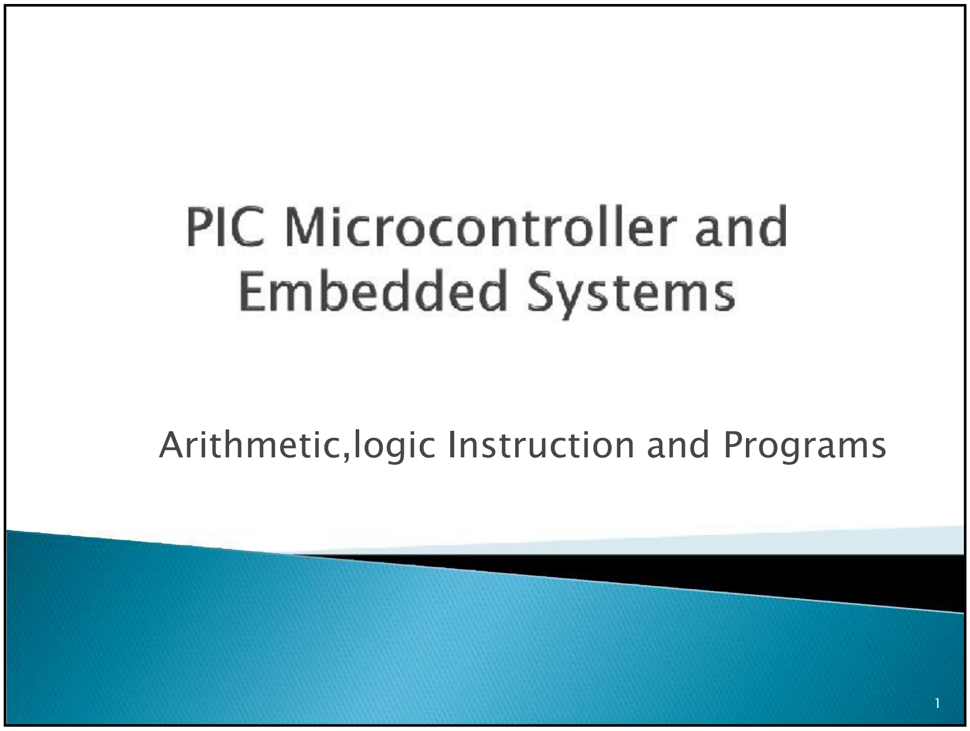
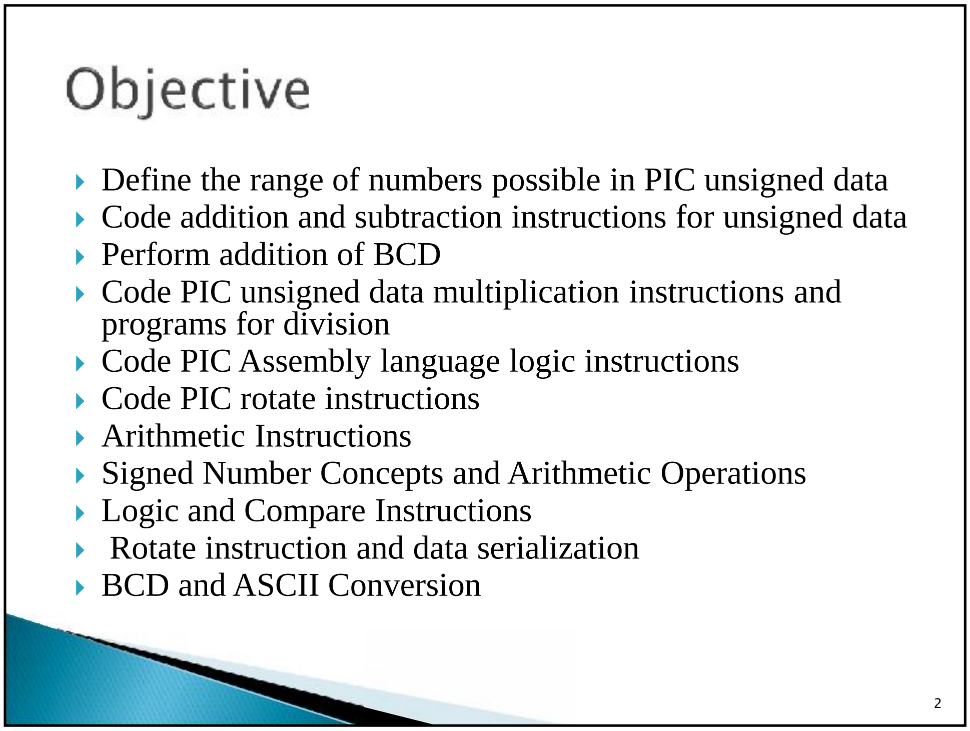


PIC Microcontroller and Embedded Systems

Arithmetic,logic Instruction and Programs



Objective

- Define the range of numbers possible in PIC unsigned data
- Code addition and subtraction instructions for unsigned data
- Perform addition of BCD
- Code PIC unsigned data multiplication instructions and programs for division
- Code PIC Assembly language logic instructions
- Code PIC rotate instructions
- Arithmetic Instructions
- Signed Number Concepts and Arithmetic Operations
- Logic and Compare Instructions
- Rotate instruction and data serialization
- BCD and ASCII Conversion

Arithmetic Instructions

- Unsigned numbers are defined as data in which all the bits are used to represent data
- No bits are set aside for neg. or pos. sign
- Addition of unsigned numbers

  ADDLW k

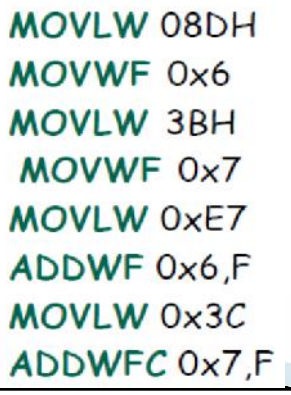
  ADDWF fileReg, d, a

- ADDWFC (adding two 16-bit numbers)

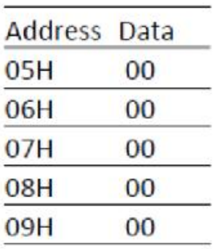
- What happens to flag register?

