```
'****************************************************************
'*  Name    : PolyMetro.BAS                                     *
'*  Author  : Godfried-Willem RAES                              *
'*  Notice  : Copyleft (c) 2012 Logosoft Public Domain          *
'*  Date    : 29.05.2012                                        *
'*  Version : 1.0                                               *
'*  Notes   : Based On Spiro-hub en Synchro-hub code model      *
'****************************************************************
'29.05.2012: code start - Version 1.0
'            Each metronome reacts to its own midi note on/off command.
'            with controllers for individual automatic tempo steering.
'            This code uses 8 timings tasks with 32 bit timing
'30.05.2012: The firmware works for note-on/offs, after solving some
'            bugs in the hardware.
'            Periodic timers not yet tested.
'            TO DO: Lookups for real-word unit timings.


Include "18F2525.inc"       'version for the Synchrochord board. (40MHz)
'Include "18F2520.inc"     'also possible.                      (40MHz)
'Include "18F25K20.inc"    'for test & debug on an Amicus board. (64MHz)

' Mapping defines for midi-events on pin outputs and inputs:
    $define Strobe      PORTC.0    ' if we leave strobe at 0, the latch
will be transparant
                                   ' if strobe goes high, the data in the
input are latched to the output and the
                                   ' outputs do not follow input changes
anymore.
    '8-bit data port
    $define M0          PORTB.4    ' metronome 0
    $define M1          PORTB.3
    $define M2          PORTB.2
    $define M3          PORTB.1
    $define M4          PORTB.0
    $define M5          PORTC.5
    $define M6          PORTC.4
    $define M7          PORTC.3
    'controle-poort - not used
    $define Pin10       PORTC.2
    $define Pin11       PORTC.1
    $define Pin12       PORTA.5
    $define Pin13       PORTA.4
    ' tweede rij    - set to input for now as we do not use them
    $define Pin14       PORTA.0
    $define Pin15       PORTA.1
    $define Pin16       PORTA.2
    $define Pin17       PORTA.3

    'red LED for debug:
    $define Debug_Led   PORTB.5   ' for testing - red led - watchdog

' configure the input and output pins:
Clear SSPCON1.5       'RC3 must be available for I/O
TRISA = %01111111     'bits set to 0 are output, 1 = input
TRISB = %11100000
TRISC = %11000110     'RC1 en RC2 zijn pwm outputs and must be set to
output
                      'RC6 en RC7 zijn USART  I/O and must be set to input
'constant definitions:
'initialisations for the midi input parser:
```

```
Symbol Midichannel = 0                               ' PolyMetrome_Channel
Symbol NoteOff_Status = 128 + Midichannel            ' 2 bytes follow
Symbol NoteOn_Status  = 144 + Midichannel
Symbol Keypres_Status = 160 + Midichannel            ' 2 bytes follow
Symbol Control_Status = 176 + Midichannel
Symbol ProgChange_Status = 192 + Midichannel         ' 1 byte message
Symbol Aftertouch_Status = 208 + Midichannel         ' 1 byte follows
Symbol Pitchbend_Status  = 224 + Midichannel         ' lsb msb follow
'application specific constants
Symbol NrTasks = 8                                   ' maximum 16
    'Symbol PWMFreq2 = PWMminF * 2                        ' PWMminF is processor
dependent.
    'Symbol PWMFreq3 = PWMminF * 3                        ' declared in the
processor include
    'Symbol PWMFreq4 = PWMminF * 4                         ' avoiding audible
artifacts
Symbol Lowtes = 36                                    ' first metronome
Symbol Hightes = 43                                   ' eighth metronome

' Setup the USART
Declare Hserial_Baud = 31250                         ' Set baud rate for the
USART to MIDI specs.
Declare Hserial_TXSTA = 0x24                         ' instead of the normal 0x20
- ?? 0x24
Declare All_Digital = True                           ' no analog inputs
'    Declare Hserial_Clear = On                      ' should clear on errors.
Bytes get lost of course...
