

```

; Filnavn FoxTimer_79.asm
;*****
; HUSK AT SÆTTE VERSIONSNUMMER I CONSTANT Ver.1,2,3,4 !
;*****
; Program til styring af rævesendere (mikkeline)
; Skrevet af OZ1LN Lasse H. P. Kjærbro.
; Der er ikke foretaget optimering af koden.
;-----
; Historik :
; 20-06-2007 : kodelstart OZ7FOX de OZ1LN
; 19-08-2008 : Version 0.70 idriftsættes i 4 rævesendere, OZ5ESB EDR Esbjerg
afdeling
; 05-07-2012 : De sidste 3 sekunder i en udsendelse skal være tavse, ønske fra
OZ1CWP
; 05-07-2012 : Rutine Test_for_3sec_TailMute er integreret i rutine Test_FoxTime
; 05-07-2012 : Denne ( endelige? ) version hedder 0.79
;*****
list P=PIC16F628
#include <P16F628.INC>

;*****
; Defaultvalues in EEPROM
;*****
org 2100
;-----
de d'4' ; EE adresse 2100 : foxhex2
de d'86' ; EE adresse 2101 : foxhex1 HEX 0486 = 1110 = 18.30
de d'5' ; EE adresse 2102 : TX_interval = 5 minutter
de d'25' ; EE adresse 2103 : TX_antal = 25, svarer til 2 timer
;*****
; Configuration
; Krystal 4096 kHz, Power on Timer, ingen Watchdog, XT-Oscillator
__CONFIG _PWRTE_ON & _WDT_OFF & _HS_OSC & _LVP_OFF & _BODEN_ON

;*****
; ParallelData ud til 2 x 24 linier LCD display (PortB0-7)
;
; -----U-----
; Button_C (4) input ----- RA2 |1 18| RA1-- Button_B (2) input
; Button_D (opt) input ---- RA3 |2 17| RA0-- Button_A (1) input
; 10Hz output, use pullup - RA4 |3 16| OSC1--XTAL
; +5V 22k/ICSP Enable ---- MCLR |4 15| OSC2--XTAL
; Ground ----- Vss |5 14| VDD-- +5 V
; LCD Display Enable ----- RB0 |6 13| RB7--LCD D7/ICSP
; CW Key active HI ----- RB1 |7 12| RB6--LCD D6/ICSP
; ButtonRow1/LCD RS ----- RB2 |8 11| RB5--LCD D5
; ButtonRow2/LCD RW ----- RB3 |9 10| RB4--LCD D4
;
; -----
;
; PIC16F628
;*****
;
; |-----|
; | k1,1 | k2,1 |
; |-----|
; | k1,2 | k2,2 |
; |-----|
; | k1,3 | k2,3 |
; |-----|
;
; Buttons
;
;-----
; k1,1 : nop
; k1,2 : nop
; k1,3 : nop
;-----
;
; k1 combinations in this section are used together with k2 buttons.
;

```

```

; k1,1 + k1,2      : ClockSet.   Brug k2 knapper til timer og minutter
; k1,2 + k1,3      : FoxTimeSet. Brug k2 knapper til timer og minutter
; k1,1 + k1,3      : Nulstil sekunder med k2,3
;
; k2,1             : Minutter opad
; k2,2             : Minutter nedad
; k2,1 + k2,3      : Timer opad
; k2,2 + k2,3      : Timer nedad
;-----
; k1,1 + k1,2 + k1,3 + k2,1 : Start kontinuerlig testudsendelse.
; k1,1 + k1,2 + k1,3 + k2,2 : TX-antal op
; k1,1 + k1,2 + k1,3 + k2,3 : TX-interval op
;-----
; Stop kontinuerlig testudsendelse ved at holde en vilkårlig knap,
; indtil pejlestregen er slut
;*****
INDF          equ      00h
FSR           equ      04h
CMCON         equ      1Fh      ; Comparator Control Address
INTCON        equ      0Bh      ; Interrupt control register
OPTION_REG    equ      81h      ; Option register
STATUS        equ      03h      ; Status register
TRISA         equ      85h      ; I/O control for port A
TRISB         equ      86h      ; I/O control for port B
PORTB         equ      06h      ; Address of port B
PORTA         equ      05h      ; Address of port A
PC            equ      02h      ; Program counter
;*****
; Variable.   HUSK laveste adresse ved PIC16F628 er 0x20 !!
;*****
hour2         equ      0x20
hour1         equ      0x21
min2          equ      0x22
min1          equ      0x23
sec2          equ      0x24
sec1          equ      0x25
loops        equ      0x26
loops2       equ      0x27
foxhex2      equ      0x28
foxhex1      equ      0x29
foxhour2     equ      0x2A
foxhour1     equ      0x2B
foxmin2      equ      0x2C
foxmin1      equ      0x2D
TEMPW        equ      0x2E      ; gem w under interruptrutine
STATUS_SAVE  equ      0x2F      ; gem Status under interruptrutine
switch       equ      0x30      ; resultat af inputtest
control      equ      0x31
pre_counter1 equ      0x32
pre_counter2 equ      0x33
tempmin      equ      0x34
clockhex1    equ      0x35
clockhex2    equ      0x36
ACCaHI       equ      0x37
ACCaLO       equ      0x38
ACCbHI       equ      0x39
ACCbLO       equ      0x3A
ACCaHI       equ      0x3B
ACCaLO       equ      0x3C
ACCaHI       equ      0x3D
ACCaLO       equ      0x3E
hour         equ      0x3F
min          equ      0x40
hextemp      equ      0x41
temp         equ      0x42
step         equ      0x43

```

```

TestData                equ      0x44
hextemp2                equ      0x45
hextemp1                equ      0x46
TX_count                equ      0x47
TX_rest                 equ      0x48
max2                    equ      0x49
max1                    equ      0x4A
MSD                     equ      0x4B
LSD                     equ      0x4C
akt2                    equ      0x4D
akt1                    equ      0x4E
Data2LCD                Equ      0x4F
SuffixNr                equ      0x50
knapper1                equ      0x51
knapper2                equ      0x52
clockhour2              equ      0x53
clockhour1              equ      0x54
clockmin2                equ      0x55
clockmin1               equ      0x56
readfoxhex2             equ      0x57
readfoxhex1             equ      0x58
Table_index             equ      0x59
TEMP                    equ      0x5A
LongDashCount           equ      0x5B
TX_interval              equ      0x5C      ; minutinterval mellem sender
TX_antal                equ      0x5D      ; antal sender
loop                    equ      0x5E
rest2                   equ      0x5F
rest1                   equ      0x60
sec50                   equ      0x61      ; test af sekund 57, 58, 59
sec57                   equ      0x62      ; test af sekund 57, 58, 59
;*****
;Portdefinitioner
;LCD
#define LcdE              PORTB,0
#define LcdRs             PORTB,2 ; command=0, character=1
#define LcdRw             PORTB,3 ; write to LCD=0, read from Lcd=1
;output
#define TX_key            PORTB,1
;switches
#define ButtonRow1        PORTB,2
#define ButtonRow2        PORTB,3

#define switch_A_in       PORTA,0
#define switch_B_in       PORTA,1
#define switch_C_in       PORTA,2
#define switch_D_in       PORTA,3
#define TestPin           PORTA,4

#define minutbit           control,0
#define set_optbit         control,1
#define set_timebit        control,2
#define TX_keybit          control,3      ; TX on, settes af Test_FoxTime
#define TestMode           control,4
#define adjustbit          control,5
#define Test_keybit        control,6      ; stop endeløs testmode
;*****
w                        equ      0
f                        equ      1
;*****
; prescaler = 256, intclock = 4096k / 4 = 1.024.000 cycles/sec
; prescaler x pre_count1 x pre_count2 = 1024.000
; pre_count1 x pre_count2 = 4000
; 200 x 20 = 4000
; Tidsgenereringen kan finjusteres på krystaltrimmeren.
; På TestPin PORTA,4 med pullupmodstand skal kunne måles 10.000 Hz