Example

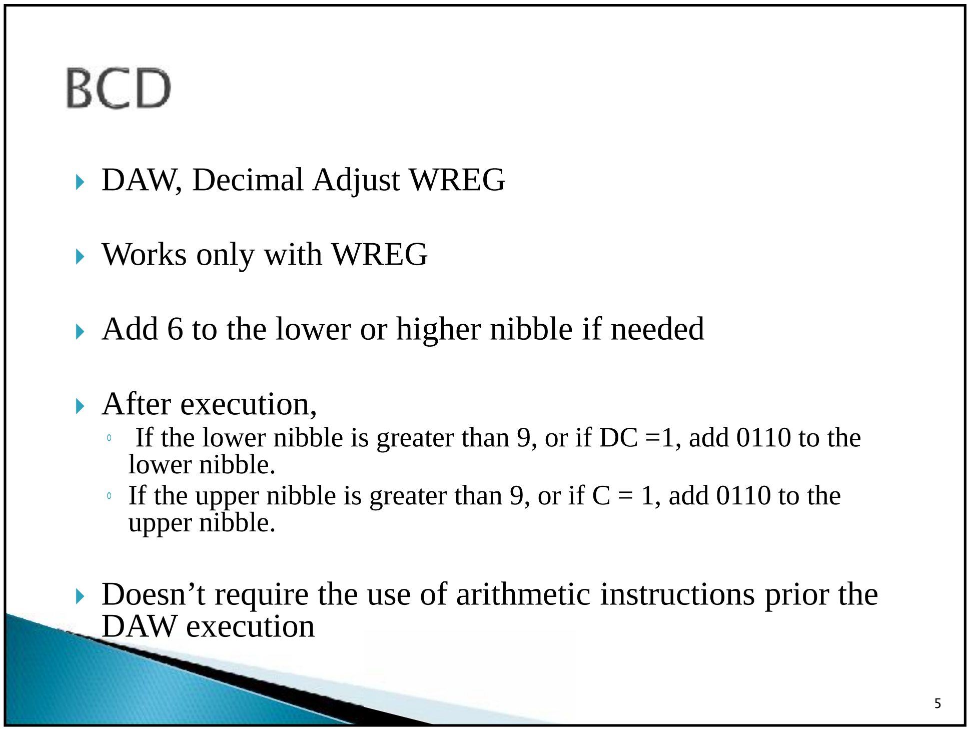
- ADD 3CE7H and 3B8DH
- Store the sum in fileReg locations 6 and 7, where location 6 should have the lower bye.



```
MOVLW 08DH
MOVWF  0x6
MOVLW  3BH
 MOVWF  0x7
MOVLW 0xE7
ADDWF 0x6,F
MOVLW 0x3C
ADDWFC 0x7,F
```

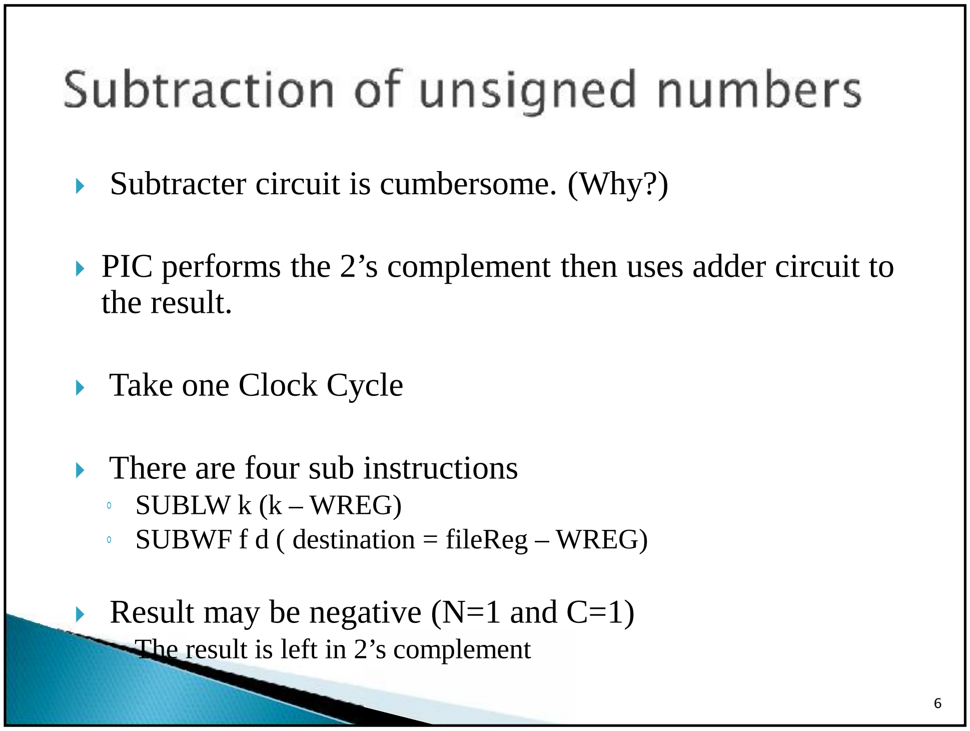


Address	Data
05H	00
06H	00
07H	00
08H	00
09H	00



BCD

- DAW, Decimal Adjust WREG
- Works only with WREG
- Add 6 to the lower or higher nibble if needed
- After execution,
  - If the lower nibble is greater than 9, or if DC =1, add 0110 to the lower nibble.
  - If the upper nibble is greater than 9, or if C = 1, add 0110 to the upper nibble.
- Doesn’t require the use of arithmetic instructions prior the DAW execution

