```
;Duty Cycle Calculation

;   CalcSPERROR     - vAbove / vBelow
;   CalcSPERRORsum
;   CalcSPERRORchg  - vInc / vDec / vChgMax
;   CalcSPERRORacc  - vAccel / vDecel
;   ---------------
;   CalcDCnew       - calculate DUTYCYCLEnew
;   ---------------
;   IncDUTYCYCLEsp  - increment DUTYCYCLEsp
;   AddWtoDUTYCYCLEsp
;   DecDUTYCYCLEsp  - decrement DUTYCYCLEsp
;   SubWfromDUTYCYCLEsp
;   ---------------
;   DecDCBIAS
;   SubWfromDCBIAS
;   IncDCBIAS
;   AddWtoDCBIAS
;   ---------------
;   PutDCnew        - load DUTYCYCLEnew into PWM

;==================
;calculate SPERROR = |SETPOINT(f) - COILTEMP(W)|
;set vAbove or vBelow in vCOILTEMP (sign of the error)
;count vAbove and vBelow events, clear every 5 seconds
CalcSPERROR:
    banksel SETPOINT

    ;clear vAbove and vBelow flags
    Clear_vAbove
    Clear_vBelow

    ;save previous SPERROR
    movf    SPERROR,W
      movwf SPERROR2
    movf    SPERROR+1,W
      movwf SPERROR2+1

    ;do 16-bit subtraction:  SPERROR = SETPOINT(f) - COILTEMP(W)
    movf    COILTEMPptr,W
    movwf   FSR             ;lo byte pointer
    movf    INDF,W          ;load W with lo(COILTEMP)
    subwf   SETPOINT,W      ;C clear if W>f
    movwf   SPERROR

    movf    SETPOINT+1,W    ;set hi byte of ...
     movwf  SPERROR+1       ;... error result
    incf    FSR             ;hi byte pointer
    movf    INDF,W          ;load W with hi(COILTEMP)
    ;check for borrow
    Skip_If_CARRY_SET       ;W>f? (C clear)
     incfsz INDF,W          ;Yes, do borrow (don't affect CARRY)
    subwf   SPERROR+1       ;C clear if COILTEMP > SETPOINT (vAbove)

    ;check if COILTEMP > SETPOINT (above)
    Skip_If_CARRY_SET       ;W(COILTEMP)>f(SETPOINT)? (C clear)
```

```
    goto    Calc_above      ;Yes, signal above SETPOINT

    ;check for COILTEMP below/at SETPOINT
    movf    SPERROR,W
    iorwf   SPERROR+1,W     ;test for ZERO
    Skip_If_ZERO
    goto    Calc_below      ;not zero, is Too Cold

    ;vAbove = vBelow = 0
    goto    Calc_sperrtst   ;test to clear error range counter

Calc_above: ;SPERROR < 0, so 2s complement it
    comf    SPERROR+1       ;complement hi(error)
    comf    SPERROR,W       ;complement lo(error)
    addlw   .1              ;2s complement
    movwf   SPERROR
    Skip_If_CARRY_CLR
      incf  SPERROR+1
    Set_vAbove              ;COILTEMP is Too Hot
    goto    Calc_sperrtst

Calc_below: ;SPERROR > 0
    Set_vBelow              ;COILTEMP is Too Cold

    ;test to clear error range counter
Calc_sperrtst:

    ;wait until after startup delay
    banksel TOKEITdelay
    movf    TOKEITdelay
    Skip_If_ZERO
    return
    movf    TOKEITdelay+1
    Skip_If_ZERO
    return

    decf    SPERRORcnt
    Skip_If_NOT_ZERO
    goto    Calc_sperradj   ;adjust DCBIAS and reset counters

    ;increment a counter
    Skip_If_NOT_vAbove
    incf    SPERRORabove    ;count vAbove event
    Skip_If_NOT_vBelow
