EECS 281, April 16, 2015

Example : Chaser: Delay loop

```
        movlw   D'50'
        movwf   j
jloop:
        movwf   k
kloop:
        decfsz  k
        goto    kloop
        decfsz  j
        goto    jloop
```

Example: adder

- Cycles output bits RB2-0 in a 3-bit counting pattern
- output the sum of RB2-0 on RB4-5
- Compare the inputs RB6-7 to RB4-5 and set the output RB3 if different.
- count up if input RA3 is low
  " down " " " " high

		7	6	5	4	3	2	1	0
PORTB:	TRISB:	1	1	0	0	0	0	0	0

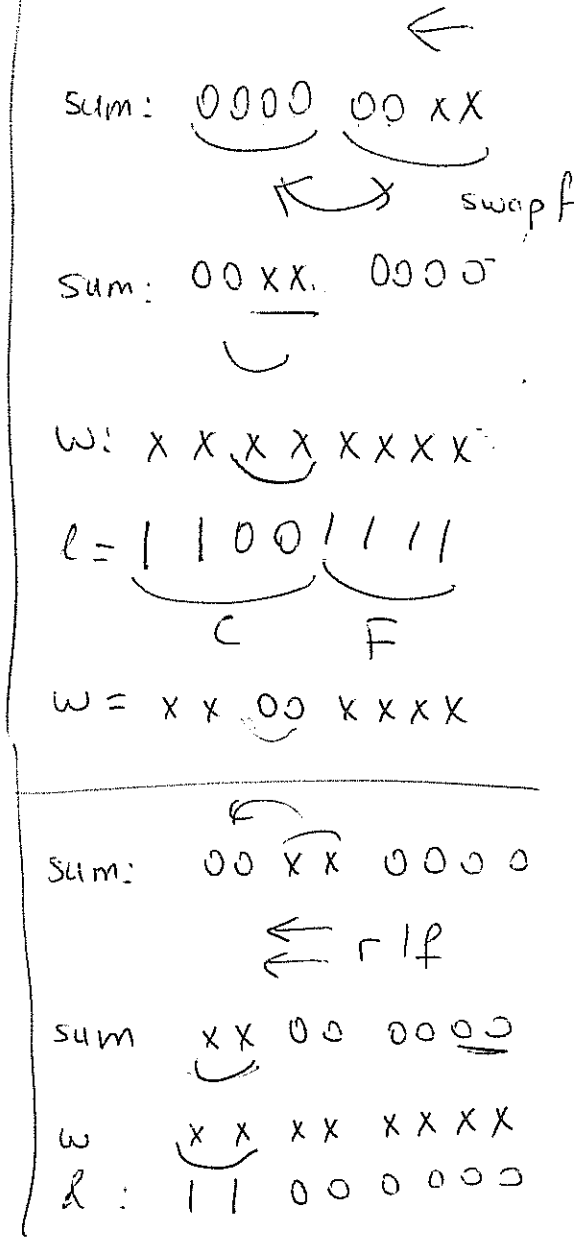
```
; let's configure PORTB
    bsf   STATUS, RP0
    movlw B'11000000'
    movwf TRISB
    bcf   STATUS, RP0
```

```
; initialize
    clrf    PORTB
    clrf    ctr

mloop:
    ; move the counter to RB2-0
```

	7	6	5	4	3	2	1	0	
count :	0	0	0	0	0	X	X	X	100 011
w :	x	x	x	x	x	x	x	x	
and 1 :	1	1	1	1	1	0	0	0	
w? or wut	x	x	x	x	x	0	0	0	

```
    movf    PORTB, w
    andlw   0xF8
    iorwf   ctr, w
    movwf   PORTB
```

```
; compute the sum
        clrf    sum
        btfsc   ctr,0
        incf    sum,f
        btfsc   ctr,1
        incf    sum,f
        btfsc   ctr,2
        incf    sum,f
        swapf   sum,f

        movf    PORTB,w
        andlw   0xCF
        iorwf   sum,w
        movwf   PORTB
; compare the sum with the input sum.
        ~~cbe~~ btfsc   STATUS,C
        rlf     sum,f
        rlf     sum,f
        andlw   0xC0
```



```
        xorwf   sum, w
        btfss   STATUS, 2
        goto    L1
        bcf     PORTB, 3
        goto    L2
L1:     bsf     PORTB, 3
L2:
        ; delay.
        (
        :
        :
        btfss   PORTA, 3
        goto    L4
        decf    ctr, f
        goto    L5
L4:     incf    ctr, f
L5:     movf    ctr, w
        andlw   0x07
        movwf   ctr
```