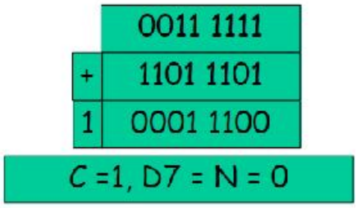


Subtraction of unsigned numbers

- Subtracter circuit is cumbersome. (Why?)
- PIC performs the 2’s complement then uses adder circuit to the result.
- Take one Clock Cycle
- There are four sub instructions
  - SUBLW k (k – WREG)
  - SUBWF f d (destination = fileReg – WREG)
- Result may be negative (N=1 and C=1)
  - The result is left in 2’s complement

Example

- MOVLW 0x23
- SUBLW 0x3F

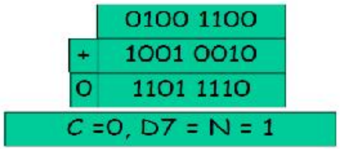


	0011 1111
+	1101 1101
1	0001 1100

C =1, D7 = N = 0

Subtract 4C – 6E

```
MYREG **EQU** 0x2
**MOVLW** 0x4C
**MOVWF**MYREG
**MOVLW**0x6E
**SUBWF** MYREG,W
**BNN** NEXT
**NEGF** WREG
NEXT
**MOVWF**MYREG
```

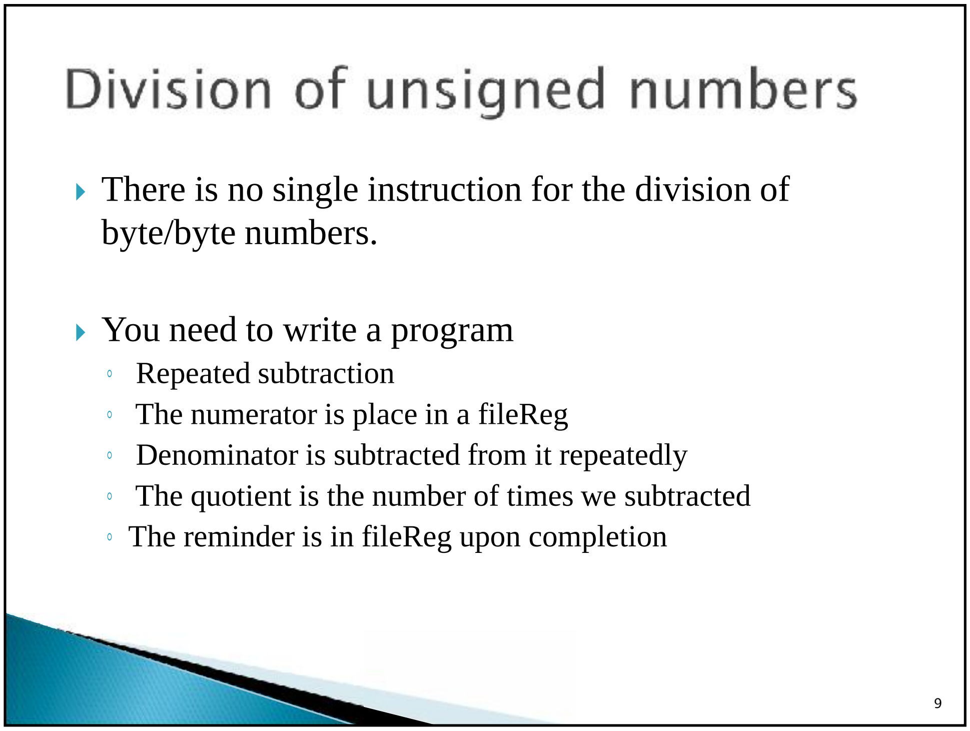


	0100 1100
+	1001 0010
0	1101 1110

C =0, D7 = N = 1

Multiplication of unsigned number

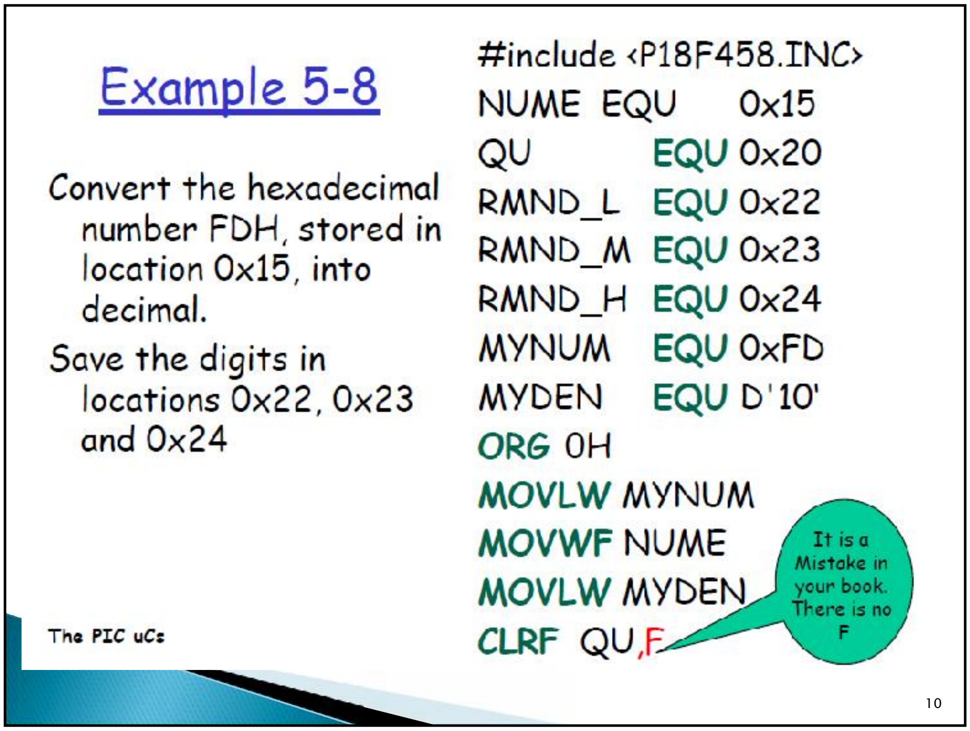
- PIC supports byte-by-byte multiplication
- One of the operand must be in WREG
- After multiplication, the result is stored in PRODH and PRODL (16 bit)
- Example
  - **MOVLW** 0x25
  - **MULLW** 0x65



