CH-1092 Belmont, Switzerland
info@didel.com, www.didel.com

Routines for the HESO protocol

HESO protocol has been described in www.didel.com/doc/DocHeso.pdf. We explain provide here the routines (source avaliable on www.didel.com/doc/DopicSources) for reading and interpreting HESO orders..html

All PIC microprocessors will execute the following routines. The variable SavBin is transmitted before the order code savec in SavOrder variable are given below. Synchronous implementation of the HESO protocol can be provided by the author.

Receive order

```
RecHeso:
        Clr     SavBin
 L$:    Call    Rec         ; W=RecData
        Sub     W,#"@",W
        Skip,CS
        Jump    Z$
        Swap    SavBin,W
        And     #2´11110000,W
        Move    W,SavBin
        Move    RecData,W
        And     #2´1111,W
        Or      W,SavBin
        Jump    L$
 Z$:    Move    RecData,W
        Move    W,SavOrder
```

Send acknowledge with and without data

Routine **SndHeso** **Send Heso parameter and order**

in: SavBin Heso or BCD parameter SavOrder 0..16´3E order code
out:
mod:

```
SndHeso:
        Swap    SavBin,W
        And     #2´1111,W
        Or      #"O",W
        Call    Snd
        Move    SavBin,W
        And     #2´1111,W
        Or      #"0",W
        Call    Snd
        Move    SavOrder,W
        Jump    Snd         ; Return from that routine
```

Convert binary/hexa 00-FF value to 00-99 BCD value

The problem is if you read an 8-bit value in e.g. an A/D converter, and you prefer to send data in BCD, betwwen 0 to 99. One solution is to trucate the binary value to 16´63 = 99, and use a binary to decimal algorithm

Program **XBinDesR.asi** 140401

Routines BinDes DesBin Convit

```
; Macros dans PPDef.asi
```

Routine **BinDes** **Binaire 0-FF en décimal**

```
; One divide by 26 (successive soustractions), the rest
; If result > 128, one add 1 so FF -> 99  Exec tim
  in:   W
  out:  W = SavHex
  mod: Tdes TdesU
BinDes:
        Move        W,Tdes
        Move        W,TdesU
        Clr         SavHex
D$:     Move        #26,W
        Sub         W,Tdes
        Skip,CS
        Jump        U$
        Inc         SavHex          ; dizaines
        Jump        D$
U$:     Add         W,Tdes
        Call        TaBDes
        Swap        SavHex
        Or          W,SavHex
        Move        SavHex,W
        TestSkip,BS TdesU:#7
        Jump        F$
        Inc         SavHex          ; en BCD
        Move        SavHex,W
        And         #2'1111,W       ; unités >9?
        Xor         #2'1010,W
        Move        #6,W            ; ne modifie pas Z
        Skip,NE
        Add         W,SavHex
  F$:   Move        SavHex,W
        Ret
```

Routine **ConVit** **Converti 94..00..07 en vitesses E1 .. 00 .. E0 avec saturation**

```
; Pas sûr de garder ces vitesses négatives, ne pas les
; Durée env 15 us
  in:   W
  out:  W
ConVit:
; On teste si négatif
        TestSkip,BC SavBin:#7
        Jump        Neg$
; Positif: Saturer à <=7 et table
        Add         #-7,W
        Skip,CC
        Clr         W
        Add         #7,W
        Move        W,SavBin
        Call        TaViPos$
        Ret
```

```
TaBDes:
        SetPage
        Move        Tdes,W
        Add         W,PCL
        t8          0,0,1,1,1,2,2,3 ; 0-7
        t8          3,3,4,4,5,5,5,6 ; 8-15
        t8          6,7,7,7,8,8,8,9 ; 16-23
        t2          9,9             ; 24 25
.If     (APC/256) .NE. (TabDes/256)
 Aie, ce module traverse la page
.Endif
```

Routine **DesBin** **Decimal 0-99 en binaire 0-FF**

```
; On multiplie par 26 les dizaines et converti les unités par
; Durée env 25 us
  in:   W
  out:  W = SavBin
  mod: Tdes TdesU
DesBin:
        Move        W,Tdes
        And         #2'1111,W
        Move        W,TdesU         ; Unités
        Clr         W
        TestSkip,BC Tdes:#4
        Add         #26,W
        TestSkip,BC Tdes:#5
        Add         #26*2-1,W
        TestSkip,BC Tdes:#6
        Add         #26*4-2,W
        TestSkip,BC Tdes:#7
        Add         #26*8-3,W
        Move        W,SavBin
        Call        TaDesB
        Add         W,SavBin
        Move        SavBin,W
        Ret
TaDesB:
        SetPage
        Move        TdesU,W
        Add         W,PCL
        t8          0,3,5,8,10,13,15,18
        t2          21,24
.If     (APC/256) .NE. (TabDes/256)
 Aie, ce module traverse la page
.Endif

TaViPos$:
        SetPage
        Move        SavBin,W
        Add         W,PCL
        t8          0,16'20,16'40,16'60,16'80,16'A0,
.If     (APC/256) .NE. (TaViPos$/256)
 Aie, ce module traverse la page
.Endif
; Pas sûr de garder ces vitesses négatives, ne pas les utili
 Neg$:
;Saturer à >= 94
        Add         #-94,W
        Skip,CS
        Clr         W
        Add         #94,W
; Maquer, 9..0 passer par la table
        And         #2'00001111,W
        Move        W,SavBin
        Call        TaViNeg$
        Ret
TaViNeg$:
        SetPage
        Move        SavBin,W
        Add         W,PCL
        t2          0,0             ; 0 1 pour 90 91
        t8          0,0,16'E1,16'C1,16'A1,16'81,16'4
.If     (APC/256) .NE. (TaViNeg$/256)
 Aie, ce module traverse la page
.Endif
```