

```

;
;           Leilani Electronics
;   program written & tested by William E. Nickles
;           CDS (Light) Current/Voltage Display
;           rev.1 2/26/2024
;
;           list      p=16f688
;           __config  h'00f4'
;           radix     hex
;
status      equ      0x03
porta       equ      0x05
portc       equ      0x07
intcon      equ      0x0b
cmcon0      equ      0x19
option_reg  equ      0x81
trisa       equ      0x85
trisc       equ      0x87
pcon        equ      0x8e
oscon       equ      0x8f
ansel       equ      0x91
wpua        equ      0x95
;
xmit        equ      0x22
temp0       equ      0x23
osdata      equ      0x24
bitctr0     equ      0x25
bitctrl1    equ      0x26
counter0     equ      0x27
counter1     equ      0x28
counter2     equ      0x29
counter3     equ      0x2a
counter      equ      0x2b
;
data1       equ      0x2c
bin_dec0    equ      0x2d
bin_dec1    equ      0x2e
lsd         equ      0x2f
nsd0        equ      0x30
nsd1        equ      0x31
nsd2        equ      0x32
msd         equ      0x33
;
c           equ      0           ; carrier bit
z           equ      2           ; zero bit
rp0         equ      5           ; bank select bit0
rp1         equ      6           ; bank select bit1
;
;   porta
;   icsp     equ      0           ; pgd
;   icsp     equ      1           ; pgc
sel         equ      2           ; selection input (na)
;mclr       equ      3           ; system reset
dout        equ      4           ; adc output
cs          equ      5           ; adc en/disable
;
;   portc
sdi         equ      0           ; driver input
sclk        equ      1           ; driver clock
le          equ      2           ; driver data update
mclk        equ      5           ; adc clock
;
;           org      0x00
;           goto     start
;
start       clrf      intcon      ; option uc
           clrf      porta
           clrf      portc
           movlw     0x07
           movwf     cmcon0
           bsf       status,rp0   ; goto bank1
           movlw     0x1f         ; configure i/o ports
           movwf     trisa

```

```

movlw 0x00
movwf trisc
clrf wpua
clrf ansel
movlw 0x88
movwf option_reg
movlw 0x03
movwf pcon
movlw 0x67
movwf osccon
bcf status,rp0 ; goto bank0
;
bsf porta,cs
bcf portc,sdi
bcf portc,sclk
bcf portc,le
bcf portc,mclk
;
cds    call cds_vi
        call convert
        movf nsd0,w
        nop
        movwf temp0
        call check#
        movf nsd1,w
        nop
        movwf temp0
        call check#
        call latch
        call wait100
        call wait100
        call wait100
        goto cds ; take another reading
;
; take adc reading
cds_vi bcf porta,cs ; enable adc
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        call clockm ; clock in start bits
        nop
        nop
        nop
        nop
        nop
        call clockm
        nop
        nop
        nop
        nop
        call clockm ; clock in null bit
        nop
        nop
        nop
        nop
        call rcv2 ; rcv 10-bit results
        movf data1,w
        nop
        movwf bin_dec1
        call rcv8
        movf data1,w
        nop

```

```

        movwf    bin_dec0
        bsf      porta,cs      ; disable adc
        return

;
;  numbers 0-9
zero     movlw   0xfd
        call    xmits
        return
one      movlw   0x61
        call    xmits
        return
two      movlw   0xdb
        call    xmits
        return
three    movlw   0xf3
        call    xmits
        return
four     movlw   0x67
        call    xmits
        return
five     movlw   0xb7
        call    xmits
        return
six      movlw   0x3f
        call    xmits
        return
seven    movlw   0xe1
        call    xmits
        return
eight    movlw   0xff
        call    xmits
        return
nine     movlw   0xe7
        call    xmits
        return

;
;  rcv 2 bits from adc
rcv2     clrf    data1
        movlw   0x02
        movwf   counter1
again3    bcf     status,c
        btfss   porta,dout
        goto    ser_in2
        bsf     status,c
ser_in2   rlf     data1,f
        call    clockm
        decfsz  counter1,f
        goto    again3
        return

;  rcv 8 bits from adc
rcv8     clrf    data1
        movlw   0x08
        movwf   counter1
again2    bcf     status,c
        btfss   porta,dout
        goto    ser_inn
        bsf     status,c
ser_inn   rlf     data1,f
        call    clockm
        decfsz  counter1,f
        goto    again2
        return

;
;  binary to deciaml converter
convert   bcf     status,z      ; clear zero flag
        clrf    lsd
        clrf    nsd0
        clrf    nsd1
        clrf    nsd2
        clrf    msd
conv1     movlw   0x00