Division of unsigned numbers

- There is no single instruction for the division of byte/byte numbers.

- You need to write a program
  - Repeated subtraction
  - The numerator is place in a fileReg
  - Denominator is subtracted from it repeatedly
  - The quotient is the number of times we subtracted
  - The reminder is in fileReg upon completion



Example 5-8

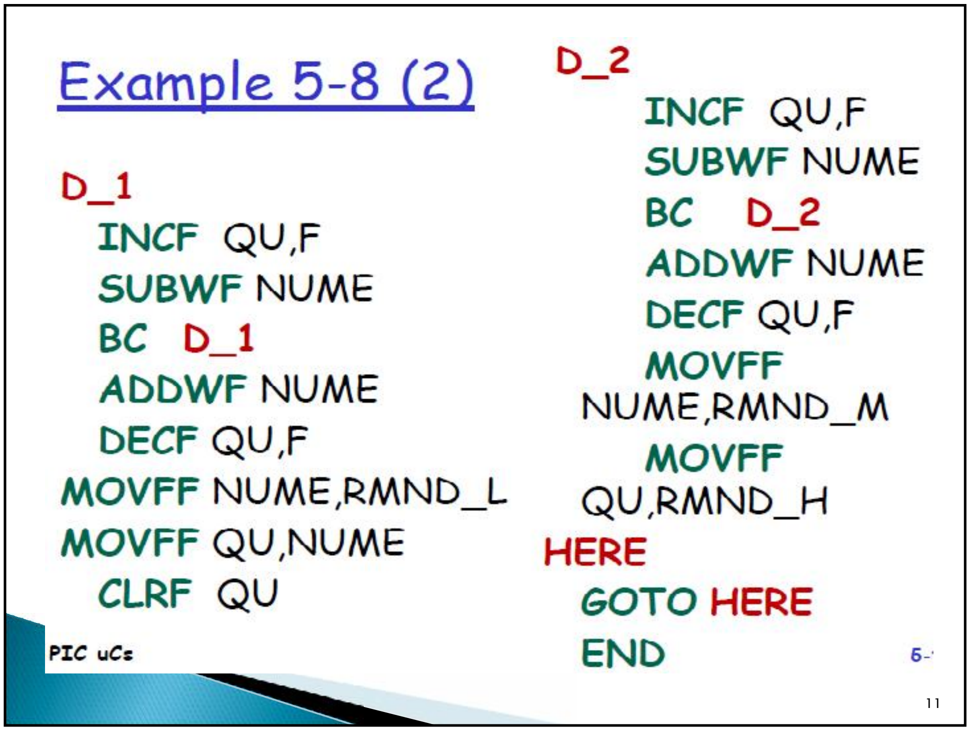
Convert the hexadecimal number FDH, stored in location 0x15, into decimal.

Save the digits in locations 0x22, 0x23 and 0x24

```
#include <P18F458.INC>
NUME  EQU      0x15
QU          EQU 0x20
RMND_L      EQU 0x22
RMND_M      EQU 0x23
RMND_H      EQU 0x24
MYNUM       EQU 0xFD
MYDEN       EQU D'10'
ORG 0H
MOVLW MYNUM
MOVWF NUME
MOVLW MYDEN
CLRF  QU,F
```