```

```

;-----
Constant      pre_count1_start = d'200'
Constant      pre_count2_start = d'20'
;-----
Constant      Ver1=0          ; programversions nummer
Constant      Ver2=7          ; programversions nummer
Constant      Ver3=9          ; programversions nummer
Constant      Ver4='a'        ; programversions suffix
;----- Program Starts here -----
org           0
goto         INIT
;----- Interrupt routine to update time -----
org          0x04
movwf       TEMPW           ; Save w
swapf      STATUS,w         ; Get status register into w
bcf        STATUS,RP0       ; Go to bank 0. SKAL KOMME FØR "movwf
STATUS_SAVE" !!!
movwf      STATUS_SAVE     ; Save status register
bcf        INTCON, GIE      ; disable Interupt (ikke
nødvendig?)
pre1
    decf    pre_counter1,f
    bcf     STATUS,Z
    movf    pre_counter1,f
    btfsc   STATUS,Z
    goto    TjekTestPin
    goto    intrpt_slut
TjekTestPin                                ; TestPin used for measuring
clockfrequency
    btfss   TestPin          ; correct measure is 10.00zero Hz
    goto    SetTestPin
    bcf     TestPin          ; clear pin, if it was set
    goto    pre2
SetTestPin
    bsf     TestPin          ; set pin, if it was cleared
    goto    pre2
pre2
    movlw   pre_count1_start
    movwf   pre_counter1    ; startværdi
    decf    pre_counter2,f
    bcf     STATUS,Z
    movf    pre_counter2,f
    btfsc   STATUS,Z
    goto    inc_sec1        ; der er gået 1 sekund
    goto    intrpt_slut
inc_sec1
    movlw   pre_count2_start
    movwf   pre_counter2    ; startværdi
    incf    sec1,f
    movf    sec1,w
    bcf     STATUS,Z
    sublw   d'10'
    btfsc   STATUS,Z
    goto    inc_sec2
    goto    intrpt_slut
inc_sec2
    clrf    sec1
    incf    sec2,f
    movf    sec2,w
    bcf     STATUS,Z
    sublw   d'6'
    btfsc   STATUS,Z
    goto    inc_clockhex1   ; der er gået et minut
    goto    intrpt_slut
inc_clockhex1
    clrf    sec2

```

```

        bsf          minutbit          ; bruges af TestMidnight
        bcf          STATUS,Z
        incf         clockhex1,f
        btfsc        STATUS,Z          ; overflow ?
        goto         inc_clockhex2
        goto         intrpt_slut
inc_clockhex2
        clrf         clockhex1
        incf         clockhex2,f
        goto         intrpt_slut
;-----
intrpt_slut
        bcf          INTCON,T0IF      ; clear T0IF bit
;-----
        swapf        STATUS_SAVE,w
        movwf        STATUS
        swapf        TEMPW,f
        swapf        TEMPW,w
        retfie
;----- End Interrupt Procedure -----
;*****
; TEXT TABLE      (RETLW 'X')
; PLACED IN PAGE 0, SO WE DONT HAVE TO BOTHER WITH PAGE BITS.
;*****
TABLE:      ADDWF     PCL,F            ;JUMP TO CHAR POINTED TO IN W
REG.
ST:         ;LABEL THE START OF TEXT STRINGS.

OZ7FOX:     EQU $ - ST
DT          "OZ7FOX ",0
OZ1LN:     EQU $ - ST
DT          "OZ1LN ",0
ed:        EQU $ - ST
DT          "de ",0
Indstil:   EQU $ - ST
DT          "Set ",0
Start_Tid: EQU $ - ST
DT          "Start: ",0
Brug_2.x:  EQU $ - ST
DT          "use 2.x ",0
Test:      EQU $ - ST
DT         "Test ",0
Udsendelse: EQU $ - ST
DT         "Transmission ",0
Stop:      EQU $ - ST
DT         "STOP:Press+Hold a button",0
af:        EQU $ - ST
DT         "af ",0
FOX:       EQU $ - ST
DT         "FOX ",0
Min:       EQU $ - ST
DT         "min ",0
Intern:    EQU $ - ST
DT         "Intern ",0
clock:     EQU $ - ST
DT         "clock ",0
Nulstil:   EQU $ - ST
DT         "Zero seconds : press 2.3",0
ProgMode   EQU $ - ST
DT         "Set TestMode Number Int ",0
TomSpace:  EQU $ - ST
DT         " ",0
TABLE_END:
;*****

```

```

****
; Initialisering
;*****
INIT
;*****
;-----
; Following routine from MicroChip used for 16F628
; to enable PORT to I/O functions
;-----
        CLRF      PORTA                ; Initialize PORTA by
                                         ; setting output data
latches
        MOVLW     0x07                  ; Turn comparators off and
        MOVWF     CMCON                 ; enable pins for I/O functions
        BCF       STATUS, RP1
        BSF       STATUS, RP0          ; Select Bank1
        MOVLW     0x1F
        MOVWF     TRISA                ; Set RA<4:0> inputs
;-----
        movlw     B'00001111'          ; Set RA<3:0> inputs, RA<4> output
        movwf     TRISA
        movlw     B'00000000'          ; Set RB<7:0> outputs
        movwf     TRISB
        bcf       STATUS, RP0          ; Bank 0
;-----
        call      InitLCD               ; Display initialisering
        call      LCDklar               ; write OK
        call      wait250ms
        call      wait250ms
        call      wait250ms
        call      wait250ms
        call      LCDclear              ; clear LCD, cursor home
        clrf      control
        clrf      switch
        clrf      hour2
        clrf      hour1
        clrf      min2
        clrf      min1
        clrf      sec2
        clrf      sec1

        call      readFoxTimeFromEEProm ; init værdi fra EE
        movf      readfoxhex2,w
        movwf     foxhex2
        movf      readfoxhex1,w
        movwf     foxhex1
        call      split_foxhex          ; 16 bit til timer og minutter
        call      readTX_intervalFromEEProm
        call      readTX_antalFromEEProm
        clrf      clockhex2
        clrf      clockhex1
;----- testværdi 12.00 =dec720=hex 02 D0
        movlw     0x02
        movwf     clockhex2
        movlw     0xD0
        movwf     clockhex1
;-----
        clrf      hour
        clrf      min
        movlw     pre_count1_start
        movwf     pre_counter1          ; startværdi
        movlw     pre_count2_start
        movwf     pre_counter2          ; startværdi
;-----
; Configure the prescaler for TIMER 0 in the PIC's OPTION register .
;-----

