```
;*****************************************************************************
; testing the LPF damping factor
; lpf.asm
;-----------------------------------------------------------------------------
;   Version: 2.0  By Dillian Wong
;-----------------------------------------------------------------------------
; Reference frequency = 12MHz, Sample cycle = 1




CLK             EQU             P3.0
DIN             EQU             P3.2
ENB             EQU             P3.4

C_REG           EQU             20H   ; C register
R_REG_H         EQU             21H   ; High byte of R register
R_REG_L         EQU             22H   ; Low byte of R register
N_REG_H         EQU             23H   ; High byte of N register
N_REG_L         EQU             24H   ; Low byte of N register

LOOP            EQU             30H
                ORG             0H
                JMP             Reset
;
;
Reset:          MOV             SP,#5FH ; set stack pointer
                SETB            ENB ; To make sure that ENB is deactivate
                MOV             C_REG,#0B3H
; C7=1 Unchanged polarity  C6=0 Enable detector B  C5=1 Enable Lock detector
; C4=1 C3=0 C2=0 OSC/8  C1=1 Enable Fv output    C0=1 Enable Fr output
                CALL            CDATA
                MOV             N_REG_L,#02EH
                MOV             N_REG_H,#04H  ; N=1070
                CALL            NDATA
                MOV             R_REG_L,#120
                MOV             R_REG_H,#0  ; R=120
                CALL            RDATA

again:
                setb            p1.0
                call            wait50ms
                clr             p1.0

                MOV             N_REG_L,#0E8H
                MOV             N_REG_H,#03H   ; N=1000
                CALL            NDATA

                call            wait50ms

                MOV             N_REG_L,#02EH
                MOV             N_REG_H,#04H  ; N=1070
                CALL            NDATA



OK:             JMP             again     ; Loop for ever



Wait50ms:
        mov     R2,     #50
        mov     R3,     #6
WL_01:  nop
        nop
        djnz    R3,     WL_01
        djnz    R2,     WL_01
        ret
```

```
                ;
; Subroutine for Register C data transfer
CDATA:          CLR             ClK      ; Serial Data Clock starts off low.
                CLR             ENB      ; Lowering ENB begins the data transfer.
                MOV             LOOP,#8 ; Eight bits to transfer.
XFER_C:         MOV             A,C_REG
                MOV             C,ACC.7 ; Move bit 7 into Carry (SPI is MSB first).
                MOV             DIN,C  ; Carry bit is sent as the Data bit.
                SETB            CLK     ; Generate Serial Data Clock rising edge.
                CLR             CLK     ; Generate Serial Data Clock falling edge.
                MOV             A,C_REG ; Accumulator is temp holder for shift operation.
                RL              A        ; Rotate left (but not through Carry!).
                MOV             C_REG,A ; Prepare bit 7 for next transfer to MC145170.
                DJNZ            LOOP,XFER_C; Decrement LOOP. Jump if not zero to XFER.
; Transfer done.
                SETB            ENB       ; Inactivate ENB.
                RET
; Subroutine for Register N data transfer
NDATA:          CLR             ClK     ; Serial Data Clock starts off low.
                CLR             ENB      ; Lowering ENB begins the data transfer.
                MOV             LOOP,#16 ; Sixteen bits to transfer.
XFER_N:         MOV             A,N_REG_H
                MOV             C,ACC.7 ; Move bit 7 into Carry (SPI is MSB first).
                MOV             DIN,C  ; Carry bit is sent as the Data bit.
                SETB            CLK     ; Generate Serial Data Clock rising edge.
                CLR             CLK     ; Generate Serial Data Clock falling edge.
                MOV             A,N_REG_L ;Accumulator is temp holder for shift operation.
                MOV             C,ACC.7  ; Get the bit 7 of N_REG_L
                MOV             A,N_REG_H ;
                RLC             A        ; Shift high byte one bit to left through Carry.
                MOV             N_REG_H,A
                MOV             A,N_REG_L
                RLC             A        ; Shift low byte one bit to left through Carry.
                MOV             N_REG_L,A
                DJNZ            LOOP,XFER_N; Decrement LOOP. Jump if not zero to XFER.
; Transfer done.
                SETB            ENB       ; Inactivate ENB.
                RET
; Subroutine for Register R data transfer
RDATA:          CLR             ClK     ; Serial Data Clock starts off low.
                CALL            SH2RL;
                CLR             ENB      ; Lowering ENB begins the data transfer.
                MOV             LOOP,#15 ; Fifteen bits to transfer.
XFER_R:         MOV             A,R_REG_H
                MOV             C,ACC.7 ; Move bit 7 into Carry (SPI is MSB first).
                MOV             DIN,C  ; Carry bit is sent as the Data bit.
                SETB            CLK     ; Generate Serial Data Clock rising edge.
                CLR             CLK     ; Generate Serial Data Clock falling edge.
                CALL            SH2RL;
                DJNZ            LOOP,XFER_R; Decrement LOOP. Jump if not zero to XFER.
; Transfer done.
                SETB            ENB       ; Inactivate ENB.
                RET
;
;
SH2RL:          MOV             A,R_REG_L ;Accumulator is temp holder for shift operation.
                MOV             C,ACC.7  ;
                MOV             A,R_REG_H ;Get the bit 7 of R_REG_L
                RLC             A        ; Shift high byte one bit to left through Carry.
                MOV             R_REG_H,A
                MOV             A,R_REG_L
                RLC             A        ; Shift low byte one bit to left through Carry.
                MOV             R_REG_L,A
                RET
                END                     ; Tell assembler code completed.
```