```
'--------------------------------------------------------------------------------
' File:   ISMapper.lib
' Author: Alex Leitman
' Description : Mapping of DS402 Drive process data to Motion Cotroller RPDO/TPDO
'
'--------------------------------------------------------------------------------

'----------- local variables ------------------
'----------------------------------------------
'--------------------------------------------------------------------------------
import_c  SET_CONTROL_PDO_DADD(byval as long,byval as long,byval as long,byval as long)
import_c  SET_STATUS_PDO_DADD(byval as long,byval as long,byval as long,byval as long)
import_c  SET_PCMD_PDO_DADD(byval as long,byval as long,byval as long,byval as long)
import_c  SET_PFB_PDO_DADD(byval as long,byval as long,byval as long,byval as long)
import_c  SET_VCMD_PDO_DADD(byval as long,byval as long,byval as long,byval as long)
import_c  SET_VFB_PDO_DADD(byval as long,byval as long,byval as long,byval as long)
import_c  SET_TCMD_PDO_DADD(byval as long,byval as long,byval as long,byval as long)
import_c  SET_TDCMD_PDO_DADD(byval as long,byval as long,byval as long,byval as long)
import_c  SET_CFB_PDO_DADD(byval as long,byval as long,byval as long,byval as long)
import_c  SET_DOUT_PDO_DADD(byval as long,byval as long,byval as long,byval as long)
import_c  SET_DIN_PDO_DADD(byval as long,byval as long,byval as long,byval as long)

import_c SDO_TIME_OUT_SET(byval as long)'int timeout
import_c SDO_TIME_OUT_GET() as long

'================================================================
' Confuguration of drive:
'RPDO #1 :0x6040.0-control word, 0x607A.0-pcmd , 0x6071.0-tcmd
'RPDO #2 :0x60FE.1 - Digital Ouputs
'RPDO #3 :---
'RPDO #4 :---
'================================================================
'TPDO #1 :0x6041.0-status word ,0x60640.0-pfb 0x60780.0-tfb
'TPDO #2 :0x60fd.0 - digila inputs, 0x6074.0 - Torque Demand
'TPDO #3 :---
'TPDO #4 :---
'================================================================
public sub IntergratedStepperMapPDO(byval drvAddress as long)
dim RPD01_COB as const long = 0x200
```

```
dim RPD02_COB as const long = 0x300
dim RPD03_COB as const long = 0x400
dim RPD04_COB as const long = 0x500
dim TPD01_COB as const long = 0x180
dim TPD02_COB as const long = 0x280
dim TPD03_COB as const long = 0x380
dim TPD04_COB as const long = 0x480
'erase MC papping
      SET_CONTROL_PDO_DADD(drvAddress,0,0,0)
      SET_STATUS_PDO_DADD(drvAddress,0,0,0)
      SET_PCMD_PDO_DADD(drvAddress,0,0,0)
      SET_PFB_PDO_DADD(drvAddress,0,0,0)
      SET_VCMD_PDO_DADD(drvAddress,0,0,0)
      SET_VFB_PDO_DADD(drvAddress,0,0,0)
      SET_TCMD_PDO_DADD(drvAddress,0,0,0)
      SET_TDCMD_PDO_DADD(drvAddress,0,0,0)
      SET_CFB_PDO_DADD(drvAddress,0,0,0)
'---------------------- Drive RPDO part ----------------------------------
      '***************** drive dynamic mapping of RPDO #1 *****************
      'disable RPDO #1 - required
      call CAN_SDO_WRITE(drvAddress,0x1400,01,32,0x80000000 + drvAddress + RPD01_COB)
'set number of elements in RPDO #1 to zero - required
      call CAN_SDO_WRITE(drvAddress,0x1600,00,8,0)
'-------------------------------------------------------------------------------
'control word
      call CAN_SDO_WRITE(drvAddress,0x1600,01,32,0x60400010) 'control word
      SET_CONTROL_PDO_DADD(drvAddress,RPD01_COB , 0 , 2) '16 bit
'-------------------------------------------------------------------------------
'pcmd
      call CAN_SDO_WRITE(drvAddress,0x1600,02,32,0x607A0020) 'pcmd
      SET_PCMD_PDO_DADD(drvAddress,RPD01_COB,2,4)    '32 bit
'-------------------------------------------------------------------------------
'tcmd
      call CAN_SDO_WRITE(drvAddress,0x1600,03,32,0x60710010) 'TCMD
      SET_TCMD_PDO_DADD(drvAddress,RPD01_COB,6, 2)