It is a Mistake in your book. There is no F

The PIC uCs



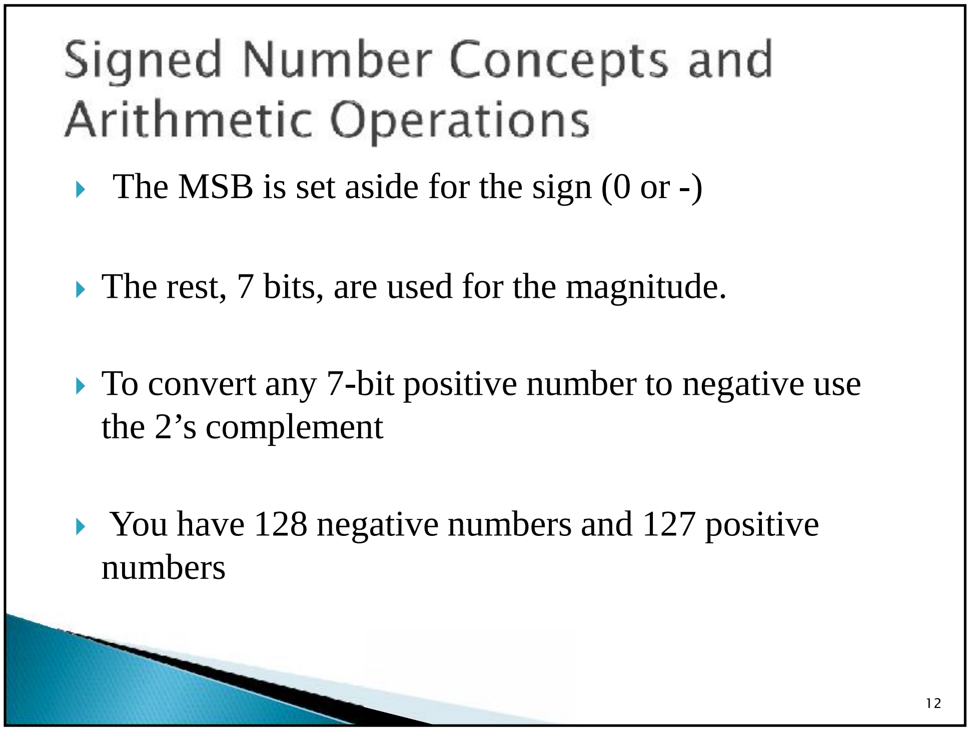
Example 5-8 (2)

```
D_1
   INCF  QU,F
   SUBWF NUME
   BC   D_1
   ADDWF NUME
   DECF QU,F
MOVFF NUME,RMND_L
MOVFF QU,NUME
   CLRF  QU
```

```
 D_2
       INCF  QU,F
       SUBWF NUME
       BC    D_2
       ADDWF NUME
       DECF QU,F
       MOVFF
    NUME,RMND_M
       MOVFF
    QU,RMND_H
HERE
    GOTO HERE
    END
```

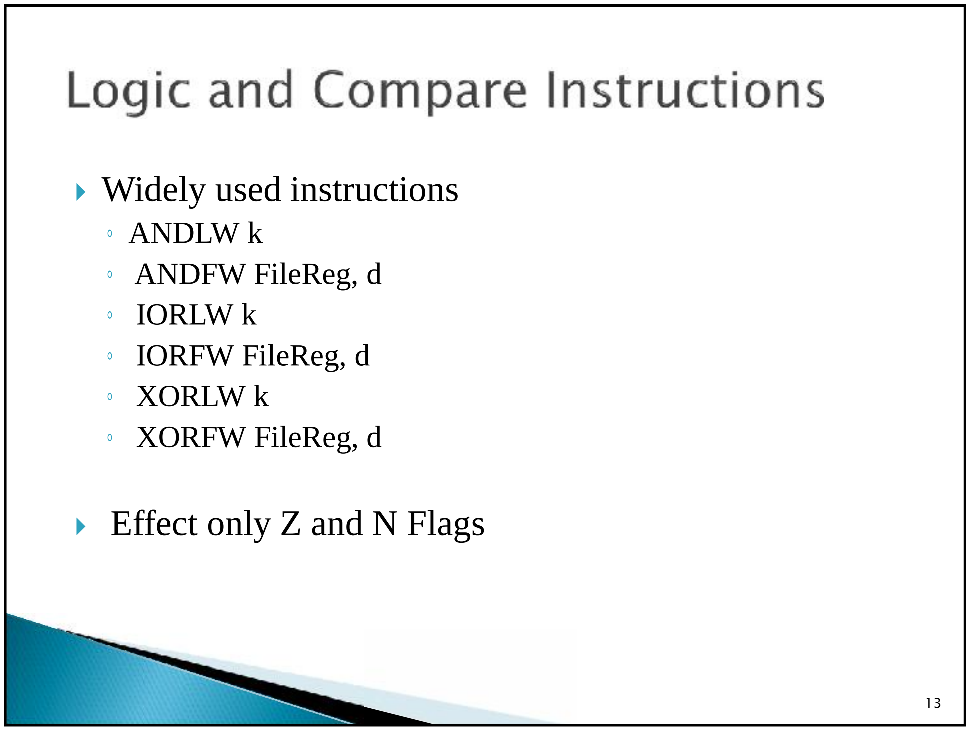
PIC uCs

5-'



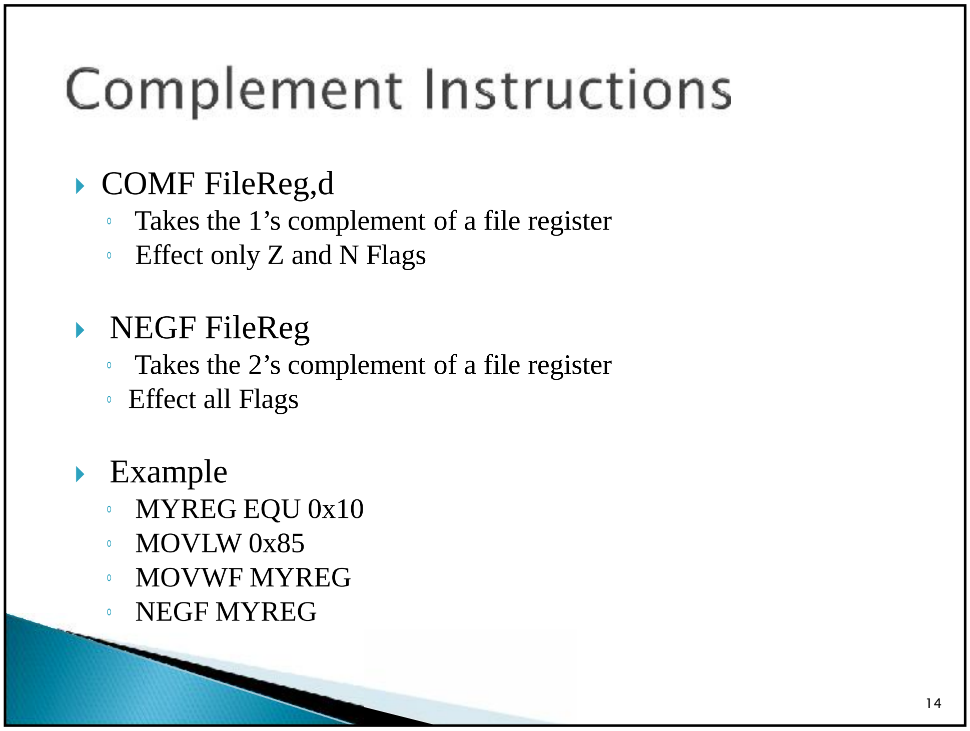
Signed Number Concepts and Arithmetic Operations