' Create variables
    Dim Cnt As Dword System
    Dim CntHw As Cnt.Word1                           'used in the timer0
interrupt, to create a 32 bit timer
    Dim CntLw As TMR0L.Word                          'this is the trick to read
both TMR0L and TMR0H
                                                     'it makes Cntlw the low word
of cnt
                                                     'We still have to copy the
contents of Lw to Cnt
    Dim Tim3 As TMR3L.Word                           ' 16 bit counter for sampler
'    Dim Sr as TMR0L.7                               '512 S/s
                                                     ' As TMR0H.1  would be 128
S/s
                                                     ' As TMR0H.2  would be 64
S/s
                                                     ' As TMR0H.3  would be 32
S/s
                                                     ' As TMR0H.4  would be 16
S/s
    Dim Bytein  As Byte System                       ' midi byte read from buffer
    Dim StBit As Bytein.7                            ' highest bit of ByteIn
    Dim i As Byte System                             ' general purpose counter
    ' midi variables
    Dim statusbyte As Byte System
    Dim noteUit As Byte System                       ' note off + release value
    Dim release As Byte System
    Dim noteAan As Byte System                       ' note on + release value
    Dim velo As Byte System
    Dim notePres As Byte System                      ' note pressure + pressure
value
    Dim pres As Byte System
    Dim Ctrl As Byte System                          ' continuous controller +
value
```

```
    Dim value As Byte System
    Dim prog As Byte System                   ' program change + program-
byte
    Dim aft  As Byte System                   ' channel aftertouch
    Dim pblsb As Byte System                  ' pitch bend lsb
    Dim pbmsb As Byte System                  ' pitch bend msb
    Dim veltim As Dword System                ' 32 bit velo
'    Dim newtim As Dword System
    Dim VelFlags As Word System               ' bits 0 - 15 used as flags
for active timers
    Dim VelFlags0 As VelFlags.Byte0           ' alias for bits 0-7
    Dim VelFlags1 As VelFlags.Byte1           ' bits 8-15 - not used
    Dim CC66 As  Byte System                  ' global on/off switch
    Dim PowerOn As CC66.0
    Dim st As Byte System
    Dim b1 As Byte System
    Dim b2 As Byte System
    Dim Lites As Byte System                  ' bits used as flags

'----------------------------------------------------------------------------
---------------
' Load the USART Interrupt handler And buffer read subroutines into memory
' Include "ADC.inc"                      ' Load the ADC macros into the
program
Include "PolyMetro_Irq.inc"            ' our own version for UART And
Timer0/3 Interrupt
'Include "Timers.inc"                  ' required for velo support with
timed pulses and periods.
'Include "DwordArrays.inc"             ' support for dword arrays.


'framework for a multitasker:
'Dim Task_rsi[NrTasks] As Word                  'task reschedule interval
(period), if 0 the task is not active
                                                'max. value limited to
65535. For longer periods, it will have to
                                                'become dword!!!
Dim Period0 As Dword
Dim Period1 As Dword
Dim Period2 As Dword
Dim Period3 As Dword
Dim Period4 As Dword
Dim Period5 As Dword
Dim Period6 As Dword
Dim Period7 As Dword
Dim Velmsb[NrTasks] As Word                     'the application for velo-
timers, is in fact just a one-shot task
Dim VelLsb[NrTasks] As Word

'make sure we initialize those pins on start up:
'fault?: there should be no executable statements outside the main program.
Low Strobe ' make latch transparant
Low M0
Low M1
Low M2
Low M3
Low M4
Low M5
Low M6
Low M7
Low Debug_Led
```

```
High Strobe ' latch the zero's
Clear CC66
Low Strobe  ' make latch transparant again

'---------------------------------------------------------------------------
---------------
' Main program starts here

MAIN:
    High Debug_Led
    DelayMS 50                      ' wait for stability
    Low Debug_Led
    Clear VelFlags0                 ' no timings tasks active
    Clear Lites                     ' no timers active on start-up
    Init_Usart_Interrupt            ' Initiate the USART serial buffer
interrupt
                                    ' this procedure is in the include
file
    Clear_Serial_Buffer             ' Clear the serial buffer and reset
its pointers
                                    ' in the include as well

                                    ' Configure Timer0 for:
'                                     Clear TMR0L and TMR0H registers
'                                     Interrupt on Timer0 overflow
'                                     16-bit operation
'                                     Internal clock source  40MHz
'                                     1:256 Prescaler : thus 40MHz / 256 =
156.250kHz
 '   Opentimer0 (Timer_INT_On & T0_16BIT & T0_SOURCE_INT & T0_PS_1_256) in
macro file.