'-------------------------------------------------------------------------------
      'update number of elements in RPDO #1 - required
      call CAN_SDO_WRITE(drvAddress,0x1600,00,8,3)
      'enable RPDO #1 - required
      call CAN_SDO_WRITE(drvAddress,0x1400,01,32,RPD01_COB + drvAddress)
```

```
       '***************** drive dynamic mapping of RPDO #2 ******************
      'disable RPDO #2 - required
      call CAN_SDO_WRITE(drvAddress,0x1401,01,32,0x80000000 + drvAddress + RPD02_COB)
     'update number of elements in RPDO #2 - required
      call CAN_SDO_WRITE(drvAddress,0x1601,00,8,0)
'-------------------------------------------------------------------------------
' Digital outputs
     'update number of elements in RPDO #2 - required
      call CAN_SDO_WRITE(drvAddress,0x1601,01,32,0x60FE0120) 'outputs
      call CAN_SDO_WRITE(drvAddress,0x60FE,02,32,0xffffffff) 'outputs mask

      SET_DOUT_PDO_DADD(drvAddress,RPD02_COB , 0 , 4) '16 bit
      call CAN_SDO_WRITE(drvAddress,0x1601,00,8,1)
      call CAN_SDO_WRITE(drvAddress,0x1401,01,32,drvAddress + RPD02_COB)
'-------------------------------------------------------------------------------

      '***************** drive dynamic mapping of RPDO #3 ******************
      'disable RPDO #3 - required
       call CAN_SDO_WRITE(drvAddress,0x1402,01,32, 0x80000000 + drvAddress + RPD03_COB)
      '***************** drive dynamic mapping of RPDO #4 ******************
      'disable RPDO #4 - required
       call CAN_SDO_WRITE(drvAddress,0x1403,01,32, 0x80000000 + drvAddress + RPD04_COB)
'----------------------- Drive TPDO part ----------------------------------

      '***************** drive dynamic mapping of TPDO #1 ******************
      'disable TPDO #1 - required
      call CAN_SDO_WRITE(drvAddress,0x1800,01,32, 0x80000000 + drvAddress + TPD01_COB)
      'set number of elements in TPDO #1 to zero - required
      call CAN_SDO_WRITE(drvAddress,0x1A00,00,8,0)
      'set TPDO #1 mapping
'-------------------------------------------------------------------------------
'status word
      call CAN_SDO_WRITE(drvAddress,0x1A00,01,32,0x60410010) 'status word
      SET_STATUS_PDO_DADD(drvAddress,TPD01_COB,0, 2)
'-------------------------------------------------------------------------------
'pfb
      call CAN_SDO_WRITE(drvAddress,0x1A00,02,32,0x60640020) 'pfb
      SET_PFB_PDO_DADD(drvAddress,TPD01_COB , 2 , 4)
'-------------------------------------------------------------------------------
'current feedback
```

```
      call CAN_SDO_WRITE(drvAddress,0x1A00,03,32,0x60780010) 'tfb Current Actual Value
      SET_CFB_PDO_DADD(drvAddress,TPD01_COB , 6 , 2)
'-------------------------------------------------------------------------------
      'update number of elements in TPDO #1 - required
      call CAN_SDO_WRITE(drvAddress,0x1A00,00,8, 3)
      'enable TPDO #1 - required
      call CAN_SDO_WRITE(drvAddress,0x1800,01,32,TPD01_COB + drvAddress)
      call CAN_SDO_WRITE(drvAddress,0x1800,02,8,1) ' enable cyclic transmission of PDO

      '***************** drive dynamic mapping of TPDO #2 ******************
      'disable TPDO #2 - required
      call CAN_SDO_WRITE(drvAddress,0x1801,01,32, 0x80000000 + drvAddress + TPD02_COB)
     'set number of elements in TPDO #1 to zero - required
      call CAN_SDO_WRITE(drvAddress,0x1A01,00,8,0)
      'set TPDO #1 mapping
'-------------------------------------------------------------------------------
'digital inputs
      call CAN_SDO_WRITE(drvAddress,0x1A01,01,32,0x60FD0020) 'digital inputs
      SET_DIN_PDO_DADD(drvAddress,TPD02_COB,0, 4)
'-------------------------------------------------------------------------------
'Torque demand
      call CAN_SDO_WRITE(drvAddress,0x1A01,02,32,0x60740010) 'toque demand that drive produces to itself