- The MSB is set aside for the sign (0 or -)
- The rest, 7 bits, are used for the magnitude.
- To convert any 7-bit positive number to negative use the 2’s complement
- You have 128 negative numbers and 127 positive numbers



Logic and Compare Instructions

- Widely used instructions
  - ANDLW k
  - ANDFW FileReg, d
  - IORLW k
  - IORFW FileReg, d
  - XORLW k
  - XORFW FileReg, d
- Effect only Z and N Flags

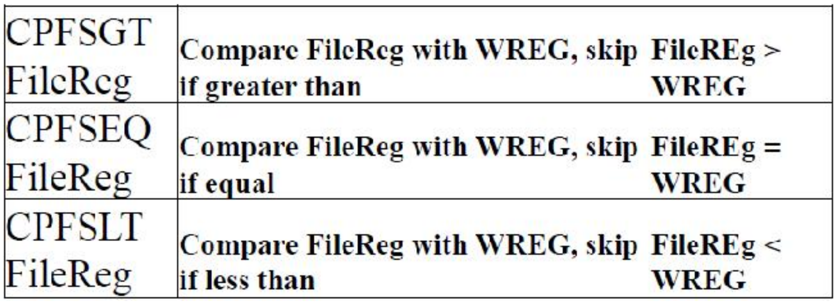


Complement Instructions

- COMF FileReg,d
  - Takes the 1’s complement of a file register
  - Effect only Z and N Flags
- NEGF FileReg
  - Takes the 2’s complement of a file register
  - Effect all Flags
- Example
  - MYREG EQU 0x10
  - MOVLW 0x85
  - MOVWF MYREG
  - NEGF MYREG

Compare Instructions

- These instructions take 1/2 cycle(s)

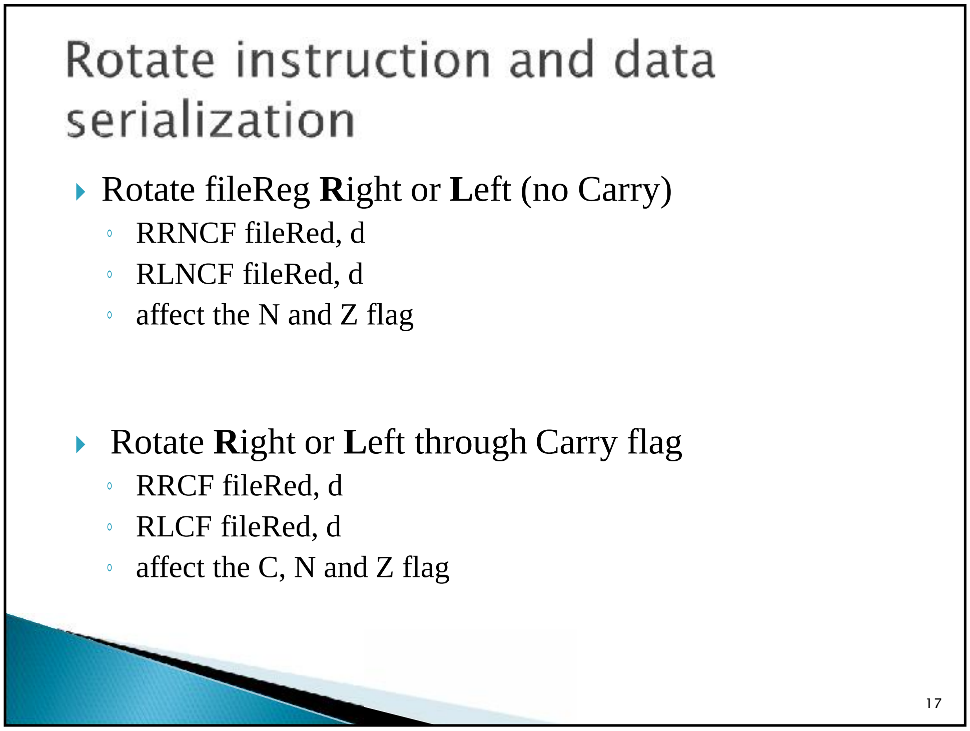


CPFSGT FilcRcg	Comparc FilcRcg with WREG, skip if greater than	FileREg > WREG
CPFSEQ FileReg	Compare FileReg with WREG, skip if equal	FileREg = WREG
CPFSLT FileReg	Compare FileReg with WREG, skip if less than	FileREg < WREG

Example

- Write code to determine if data on PORTB contains the value 99H. If so, write letter ‘Y’ to PORTC; otherwise, make PORTC=‘N’

```
        CLRF TRISC
        MOVLW A’N‘
        MOVWF PORTC
        SETF TRISB
        MOVLW 0x99
        CPFSEQ PORTB
        BRA OVER
        MOVLW A’Y‘
        MOVWF PORTC
OVER ……….
```