    incf    SPERRORbelow    ;count vBelow event
    return

Calc_sperradj:
    ;if SPERRORabove > SPERRORbelow then decrement DCBIAS
    movf    SPERRORabove,W
    subwf   SPERRORbelow,W
    Skip_If_CARRY_CLR
    goto    Calc_sperradj2
    call    DecDCBIAS
    goto    Calc_sperrclr
```

```
Calc_sperradj2:
    ;if SPERRORbelow > SPERRORabove then increment DCBIAS
    movf    SPERRORbelow,W
    subwf   SPERRORabove
    Skip_If_CARRY_SET
    call    IncDCBIAS

Calc_sperrclr:
    ;clear the counters
    clrf    SPERRORabove
    clrf    SPERRORbelow
    movlw   cSPERRORcnt
    movwf   SPERRORcnt
    return

;------------------
;Calculate the sum of SPERROR
;set to zero at/cross SETPOINT
CalcSPERRORsum:
    banksel SPERRORsum

    ;wait until after start up
    Skip_If_NOT_vDCmax
    return
    movf    SPERROR+1
    Skip_If_ZERO
    return

    ;test for at or crossed SETPOINT
    Skip_If_NOT_vAbove
    goto    css1
    Skip_If_NOT_vBelow
    goto    css2
    goto    css3                ;neither set, at SETPOINT

css1:   ;vAbove
    Skip_If_Old_NOT_vAbove
    goto    css4                ;same as before, add value
    goto    css3                ;crossed SETPOINT

css2:   ;vBelow
    Skip_If_Old_NOT_vBelow
    goto    css4                ;same as before, add value

    ;... crossed SETPOINT ...

css3:   ;at or crossed SETPOINT
    clrf    SPERRORsum
    clrf    SPERRORsum+1

css4:   ;add the error into sum
    movf    SPERROR,W
    addwf   SPERRORsum,W
    Skip_If_CARRY_CLR
    movlw   0xFF                ;limit to max value
    movwf   SPERRORsum
```

```
    return

;------------------
;Calculate the change in SPERROR:  f(<previous SPERROR>) - W(SPERROR)
;set vInc (error increasing) or vDec (error decreasing)
;If vDCmax Then save maximum change
;set vChgMax if SPERRORchg > cCHANGEmax
CalcSPERRORchg:

    banksel SPERRORchg
    Clear_vInc
    Clear_vDec
    Clear_vChgMax

    ;set SPERRORchg2 to previous SPERRORchg
    movf    SPERRORchg,W
      movwf SPERRORchg2

    ;8-bit subtract: SPERRORchg = |f(<previous SPERROR>) - W(SPERROR)|
    movf    SPERROR,W
    subwf   SPERROR2,W      ;(f)-(W): C clear if W>f i.e. temp increasing
    movwf   SPERRORchg
    ;check for borrow
    Skip_If_CARRY_CLR       ;W>f? (C clear)
    goto    calc_chg2

    ;SPERRORchg < 0, so 2s complement it
    comf    SPERRORchg,W    ;complement lo(error)
    addlw   .1              ;2s complement
    Skip_If_CARRY_CLR
     movlw  0xFF
    movwf   SPERRORchg

    ;check if SPERROR > previous SPERROR
calc_chg2:
    movf    SPERROR,W
    subwf   SPERROR2,W
    Skip_If_CARRY_SET       ;W(SPERRORchg)>f(previous SPERRORchg)? (C clear)
      goto  chg_inc         ;Yes, error increasing
    Skip_If_ZERO
      goto  chg_dec         ;not zero, is cooling
    return                  ;flat slope

chg_inc:    ;SPERROR is increasing
    Set_vInc
    goto    chg_max