```

```

; Description of the OPTION register, from the PIC16F628 data sheet:
; bit 7: RBPU: PORTB Pull-up Enable bit
;         1 = PORTB pull-ups are disabled
;         0 = PORTB pull-ups are enabled by individual port latch values
; bit 6: INTEDG: Interrupt Edge Select bit
;         1 = Interrupt on rising edge of RB0/INT pin
;         0 = Interrupt on falling edge of RB0/INT pin
; bit 5: T0CS: TMR0 Clock Source Select bit
;         1 = Transition on RA4/T0CKI pin
;         0 = Internal instruction cycle clock (CLKOUT)
; bit 4: T0SE: TMR0 Source Edge Select bit
;         1 = Increment on high-to-low transition on RA4/T0CKI pin
;         0 = Increment on low-to-high transition on RA4/T0CKI pin
; bit 3: PSA: Prescaler Assignment bit
;         1 = Prescaler is assigned to the WDT
;         0 = Prescaler is assigned to the Timer0 module
; bit 2-0: PS2:PS0: Prescaler Rate Select bits, here shown for TMR0 :
;         000 = 1 : 2
;         ... 111 = 1 : 256
; Note: to count EVERY pulse (1 : 1) with TMR0, the prescaler
;        must be assigned to the WATCHDOG TIMER (WDT) !
;-----
        bsf     STATUS, RP0      ; Select Bank1
        movlw   B'00001111'     ; pull-up, count intern Takt, xxx - prescaler
        movwf   OPTION_REG
        bcf     STATUS, RP0      ; Select Bank0
        clrf    TMR0

        bsf     INTCON, T0IE     ; enable Timer0 interrupt
        bsf     INTCON, GIE      ; enable Interrupt

        goto    MainStart
;*****
; Hovedprogram Start
;*****
MainStart
        call    TestInput        ; aflæser knapper og vælger subroutine
        call    TestMidnight     ; resetter intern clock ved 24.00
        call    Test_FoxTime     ; afgør om det er sendetid
        call    Test_Suffix      ; vælger a,u,v,h,5
        call    split_clockhex   ; 16 bit til timer og minutter
;        call    split_foxhex     ; 16 bit til timer og minutter
        call    send_to_LCD1
        call    send_to_LCD2
;        bcf     TX_keybit        ; TEST : bcf = sender ikke
        btfsc   TX_keybit        ; må der sendes ?
        call    KeyCall_Std
        btfsc   TestMode         ; skal der sendes kontinuerligt ?
        call    KeyCall_Test
        goto    MainStart
;*****
; Hovedprogram Slut
;*****
;*****
; TestInput. Læser knapper
; Mellemresultat i switch
; Data ud i knapper1 og knapper2
;*****
TestInput
        bcf     ButtonRow1
        bcf     ButtonRow2
;-----
        bsf     ButtonRow1      ; HI = aktiv
        call    Hent_switch
        movwf   knapper1

```

```

        call    wait1ms
        bcf     ButtonRow1
;-----
        bsf     ButtonRow2                ; HI = aktiv
        call    Hent_switch
        movwf   knapper2
        call    wait1ms
        bcf     ButtonRow2
;-----
        goto    Test_Knapper
;-----
; Hent_switch    ; resultat i w
;-----
Hent_switch
    clrf       switch
    btfsc      switch_A_in
    bsf        switch,0
    btfsc      switch_B_in
    bsf        switch,1
    btfsc      switch_C_in
    bsf        switch,2
    btfsc      switch_D_in
    bsf        switch,3
    movf       switch,w
    return
;*****
; Test_Knapper udfører hvad der står i knapper1
;*****
Test_Knapper
    movf       knapper1,w
    bcf        STATUS,Z
    sublw      d'3'                ; Clock justering
    btfsc      STATUS,Z            ; er knapper1 <> 3 ? skip next
    goto       Knapper2a

    movf       knapper1,w
    bcf        STATUS,Z
    sublw      d'5'                ; Zero seconds
    btfsc      STATUS,Z            ; er knapper1 <> 5 ? skip next
    goto       Knapper2d

    movf       knapper1,w
    bcf        STATUS,Z
    sublw      d'6'                ; FoxTime justering
    btfsc      STATUS,Z            ; er knapper1 <> 6 ? skip next
    goto       Knapper2b

    movf       knapper1,w
    bcf        STATUS,Z
    sublw      d'7'                ; Prog+test Mode (alle knapper trykket)
    btfsc      STATUS,Z            ; er knapper1 <> 7 ? skip next
    goto       Knapper2c

    movf       knapper1,w
    bcf        STATUS,Z
    btfsc      STATUS,Z            ; er knapper1 = 0 ? skip next
    goto       Knapper2c
    return
;*****
; Knapper2 udfører hvad der står i knapper2
;*****
Knapper2a                                ; Clock justering
    bcf        TX_keybit            ; så sendes der ikke under indstillinger
    call       SetClockToLCD1

    movf       knapper2,w

```

```

    bcf          STATUS,Z
    sublw       d'1'
    btfsc       STATUS,Z      ;er knapper1 <> 1 ? skip next
    goto        IncClockTimeSlow

    movf        knapper2,w
    bcf          STATUS,Z
    sublw       d'2'
    btfsc       STATUS,Z      ;er knapper1 <> 2 ? skip next
    goto        DecClockTimeSlow

    movf        knapper2,w
    bcf          STATUS,Z
    sublw       d'5'
    btfsc       STATUS,Z      ;er knapper1 <> 5 ? skip next
    goto        IncClockTimeFast

    movf        knapper2,w
    bcf          STATUS,Z
    sublw       d'6'
    btfsc       STATUS,Z      ; er knapper1 <> 6 ? skip next
    goto        DecClockTimeFast
;-----
    goto        TestInput
;*****
Knapper2b                                ; FoxTime justering
    bcf          TX_keybit           ; så sendes der ikke under indstillinger
    call        SetFoxTimeToLCD2

    movf        knapper2,w
    bcf          STATUS,Z
    sublw       d'1'
    btfsc       STATUS,Z      ; er knapper1 <> 1 ? skip next
    goto        IncFoxTimeSlow

    movf        knapper2,w
    bcf          STATUS,Z
    sublw       d'2'
    btfsc       STATUS,Z      ; er knapper1 <> 2 ? skip next
    goto        DecFoxTimeSlow

    movf        knapper2,w
    bcf          STATUS,Z
    sublw       d'5'
    btfsc       STATUS,Z      ; er knapper1 <> 5 ? skip next
    goto        IncFoxTimeFast

    movf        knapper2,w
    bcf          STATUS,Z
    sublw       d'6'
    btfsc       STATUS,Z      ; er knapper1 <> 6 ? skip next
    goto        DecFoxTimeFast
;-----
    goto        TestInput
;*****
Knapper2c                                ; Prog+test Mode (alle knapper trykket)
    call        ProgrModeToLCD2
    call        send_to_LCD1
    bcf          TX_keybit           ; så sendes der ikke under indstillinger

    movf        knapper2,w
    bcf          STATUS,Z
    sublw       d'1'              ; TestMode
    btfsc       STATUS,Z      ; er knapper1 <> 4 ? skip next
    goto        KeyCall_Test

```

```

    movf    knapper2,w
    bcf     STATUS,Z
    sublw   d'2'
    btfsc   STATUS,Z          ; er knapper1 <> 2 ? skip next
    call    IncTX_antal

    movf    knapper2,w
    bcf     STATUS,Z
    sublw   d'4'
    btfsc   STATUS,Z          ; er knapper1 <> 1 ? skip next
    call    IncTXinterval

    call    send_to_LCD1
    call    wait100ms
    goto    TestInput
;*****
Knapper2d                                ; Zero seconds
    call    NulstilToLCD1
    call    send_to_LCD2
    bcf     TX_keybit          ; så sendes der ikke under indstillinger

    movf    knapper2,w
    bcf     STATUS,Z
    sublw   d'4'              ; Nulstil sekunder
    btfsc   STATUS,Z          ; er knapper1 <> 4 ? skip next
    goto    ZeroSec
    call    send_to_LCD2
    call    wait100ms
    goto    TestInput
;*****
ZeroSec
    clrf    sec1
    clrf    sec2
    goto    TestInput
;*****
; SetInternClock - rutiner
;*****
IncClockTimeSlow
    movlw   d'1'
    goto    IncClockTime
;*****
IncClockTimeFast
    movlw   d'60'
    goto    IncClockTime
;-----
IncClockTime
    addwf   clockhex1,f      ; læg minutter til Lbyte
    btfsc   STATUS,C          ; er der ikke carry, skip næste
    incf    clockhex2,f      ; læg 1 til Hbyte
;-----test_clockmax-----
    movf    clockhex2,w
    bcf     STATUS,C
    sublw   d'5'
    btfsc   STATUS,C          ; større end 6, skip næste
    goto    videre
    goto    ZeroClockTime
;-----
videre
    movf    clockhex2,w
    bcf     STATUS,C
    sublw   d'4'
    btfsc   STATUS,C          ; større end 5, skip næste
    goto    ClockTimeEnd
;-----
    movf    clockhex1,w
    bcf     STATUS,C