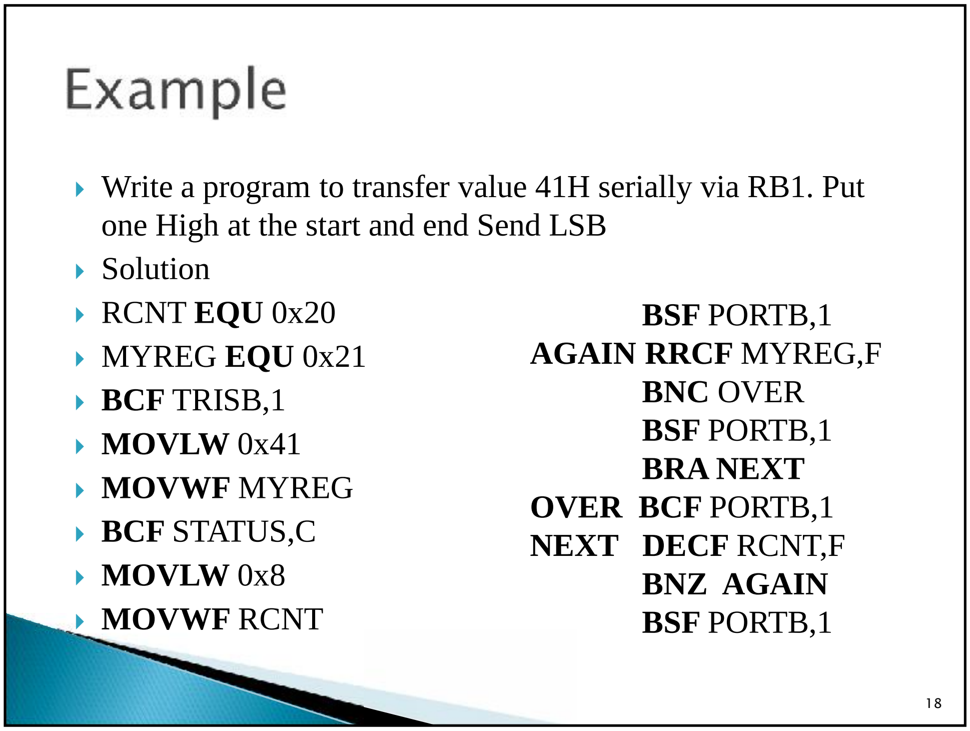


Rotate instruction and data serialization

- Rotate fileReg **R**ight or **L**eft (no Carry)
  - RRNCF fileRed, d
  - RLNCF fileRed, d
  - affect the N and Z flag

- Rotate **R**ight or **L**eft through Carry flag
  - RRCF fileRed, d
  - RLCF fileRed, d
  - affect the C, N and Z flag

17

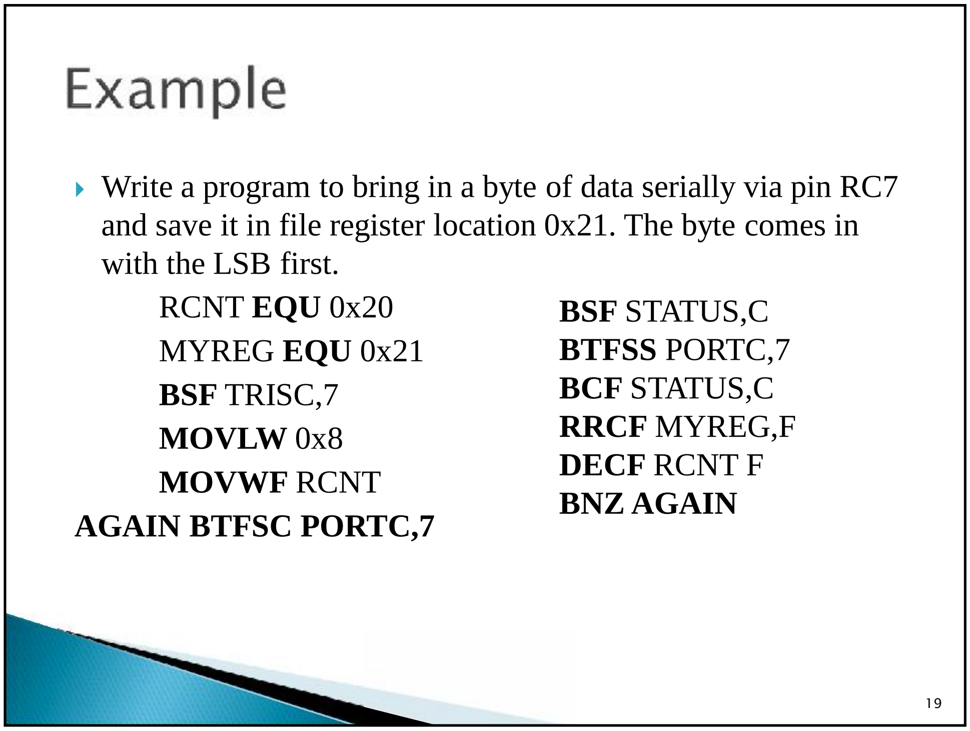


Example

- Write a program to transfer value 41H serially via RB1. Put one High at the start and end Send LSB
- Solution
- RCNT **EQU** 0x20
- MYREG **EQU** 0x21
- **BCF** TRISB,1
- **MOVLW** 0x41
- **MOVWF** MYREG
- **BCF** STATUS,C
- **MOVLW** 0x8
- **MOVWF** RCNT

```
          BSF PORTB,1
AGAIN RRCF MYREG,F
          BNC OVER
          BSF PORTB,1
          BRA NEXT
OVER  BCF PORTB,1
NEXT   DECF RCNT,F
          BNZ  AGAIN
          BSF PORTB,1
```

