**Lab Session 8**

**Objectives**

**COMPARE and ASCII & BCD Conversion of unsigned numbers**

**Theory:**

**COMPARE OF UNSIGNED NUMBERS**

**CMP** dest,source ;compare dest and source.

The **operands** themselves remain **unchanged**.

The dest operand can be in register or memory. The source operand can be in register, memory or an immediate number.

CMP instruction compares two operands and changes the flags accordingly. Although CF,AF,SF,PF,ZF and OF flags reflect the result of the comparison, only the CF and ZF are affected.

|  |  |  |
| --- | --- | --- |
| **Compare operands** | **CF** | **ZF** |
| Destination >source | 0 | 0 |
| Destination = source | 0 | 1 |
| Destination < source | 1 | 0 |

Flag settings of the CMP instruction.

Ex: DATA1 DW 235FH

…

MOV AX,CCCCH

CMP AX,DATA1 ;compare CCCC with 235F

JNC OVER ;jump if CF=0

SUB AX,AX

OVER: INC DATA1

|  |  |
| --- | --- |
| **Digit** | **BCD** |
| 0 | 0000 |
| 1 | 0001 |
| 2 | 0010 |
| 3 | 0011 |
| 4 | 0100 |
| 5 | 0101 |
| 6 | 0110 |
| 7 | 0111 |
| 8 | 1000 |
| 9 | 1001 |

**BCD(Binary Coded Decimal and ASCII (American Standard Code for Information Interchange)**

**Instructions**

* + Binary representation of 0 to 9 (used by human beings) is called BCD.
  + There are two types of BCD numbers,

(1) unpacked BCD (2) packed BCD

**Unpacked BCD:** 1 byte is used to store 4 bit BCD code. E.g. 0000 1001 is unpacked BCD for 9.

**Packed BCD:** 1 byte is used to store two 4 bit BCD codes. E.g. 0101 1001 is packed BCD for 59. More efficient in storing data.

**ASCII numbers:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **ASCII(Hex)** | **Binary** | **BCD (Unpacked)** |
| 0 | 30 | 011 0000 | 0000 0000 |
| 1 | 31 | 011 0001 | 0000 0001 |
| 2 | 32 | 011 0010 | 0000 0010 |
| 3 | 33 | 011 0011 | 0000 0011 |
| 4 | 34 | 011 0100 | 0000 0100 |
| 5 | 35 | 011 0101 | 0000 0101 |
| 6 | 36 | 011 0110 | 0000 0110 |
| 7 | 37 | 011 0111 | 0000 0111 |
| 8 | 38 | 011 1000 | 0000 1000 |
| 9 | 39 | 011 1001 | 0000 1001 |

* **ASCII to BCD Conversion**

**ASCII to Unpacked BCD Conversion**

* In order to convert ASCII to BCD the programmer must get rid of tagged “011” in the higher four bits of the ASCII.
* To do that each ASCII number is ANDed with ‘0000 1111’ (0FH).

Ex: ASC DB ‘9562481273’

ORG 0010H

UNPACK DB 10 DUP(?)

…

MOV CX,5 ;CX is the loop counter

MOV BX,OFFSET ASC ;BX points to ASCII data

MOV DI,OFFSET UNPACK ;DI points to unpacked BCD data

AGAIN: MOV AX,WORD PTR [BX] ;move next 2 ASCII numbers to AX

AND AX,0F0F ;remove ASCII 3s (011)

MOV WORD PTR [DI],AX ;store unpacked BCD

ADD DI,2 ;point to next unpacked BCD data

ADD BX,2 ;point to next ASCII data

LOOP AGAIN

**ASCII to packed BCD Conversion**

To convert ASCII to packed BCD, it is first converted to unpacked BCD (to get rid of the 3) and then combined to make packed BCD.

Key ASCII Unpacked BCD Packed BCD

4 34 00000100

7 37 00000111 01000111 or 47H

ORG 0010H

VAL\_ASC DB ‘47’ VAL\_BCD DB ?

…

;reminder: the DB will put 34 in 0010H location and 37 in 0011H.

|  |  |
| --- | --- |
| MOV AX,WORD PTR VAL\_ASC ;AH=37 AL=34  AND AX,0F0FH ;mask 3 to get unpacked BCD  XCHG AH,AL ;swap AH and AL |  |
| MOV CL,4 ;CL=04 to shift 4 times  SHL AH,CL ;shift left AH to get AH=40H OR AL,AH ;OR them to get packed BCD MOV VAL\_BCD,AL save the result |  |

**Packed BCD to ASCII Conversion**

To convert packed BCD to ASCII, it must be first converted to unpacked and then the unpacked BCD is tagged with 011 0000 (30H).

Packed BCD Unpacked BCD ASCII

29H 02 & 09 32 & 39

0010 1001 0000 0010 & 0000 1001 0011 0010 & 0011 1001

Ex:

VAL1\_BCD DB 29H VAL3\_ASC DW ?

….

MOV AL,VAL1\_BCD

MOV AH,AL ;copy AL to AH. Now AH=29 and AL=29

AND AX,F00FH ;mask 9 from AH and 2 from AL MOV CL,04 ;CL=04 for shift

SHR AH,CL ;shift right AH to get unpacked BCD OR AX,3030H combine with 30 to get ASCII

XCHG AH,AL ;swap for ASCII storage convention MOV VAL3\_ASC,AX ;store the ASCII