Experiment 7

7 Flow Control Instructions

Introduction:

In this experiment you will practice how to control the flow of an assembly language program using the compare instruction, the different jump instructions and the loop instructions.

Objectives:

1- Compare Instruction.
2- Jump Instructions.
3- Loop Instructions.

7.1 Compare instruction:

The compare instruction is used to compare two numbers. At most one of these numbers may reside in memory. The compare instruction subtracts its source operand from its destination operand and sets the value of the status flags according to the subtraction result. The result of the subtraction is not stored anywhere. The flags are set as indicated in **Error! Reference source not found.**1.

Instruction	Example	Meaning
CMP	CMP AX, BX	If (AX = BX) then ZF ← 1 and CF ← 0
		If (AX < BX) then ZF ← 0 and CF ← 1
		If (AX > BX) then ZF ← 0 and CF ← 0

Table 7. 1: The Compare Instruction of the 8086 Microprocessor

7.2 Jump Instructions

The jump instructions are used to transfer the flow of the program to the indicated operator. When the jump address is within the same segment, the jump is called intra-segment jump. When this address is outside the current segment, the jump is called inter-segment jump. All the jump instructions are summarized in
Table **A1.**. through to Table 7. 6. Table 7.7 lists the possible addressing modes used with the jump instructions.

7.2.1 The Unconditional Jump

The unconditional jump may be used to make infinite loops. Though the use of such instructions is not recommended in high level languages, due to the availability of program control structures, in assembly however, most of the time one must use the unconditional jump.

Instruction	Meaning (Jump If)	Condition
JMP	unconditional	None

Table 7. 2: Unconditional Jump Instruction

7.2.2 The Conditional Unsigned Jump Instructions

Used after comparisons of unsigned numbers.

Instruction		Meaning (Jump If)	Condition
JA	JNBE	above (not below or equal)	CF = 0 and ZF = 0
JAE	JNB	above or equal (not below)	CF = 0
JB	JNAE	below (not above or equal)	CF = 1
JBE	JNA	below or equal (not above)	CF = 1 or ZF = 1

Table 7. 3: Conditional Unsigned Jump Instructions

7.2.3 The Conditional Signed Jump Instructions

Instruction		Meaning (Jump If)	Condition
JG	JNLE	greater (not lower or equal)	ZF = 0 and SF = OF
JGE	JNL	greater or equal (not lower)	SF = OF
JL	JNGE	lower (not greater or equal)	(SF xor OF) = 1 i.e. SF ≠ OF
JLE	JNG	lower or equal (not greater)	(SF xor OF or ZF) = 1
JCXZ	LOOP	CX register is zero	(CF or ZF) = 0

Table 7. 4: Conditional Signed Jump Instructions

7.2.4 The Conditional Signed/Unsigned Jump Instructions

These instructions are used for comparing both signed and unsigned numbers.

Instruction		Meaning	Condition
JE	JZ	Jump if zero flag set. i.e. last result equal 0	ZF = 1
JNE	JNZ	Jump if zero flag not set. i.e. last result not equal 0	ZF = 0

Table 7. 5: Signed and Unsigned Conditional Jump Instruction

7.2.5 Flag Based Jump Instructions

These instructions test individual flag bits. Many instructions have alternatives, but are not always used in programming.

Instruction		Meaning (Jump If)	Condition
JC		Carry	CF = 1
JNC		no carry	CF = 0
JNO		no overflow	OF = 0
JO		Overflow	OF =1
JNP	**JPO**	no parity (parity odd)	PF = 0
JP	**JPE**	parity (parity even)	PF = 1
JNS		no sign	SF = 0
JS		Sign	SF = 1
JZ	**JE**	Zero	ZF = 1
JNZ		non-zero	ZF = 0

Table 7. 6: Flag Based Jump Instructions

7.3 Addressing Modes

Label Pointer	Range	Addressing Mode	Specified By	Encoded As	Directive
Short	+127/-128 bytes IP ← IP + Offset	Immediate	Word	Differentially*	SHORT
Near	Intra-segment IP ← Address	Immediate	Word	Differentially	NEAR PTR
		Register	Word	Absolute address	
		Memory	Word	Absolute address	
Far	Inter-segment IP ← Address CS ← Segment	Immediate	Double Word	Absolute address	FAR PTR
		Memory	Double Word	Absolute address	

*Differentially = Difference between current and next address.