```

```

        sublw    d'159'                ; (5 x 60)+160=1440=1 døgn
        btfsc    STATUS,C              ; større end 159, skip næste
        goto     ClockTimeEnd
;-----
ZeroClockTime
        clrf     clockhex2              ; set HIbyte og LObyte til 0
        clrf     clockhex1
        goto     ClockTimeEnd
;*****
DecClockTimeSlow
        movlw    d'1'
        goto     DecClockTime
;*****
DecClockTimeFast
        movlw    d'60'
        goto     DecClockTime
;-----
DecClockTime
        subwf     clockhex1,f           ; træk minutter fra
        btfsc     STATUS,C              ; skip næste, hvis resultat er mindre end 0
        goto     ClockTimeEnd
;-----
        movlw    d'1'
        subwf     clockhex2,f           ; træk 1 fra
        btfsc     STATUS,C              ; skip næste, hvis resultat er mindre end 0
        goto     ClockTimeEnd
;-----
        movlw    0x05                   ; sæt HIbyte og LObyte til 23.59
        movwf     clockhex2
        movlw    0x9F                   ; hex 059F : 5x60 + 159 = 1439 = 23.59
        movwf     clockhex1
        goto     ClockTimeEnd
;-----
ClockTimeEnd
        call      split_clockhex
        call      send_to_LCD2
        call      wait250ms
        goto     TestInput
;*****
; SetFoxTime - rutiner
;*****
IncFoxTimeSlow
        movlw    d'1'
        goto     IncFoxTime
;*****
IncFoxTimeFast
        movlw    d'60'
        goto     IncFoxTime
;-----
IncFoxTime
        addwf     foxhex1,f             ; læg minutter til LObyte
        btfsc     STATUS,C              ; er der ikke carry, skip næste
        incf      foxhex2,f             ; læg 1 til HIbyte
        goto     TestFoxMax
;-----TestFoxMax-----
TestFoxMax
        movf      foxhex2,w
        bcf        STATUS,C
        sublw     d'5'
        btfsc     STATUS,C              ; større end 6, skip næste
        goto     videre2
        goto     ZeroFoxTime
;-----
videre2
        movf      foxhex2,w
        bcf        STATUS,C

```

```

        sublw    d'4'
        btfsc    STATUS,C                ; større end 5, skip næste
        goto     FoxTimeEnd
;-----
        movf     foxhex1,w
        bcf      STATUS,C
        sublw    d'159'                  ; (5 x 60)+159=1439=1 døgn
        btfsc    STATUS,C                ; større end 159, skip næste
        goto     FoxTimeEnd
;-----
ZeroFoxTime
        clrf     foxhex2                  ; set HIbyte og LObyte til 0
        clrf     foxhex1
        goto     FoxTimeEnd
;*****
DecFoxTimeSlow
        movlw    d'1'
        goto     DecFoxTime
;*****
DecFoxTimeFast
        movlw    d'60'
        goto     DecFoxTime
;-----
DecFoxTime
        subwf     foxhex1,f              ; træk minutter fra
        btfsc     STATUS,C              ; skip næste, hvis resultat er mindre end 0
        goto     FoxTimeEnd
;-----
        movlw     d'1'
        subwf     foxhex2,f              ; træk 1 fra
        btfsc     STATUS,C              ; skip næste, hvis resultat er mindre end 0
        goto     FoxTimeEnd
;-----
        movlw     0x05                   ; sæt HIbyte og LObyte til 23.59
        movwf     foxhex2
        movlw     0x9F                   ; hex 059F : 5x60 + 159 = 1439 = 23.59
        movwf     foxhex1
        goto     FoxTimeEnd
;-----
FoxTimeEnd
        call      writeFoxTime2EEProm
        call      wait100ms              ; ellers går der kage i det ?? 120808
        call      testFoxTimeFromEEProm
;-----
        call      split_foxhex          ; for at se data efter write
        call      send_to_LCD1
        goto     TestInput
;*****
; Indstil TXinterval. Min 2 og Max 10 minutter
;*****
IncTXInterval
        incf      TX_interval,f
        movf      TX_interval,w
        bcf       STATUS,C
        sublw     d'10'
        btfsc     STATUS,C              ; større end 10, skip næste
        goto     writeTX_interval2EEProm
        movlw     d'2'
        movwf     TX_interval
        goto     writeTX_interval2EEProm
;-----
writeTX_interval2EEProm
        movlw     d'2'                  ; EE adresse 2102
        bsf       STATUS, RP0           ; Bank 1
        movwf     EEADR                 ; writeaddress in EEPROM
        bcf       STATUS, RP0           ; Bank 0

```

```

    movf    TX_interval,w
    bsf     STATUS, RP0                ; Bank 1
    movwf   EEDATA
    bcf     STATUS, RP0                ; Bank 0
    call    EE_write
    call    wait100ms
    call    readTX_intervalFromEEPROM
    return
;*****
; Indstil TX_antal. Min 1 og Max 37 udsendelser
;*****
IncTX_antal
    incf    TX_antal,f
    movf    TX_antal,w
    bcf     STATUS,C
    sublw   d'37'
    btfsc   STATUS,C                  ; større end 37, skip næste
    goto    writeTX_antal2EEPROM
    movlw   d'1'
    movwf   TX_antal
    goto    writeTX_antal2EEPROM
;-----
writeTX_antal2EEPROM
    movlw   d'3'                      ; EE adresse 2103
    bsf     STATUS, RP0                ; Bank 1
    movwf   EEADR                     ; writeaddress in EEPROM
    bcf     STATUS, RP0                ; Bank 0
    movf    TX_antal,w
    bsf     STATUS, RP0                ; Bank 1
    movwf   EEDATA
    bcf     STATUS, RP0                ; Bank 0
    call    EE_write
    call    wait100ms
    call    readTX_antalFromEEPROM
    return
;*****
; Skriv FoxTime til EEPROM
; EEADR holder adressen, EEDATA holder data
;*****
writeFoxTime2EEPROM
    movlw   d'0'                      ; EE adresse 2100
    bsf     STATUS, RP0                ; Bank 1
    movwf   EEADR                     ; writeaddress in EEPROM
    bcf     STATUS, RP0                ; Bank 0
    movf    foxhex2,w
    bsf     STATUS, RP0                ; Bank 1
    movwf   EEDATA
    bcf     STATUS, RP0                ; Bank 0
    call    EE_write
    call    wait100ms
;-----
    movlw   d'1'                      ; EE adresse 2101
    bsf     STATUS, RP0                ; Bank 1
    movwf   EEADR                     ; writeaddress in EEPROM
    bcf     STATUS, RP0                ; Bank 0
    movf    foxhex1,w
    bsf     STATUS, RP0                ; Bank 1
    movwf   EEDATA
    bcf     STATUS, RP0                ; Bank 0
    call    EE_write
    call    wait100ms
    return
;*****
; hent data from EEPROM til variable, output readfoxhex1+2
;*****
readFoxTimeFromEEPROM