chg_dec:    ;SPERROR is decreasing
    Set_vDec

    ;save maximum change during DCmax
chg_max:
    Skip_If_vDCmax
    goto    chg_tst
    movf    SPERRORchg,W
    subwf   CHANGEmax,W
```

```
    Skip_If_CARRY_CLR
    goto    chg_tst
    movf    SPERRORchg,W
    movwf   CHANGEmax

    ;test for max allowed SPERRORchg (same as COILTEMP change)
chg_tst:
    movf    SPERRORchg,W
    sublw   cCHANGEmax
    Skip_If_CARRY_SET
    Set_vChgMax             ;set flag = change is too large
    return

;------------------
;calculate SPERROR acceleration
;set vAccel (change increasing) or vDecel (change decreasing) in vCOILTEMP
CalcSPERRORacc:
    banksel SPERRORchg
    Clear_vAccel
    Clear_vDecel

    ;test if last two changes increasing
    Skip_If_Old_vInc
    goto    acc0
    Skip_If_vInc
    return              ;reversed direction
    ;still increasing
    goto    acc1

acc0:
    ;test if last two changes decreasing
    Skip_If_Old_vDec
    return
    Skip_If_vDec
    return              ;reversed direction
    ;still decreasing
    ;goto   acc1

acc1:   ;determine accelerate/decelerate
    movf    SPERRORchg,W
    subwf   SPERRORchg2,W
    ;check if SPERRORchg > previous SPERRORchg
    Skip_If_CARRY_SET       ;if W(chg)>f(chg2) then C clear
    goto    acc3

acc2:   ;error decelerating
    Set_vDecel
    return

acc3:   ;error accelerating
    Set_vAccel
    return

;==================
;Calculate the new DUTYCYCLE value
;DCBIAS: duty cycle for no more than 1°F/sec increase
```

```
;SPERROR = |SETPOINT(f) - COILTEMP(W)|
;   vAbove  set if COILTEMP > SETPOINT
;   vBelow  set if COILTEMP < SETPOINT
;SPERRORchg = |previous SPERROR - SPERROR|
;   vInc    set if SPERROR increasing
;   vDec    set if SPERROR decreasing
;   vChgMax set if SPERRORchg > max allowed value (CHANGEmax)
;   vAccel  set if SPERRORchg > SPERRORchg2
;   vDecel  set if SPERRORchg < SPERRORchg2
;DUTYCYCLEsp:  calculated value to maintain COILTEMP = SETPOINT
;DUTYCYCLEnew: 10-bit value to load into PWM duty cycle register
;------------------
CalcDCnew:
    ;clear new value flag
    Clear_vDCnew                ;set flag if new value calculated

;==================
;if vDCmax then test for too much change
    Skip_If_vDCmax              ;if vDCmax ...
    goto    run_PID
    Skip_If_vChgMax             ;... AND vChgMax ...
    goto    run_PID
    goto    calcDCnew_zero      ;... turn off heater

;==================
;Run the PID algorithm - calculate adjustment and apply to DCBIAS
run_PID:
    call    eePutDCBIAS         ;put current DCBIAS in eePROM

    ;set DUTYCYCLEsp to DCBIAS +/- DCVAR
    banksel DCBIAS
    movf    DCBIAS,W
      movwf DUTYCYCLEsp
    movf    DCBIAS+1,W
      movwf DUTYCYCLEsp+1

;==================
case_vAbove:
    Skip_If_vAbove
    goto    case_vBelow

        ;=== handle temp too high
        movf    SPERROR+1       ;if SPERROR > cSPABOVElo,hi ...