Table 7. 7: Jump Instructions and Addressing Modes

7.4 Examples

Instruction	Example	Meaning
JMP	JMP FAR PTR [BX]	IP ← [BX], CS ←[BX+2]
JNZ	JNZ END	If (ZF=0) Then IP ← Offset of END
JE	JE FIRST	If (ZF=1) Then IP ← Offset of FIRST
JC	JC SECOND	If (CF=1) Then IP ← Offset of SECOND

Table 7. 8: Examples of Jump Instructions

7.4.1 The LOOP Instructions

The LOOP instruction is a combination of a DEC and JNZ instructions. It causes execution to branch to the address associated with the LOOP instruction. The branching occurs a number of times equal to the value stored in the CX register. All LOOP instructions are summarized in Table 7..

Instruction	Example	Meaning
LOOP	LOOP Label1	If (CX≠0) then IP ← Offset Label1
LOOPE **LOOPZ**	LOOPE Label1	If (CX≠0 and ZF = 0) then IP ← Offset Label1
LOOPNE **LOOPNZ**	LOOPNZ Label1	If (CX≠0 and ZF = 0) then IP ← Offset Label1

Table 7.9: Summary of the LOOP Instructions.

7.4.2 Some HLL Program Structures

Like the conditional and unconditional jump instructions, which can be used to simulate the IF-Then-Else structure of high level programming languages, the LOOP instructions can be used to simulate the Repeat-Until and While-Do loops. These are used as shown in the following (Table 7.10).

Structure	Repeat-Until	While-Do
Code	; Repeat until CX = 0 - MOV CX, COUNT Again: - - LOOP Again - -	; While (CX ≠ 0) Do - MOV CX, COUNT Again: JZ Next - LOOP Again Next: - -

Table 7.10: The Loop Program Structure.

7.5 Lab Work

Part I:

1. Write a program that reads signed and unsigned numbers. Make sure you test the '+' or '-' signs when you read the number from the keyboard. Convert your program to make it look like a procedure, call it **ReadSigned**
2. Write a program that displays a signed number. You have to test the sign bit to see if your number is negative or positive. Convert your program to make it look like a procedure, call it **DisplaySigned**
3. Write a program that reads a list of signed numbers, then writes them back on the screen.

Part II:

4. Write a program that reads a list of 10 signed numbers, then finds the largest/smallest number and displays it on the screen.
5. Write a program that reads a list of 10 signed numbers, then sorts them in ascending/descending order based on the user's choice.
6. Write a program that reads a list of 10 signed numbers, then replaces all negative numbers by zeros, and finally displays the list back on the screen.

7. Show all your work to the instructor, and submit it at the end of the lab session.

Appendix-A-

Type	Instruction		Meaning (Jump If)	Condition
Unconditional	JMP		unconditional	None
Conditional Signed Instructions	JA	JNBE	above (not below or equal)	CF = 0 and ZF = 0
	JAE	JNB	above or equal (not below)	CF = 0
	JB	JNAE	below (not above or equal)	CF = 1
	JBE	JNA	below or equal (not above)	CF = 1 or ZF = 1
Conditional Unsigned Instructions	JG	JNLE	greater (not lower or equal)	ZF = 0 and SF = OF
	JGE	JNL	greater or equal (not lower)	SF = OF
	JL	JNGE	lower (not greater or equal)	(SF xor OF) = 1 i.e. SF ≠ OF
	JLE	JNG	lower or equal (not greater)	(SF xor OF or ZF) = 1
	JCXZ	LOOP	CX register is zero	(CF or ZF) = 0
Signed and Unsigned	JE	JZ	equal (zero)	ZF = 1
	JNE	JNZ	not equal (not zero)	ZF = 0
Conditional Flags Instructions	JC		Carry	CF = 1
	JNC		no carry	CF = 0
	JNO		no overflow	OF = 0
	JO		overflow	OF =1
	JNP	JPO	no parity (parity odd)	PF = 0
	JP	JPE	parity (parity even)	PF = 1
	JNS		no sign	SF = 0
	JS		sign	SF = 1
	JZ		zero	ZF = 1
	JNZ		non-zero	ZF = 0

Table A1.1: Summary of the Jump Instructions