      SET_TDCMD_PDO_DADD(drvAddress,TPD02_COB , 4 , 2)
'-------------------------------------------------------------------------------
      'update number of elements in TPDO #1 - required
      call CAN_SDO_WRITE(drvAddress,0x1A01,00,8, 2)
      'enable TPDO #1 - required
      call CAN_SDO_WRITE(drvAddress,0x1801,01,32,TPD02_COB + drvAddress)

      '***************** drive dynamic mapping of TPDO #3 ******************
      'disable TPDO #3 - required
      call CAN_SDO_WRITE(drvAddress,0x1802,01,32, 0x80000000 + drvAddress + TPD03_COB)
      '***************** drive dynamic mapping of TPDO #4 ******************
      'disable TPDO #4 - required
      call CAN_SDO_WRITE(drvAddress,0x1803,01,32, 0x80000000 + drvAddress + TPD04_COB)
end sub

' Save drive params to EEPROM
' Input parameter
' node : physical address 1,2,3...127
```

```
' usage example:
' call IS_SAVE(127)
public sub IS_SAVE(byval node as long)
  call can_sdo_write(node,0x1010,1,32,0x65766173)
end sub' Change drive address

' Input parameter
' oldaaddress,oldaaddress : physical address 1,2,3...127
' usage example:
' call IS_SET_ADDRESS(127, 1)
public sub IS_SET_ADDRESS(byval oldaaddress as long, byval newaddress as long)
  dim ret as long
  dim na as long
  call can_sdo_write(oldaaddress,0x2F1B,0,16,newaddress)
  call IS_SAVE(oldaaddress)
  sleep 5000 'todo replace with eeprom status read
  ret = user DS402_NMT_RESET_NODE(oldaaddress,0,0)
  sleep 100
  call can_reset
end sub
' Download firmware file to Integrated stepper
' Node : physical address 1,2,3...127
' Input parameter - file name, example ism.dat
' usage example:
' call IS_UPGRADE(2, "ism.dat")
public sub IS_UPGRADE(byval node as long, byval fname as string)
' for monitoring use
' candump -td can0,602:7ff,582:7ff,0:7ff,67f:7ff,5ff:7ff
  dim fdesc as long = 1
  dim verybigstring as string
  dim fsize as long
  dim ret as long
  dim lStartTime as long
  dim lEraseTime as long
  dim lEraseTimeout as const long = 10000
  dim bEraseSuccess as long = 0
  dim lTimeout as long
' save original timeout version
  lTimeout = SDO_TIME_OUT_GET()
  SDO_TIME_OUT_SET(2000)
```

```
  lStartTime = sys.clock
  try
    close #fdesc
  catch else

  end try
  try
  open fname mode="r" as #fdesc
  fsize=loc(fdesc)
  verybigstring=input$(fsize,#fdesc)
  close #fdesc
  ret = user DS402_NMT_RESET_NODE(node,0,0)
  ret = user DS402_NMT_RESET_NODE(127,0,0)
  sleep 100
  ret = can_sdo_read(127,0x1000,0)
  print "Erasing flash"
'SDO 2001 subindex 2 write 1 will erase flash
  call can_sdo_write(127,0x2001,2,16,1)
  lEraseTime = sys.Clock + lEraseTimeout
' read subindex 1 for status. If returns 0 flash erased successfully
  while lEraseTime > sys.clock
    try
      if (can_sdo_read(127,0x2001,1) BAND 0xff ) = 0 then
        bEraseSuccess = 1
      end if
    catch 20152 ' timeout drive irresponsive durity erase

    end try
    if bEraseSuccess then
      goto erased
    end if
  end while
  print "Flash erase failed"

erased:
  if bEraseSuccess then

    print "Flash erased"
    print "Downloading ";fname;" Size :";fsize
```

```
    try
      call CAN_SDO_WRITE_STRING(127,0x2800,0,verybigstring)
    catch 0

    end try
  ' after program download read IDN 2000
  ' if retuns 0 programed successfully and write 1 to jump to program.

    if (can_sdo_read(127,0x2000,0) BAND 0xff ) = 0 then
      print "Download Success, took ";(sys.Clock - lStartTime)/1000;"secs"
      call can_sdo_write(127,0x2000,0,16,1)
      sleep 1000
      ret = USER USER_CAN_RESET_PORT(0,0,0)
    else
      print "Download failed"
    end if
  end if
  catch 0
  end try
'restore timeout
  SDO_TIME_OUT_SET(lTimeout)

end sub
' --- Get Drive Version version ------------
public function IS_Ver(byval node as long) as string
  IS_Ver = can_sdo_read_string(node,0x100a,0)
end function
```