        Skip_If_ZERO
        goto    calcDCnew_zero  ;... then power off
        movf    SPERROR,W
        sublw   aERRORmax
        Skip_If_CARRY_SET
        goto    calcDCnew_zero  ;... then power off

        ;=== adjust DUTYCYCLEsp by +/- (PID value)

        ;P:     Pgain factor = max(SPERROR/8, 7)
        rrf     SPERROR,W
        movwf   TEMPVAR
        rrf     TEMPVAR
```

```
        rrf     TEMPVAR,W
        andlw   b'00011111'
        Skip_If_NOT_ZERO
        movlw   aPgain
        call    Sub2WfromDUTYCYCLEsp

        ;I:     Igain factor = max(SPERROR/8, 7)
        rrf     SPERRORsum,W
        movwf   TEMPVAR
        rrf     TEMPVAR
        rrf     TEMPVAR,W
        andlw   b'00011111'
        Skip_If_NOT_ZERO
        movlw   aIgain
        call    Sub2WfromDUTYCYCLEsp

        ;D:     DCVAR = DCVAR +/- f(SPERRORchg)
        Skip_If_vInc
        goto    vA2

            ;increasing error = max(SPERROR/16, 15)
            swapf   SPERRORchg,W
            andlw   b'00001111'
            Skip_If_NOT_ZERO
            movlw   aDgainInc
            call    Sub2WfromDUTYCYCLEsp
            goto    calcDCnew_sp            ;DCnew = DCsp

vA2:        ;decreasing error = max(SPERROR/16, 7)
            swapf   SPERRORchg,W
            andlw   b'00001111'
            Skip_If_NOT_ZERO
            movlw   aDgainDec
            call    Add2WtoDUTYCYCLEsp
            goto    calcDCnew_sp            ;DCnew = DCsp
        ;==================

;==================
case_vBelow:
    ;test for at SETPOINT
    Skip_If_vBelow
    goto    calcDCnew_sp        ;at SETPOINT! DCnew = DCsp (DCBIAS)

        ;=== handle overshoot on startup
        movf    SPERROR,W       ;if SPERROR > SPERRORtrip ...
        subwf   SPERRORtrip,W
        movf    SPERROR+1,W
        Skip_If_CARRY_SET
         incfsz SPERROR+1,W
        subwf   SPERRORtrip+1,W
        Skip_If_CARRY_SET
        goto    calcDCnew_max   ;(keep cranking)

        ;=== handle temp 15F below SETPOINT
        movf    SPERROR+1       ;68°F low
        Skip_If_ZERO
```

```
        goto    calcDCnew_max   ;then bump power
        movf    SPERROR,W       ;test for startup
        sublw   bERRORmax
        Skip_If_CARRY_SET
        goto    calcDCnew_sp4   ;... then bump power

        ;=== adjust DUTYCYCLEsp by +/- (PID value)

        ;P:     Pgain factor
        rrf     SPERROR,W
        movwf   TEMPVAR
        rrf     TEMPVAR
        rrf     TEMPVAR,W
        andlw   b'00011111'
        Skip_If_NOT_ZERO
        movlw   bPgain
        call    Add2WtoDUTYCYCLEsp

        ;I:     Igain factor
        rrf     SPERRORsum,W
        movwf   TEMPVAR
        rrf     TEMPVAR
        rrf     TEMPVAR,W
        andlw   b'00011111'
        Skip_If_NOT_ZERO
        movlw   bIgain
        call    Add2WtoDUTYCYCLEsp

        ;Dgain factor
        Skip_If_vInc
        goto    vB2

            ;increasing error
            swapf   SPERRORchg,W
            andlw   b'00001111'
            Skip_If_NOT_ZERO
            movlw   bDgainInc
            call    Add2WtoDUTYCYCLEsp
            goto    calcDCnew_sp            ;DCnew = DCsp

vB2:        ;decreasing error