```

```

        movlw    d'0'                ; EEprom 2100 :foxhex2 starttid
        call     EE_read
        movwf    readfoxhex2
        movlw    d'1'                ; EEprom 2101 :foxhex1 starttid
        call     EE_read
        movwf    readfoxhex1
        return
;*****
; henter readFoxTimedata fra EEprom og sammenligner med
; variable FoxTimeData. Hvis ikke ens kaldes write-rutine.
;*****
testFoxTimeFromEEprom
        call     readFoxTimeFromEEprom
testreadfoxhex2
        movf     readfoxhex2,w
        bcf      STATUS,Z
        subwf    foxhex2,w
        btfsc    STATUS,Z            ; er de forskellige ? så skip next
        goto     testreadfoxhex1
        goto     writeError
testreadfoxhex1
        movf     readfoxhex1,w
        bcf      STATUS,Z
        subwf    foxhex1,w
        btfsc    STATUS,Z            ; er de forskellige ? så skip next
        return
        goto     writeError
;-----
writeError
        call     send_to_LCD1
        call     LCDerrorMsg
        call     writeFoxTime2EEprom
        return
;*****
; readTX_intervalFromEEprom og readTX_antalFromEEprom læser EE
; og fører resultatet til TX_interval og TX_antal henholdsvis.
;*****
readTX_intervalFromEEprom
        movlw    d'2'                ; EEprom 2102 :TX interval
        call     EE_read
        movwf    TX_interval ; minutinterval mellem sendinger
        return
;-----
readTX_antalFromEEprom
        movlw    d'3'                ; EEprom 2103 : TX antal
        call     EE_read              ; result in w
        movwf    TX_antal             ; antal udsendelser
        return
;*****
; Test_Suffix tester, om minut-enerne er 0,5/1,6/2,7/3,8 eller 4,9
; output suffix 1,2,3,4 og 5 svarer til ræv a,u,v,4 og 5
;*****
Test_Suffix
        movf     foxmin1,w
        movwf    SuffixNr
        movlw    d'5'
        bcf      STATUS,Z
        subwf    SuffixNr,w
        btfsc    STATUS,C            ;er SuffixNr > 4 ? skip next
        goto     Test_Suffix_slut
        incf     SuffixNr,f          ;lægger 1 til minuttallet
        return
;-----
Test_Suffix_slut                ;SuffixNr er større end 4
        movlw    d'4'                ;så vi trækker 5 fra
        subwf    SuffixNr,f

```

```

        return
;*****
; Test_FoxTime
;*****
Test_FoxTime
        bcf                TX_keybit                ;clear TX_keybit før test af
sendetid
;-----
Test_for_3sec_TailMute                ; sidste 3 sekunder skal være tavse.

Test_50sec
        movf                sec2,w
        movwf               sec50
        movlw               d'5'
        bcf                STATUS,Z
        subwf               sec50,w
        btfscl               STATUS,C                ;er sec2 > 4 ? skip next
        goto               Test_57sec
        goto               Test_FoxTime1            ;sekunder er under 50, fortsæt

Test_57sec
        movf                sec1,w
        movwf               sec57
        movlw               d'7'
        bcf                STATUS,Z
        subwf               sec57,w
        btfscl               STATUS,C                ;er sec1 > 6 ? skip next
        return              ;sekunder er over 57, TX_keybit=0
        goto               Test_FoxTime1            ;sekunder er under 57, fortsæt
;-----
Test_FoxTime1
        movf                TX_antal,w
        movwf               TX_count
        movf                foxhex2,w                ; Foxtid
        movwf               hextemp2
        movf                foxhex1,w
        movwf               hextemp1

Test_FoxTime2
        movf                hextemp2,w                ; tempFoxtid
        movwf               ACCbHI
        movf                hextemp1,w
        movwf               ACCbLO

        movf                clockhex2,w                ; urtid
        movwf               ACCaHI
        movf                clockhex1,w
        movwf               ACCaLO

firstByteTest
;*****
; D_sub : Double Precision Subtraction ( ACCb - ACCa -> ACCb )
;-----
        call               D_sub
        bcf                STATUS,Z
        movf                ACCbHI,f                ; er den 0 ?
        btfscl               STATUS,Z
        goto               nextByteTest
        goto               add5min

nextByteTest
        bcf                STATUS,Z
        movf                ACCbLO,f                ; er den 0 ?
        btfscl               STATUS,Z
        goto               set_TX_keybit            ; Hi og LO-byte ens, der skal sendes
        goto               add5min
;-----
add5min
        decf                TX_count,f
        bcf                STATUS,Z

```

```

        movf    TX_count,f                ; er den 0 ?
        btfsc   STATUS,Z
        return                                ; ja den er nul, ud
        movlw   0x00
        movwf   ACCaHI
        movf    TX_interval,w
        movwf   ACCaLO
;-----
        movf    hextemp2,w                ; tempFoxtid
        movwf   ACCbHI
        movf    hextemp1,w
        movwf   ACCbLO
;*****
; Double Precision Addition ( ACCb + ACCa -> ACCb ) call D_add
;-----
        call    D_add                      ; TX-interval add til ACCb
        movf    ACCbHI,w
        movwf   hextemp2
        movf    ACCbLO,w
        movwf   hextemp1
        goto    Test_FoxTime2      ; test en gang til med + 5 minutter
;-----
set_TX_keybit
        bsf     TX_keybit
        return
;*****
; split_clockhex deler clockhex2+clockhex1 med 60.
; input clockhex2 og clockhex1,
; output clock-hour2, -hour1, -min2 og -min1
;*****
split_clockhex
        movf    clockhex2,w
        movwf   ACCbHI
        movf    clockhex1,w
        movwf   ACCbLO
        clrf    hour2
        clrf    hour1
        clrf    min2
        clrf    min1
        call    split_to_hours_and_minutes
        movf    hour2,w
        movwf   clockhour2
        movf    hour1,w
        movwf   clockhour1
        movf    min2,w
        movwf   clockmin2
        movf    min1,w
        movwf   clockmin1
        return
;*****
; split_foxhex deler foxhex2+foxhex1 med 60.
; input foxhex2 og foxhex1,
; output fox-hour2, -hour1, -min2 og -min1
;*****
split_foxhex
        call    readFoxTimeFromEEPROM
        movf    readfoxhex2,w
        movwf   ACCbHI
        movf    readfoxhex1,w
        movwf   ACCbLO
        clrf    hour2
        clrf    hour1
        clrf    min2
        clrf    min1
        call    split_to_hours_and_minutes
        movf    hour2,w

```

```

        movwf    foxhour2
        movf     hour1,w
        movwf    foxhour1
        movf     min2,w
        movwf    foxmin2
        movf     min1,w
        movwf    foxmin1
        return
;*****
; split_to_hours_and_minutes : ACCbHI - ACCbLO -> timer og minutter
; 1 : division med 60 = hextimer, divideres med 10 = hour2 og hour1
; 2 : rest2 og rest1 = hexminutter, divideres med 10 = min2 og min1
;*****
split_to_hours_and_minutes
        movlw    d'0'
        movwf    ACCaHI
        movlw    d'60'
        movwf    ACCaLO
        call     D_div
        movf     ACCcHI,w          ; rest skal deles til minutter senere
        movwf    rest2
        movf     ACCcLO,w          ; rest skal deles til minutter senere
        movwf    rest1
;-----
        movlw    d'0'
        movwf    ACCaHI
        movlw    d'10'
        movwf    ACCaLO
        call     D_div
        movf     ACCbLO,w
        movwf    hour2
        movf     ACCcLO,w
        movwf    hour1
;-----
        movf     rest2,w          ; minutter i HEX
        movwf    ACCbHI
        movf     rest1,w          ; minutter i HEX
        movwf    ACCbLO
        movlw    d'0'
        movwf    ACCaHI
        movlw    d'10'
        movwf    ACCaLO
        call     D_div
        movf     ACCbLO,w
        movwf    min2
        movf     ACCcLO,w
        movwf    min1
        return
;*****
;          AN526    Double Precision Division    ( 16/16 -> 16 )
;          ( Optimized for Code Size : Looped Code )
;*****
;          ( ACCb/ACCa -> ACCb with remainder in ACCc ) : 16 bit output
; Quotient in ACCb (ACCbHI,ACCbLO), Remainder in ACCc (ACCcHI,ACCcLO).
;
;          (a) Load the Denominator in location ACCaHI & ACCaLO ( 16 bits )
;          (b) Load the Numerator in location ACCbHI & ACCbLO ( 16 bits )
;          (c) CALL D_div
;          (d) The 16_bit result is in location ACCbHI & ACCbLO
;          (e) The 16 bit Remainder is in locations ACCcHI & ACCcLO
;*****
D_div
        call     setup
        clrf     ACCcHI
        clrf     ACCcLO
dloop