         Clear T1CON
         Clear IntConBits_T0IF     ' clear interrupt flag
         Set INTCONBITS_T0IE       ' enable interrupt on overflow
         T0CON = %10000111          ' bit 7 = enable/disable
                                    ' bit 6 = 1=8 bot, 0=16 bit
                                    ' bit 5 = 1 pin input, 0= Internal
Clk0
                                    ' bit 4 = HL or LH transition when
bit5 =1
                                    ' bit 3 = 1= bypass prescaler, 0=
input from prescaler
                                    ' bit 2-0 = prescaler select: 111=
1:256
                                    ' Setup the High priorities for the
interrupts

    ' open and start timer3 for sampling:  (not used here)
    Clear T3CON
    Clear PIR2BITS_TMR3IF   ' clear IRQ flag
    Set PIE2BITS_TMR3IE     ' irq on
    Clear Tim3              ' Clear TMR3L And TMR3H registers
    Set RCONbits_IPEN       ' Enable priority interrupts
    Clear IPR2bits_TMR3IP   ' Set Timer3 as a low priority interrupt source
    ' we can also set T3Con in one instruction as:
    T3CON = %10110001              ' oef, now it works...
                            ' bit 7 = 16 bit mode
                            ' bit 6,3 = 0, 0
                            ' bit 5,4 = 1:8 prescale
                            ' bit 2 = 0
                            ' bit 1 = 0 Internal clock = Fosc/4
```

```
                              ' bit 0 : 1= enable timer 3, 0= disable
                              ' maximum count = 52.42ms, 1 tick =0.8uS,
lowest freq.=19Hz
    ' start the main program loop:
LOOP:
                                       ' Create an infinite loop
        Bytein = HRSIn                 ' Read data from the serial buffer,
with no timeout
                                       ' Start the midi parser.
        Midi_Parse:
            If Bytein > Control_Status Then    ' here higher statusses are
not implemented.
                If Bytein > 253 Then       '254 = midiclock, 255= reset
                                           'midiclock can interrupt all
other msg's...
                                           '255 had to be intercepted since
thats what we
                                           'get when no new byte flows in
(?)
                Else
                    Clear statusbyte       'reset the status byte
                End If
                GoTo Check_Timers          'throw away
            EndIf
            If StBit =1 Then               'should be faster than If Bytein
> 127 Then
                                           'status byte received, bit 7 is
set
                Clear statusbyte           'if on another channel, the
statusbyte needs a reset
                Select Bytein              'eqv to Select case ByteIn
                       Case NoteOff_Status
                            statusbyte = Bytein
                            Set noteUit '= 255          'reset value.
Cannot be 0 !!!
                            Set release '= 255          '0 is a valid midi
note!
                       Case NoteOn_Status
                            statusbyte = Bytein
                            Set noteAan '= 255
                            Set velo '= 255
'                       Case Keypres_Status
'                            statusbyte = Bytein
'                            Set notePres '= 255
'                            Set pres '= 255
                       Case Control_Status
                            statusbyte = Bytein
                            Set Ctrl '= 255
                            Set value '= 255
'                       Case ProgChange_Status
'                            statusbyte = Bytein
'                            prog = 255
'                       Case Aftertouch_Status
'                            statusbyte = Bytein
'                            aft = 255
'                       Case Pitchbend_Status
'                            statusbyte = Bytein
'                            pblsb = 255
'                            pbmsb = 255
                End Select
            Else                                        'midi byte is 7
bits
                Select statusbyte
                       Case 0                           'not a message for
this channel
                            GoTo Check_Timers           'disregard
                       Case NoteOff_Status
                            If noteUit = 255 Then
                                noteUit = Bytein
                            Else
                                release = Bytein        'message complete,
so we can do the action...
                                Select noteUit
                                       Case Lowtes
                                            Low M0      ' clear should be
only a single machine cycle. Low sets TRIS also.
