Sommario

DEVICE CANOPEN

1. Introduction

The CANOPEN device is an object that can be used for configuring, monitoring and managing a CANOpen network. CANopen is a communication protocol based on CAN bus. For every reference to CiA documents please refer to Protocol: the international organization that develops and promotes the protocols based on CAN bus (http://www.can-cia.org/cia).

1.1 Device declaration

In the configuration section INTDEVICE unit must be declared so that you have the hardware necessary to use the CANOPEN device. In the INTDEVICE section of the unit of configuration must be added the following definition:

```
;--------------------------------
; Device declaration
;--------------------------------
INTDEVICE
...
<device_name>  CANOPEN TCamp  Speed Port
```

Where:

<device_name>	the name given to the device
CANOPEN	keyword that identifies the device analog positioner
TCamp	sample time device (1÷255 ms)
Speed	indicates the value in Kbps of speed. Accepted values are 1000, 500, 250 and 125. On some hardware, you can also set the value 0. In this case the CANbus speed is set with the dip-switches. The value of the selected speed will be visible in the parameter speed
Port	indicates the CAN unit to which the device is to be referred. If your hardware is slated to only one port CAN use the value 0

Attention: It is necessary that each definition are present on the same line.

1.2 Operation

The main features of the CANOPEN device are:

- CANbus network monitoring: some switches allow you to control the errors of the CAN network, to estimate traffic bus, check some typical CANbus device States
- receiving and decoding of emergency messages typical of the CANOpen protocol
- managing QDO (Qem Data Object) they are objects that do not have a direct relationship with the Protocol, but enhance ensure the possibility of CANOpen communication library configuration. For example the QDO are used to define the refresh rate of the analog inputs, to define the Division factor of position read from a drive, etc.

Here is a list of currently available QDO:

index	Name	Default value	Access	Range of values	Short description
1	aiperiod	100	RW	0÷32767	Update time analog and count inputs expressed in ms
2	setdrivemode	-	RW	0÷255	Sets the drive to work with indicated mode and DS402 specification
3	targetpos	-	W	-	Sets the source location for the drive. Your writing should be done with the drive not operational. This parameter is automatically set with the QDO setdrivemode.
4	divfactor	2	RW	0÷4	Sets the divider for positions. The value is the divisor expressed as an exponent of 2. (example: 2 means $2^2 = 4$)
5	Slave Link Status	-	R		Indicates the linkup of the slaves. The State of bit 0 corresponds to the status of links of the slave with address 1
6	Slave Link Up	-	W		Adds to the CANOpen network one slave in link down. Bit 0 corresponds to the slave with address 1
7	Sync Mode		RW	0÷1	Sets the message SYNC mode (0 for DS401, 1 for DS402 in interpolation mode)
8	Guard Time		RW	-	Sets the Guard Time of the NodeGuarding protocol
9	SDO timeout		RW	-	Sets the timeout value of the SDO messages

1.2.1 "Slave Link Status" use example

Example of a unit created for Qview 6 with local declaration and CANOPEN device declared as REFERENCE.

```
CONST

    READ_LINK_STATUS__TIME    1000

GLOBAL
    ErrorCode        B OUT
    SlavesLinkDown        L OUT    ; link down state (bit0=slaveID1, bit1=slaveID2, ecc)

TIMER
    tmLinkStatus

INTDEVICE
    CanOpen            CANOPEN        REFERENCE

BEGIN

MAIN:


    IF tmLinkStatus
        tmLinkStatus = READ_LINK_STATUS__TIME

                CanOpen.index = 5
        CanOpen.READQDO
        WAIT CanOpen.st_send
        IF NOT CanOpen.qdoerr
            SlavesLinkDown = CanOpen.data
                        ErrorCode = 0
        ELSE
            ErrorCode = 1
        ENDIF
    ENDIF

    WAIT 1
    JUMP MAIN

END
```

1.3 Parameters table

Name	Dimension	Default value	Access type	Unit of measure	Valid range	Write conditions	Description
speed	Word	-	R	Kbps	1000, 500, 250, 125 o 0	-	**Transmission speed** Shows the relevant setting on the device for the transmission speed.
maxrxerr	Word	0	RW	-	-	-	**Maximum number of receive errors** Shows the maximum number of receive errors.
maxtxerr	Word	0	RW	-	-	-	**Maximum number of errors in transmission** Shows the maximum number of errors in transmission.
busload	Word	0	R	Percentage	0÷100	-	**Bus load** Indicates the network traffic load as a percentage, the value is updated every 500 ms.
maxtraffic	Word	0	RW	Percentage	0÷100	-	**Maximum load detected in bus** Shows the maximum percentage of traffic detected on the bus from the last reset of this parameter.
errflags	Word	0	RW	-	0÷32768	-	**Error flag indicator** The value is the result of the conversion to decimal values of some bit flags, at every mistake every flag is placed at 1, to clear them you can set the parameter to 0. All of these error messages are sent from the chip that manages communication CAN. **bit0** overrun error **bit1** rx buffer full error **bit2** idle character detect **bit3** bus-off state **bit4** transmitter error passive **bit5** receiver error passive **bit6** transmitter warning **bit7** receiver warning **bit8** acknowledge error **bit9** CRC error **bit10** form error **bit11** stuff bit error **bit12** bit 0 error **bit13** bit 1 error **bit14** received message queue full **bit15** transmit message queue full **bit16** receiver busy
unit	Byte	0	RW	-	-	-	**CANOpen unit's identification number** Identifies the number of CANOpen drive that will use the READSDO and WRITESDO commands.
index	Long	0	RW	-	0÷65535	-	**Index of Dictionary objects** Identifies the index of the Dictionary objects.