```

```

        bcf      STATUS,C
        rlf      ACCdLO, F
        rlf      ACCdHI, F
        rlf      ACCcLO, F
        rlf      ACCcHI, F
        movf     ACCaHI,W
        subwf    ACCcHI,W          ; check if a>c
        btfss    STATUS,Z
        goto     nochk
        movf     ACCaLO,W
        subwf    ACCcLO,W          ; if msb equal then check lsb
nochk
        btfss    STATUS,C          ; carry set if c>a
        goto     nogo
        movf     ACCaLO,W          ; c-a into c
        subwf    ACCcLO, F
        btfss    STATUS,C
        decf     ACCcHI, F
        movf     ACCaHI,W
        subwf    ACCcHI, F
        bsf      STATUS,C          ; shift a 1 into b (result)
nogo
        rlf      ACCbLO, F
        rlf      ACCbHI, F
        decfsz   loop, F          ; loop untill all bits checked
        goto     dloop
        retlw    0
;*****
setup
        movlw    .16              ; for 16 shifts
        movwf    loop
;-----
        movf     ACCbHI,W          ; move ACCb to ACCd
        movwf    ACCdHI
        movf     ACCbLO,W
        movwf    ACCdLO
;-----
        clrf     ACCbHI
        clrf     ACCbLO
        retlw    0
;*****
; Double Precision Subtraction ( ACCb - ACCa -> ACCb )
;-----
D_sub
        call     neg_A ; At first negate ACCa; Then add
;
;*****
; Double Precision Addition ( ACCb + ACCa -> ACCb ) call D_add
;-----
D_add
        movf     ACCaLO,W
        addwf    ACCbLO, F ; add lsb
        btfsc    STATUS,C ; add in carry
        incf     ACCbHI, F
        movf     ACCaHI,W
        addwf    ACCbHI, F ; add msb
        retlw    0
;-----
neg_A
        comf     ACCaLO, F ; negate ACCa ( -ACCa -> ACCa )
        incf     ACCaLO, F
        btfsc    STATUS,Z
        decf     ACCaHI, F
        comf     ACCaHI, F
        retlw    0

```

```

;*****
; Tidsforsinkelse i 1 msec gange loops.
; 4096kHz extern frekvens betyder 1024kHz intern takt
; 1 ms varer 1024 instrukser
; 92 sløjfer a 11 instrukser er 1012 instrukser = 0.9775 ms
; resterende 12 instrukser til indgang og udgang
;*****
wait_ms_x_loops
top
    movlw    .92                ; timing adjustment
    movwf    loops2
top2
    nop                        ; inner loop start (loops2)
    nop
    nop
    nop
;    nop
;    nop
;    nop
    nop
    nop
    nop
    decfsz   loops2, F          ; inner loops complete?
    goto     top2              ; no, go again

    decfsz   loops, F           ; outer loops complete?
    goto     top               ; no, go again
    retlw    0                 ; yes, return with 0 in w
;*****
wait1ms
    movlw    D'1'              ; 1 ms Pause
    movwf    loops
    call     wait_ms_x_loops
    return

wait2ms
    movlw    D'2'              ; 2 ms Pause
    movwf    loops
    call     wait_ms_x_loops
    return

wait50ms
    movlw    D'50'             ; 50 ms Pause
    movwf    loops
    call     wait_ms_x_loops
    return

wait100ms
    movlw    D'100'            ; 100 ms Pause
    movwf    loops
    call     wait_ms_x_loops
    return

wait250ms
    movlw    D'250'            ; 250 ms Pause
    movwf    loops
    call     wait_ms_x_loops
    return
;-----
dit
    movlw    D'70'             ; 85 ms Pause
    movwf    loops
    call     wait_ms_x_loops
    return

```

```

sendDit
    bsf          TX_key
    call    dit
    bcf          TX_key
    call    send_Space
    return

send_Dash
    bsf          TX_key
    call    dit
    call    dit
    call    dit
    bcf          TX_key
    btfss    Test_keybit          ; er vi i TestMode, så skip neste ?
    call    Test_FoxTime
    btfss    TX_keybit            ; er det sendetid, så skip neste ?
    goto    MainStart            ; afbrydelse, hvis minuttet er gået.
    return

sendDash
    call    send_Dash
    call    send_Space
    return

send_Space                                ; mellemrum mellem tegnelementer
    bcf          TX_key
    call    dit
    bcf          TX_key
    return

sendSpace                                ; ordmellemrum
    bcf          TX_key
    call    dit
    call    dit
    bcf          TX_key
    return

sendLongDash    ; tid tilpasses, så call og pejlestreg er ca 16 sec
    movlw    d'110'          ; (110) antal prikker til pejlestreg
    movwf    LongDashCount
    bsf          TX_key

sendLongDash2
    call    dit
    btfss    Test_keybit          ; er vi i TestMode, så skip neste ?
    call    Test_FoxTime
    btfss    TX_keybit            ; er det sendetid, så skip neste ?
    goto    MainStart            ; afbrydelse, hvis minuttet er gået.
    decfsz   LongDashCount,f
    goto    sendLongDash2
    call    sendSpace
    return

;*****
; KeyCall sender morse på PortA,0
;*****
KeyCall_Test
    call    TestTransmitToLCD1
    call    TestTransmitToLCD2
    bsf          TX_keybit
    bsf          Test_keybit
    call    KeyCall

    call    TestInput

    bcf          STATUS,Z
    movf    knapper1,f
    btfss    STATUS,Z          ; er knapper1 = 0 ? skip next
    goto    ClearTestMode

    bcf          STATUS,Z
    movf    knapper2,f

```

```

        btfss    STATUS,Z          ; er knapper2 = 0 ? skip next
        goto    ClearTestMode
        goto    KeyCall_Test

ClearTestMode
        bcf      Test_keybit
        bcf      TX_keybit
        goto    MainStart
;-----
KeyCall_Std
        call     TransmitLCD1
        goto    KeyCall

KeyCall
        call     sendDash          ; 0
        call     sendDash
        call     sendDash
        call     sendSpace

        call     sendDash          ; Z
        call     sendDash
        call     sendDit
        call     sendDit
        call     sendSpace

        call     sendDash          ; 7
        call     sendDash
        call     sendDit
        call     sendDit
        call     sendDit
        call     sendSpace

        call     sendDit           ; F
        call     sendDit
        call     sendDash
        call     sendDit
        call     sendSpace

        call     sendDash          ; 0
        call     sendDash
        call     sendDash
        call     sendSpace

        call     sendDash          ; X
        call     sendDit
        call     sendDit
        call     sendDash
        call     sendSpace

        call     send_Space
        call     sendSpace

test_1
        movf     SuffixNr,w
        bcf      STATUS,Z
        sublw    d'1'
        btfss    STATUS,Z
        goto     test_2
        call     sendDit           ; A
        call     sendDash
        call     sendSpace

test_2
        movf     SuffixNr,w
        bcf      STATUS,Z
        sublw    d'2'
        btfss    STATUS,Z
        goto     test_3

```

```

        call    sendDit
        call    sendDit          ; U
        call    sendDash
        call    sendSpace
test_3
        movf    SuffixNr,w
        bcf     STATUS,Z
        sublw   d'3'
        btfss   STATUS,Z
        goto    test_4
        call    sendDit
        call    sendDit
        call    sendDit          ; V
        call    sendDash
        call    sendSpace
test_4
        movf    SuffixNr,w
        bcf     STATUS,Z
        sublw   d'4'
        btfss   STATUS,Z
        goto    test_5
        call    sendDit
        call    sendDit
        call    sendDit
        call    sendDit          ; H
test_5
        movf    SuffixNr,w
        bcf     STATUS,Z
        sublw   d'5'
        btfss   STATUS,Z
        goto    test_6
        call    sendDit
        call    sendDit
        call    sendDit
        call    sendDit          ; 5
        call    sendDit
test_6
                                ;ingen højere
        call    sendSpace
        call    sendLongDash
        call    sendSpace

        return
;*****
; Binary To BCD Conversion Routine
;
; This routine converts the 8 bit binary number in the W Register
; to a 2 digit BCD number.
; The least significant digit is returned in location LSD and
; the most significant digit is returned in location MSD.
;
; Performance :
; Program Memory : 10
; Clock Cycles : 81 (worst case when W = 63 Hex )
; ( i.e max Decimal number 99 )
;
; Program: BIN8BCD.ASM
; Revision Date:
; 1-13-97 Compatibility with MPASMWIN 1.40
;
;*****
Bin8BCD
        clrf    MSD
        movwf   LSD
gtenth
        movlw   .10
        subwf   LSD,W