                                            Clear Lites.0
                                            Clear VelFlags0.0
                                       Case Lowtes + 1
                                            Low M1
                                            Clear Lites.1
                                            Clear VelFlags0.1
                                       Case Lowtes + 2
                                            Low M2
                                            Clear Lites.2
                                            Clear VelFlags0.2
                                       Case Lowtes + 3
                                            Low M3
                                            Clear Lites.3
                                            Clear VelFlags0.3
                                       Case Lowtes + 4
                                            Low M4
                                            Clear Lites.4
                                            Clear VelFlags0.4
                                       Case Lowtes + 5
                                            Low M5
                                            Clear Lites.5
                                            Clear VelFlags0.5
                                       Case Lowtes + 6
                                            Low M6
                                            Clear Lites.6
                                            Clear VelFlags0.6
                                       Case Lowtes + 7    ' = Hightes
                                            Low M7
                                            Clear Lites.7
                                            Clear VelFlags0.7
                                End Select
                                Set noteUit '= 255
'reset
                            EndIf
                            GoTo Check_Timers
                       Case NoteOn_Status
                            If noteAan = 255 Then
                                noteAan = Bytein
                            Else
                                velo = Bytein
                                If velo = 0 Then
                                    Select  noteAan
                                       Case Lowtes
                                            Low M0      ' clear should be
onloy a single machine cycle. Low sets TRIS also.
```

```
                                                  Clear Lites.0
                                                  Clear VelFlags0.0
                                             Case Lowtes + 1
                                                  Low M1
                                                  Clear Lites.1
                                                  Clear VelFlags0.1
                                             Case Lowtes + 2
                                                  Low M2
                                                  Clear Lites.2
                                                  Clear VelFlags0.2
                                             Case Lowtes + 3
                                                  Low M3
                                                  Clear Lites.3
                                                  Clear VelFlags0.3
                                             Case Lowtes + 4
                                                  Low M4
                                                  Clear Lites.4
                                                  Clear VelFlags0.4
                                             Case Lowtes + 5
                                                  Low M5
                                                  Clear Lites.5
                                                  Clear VelFlags0.5
                                             Case Lowtes + 6
                                                  Low M6
                                                  Clear Lites.6
                                                  Clear VelFlags0.6
                                             Case Lowtes + 7    ' = Hightes
                                                  Low M7
                                                  Clear Lites.7
                                                  Clear VelFlags0.7
                                          End Select
                                          Set noteAan '= 255
'reset !!!
                                          GoTo Check_Timers              'jump
out
                                      EndIf
                                          Select noteAan
                                             Case Lowtes
                                                  High M0      ' clear should be
only a single machine cycle. Low sets TRIS also.
                                                  'Set Lites.0
                                             Case Lowtes + 1
                                                  High M1
                                                  'Set Lites.1
                                             Case Lowtes + 2
                                                  High M2
                                                  'Set Lites.2
                                             Case Lowtes + 3
                                                  High M3
                                                  'Set Lites.3
                                             Case Lowtes + 4
                                                  High M4
                                                  'Set Lites.4
                                             Case Lowtes + 5
                                                  High M5
                                                  'Set Lites.5
                                             Case Lowtes + 6
                                                  High M6
                                                  'Set Lites.6
                                             Case Lowtes + 7    ' = Hightes
                                                  High M7
```

```
                                    'Set Lites.7
                              End Select
                           Set noteAan '= 255                   'reset
                       EndIf
                       GoTo Check_Timers
                  Case Keypres_Status           'may be used for speed
modulation
                       If notePres = 255 Then
                           notePres = Bytein
                       Else
                           pres = Bytein
                           GoSub KeyPres
                       EndIf
                       GoTo Check_Timers
                  Case Control_Status    'this is where the action
takes place for controllers
                       If Ctrl = 255 Then
                           Ctrl = Bytein
                       Else
                           value = Bytein
                           GoSub Controller
                       EndIf
                       GoTo Check_Timers
'                  Case ProgChange_Status
'                       If prog = 255 Then                'single byte
message
'                            prog = Bytein                'weak
coding...
