


ret

found:
mov ah,9
mov dx,offset msg1
int 21h


ret

arr db 1,2,3,4,5
key db 9
msg1 db "Key is found$"
```