```

```

        BTFSS     STATUS, C
        goto      over
        movwf     LSD
        incf      MSD, F
        goto      gtenth
over
        retlw     0
;*****
; LCD rutiner henter tekststrenger i Tabel på page 1
;*****
LCDerrorMsg
        call      LCDline2
        movlw     'E'
        movwf     Data2LCD
        call      SendData2LCD
        movlw     'R'
        movwf     Data2LCD
        call      SendData2LCD
        movlw     'R'
        movwf     Data2LCD
        call      SendData2LCD
        movlw     'O'
        movwf     Data2LCD
        call      SendData2LCD
        movlw     'R'
        movwf     Data2LCD
        call      SendData2LCD
        movlw     ' '
        movwf     Data2LCD
        call      SendData2LCD
        call      wait250ms
        call      wait250ms
        call      wait250ms
        call      wait250ms
        call      wait250ms
        call      wait250ms
        call      wait250ms
        call      wait250ms
        call      wait250ms
        return
;-----
LCDklar
        call      LCDline1
        movlw     OZ7FOX
        call      LCD_TEXT
        movlw     ed
        call      LCD_TEXT
        movlw     OZ1LN
        call      LCD_TEXT
        movlw     TomSpace
        call      LCD_TEXT

        movlw     Ver1
        andlw     H'0F'
        addlw     B'00110000'
        movwf     Data2LCD
        call      SendData2LCD

        movlw     '.'
        movwf     Data2LCD
        call      SendData2LCD

        movlw     Ver2
        andlw     H'0F'
        addlw     B'00110000'
        movwf     Data2LCD
        call      SendData2LCD

```

```

        movlw    Ver3
        andlw    H'0F'
        addlw    B'00110000'
        movwf    Data2LCD
        call     SendData2LCD
        movlw    Ver4
        movwf    Data2LCD
        call     SendData2LCD
        movlw    TomSpace
        call     LCD_TEXT
        movlw    TomSpace
        call     LCD_TEXT
        return
;-----
SetFoxTimeToLCD2
        call     LCDline2
        movlw    Indstil
        call     LCD_TEXT
        movlw    FOX
        call     LCD_TEXT
        movlw    Start_Tid
        call     LCD_TEXT
        movlw    Brug_2.x
        goto     LCD_TEXT
;-----
SetClockToLCD1
        call     LCDline1
        movlw    Indstil
        call     LCD_TEXT
        movlw    Intern
        call     LCD_TEXT
        movlw    clock
        call     LCD_TEXT
        movlw    Brug_2.x
        goto     LCD_TEXT
;-----
NulstilToLCD1
        call     LCDline1
        movlw    Nulstil
        goto     LCD_TEXT
;-----
ProgrModeToLCD2
        call     LCDline2
        movlw    ProgMode
        goto     LCD_TEXT
;-----
TestTransmitToLCD1
        call     LCDline1
        movlw    Test
        call     LCD_TEXT
        movlw    Udsendelse
        call     LCD_TEXT
        movlw    OZ7FOX
        call     LCD_TEXT
        movlw    TomSpace
        call     LCD_TEXT
        movf     SuffixNr,w
        andlw    H'0F'
        addlw    B'00110000'
        movwf    Data2LCD
        call     SendData2LCD
;-----
TestTransmitToLCD2
        call     LCDline2
        movlw    Stop
        call     LCD_TEXT

```

```

        return
;-----
TransmitLCD1
    call    LCDline1

    movlw   Udsendelse
    call    LCD_TEXT

    movf    TX_count,w
    subwf   TX_antal,w
    movwf   TX_rest
    incf    TX_rest,w

    call    Bin8BCD
    movf    MSD,w
    movwf   akt2
    movf    LSD,w
    movwf   akt1

    movf    akt2,w
    andlw   H'0F'
    addlw   B'00110000'
    movwf   Data2LCD
    call    SendData2LCD      ; and print in display

    movf    akt1,w
    andlw   H'0F'
    addlw   B'00110000'
    movwf   Data2LCD
    call    SendData2LCD      ; and print in display

    movlw   TomSpace
    call    LCD_TEXT

    movlw   af
    call    LCD_TEXT

    movf    TX_antal,w
    call    Bin8BCD
    movf    MSD,w
    movwf   max2
    movf    LSD,w
    movwf   max1

    movf    max2,w
    andlw   H'0F'
    addlw   B'00110000'
    movwf   Data2LCD
    call    SendData2LCD      ; and print in display

    movf    max1,w
    andlw   H'0F'
    addlw   B'00110000'
    movwf   Data2LCD
    call    SendData2LCD      ; and print in display

    movlw   TomSpace
    call    LCD_TEXT
    movlw   TomSpace
    call    LCD_TEXT
    movlw   '.'
    movwf   Data2LCD
    call    SendData2LCD

    return
;-----

```

```

send_to_LCD1
    call    LCDline1

    movf    foxhour2,w
    andlw   H'0F'
    addlw   B'00110000'
    movwf   Data2LCD
    call    SendData2LCD        ; and print in display

    movf    foxhour1,w
    andlw   H'0F'
    addlw   B'00110000'
    movwf   Data2LCD
    call    SendData2LCD        ; and print in display

    movlw   ':'
    movwf   Data2LCD
    call    SendData2LCD

    movf    foxmin2,w
    andlw   H'0F'
    addlw   B'00110000'
    movwf   Data2LCD
    call    SendData2LCD        ; and print in display

    movf    foxmin1,w
    andlw   H'0F'
    addlw   B'00110000'
    movwf   Data2LCD
    call    SendData2LCD        ; and print in display

    movlw   TomSpace
    call    LCD_TEXT

    movlw   FOX
    call    LCD_TEXT

    movf    SuffixNr,w
    andlw   H'0F'
    addlw   B'00110000'
    movwf   Data2LCD
    call    SendData2LCD

    movlw   TomSpace
    call    LCD_TEXT

    movlw   TomSpace
    call    LCD_TEXT

    movf    TX_antal,w
    call    Bin8BCD
    movf    MSD,w
    movwf   max2
    movf    LSD,w
    movwf   max1

    movf    max2,w
    andlw   H'0F'
    addlw   B'00110000'
    movwf   Data2LCD
    call    SendData2LCD        ; and print in display

    movf    max1,w
    andlw   H'0F'
    addlw   B'00110000'
    movwf   Data2LCD

```

```

        call    SendData2LCD      ; and print in display

        movlw   TomSpace
        call    LCD_TEXT

        movlw   'x'
        movwf   Data2LCD
        call    SendData2LCD

        movlw   TomSpace
        call    LCD_TEXT

        movf    TX_interval,w
        call    Bin8BCD
        movf    MSD,w
        movwf   akt2
        movf    LSD,w
        movwf   akt1

        movf    akt2,w
        andlw   H'0F'
        addlw   B'00110000'
        movwf   Data2LCD
        call    SendData2LCD      ; and print in display

        movf    akt1,w
        andlw   H'0F'
        addlw   B'00110000'
        movwf   Data2LCD
        call    SendData2LCD      ; and print in display

        movlw   TomSpace
        call    LCD_TEXT

        movlw   Min                      ; står for minutter
        call    LCD_TEXT

        return
;-----
send_to_LCD2
        call    LCDline2

;        movf    knapper1, w      ;test
;        andlw   H'0F'
;        addlw   B'00110000'
;        movwf   Data2LCD
;        call    SendData2LCD      ; and print in display

;        movlw   TomSpace          ;test
;        call    LCD_TEXT

;        movf    knapper2, w      ;test
;        andlw   H'0F'
;        addlw   B'00110000'
;        movwf   Data2LCD
;        call    SendData2LCD      ; and print in display

;        movlw   TomSpace          ;test
;        call    LCD_TEXT

        movlw   Intern
        call    LCD_TEXT
        movlw   clock
        call    LCD_TEXT
        movf    clockhour2, w
        andlw   H'0F'