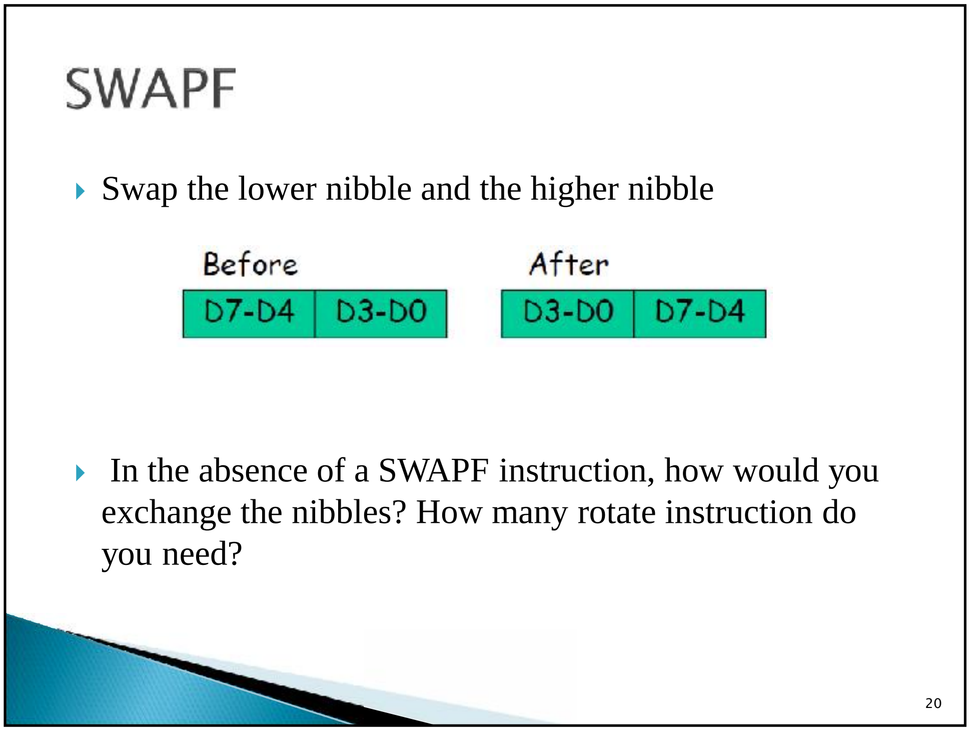
18



Example

- Write a program to bring in a byte of data serially via pin RC7 and save it in file register location 0x21. The byte comes in with the LSB first.

```
        RCNT EQU 0x20
        MYREG EQU 0x21
        BSF TRISC,7
        MOVLW 0x8
        MOVWF RCNT
AGAIN BTFSC PORTC,7
        BSF STATUS,C
        BTFSS PORTC,7
        BCF STATUS,C
        RRCF MYREG,F
        DECF RCNT F
        BNZ AGAIN
```

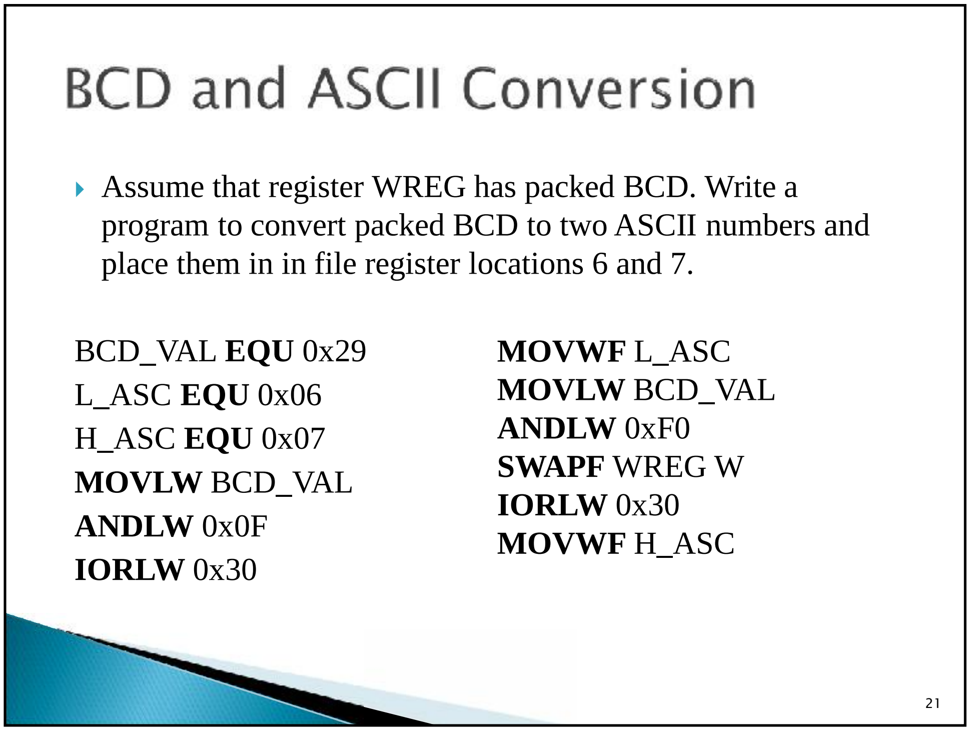


SWAPF

- Swap the lower nibble and the higher nibble

Before		After	
D7-D4	D3-D0	D3-D0	D7-D4

- In the absence of a SWAPF instruction, how would you exchange the nibbles? How many rotate instruction do you need?



BCD and ASCII Conversion

- Assume that register WREG has packed BCD. Write a program to convert packed BCD to two ASCII numbers and place them in in file register locations 6 and 7.

```
BCD_VAL EQU 0x29
L_ASC EQU 0x06
H_ASC EQU 0x07
MOVLW BCD_VAL
ANDLW 0x0F
IORLW 0x30
MOVWF L_ASC
MOVLW BCD_VAL
ANDLW 0xF0
SWAPF WREG W
IORLW 0x30
MOVWF H_ASC
```