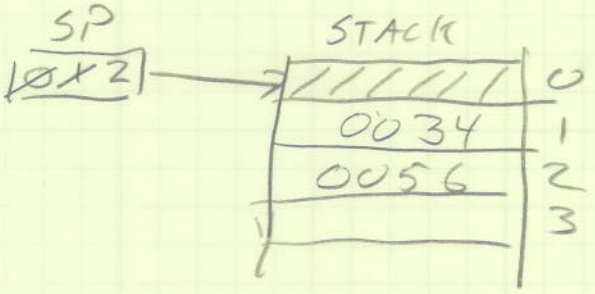
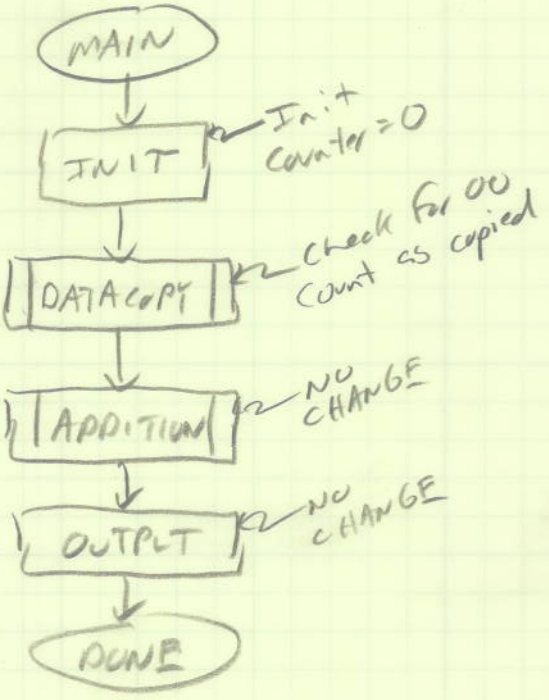
7.10

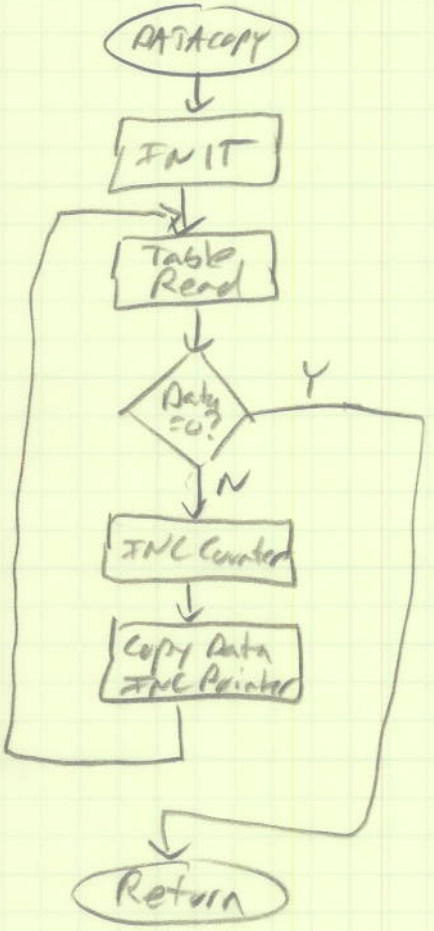
```
0032        RCALL   SUB1
0034        MOVWF   REG1

0050  SUB1  MOVF    REG2,W
0052        CALL    SUB2
0056        COMF    REG1,F
```