```

```

    addlw    B'00110000'
    movwf    Data2LCD
    call     SendData2LCD      ; and print in display

    movf     clockhour1, w
    andlw    H'0F'
    addlw    B'00110000'
    movwf    Data2LCD
    call     SendData2LCD      ; and print in display

    movlw    ':'
    movwf    Data2LCD
    call     SendData2LCD

    movf     clockmin2, w
    andlw    H'0F'
    addlw    B'00110000'
    movwf    Data2LCD
    call     SendData2LCD      ; and print in display

    movf     clockmin1, w
    andlw    H'0F'
    addlw    B'00110000'
    movwf    Data2LCD
    call     SendData2LCD      ; and print in display

    movlw    ':'
    movwf    Data2LCD
    call     SendData2LCD

    movf     sec2, w
    andlw    H'0F'
    addlw    B'00110000'
    movwf    Data2LCD
    call     SendData2LCD      ; and print in display

    movf     sec1, w
    andlw    H'0F'
    addlw    B'00110000'
    movwf    Data2LCD
    call     SendData2LCD      ; and print in display

    movlw    TomSpace
    call     LCD_TEXT

    movf     pre_counter2, w
    andlw    H'0F'
    addlw    B'00110000'
    movwf    Data2LCD
    call     SendData2LCD      ; and print in display

    movlw    TomSpace
    call     LCD_TEXT

    return

```

```

;-----

```

```

LCD_TEXT:

```

```

    MOVWF    TEMP                ; TEMP HOLDS TEXT START ADDRESS.
    CALL     TABLE
    ANDLW    H'FF'
    BTFSC    STATUS,Z            ; AT END OF MESSAGE? (0 RETURNED AT END)
    RETURN                                ; YES. ALL DONE.

    movwf    Data2LCD
    call     SendData2LCD

    MOVF     TEMP,W              ; POINT TO NEXT CHARACTER.
    ADDLW    1

```

```

        GOTO     LCD_TEXT
        return
;*****
LCDclear
        movlw    B'00000001'        ; LCD clear og cursor home
        call     control4bit
        return

LCDline1
        movlw    B'10000000'        ; LCD line 1
        call     control4bit
        return

LCDline2
        movlw    B'11000000'        ; LCD clear og cursor home
        call     control4bit
        return
;*****
; en controlbyte 8-bits overføres.
; PortB bit0 : En = HI, then En = LO to get data from port
; PortB bit2 : RS = LO (write to controlregister)
; PortB bit3 : RW = LO (write)
;*****
Control8Bit
        movwf    PORTB
        bsf      LcdE                ; Enable LCD
        nop
        bcf      LcdE
        call     wait2ms
        return
;*****
; en controlByte fra Data2LCD til Display overføres,
; først HI-nible (4 bit) og derefter LO-nible (4bit)
control4bit
        movwf    Data2LCD
        movf     Data2LCD, w
        andlw    H'F0'
        movwf    PORTB                ; HI-nible Data skrives
        bsf      LcdE                ; Enable LcdBus
        nop
        bcf      LcdE                ; Disable LcdBus
        swapf    Data2LCD, w          ; swap Data
        andlw    H'F0'
        movwf    PORTB                ; LO-nible Data skrives
        bsf      LcdE                ; Enable LcdBus
        nop
        bcf      LcdE                ; Disable LcdBus
        call     wait2ms
        return
;*****
; en Databyte fra Data2LCD til Display overføres
; først HI-nible (4 bit) og derefter LO-nible (4bit)
SendData2LCD
        movf     Data2LCD, w
        andlw    H'F0'
        movwf    PORTB                ; HI-nible Data skrives
        bsf      LcdRs                ; Data
        bsf      LcdE                ; Enable LcdBus
        nop
        nop
        bcf      LcdE                ; Disable LcdBus
        bcf      LcdRs                ;
        swapf    Data2LCD, w
        andlw    H'F0'
        movwf    PORTB                ; LO-nible Data skrives
        bsf      LcdRs                ; Data

```

```

        bsf                LcdE
        nop
        nop
        bcf                LcdE            ; Disable LcdBus
        bcf                LcdRs        ;
        call    wait1ms
        return

;*****
; Initialisering af LCD-Display
;*****
InitLCD
        call    wait250ms            ; At least 15 ms wait after power up
        movlw   B'00110000'          ; 1 FunctSet bit4:8bit,bit3:2line,bit2:5x8
        call    Control8Bit
        call    wait50ms
        movlw   B'00110000'          ; 2 FunctSet bit4:8bit,bit3:2line,bit2:5x8
        call    Control8Bit
        movlw   B'00110000'          ; 3 FunctSet bit4:8bit,bit3:2line,bit2:5x8
        call    Control8Bit
        movlw   B'00100000'          ; 4 FunctSet bit4:4bit,bit3:2line,bit2:5x8
        call    Control8Bit
        movlw   B'00000001'          ; LCD clear og cursor home
        call    control4bit
        movlw   B'00101000'          ; 5 FunctSet bit4:4bit,bit3:2line,bit2:5x8
        call    Control8Bit
        call    wait2ms
        movlw   B'00001000'          ; 6 bit2:display,bit1:cursor,bit0:blink.(0=off)
        call    control4bit
        call    wait2ms
        movlw   B'00000110'          ; 7 Entry mode, move right, display-shift off
        call    control4bit
        call    wait2ms
        movlw   B'00000011'          ; 8 cursor home, cursor home
        call    control4bit
        call    wait2ms
        movlw   B'00001100'          ; 9 bit2:display,bit1:cursor,bit0:blink. (1=on)
        call    control4bit
        call    wait2ms
        return

;*****
; TestMidnight tester minutbit, som sættes fra interruptrutinen.
; Hvis minutbit=1 testes for max antalminutter i et døgn,
; clockhex1+2 og sec1+2 resettes ved overflow (midnat)
;*****
TestMidnight
        btfss    minutbit
        return
        bcf      minutbit
        goto     test_max_count
test_max_count
        movf     clockhex2,w
        bcf      STATUS,C
        sublw    d'4'
        btfsc    STATUS,C
        return
        goto     test_clockhex1
test_clockhex1
        movf     clockhex1,w
        bcf      STATUS,C
        sublw    d'159'            ; (5 x 60)+160=1440=1 døgn
        btfsc    STATUS,C
        return
        goto     reset_all
reset_all
        clrf     clockhex2

```

```

        clrfs    clockhex1
        clrfs    sec2
        clrfs    sec1
        return
;*****
; Procedure EE_write writes data to EEPROM.
; Input : EEprom-adresse in EEADR, data in EEDATA.
;*****
EE_write
        bsf      STATUS,RP0          ; select bank 1
        bsf      EECON1,WREN         ; set write enable
        bcf      INTCON, GIE         ; disable interrupts
        movlw    55
        movwf    EECON2
        movlw    0AA
        movwf    EECON2
        bsf      EECON1,WR           ; init write cycle
        bcf      EECON1,WREN         ; clear write enable
        bsf      INTCON, GIE         ; enable interrupts
        bcf      STATUS,RP0          ; select page 0
        return
;*****
; Procedure EE_read reads in EEPROM.
; Input  : adress in w
; Output : data in EEDATA and w.
;*****
EE_read
        bsf      STATUS,RP0          ; select bank 1
        movwf    EEADR
        bsf      EECON1,RD           ; set read
        movf     EEDATA,w            ; result in EEDATA and W
        bcf      STATUS,RP0          ; select bank 0
        return
;-----
        END

```