ctr: x x x x x x x x

```
; adder.asm
; Steven L. Garverick

; Cycles output bits RB2-0 in a 3-bit counting pattern
; Outputs the sum of RB2-0 on RB4,5
; Compares inputs RB6,7 to RB4,5 and sets output RB3 if different

; Counts up if input RA3 is low
; Counts down if input RA3 is high
; Uses a double-loop to create long delays before inc/dec RB2-0
; The loop delay is approx (1 + (RA2-0)) * 16 * 256 * 3 CPU cycles
; With an RC oscillator, about 100 kHz, a CPU cycle is 40 usec
; Therefore, the loop delay ranges from about 492 msec to 3.93 sec

; CPU configuration
; (16F84 with RC osc, watchdog timer off, power-up timer on)

        processor 16f84a
        include <p16F84a.inc>
        __config _RC_OSC & _WDT_OFF & _PWRTE_ON

; file register variables

ctr equ 0x0C    ; counter for output words
sum equ 0x0D    ; sum of counter bits (temporary storage)
octr equ 0x0E   ; outer-loop counter for delays
ictr equ 0x0F   ; inner-loop counter for delays

; beginning of program code

        org     0x00    ; reset at address 0
reset:  goto    init    ; skip reserved program addresses

        org     0x08    ; beginning of user code
init:
; on reset, all ports are inputs
; this code sets up RB5-0 as outputs
        bsf     STATUS,RP0      ; switch to bank 1 memory
        movlw   B'11000000'     ; RB7-6 are inputs, RB5-0 are outputs
        movwf   TRISB           ; set the I/O direction for PORTB
        bcf     STATUS,RP0      ; return to bank 0 memory

; initialize state variables
        clrf    PORTB           ; set all PORTB outputs to 0
        clrf    ctr             ; start with the counter at 0

mloop:  ; here begins the main program loop

; move the counter to RB2-0
        movf    PORTB,W ; load PORTB into W
        andlw   0xF8    ; and-out bits 2-0
        iorwf   ctr,W   ; or-in the counter
        movwf   PORTB   ; copy the result back to PORTB

; compute the expected sum and carry bits and copy to RB5-4
        clrf    sum     ; start with zero sum
        btfsc   ctr,0   ; skip increment if RB0=0
        incf    sum,F   ; increment the sum
        btfsc   ctr,1   ; skip increment if RB1=0
        incf    sum,F   ; increment the sum
        btfsc   ctr,2   ; skip increment if RB2=0
        incf    sum,F   ; increment the sum
        swapf   sum,F   ; move bits 1-0 of the sum to 5-4
```

```
        movf    PORTB,W ; load PORTB into W
        andlw   0xCF    ; and-out bits 5-4
        iorwf   sum,W   ; or-in the sum to bits 5-4
        movwf   PORTB   ; copy the result back to PORTB

; compare expected sum to actual sum and set/clear error flag
        rlf     sum,F   ; rotate expected sum bits to position 7-6
        rlf     sum,F
        movf    PORTB,W ; load PORTB into W
        andlw   0xC0    ; and-out bits 5-0, i.e. keep 7-6
        xorwf   sum,W   ; compare expected to actual sum
        btfss   STATUS,Z; if (Z) bcf RB3; else bsf RB3
        goto    L1
        bcf     PORTB,3 ; clear the error flag
        goto    L2
L1:     bsf     PORTB,3 ; set the error flag
L2:

; insert a double-loop delay using RA2-0
        ; initialize the outer-loop counter (octr)
        movf    PORTA,W ; load PORTA into W
        andlw   0x07    ; and-out bits 7-3, i.e. keep 2-0
        movwf   octr    ; copy the result to the loop counter
        incf    octr,F  ; add 1 to its initial value
        swapf   octr,F  ; swap nibbles to multiply by 16
        ; initialize the inner-loop counter (W)
        clrf    ictr    ; initialize the inner-loop counter (ictr)
L3:     ; inner loop
        decfsz  ictr,F  ; if (--ictr != 0) goto L3
        goto    L3
        ; outer loop
        decfsz  octr,F  ; if (--octr != 0) goto L3
        goto    L3

; test RA3 and inc/dec the 3-bit counter
        btfss   PORTA,3 ; if (RA3) decf ctr; else incf ctr
        goto    L4
        decf    ctr,F   ; decrement ctr if RA3=1
        goto    L5
L4:     incf    ctr,F   ; increment ctr if RA3=0
L5:     movf    ctr,W   ; load ctr into W
        andlw   0x07    ; and-out bits 7-3 to insure mod-8 counting
        movwf   ctr     ; copy the result back to ctr

; repeat the endless loop
        goto    mloop
        end             ; end of program code
```