7.20





```
;*******************************************************************************
; This program copies unsigned data bytes from program memory into data registers
; and adds the bytes.  The 16-bit sum is displayed at PORTB and PORTC.
; IP7-4 has been modified so the number of bytes is variable, but the data string
; is terminated in the byte 00.  Modifications are shown in RED.
;*******************************************************************************

      Title "7-20 Copying and Adding Bytes Modified"
      List p=18F452, f =inhx32
      #include <p18F452.inc>  ;The header file

BUFFER      EQU   0x10        ;Beginning data register address
COUNTER     EQU   0X01        ;Counter for number of bytes

            ORG   00
            GOTO  MAIN

            ORG   0x20
MAIN: MOVLW 0x00        ;Initialize Ports B & C as an output ports
      MOVWF TRISB
      MOVWF TRISC
      LFSR  FSR0,BUFFER ;Set up FSR0 as pointer for data registers
      MOVLW 00          ;Init COUNTER
      MOVWF COUNTER
      CALL  DATACOPY    ;Call DATACOPY Subroutine
      LFSR  FSR0,BUFFER ;Set up FSR0 as pointer for data registers
      CALL  ADDITION    ;Call ADDITION Subroutine
      MOVWF PORTC       ;Display low-order byte of sum at PORTC
      MOVFF CYREG,PORTB ;Display high-order byte of sum at PORTB
      SLEEP

DATACOPY: ;*****************************************************************
        ; Function: This subroutine copies data bytes from program memory at
        ;           SOURCE to data registers at BUFFER.
        ; Input:    Pointer to BUFFER in FSR0.
        ; Output:   Number of bytes copied in COUNTER.
        ;*****************************************************************

      MOVLW UPPER SOURCE      ;Set up TBLPTR pointing to Source
      MOVWF TBLPTRU
      MOVLW HIGH SOURCE
      MOVWF TBLPTRH
      MOVLW LOW SOURCE
      MOVWF TBLPTRL
NEXT: TBLRD*+                 ;Copy byte to Table Latch and increment pointer
      MOVF  TABLAT,W    ;Copy byte from Table Latch to W
      BZ    LAST        ;If byte = 0, then done
      INCF  COUNTER,F   ;Increment counter
      MOVWF POSTINC0    ;Copy byte from W to BUFFER and increment FSR0
      BRA   NEXT        ;Go back to copy next byte
LAST: RETURN

CYREG EQU   0x02

ADDITION: ;*****************************************************************
        ; Function: This subroutine adds bytes stored in data registers BUFFER
        ; Input:    Pointer to BUFFER in FSR0 and the number of bytes to be
        ;           copied in COUNTER.
        ; Output:   The 16-bit sum in CYREG and WREG.
        ;*****************************************************************
```

```
      CLRF  CYREG       ;Clear carry register and W for sum
      MOVLW 0x00
NXT:  ADDWF POSTINC0,W  ;Add byte and increment FSR0
      BNC   SKIP        ;Check for carry: if no carry jump to SKIP
      INCF  CYREG,F           ;If there is carry, increment CYREG
SKIP: DECF  COUNTER,F   ;Decrement counter
      BNZ   NXT         ;Is addition complete?  If not go back

SOURCE:     DB    0xF6,0x67,0x7F,0xA9,0x72,0x00 ;Added 00 as last byte

      END
```

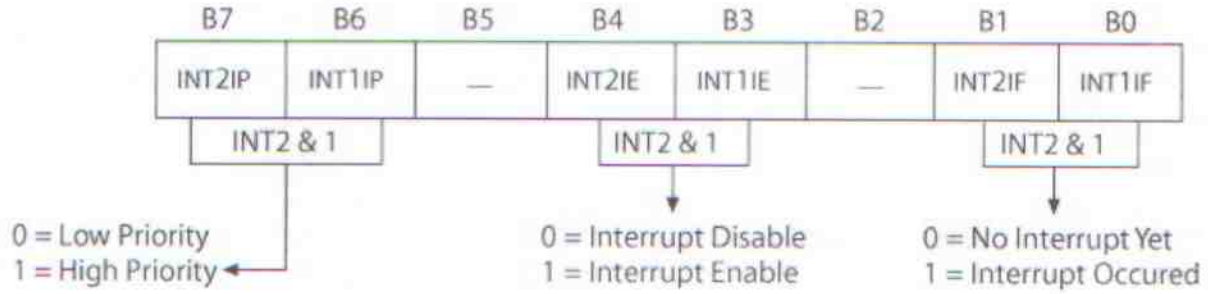
9.4


```
REGB    EQU   0x01      ; Register to save INPUT
        ORG   0x20
        MOVLW B'11110000' ; RB7-4 Input, RB3-0 output
        MOVWF TRISB
        BCF   INTCON2,7 ; Enable All-up resistors
LOOP    MOVFF PORTB,REGB ; Read switches
        COMF  REGB,F    ; Complement switch reading
        SWAPF REGB,F    ; Swap switches to LEDs
        MOVFF REGB,PORTB ; Turn on LEDs
        BRA   LOOP
        END
```

10.19

```
        ORG     0x0008      ;High priority interrupt vector
        BTFSC   INTCON3,INTIIF  ;Check INT1 Flag - skip if clear
        GOTO    INT1_ISR
        BTFSC   PIR1,TMR1IF ;Check Timer1 Flag - skip if clear
        GOTO    TMR1_ISR
        RETFIE  FAST        ;Return - False alarm
;
```

INTCON3 Register:



PIR1 Register: