

EEE 410 – Microprocessors I

Fall 05/06 – Lecture Notes # 8

Outline of the Lecture

- Arithmetic and Logic Instructions and Programs
- Unsigned Addition and Subtraction

UNSIGNED ADDITION AND SUBTRACTION

- Unsigned numbers are defined as data in which all the bits are used to represent data and no bits are set aside for the positive and negative sign.

For 8-bit, data operand can be between 00H and FFH (0 to 255 decimal)

For 16-bit, data operand can be between 00H and FFFFH (0 to 65535 decimal)

➤ ADD: Addition of unsigned numbers

Format: ADD destination,source ;dest. operand = dest. operand + source operand

Ex: Show how the flag register is affected by the following addition

```
MOV AL,0F5H
ADD AL,0BH
```

Solution:

	F5	1111 0101
	+ 0B	+ 0000 1011
	100H	0000 0000

After the addition AL will contain 00 and the flags are as follows.

CF = 1 since there is a carry out from d7

SF = 0 the status of d7 of the result

PF = 1

AF = 1

ZF = 1

➤ ADC: Add with carry

Format: ADC destination,source ; dest .operand = dest. operand + source + CF

If CF=1 prior to this instruction, then after execution of this instruction, source is added to destination plus 1. If CF=0, source is added to destination plus 0. Used widely in multibyte and multiword additions.

➤ Addition of individual byte data

Ex: Write a program to calculate the total sum of 5 bytes of data. Each byte represents the daily wages of a worker. This person does not make more than \$255 (FFH) a day. The decimal data is as follows: 125, 235, 197, 91, and 48.

- Note that these numbers are converted to hex by the assembler as follows: 125=7DH, 235=EBH, 197=C5H, 91=5BH, 48=30H.
- In this program following lines of the program can be replaced:

Replace these lines

```
BACK:    ADD  AL,[SI]
         ADC  AH,00
         INC  SI
```

with these lines

```
BACK:    ADD AL,[SI]
         JNC OVER      ;add 1 to AH if CF=1
         INC AH
OVER:    INC SI
```

```
Assemble link and debug:  -U CS:0 1D  (Assume DS=199C)
                          -D 199C:0 F
                          -G
                          -D 199C:0 F
```

```
; This program adds 5 unsigned byte numbers.
.MODEL SMALL
.STACK 64
.DATA
COUNT    EQU    05
DATA      DB    125,235,197,91,48
          ORG    0008H
SUM       DW    ?
.CODE
MAIN:     MOV AX, @DATA
          MOV DS,AX
          MOV CX,COUNT      ;CX is the loop counter
          MOV SI,OFFSET DATA ;SI is the data pointer
          MOV AX,00         ;AX will hold the sum
BACK:     ADD AL,[SI]       ;add the next byte to AL
          ADC AH,00         ;add 1 to AH if CF =1
          INC SI            ;increment data pointer
          DEC CX            ;decrement loop counter
          JNZ BACK         ;if not finished, go add next byte
          MOV SUM,AX       ;store sum
          MOV AH,4CH
          INT 21H          ;go back to DOS
          END MAIN
```

➤ Addition of individual word data

Ex: Write a program to calculate the total sum of 5 words of data. Each data value represents the yearly wages of a worker. This person does not make more than \$65535 (FFFFH) a year. The decimal data is as follows: 27345, 28521, 29533, 30105, and 32375.

```
Assemble link and debug :  -U CS:0 24
                          -D 199C:0 1F  (Assume DS=199C)
                          -G
                          -D 199C:0 1F
```

; This program is an example for Multiword addition

```
.MODEL SMALL
.STACK 64
.DATA
DATA1      DQ    548FB9963CE7H
           ORG    0010H
DATA2      DQ    3FCD4FA23B8DH
           ORG    00020H
DATA3      DQ    (?)
.CODE
MAIN:      MOV AX, @DATA
           MOV DS,AX
           CLC                                ;clear carry before the first addition
           MOV SI,OFFSET DATA1             ;SI is the data pointer for operand1
           MOV DI,OFFSET DATA2             ;DI is the data pointer for operand2
           MOV BX,OFFSET DATA3             ;BX is the data pointer for the sum
           MOV CX,04                        ;CX is the loop counter
BACK:      MOV AX,[SI]                       ;move the first operand to AX
           ADC AX,[DI]                       ;add the second operand to AX
           MOV [BX],AX                      ;store the sum
           INC SI                            ;point to next word of operand1
           INC SI
           INC DI                            ;point to next word of operand2
           INC DI
           INC BX                            ;point to next word of sum
           INC BX
           LOOP BACK                        ;if not finished, continue adding
           MOV AH,4CH
           INT 21H                          ;go back to DOS
           END MAIN
```

➤ Addition of multiword numbers

Ex: Write a program that adds the following two multiword numbers and saves the result:
DATA1 = 548FB9963CE7H and DATA2 = 3FCD4FA23B8DH

Assemble link and debug :
-U CS:0 22
-D 199C:0 2F
-G
-D 199C:0 2F

Note: **LOOP BACK** ;is the equivalent of the following two instructions

 DEC CX
 JNZ BACK