Name	Dimension	Default value	Access type	Unit of measure	Valid range	Write conditions	Description
subindex	Word	0	RW	-	0÷255	-	**Sub-Index of Dictionary objects** Identifies the sub-index of the Dictionary objects in the case that the object is of complex type.
length	Word	0	RW	bytes	-	-	**Dimension SDO** Indicates the size in bytes of the SDO read or written.
data	Long	0	RW	-	-	-	**SDO or QDO value** Indicates the QDO or SDO value to be read or written.
string1÷16	Byte	0	RW	-	-	-	**SDO value** Indicates the SDO value read or written in the case that the data is of type VISIBLE_STRING or OCTET_STRING.
sdoerr	Byte	0	R	-	-	-	**Error while the SDO processing** Indicates an error made while of the SDO reading or writing. The value must be read after *st_send* is placed to 1. The error codes are: **6** Invalid unit **10** invalid group number **11** invalid SDO **12** communication error **13** length invalid SDO **17** wrong unit **28** invalid message length **29** transmission error **34** not valid object length
qdoerr	Byte	0	R	-	-	-	**Error while QDO processing** Indicates an error made while of the QDO reading or writing. The value must be read after *st_send* is placed to 1. The error codes are: **1** object not available **2** error in the QDO running **3** value of the QDO used beyond the limits
sdoabort	Word	0	R	-	-	-	**SDO abort code** CAN protocol error messages. (Not yet implemented)
emcyunit	Byte	0	R	-	-	-	**Emergency unit** Indicates the number of units located in emergency.
emcycode	Word	0	R	-	-	-	**Emergency message code** Indicates the emergency message code.
emcyreg	Byte	0	R	-	-	-	**Error log emergencies** Emergency message error log.
emcyman1÷5	Byte	0	R	-	-	-	**Error constructor** These parameters allow you to read the value of the “manufacturer specific error register” field of the emergency message.
errcode	Byte	0	R	-	0÷100	st_error=1	**Error identification code** Indicates the type of failure intervened in the system. When *st_error* = 1 is present on the *errcode* variable the type of failure intervened and in the *errvalu* variable and e an indication on the cause of the error.
errvalue	Byte	0	R	-	0÷100	-	**Identifying code of the cause of the error** Indicates the cause of the error in the system. The code is valid only if *st_error* = 1.
wrncode	Byte	0	R	-	0÷100	-	**Identification code warning** Indicates the type of warning in the system. The *st_warning* state indicates a minor event that guarantees the operation of the device. When *st_warning* is equal to 1, are present on the *wrncode* variable the type of warning intervened (see the table) and in the *wrnvalue* variable an indication as to the cause of the warning.

Name	Dimension	Default value	Access type	Unit of measure	Valid range	Write conditions	Description
wrnvalue	Byte	0	R	-	0÷100	-	**Identification code of the cause of the warning** Indicates the cause of the warning in the system. **1**: attempt to write access on a parameter where the conditions for writing were not met, **2**: attempt to execute a command when the conditions weren't met.

1.4 Status table

Noae	Short description	Descrizption
st_send	1	**SDO or QDO commands executed** When the State takes the value 1 indicates that the SDO command (*READSDO* or *WRITESDO*) or QDO (*READQDO* or *WRITEQDO*) it's executed.
st_emcy	0	**State of emergency** The device has received an emergency message when the State is worth 1.
st_error	0	**Error presence** Indicates the error status of the device, to recognize the type of error you must refer to the *errcode* and *errvalue* variables: **0**: error not present, **1**: error present
st_warning	0	**The presence of a warning** Indicates the device warning status, to recognize the type of warning you must refer to the *wrncode* eand *wrnalue* variables: **0**: warning not present, **1**: warning present

1.5 Commands table

Name	Condition	Description
READSDO	st_send=1	**Reading an SDO** Read command. The following parameters must be set: *unit*, *index*, *subindex*. Upon activation of the *st_send* state you can verify the parameters to check for an error. If the operation went successful then in *data* or *string1÷16* parameters, you will find the value of the object and in the *length* parameter you find the size of the data.
WRITESDO	st_send=1	**Writing an SDO** Command of writing an SDO. The following parameters must be set: *unit*, *index*, *subindex*, *length* and *data*. Upon activation of the *st_send* state you can verify the parameters to check for an error.
READQDO	st_send=1	**Reading an SDO** Read command a QDO. The following parameters must be set: *unit*, *index*, *subindex*. Upon activation of the *st_send* state you can verify the *qdoerr* parameter to check for an error.
WRITEQDO	st_send=1	**Writing an SDO** A write command QDO. The following parameters must be set: *unit*, *index*, *subindex*, *length* and *data*. Upon activation of the *st_send* state you can check the *qdoerr* parameter to check for an error.
READEMCY	st_emcy=1	**Reading an emergency message** It commands an emergency message reading. When the *st_emcy* state becomes active means that the device has received an error message, and you can read it with this command. The *emcyunit*, *emcycode*, *emcyreg* and *emcyman1÷5* parameters represent the data in your message.
RSERR	st_error=1	**Reset error state** Reset the *st_error* state.
RSWRN	st_warning=1	**Reset warning state** Reset the *st_warning* state.