'                            GoSub ProgChange
'                       EndIf
                End Select
            EndIf

Check_Timers:
        ' here we check the Task counters and compare them with the 32 bit
cnt value
        ' using the Velflags dword variable:
        If VelFlags0 > 0 Then       'if any bit is set here, there is a
timer running
            If VelFlags0.0 = 1 Then
                    veltim.Word1 = Velmsb[0]
                    veltim.Word0 = VelLsb[0]
                    Cnt.Word0 = CntLw                    'read counter
                    If Cnt >= veltim Then GoSub Task0    'note 0
            EndIf
            If VelFlags0.1 = 1 Then
                    veltim.Word1 = Velmsb[1]
                    veltim.Word0 = VelLsb[1]
                    Cnt.Word0 = CntLw                    'read counter
                    If Cnt >= veltim Then GoSub Task1    'note 1
            EndIf
            If VelFlags0.2 = 1 Then
                    veltim.Word1 = Velmsb[2]
                    veltim.Word0 = VelLsb[2]
                    Cnt.Word0 = CntLw                    'read counter
                    If Cnt >= veltim Then GoSub Task2    'note 2
            EndIf
            If VelFlags0.3 = 1 Then
                    veltim.Word1 = Velmsb[3]
                    veltim.Word0 = VelLsb[3]
                    Cnt.Word0 = CntLw                    'read counter
                    If Cnt >= veltim Then GoSub Task3    'note 3
            EndIf
            If VelFlags0.4 = 1 Then
                    veltim.Word1 = Velmsb[4]
                    veltim.Word0 = VelLsb[4]
                    Cnt.Word0 = CntLw                    'read counter
                    If Cnt >= veltim Then GoSub Task4    'note 4
            EndIf
            If VelFlags0.5 = 1 Then
                    veltim.Word1 = Velmsb[5]
                    veltim.Word0 = VelLsb[5]
                    Cnt.Word0 = CntLw                    'read counter
                    If Cnt >= veltim Then GoSub Task5    'note 5
            EndIf
            If VelFlags0.6 = 1 Then
                    veltim.Word1 = Velmsb[6]
                    veltim.Word0 = VelLsb[6]
                    Cnt.Word0 = CntLw                    'read counter
                    If Cnt >= veltim Then GoSub Task6    'note 6
            EndIf
            If VelFlags0.7 = 1 Then
                    veltim.Word1 = Velmsb[7]
                    veltim.Word0 = VelLsb[7]
                    Cnt.Word0 = CntLw                    'read counter
                    If Cnt >= veltim Then GoSub Task7    'note 7
            EndIf
       'Else
       '     If CntHw > 0xFF Then Clear CntHw
        EndIf
GoTo LOOP                              ' end of the main loop

KeyPres:
        'the note to which the pressure should be applied is passed in
NotePres, the value in Pres
    Set notePres '= 255
Return

ProgChange:
    ' we could use this for presets such as for Logos 3/5 or Fall95.
    Set prog '= 255                'this is not realy required
Return

Pitchbend:
                 'only implemented on dsPIC based robots
    Set pblsb '= 255
Return

Aftertouch:
    'this is the channel aftertouch, affecting all notes
    Set aft '= 255                               'not mandatory
Return

Controller:
    Select Ctrl
           ' msb controllers voor tempo kontrole: lowtes to hightes
           ' lsb controllers voor tempo kontrole: lowtes + 32 to hightes +
32
           ' we have to calculate values for Period0 ... Period7 (dword)
           Case Lowtes                      ' 36
                'Period0 = (~value & 127) << 9
                Period0 = value << 13 '9      ' this is in tick units
           Case Lowtes + 1
                Period1 = value << 13 '9
           Case Lowtes + 2
                Period2 = value << 13 '9
           Case Lowtes + 3
                Period3 = value << 13 '9
           Case Lowtes + 4
                Period4 = value << 13 '9
           Case Lowtes + 5
                Period5 = value << 13 '9
           Case Lowtes + 6
                Period6 = value << 13 '9
           Case Lowtes + 7                ' 43
                Period7 = value << 13 '9
           Case Lowtes + 32               ' 68  ' lsb's
                Period0 = Period0 + (value << 6)          ' should give a
0.9ms resolution
           Case Lowtes + 33
                Period1 = Period1 + (value << 6)
           Case Lowtes + 34                 ' lsb's
                Period2 = Period2 + (value << 6)
           Case Lowtes + 35
                Period3 = Period3 + (value << 6)
           Case Lowtes + 36                 ' lsb's
                Period4 = Period4 + (value << 6)
           Case Lowtes + 37
                Period5 = Period5 + (value << 6)