            swapf   SPERRORchg,W
            andlw   b'00001111'
            Skip_If_NOT_ZERO
            movlw   bDgainDec
            call    Sub2WfromDUTYCYCLEsp
            goto    calcDCnew_sp            ;DCnew = DCsp

        ;==================

;==================
;calcDCnew_sp04:    ;set DUTYCYCLEnew to DCBIAS / 4
;   banksel DUTYCYCLEsp
;   ;divide DCBIAS (in DUTYCYCLEsp) by 2
;   Clear_CARRY
;   rrf     DUTYCYCLEsp+1
```

```
;   rrf     DUTYCYCLEsp
;calcDCnew_sp02:    ;set DUTYCYCLEnew to DCBIAS / 2
;   banksel DUTYCYCLEsp
;   ;divide DCBIAS (in DUTYCYCLEsp) by 2
;   Clear_CARRY
;   rrf     DUTYCYCLEsp+1
;   rrf     DUTYCYCLEsp
;   goto    calcDCnew_sp
;------------------
calcDCnew_sp4:  ;set DUTYCYCLEnew to DCBIAS * 4
    banksel DUTYCYCLEsp
    ;multiply DCBIAS (in DUTYCYCLEsp) by 2
    Clear_CARRY
    rlf     DUTYCYCLEsp
    rlf     DUTYCYCLEsp+1
calcDCnew_sp2:  ;set DUTYCYCLEnew to DCBIAS * 2
    banksel DUTYCYCLEsp
    ;multiply DCBIAS (in DUTYCYCLEsp) by 2
    Clear_CARRY
    rlf     DUTYCYCLEsp
    rlf     DUTYCYCLEsp+1
;------------------
;at SETPOINT, maintain bias setting
calcDCnew_sp:   ;set DUTYCYCLEnew = DUTYCYCLEsp
    banksel DUTYCYCLEsp
    movf    DUTYCYCLEsp,W
      movwf DUTYCYCLEnew
    movf    DUTYCYCLEsp+1,W
      movwf DUTYCYCLEnew+1
    Clear_vDCzero
    Clear_vDCmax
    Set_vDCsp
    Set_vDCnew
    return
;------------------
calcDCnew_zero: ;set DUTYCYCLEnew to zero
    banksel DUTYCYCLEnew
    clrf    DUTYCYCLEnew
    clrf    DUTYCYCLEnew+1
    Set_vDCzero
    Clear_vDCmax
    Clear_vDCsp
    Set_vDCnew
    return
;------------------
calcDCnew_max:  ;set DUTYCYCLEnew to DCMAX
    banksel DUTYCYCLEnew
    movlw   cDCMAXlo
      movwf DUTYCYCLEnew
    movlw   cDCMAXhi
      movwf DUTYCYCLEnew+1
    Clear_vDCzero
    Set_vDCmax
    Clear_vDCsp
    Set_vDCnew
    return
```

```
;==================
;increment DUTYCYCLEsp, limit to DCmax (1023.)
IncDUTYCYCLEsp:
    movlw   .1
    goto    AddWtoDUTYCYCLEsp
;------------------
Add2WtoDUTYCYCLEsp:
    movwf   TEMPVAR
    Skip_If_NOT_vInc
    rlf     TEMPVAR,W       ;*2 if accelerating
;------------------
;Add (W) to DUTYCYCLEsp, limit to DCmax (1023.)
AddWtoDUTYCYCLEsp:
    banksel DUTYCYCLEsp
    addwf   DUTYCYCLEsp
    Skip_If_CARRY_CLR
      incf  DUTYCYCLEsp+1
    ;test for max value
    movf    DUTYCYCLEsp,W
    sublw   cDCMAXlo
    movf    DUTYCYCLEsp+1,W
    Skip_If_CARRY_SET
     incfsz DUTYCYCLEsp+1,W     ;borrow
    sublw   cDCMAXhi
    Skip_If_CARRY_CLR           ;if W(DUTYCYCLEsp) > DCMAX, C clear
     return
    ;set DUTYCYCLEsp to maximum value
    movf    cDCMAXlo,W
      movwf DUTYCYCLEsp
    movf    cDCMAXhi,W
      movwf DUTYCYCLEsp+1
    return

;------------------
;decrement DUTYCYCLEsp, minimum value = 0
DecDUTYCYCLEsp:
    movlw   .1
    goto    SubWfromDUTYCYCLEsp
;------------------
Sub2WfromDUTYCYCLEsp:
    movwf   TEMPVAR
    Skip_If_NOT_vInc
    rlf     TEMPVAR,W       ;*2 if accelerating
;subtract (W) from DUTYCYCLEsp, limit to 0
SubWfromDUTYCYCLEsp:
    banksel DUTYCYCLEsp
    subwf   DUTYCYCLEsp
    movlw   .1
    Skip_If_CARRY_SET           ;W > DUTYCYCLEsp? ...