```

```

    subwf    bin_dec0,w
    btfss    status,z
    goto     conv0
    bcf      status,z
    movlw    0x00
    subwf    bin_dec1,w
    btfsc    status,z
    goto     done1
    movlw    0xff
    nop
    movwf    bin_dec0
    decf     bin_dec1,f
    goto     conv2
conv0:
conv2:
    decf     bin_dec0,f
    incf     lsd,f      ; increment lsd
    bcf      status,z   ; clear zero bit before next calculation
    movlw    0x0a
    subwf    lsd,w      ; is lsd '10' ?
    btfss    status,z
    goto     conv1      ; no, keep going..
    clrf     lsd        ; yes, set lsd to '0' and increment nsd0
    incf     nsd0,f
    bcf      status,z   ; clear zero bit before next calculation
    movlw    0x0a
    subwf    nsd0,w     ; is nsd0 '10' ?
    btfss    status,z
    goto     conv1      ; no, keep going..
    clrf     nsd0       ; yes, set nsd0 to '0' and increment nsd1
    incf     nsd1,f
    bcf      status,z   ; clear zero bit before next calculation
    movlw    0x0a
    subwf    nsd1,w     ; is nsd1 '10' ?
    btfss    status,z
    goto     conv1      ; no, keep going..
    clrf     nsd1       ; yes, set nsd1 to '0' and increment nsd2
    incf     nsd2,f
    bcf      status,z   ; clear zero bit before next calculation
    movlw    0x0a
    subwf    nsd2,w     ; is nsd2 '10' ?
    btfss    status,z
    goto     conv1      ; no, keep going..
    clrf     nsd2       ; yes, set nsd2 to '0' and increment msd
    incf     msd,f
    bcf      status,z   ; clear zero bit before next calculation
    movlw    0x09
    subwf    msd,w      ; is msd '9' ?
    btfss    status,z
    goto     conv1      ; no, keep going..
done1:
    bcf      status,z   ; yes, counter maxed out,
    return
;
check#:
    bcf      status,z   ; clear zero bit before calculating
    movlw    0x01       ; test #1
    subwf    temp0,w
    btfss    status,z   ; is 'temp0' equal to '1' ?
    goto     check2     ; no, check #2
    call     one        ; yes, display #1
    goto     done0       ; done..
check2:
    bcf      status,z   ; clear zero bit before next calculation
    movlw    0x02       ; test #2
    subwf    temp0,w
    btfss    status,z   ; is 'temp0' equal to '2' ?
    goto     check3     ; no, check #3
    call     two        ; yes, display #2
    goto     done0
check3:
    bcf      status,z   ; clear zero bit before next calculation
    movlw    0x03       ; test #3
    subwf    temp0,w
    btfss    status,z   ; is 'temp0' equal to '3' ?
    goto     check4     ; no, check #4
    call     three      ; yes, display #3

```

```

goto    done0
check4  bcf     status,z      ; clear zero bit before next calculation
        movlw   0x04         ; test #4
        subwf   temp0,w
        btfss   status,z      ; is 'temp0' equal to '4' ?
        goto    check5       ; no, check #5
        call    four         ; yes, display #4
        goto    done0
check5  bcf     status,z      ; clear zero bit before next calculation
        movlw   0x05         ; test #5
        subwf   temp0,w
        btfss   status,z      ; is 'temp0' equal to '5' ?
        goto    check6       ; no, check #6
        call    five         ; yes, display #5
        goto    done0
check6  bcf     status,z      ; clear zero bit before next calculation
        movlw   0x06         ; test #6
        subwf   temp0,w
        btfss   status,z      ; is 'temp0' equal to '6' ?
        goto    check7       ; no, check #7
        call    six          ; yes, display #6
        goto    done0
check7  bcf     status,z      ; clear zero bit before next calculation
        movlw   0x07         ; test #7
        subwf   temp0,w
        btfss   status,z      ; is 'temp0' equal to '7' ?
        goto    check8       ; no, check #8
        call    seven        ; yes, display #7
        goto    done0
check8  bcf     status,z      ; clear zero bit before next calculation
        movlw   0x08         ; test #8
        subwf   temp0,w
        btfss   status,z      ; is 'temp0' equal to '8' ?
        goto    check9       ; no, check #9
        call    eight        ; yes, display #8
        goto    done0
check9  bcf     status,z      ; clear zero bit before next calculation
        movlw   0x09         ; test #9
        subwf   temp0,w
        btfss   status,z      ; is 'temp0' equal to '9' ?
        goto    check0       ; no, display #0
        call    nine         ; yes, display #9
        goto    done0
check0  call    zero
done0   bcf     status,z      ; done..
        return

;
;  xmit 8 bits (display driver)
xmits   movwf   osdata
        movlw   0x08
        movwf   bitctr0
oled0   rrf     osdata,f
        btfss   status,c
        goto    oled1
        bsf     portc,sdi
oled1   goto    oled2
oled2   bcf     portc,sdi
        call    clocks
        decfsz  bitctr0,f
        goto    oled0
        return

;
clocks  bsf     portc,sclk    ; toggle driver clock
        nop
        nop
        nop
        nop
        nop
        bcf     portc,sclk
        return
clockm  bsf     portc,mclk    ; toggle adc clock

```

```

nop
nop
nop
nop
nop
nop
    bcf     portc,mclk
    return
latch    bsf     portc,le      ; update driver data
nop
nop
nop
nop
nop
    bcf     portc,le
    return
;   brief wait...
wait100  movlw   0xff
        movwf   counter2
repeat3  movlw   0xff
        movwf   counter3
loop3    decfsz  counter3,f
        goto    loop3
        decfsz  counter2,f
        goto    repeat3
        return
;
end

```