           Case Lowtes + 38                 ' lsb's
                Period6 = Period6 + (value << 6)
           Case Lowtes + 39                 ' 75
                Period7 = Period7 + (value << 6)
           Case 65
                ' this should start all metronomes in sync
                If Period0 > 0 Then Set VelFlags0.0 : Set Lites.0
                If Period1 > 0 Then Set VelFlags0.1 : Set Lites.1
                If Period2 > 0 Then Set VelFlags0.2 : Set Lites.2
                If Period3 > 0 Then Set VelFlags0.3 : Set Lites.3
                If Period4 > 0 Then Set VelFlags0.4 : Set Lites.4
                If Period5 > 0 Then Set VelFlags0.5 : Set Lites.5
                If Period6 > 0 Then Set VelFlags0.6 : Set Lites.6
                If Period7 > 0 Then Set VelFlags0.7 : Set Lites.7
           Case 66
                'on/off for the robot
                If value = 0 Then
                    Clear PowerOn   'CC66.0
                    GoSub PowerDown
                Else
                    Set PowerOn     'CC66.0
                EndIf
           Case 123
                GoSub AllNotesOff
    End Select
    Set Ctrl '= 255                                  'mandatory reset
Return

AllNotesOff:
              Clear VelFlags         'stop all running timers
              Clear M0
              Clear M1
              Clear M2
              Clear M3
              Clear M4
              Clear M5
              Clear M6
              Clear M7
              Low Debug_Led
              Clear Lites
Return

PowerDown:
              Clear VelFlags         'stop all running timers
              Clear M0
              Clear M1
              Clear M2
              Clear M3
              Clear M4
              Clear M5
              Clear M6
              Clear M7
              Low Debug_Led
              Clear Lites
              Clear CC66
Return

Task0:
      If Lites.0 = 0 Then
             Clear VelFlags0.0    'stop task, as lite is switched off
             Clear M0 '= 0
      Else
          'reload task0
          'Set VelFlags0.0         'can just stay set
          Cnt.Word0 = CntLw
          veltim = Cnt + Period0 ' Task_rsi[0]      'add the period
duration
          Velmsb[0] = veltim.Word1
          VelLsb[0] = veltim.Word0
          'Toggle
          btg M0
      EndIf
Return

Task1:
      If Lites.1 = 0 Then
             Clear VelFlags0.1         'stop task, as lite is switched off
             Clear M1' = 0
      Else
          Cnt.Word0 = CntLw
          veltim = Cnt + Period1 'Task_rsi[1]      'add the period duration
in Task_rsi[1]
          Velmsb[1] = veltim.Word1
          VelLsb[1] = veltim.Word0
          'Toggle
          btg M1
      EndIf
Return

Task2:
      If Lites.2 = 0 Then
             Clear VelFlags0.2    'stop task, as lite is switched off
             Clear M2
      Else
          Cnt.Word0 = CntLw
          veltim = Cnt + Period2 'Task_rsi[2]      'add the period duration
          Velmsb[2] = veltim.Word1
          VelLsb[2] = veltim.Word0
          btg M2
      EndIf
Return

Task3:
      If Lites.3 = 0 Then
             Clear VelFlags0.3    'stop task, as lite is switched off
             Clear M3
      Else
          Cnt.Word0 = CntLw
          veltim = Cnt + Period3 'Task_rsi[3]      'add the period duration
          Velmsb[3] = veltim.Word1
          VelLsb[3] = veltim.Word0
          btg M3
      EndIf
Return

Task4:
      If Lites.4 = 0 Then
             Clear VelFlags0.4    'stop task, as lite is switched off
             Clear M4
      Else
          Cnt.Word0 = CntLw
          veltim = Cnt + Period4 'Task_rsi[4]      'add the period duration
          Velmsb[4] = veltim.Word1
          VelLsb[4] = veltim.Word0
          btg M4
      EndIf
Return

Task5:
      If Lites.5 = 0 Then
             Clear VelFlags0.5    'stop task, as lite is switched off
             Clear M5
      Else
          Cnt.Word0 = CntLw
          veltim = Cnt + Period5 ' Task_rsi[5]      'add the period
duration
          Velmsb[5] = veltim.Word1
          VelLsb[5] = veltim.Word0
          btg M5
      EndIf
Return

Task6:
      If Lites.6 = 0 Then
             Clear VelFlags0.6    'stop task, as lite is switched off
             Clear M6
      Else
          Cnt.Word0 = CntLw
          veltim = Cnt + Period6     'add the period duration
          Velmsb[6] = veltim.Word1
          VelLsb[6] = veltim.Word0
          btg M6
      EndIf
Return

Task7:
      If Lites.7 = 0 Then
            Clear VelFlags0.7    'stop task, as lite is switched off
            Clear M7
      Else
          Cnt.Word0 = CntLw
          veltim = Cnt + Period7     'add the period duration
          Velmsb[7] = veltim.Word1
          VelLsb[7] = veltim.Word0
          btg M7
      EndIf
Return
'[EOF]
```