      subwf DUTYCYCLEsp+1       ;... Yes, do borrow
    Skip_If_CARRY_CLR
      return                    ;... No, done, exit
    ;set DUTYCYCLEsp to minimum value
    clrf    DUTYCYCLEsp
    clrf    DUTYCYCLEsp+1
```

```
    return

;==================
;Decrement DCBIAS if vAbove - limit to zero
DecDCBIAS:
    movlw   .1
;Decrease DCBIAS by W
SubWfromDCBIAS:
    banksel DCBIAS
    subwf   DCBIAS
    movlw   .1
    Skip_If_CARRY_SET       ;lo(W) > lo(DCBIAS)? ...
      subwf DCBIAS+1        ;... Yes, do borrow
    Skip_If_CARRY_SET       ;if underflow ...
    goto    dec_DC0         ;... then reset to minimum
    ;if DCBIAS too small then ...
    movf    DCBIAS,W
    sublw   cDCBIASminlo
    movf    DCBIAS+1,W
    Skip_If_CARRY_SET
    incfsz  DCBIAS+1,W      ;borrow
    sublw   cDCBIASminhi
    Skip_If_CARRY_SET       ;if W(DCBIAS) > DCMAX, C clear
      return
    ;... set DCBIAS to minimum value
dec_DC0:
    movlw   cDCBIASminlo
      movwf DCBIAS
    movlw   cDCBIASminhi
      movwf DCBIAS+1
    return

;Increment DCBIAS if vBelow - limit to cDCBIASmax(lo,hi)
IncDCBIAS:
    movlw   .1
;Increase DCBIAS by W
AddWtoDCBIAS:
    banksel DCBIAS
    addwf   DCBIAS
    movlw   .1
    Skip_If_CARRY_CLR
      addwf DCBIAS+1    ;add carry
    Skip_If_CARRY_CLR
    goto    inc_DC0     ;if overflow then set to max
    ;test for above max value: DCMAX - DCBIAS
    movf    DCBIAS,W
    sublw   cDCMAXlo
    movf    DCBIAS+1,W
    Skip_If_CARRY_SET
     incfsz DCBIAS+1,W      ;borrow
    sublw   cDCMAXhi
    Skip_If_CARRY_CLR       ;if W(DCBIAS) > DCMAX, C clear
      return
    ;set DCBIAS to maximum value
inc_DC0:
    movlw   cDCBIASmaxlo
```

```
      movwf DCBIAS
    movlw   cDCBIASmaxhi
      movwf DCBIAS+1
    return

;==================
;store DUTYCYCLEnew,DUTYCYCLEnew+1 in CCPR1CON, CCPR1L
;use TEMPVAR to preserve DUTYCYCLEnew
PutDCnew:
    ;CCP1CON<5:4> is two LSB bits
    banksel DUTYCYCLEnew
    movf    DUTYCYCLEnew,W
    movwf   TEMPVAR
    movf    DUTYCYCLEnew+1,W
    movwf   TEMPVAR+1
    swapf   TEMPVAR,W       ;bits <1:0> goto W<5:4>
    andlw   b'00110000'     ;mask two LSBs
    iorlw   b'00001100'     ;configure for PWM mode active-high
    banksel CCP1CON
    movwf   CCP1CON         ;set 2 LSBs of DUTYCYCLE
    ;CCPR1L is hi byte
    banksel TEMPVAR
    rrf     TEMPVAR+1       ;right shift DUTYCYCLE(hi)
    rrf     TEMPVAR         ;into DUTYCYCLE(lo)
    rrf     TEMPVAR+1       ;right shift DUTYCYCLE(hi)
    rrf     TEMPVAR,W       ;into DUTYCYCLE(lo), load W
    banksel CCPR1L
    movwf   CCPR1L          ;set upper 8 bits of DUTYCYCLE
    return

;end    *****
```