Tech-note

CANopen HMI Communication Driver for JMobile

This Technical Note contains the information needed to connect the HMI to control devices using the CANopen protocol with HMI profile.

Exor International S.p.A.
ptn0427
Ver. 1.0

Contents

CANopen HMI Driver

The CANopen HMI communication driver has been designed to connect UniOP HMI products to a CANopen network. A new device communication profile has been developed for the HMI. This profile takes advantage from the advanced user interface features of the UniOP products, while retaining the simple networking concept supported by the CANopen network.
The basic idea is create a client/server communication structure where the HMI is the client and the CANopen controller is the server.

Connection to CANopen network requires the optional CANopen communication module.

Platform	Optional module
UniOP eTOP Series 400	TCM09
UniOP eTOP Series 500	PLCM01

Please note that changes in the controller protocol or hardware, which may interfere with the functionality of this driver, may have occurred since this documentation was created. Therefore, always test and verify the functionality of the application. To accommodate developments in the controller protocol and hardware, drivers are continuously updated. Please ensure that the latest driver is used in the application.

UniOP CANopen HMI Profile

In this communication model the HMI initiates the communication sessions, acting as a source of messages.
The basic messages are PDO messages with the standard size of 8 bytes.
The COB-ID of the messages is defined in a way that makes clear, from the well-known CANopen rules, what is the target of the PDO message.
The format of the PDO message has been defined according to a custom application layer protocol. This application layer protocol defines a device-independent communication profile optimized for HMI applications.
When the CANopen master controller receives the PDO message, it will interpret its contents and produce a PDO message with the response addressed to the HMI device.

The definition of this client/server relationship is independent of the CANopen Master in the sense that it can easily be supported in any particular CANopen master system. The resulting solution is easily portable to any CANopen master.

JMobile Studio offers a user interface that adapts itself to show the typical addressing model of CANopen master controller where the panel is going to be connected.
Adapting to different masters is possible using a profile customization file that may contain data definitions for different controller types.

Profile Details

This chapter provides the specification of the HMI profile and describes the subset of the request/response formats used by this implementation of the protocol.
The communication driver in the HMI generates PDO messages initiating communication request sessions as soon as the HMI runtime requires data from the protocol.
The panel is using the first transmit PDO identified by the COB-ID 0x180 combined with the Node Number assigned to the panel.

The communication profile uses only one transmit PDO and one receive PDO; the limited number of bytes available in standard PDO message maybe limiting, in some cases, the driver capabilities especially in terms of performance.

Request Format: HMI to Controller (Transmit PDO)

The PDO message transmitted by the HMI is formatted according to Table 1.

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Offset Low	Offset High	Data 0	Data 1	Data 2	Data 3	Data Length and Job Number	Operation Type and Controller ID

Table 1

The request frame includes the following elements:

Offset Low Low byte of the offset (16 bits address) for the requested block of data

Offset High High byte of the offset (16 bits address) for the requested block of data

Data 0 … Data 3 Data for Write Operations; not used in Read Operations

Data Length and Job Number Contains:
- number of requested bytes
- job Number indicator;

Table 2 shows the details of the bit assignment

Operation Type and Controller ID Contains:
- type of operation requested
- the Controller ID that identifies the target of the message;

Table 3 shows the details of the bit assignment

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Data Length [1]	Data Length [0]	Job Number [5]	Job Number [4]	Job Number [3]	Job Number [2]	Job Number [1]	Job Number [0]

Table 2

The “Data Length” parameter is coded in 2 bits and takes values between 1 and 4 according to the following rules:

00	1 bytes
01	2 bytes
10	3 bytes
11	4 bytes

Note that the elementary size of each data item depends on the Controller memory organization.
The “Job Number” occupies 6 bits and can have values between 0 and 63; the “Job number” parameter is placed as last element in the PDO to ensure data consistency; the PLC program running the controller should constantly monitor the value of the “Job Number” parameter and consider the received message as valid only when detecting a change in the value of the “Job Number” field. “Job Number” is automatically increased at each new communication session (new request frame).

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Operation Type	Controller ID [6]	Controller ID [5]	Controller ID [4]	Controller ID [3]	Controller ID [2]	Controller ID [1]	Controller ID [0]

Table 3

The “Operation Type” uses one bit with the following definition:

0	Read	data is transferred from controller to UniOP
1	Write	data is transferred from UniOP to controller

The “Controller ID” uses 6 bits; it represents the Node Number in the CANopen network of the master controller addressed by the current request.
This parameter is required in case the CAN network has more than one master controller; the CANopen standard defines in fact the COB-ID of the messages in a way that all the partners of the bus known the originator. In case more than one master device is present in the same network, the “Controller ID” field will specify the target of each individual request message. Only the master controller that recognizes in this field its own Node ID will consider the message and process the PDO contents.

Response Format: Controller to Panel (Receive PDO)

The PDO message returned by the controller must be formatted as defined in Table 4.

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Status Flag / Error Code	Dummy – Always 0	Data 0	Data 1	Data 2	Data 3	Data Length and Job Number	Operation Type and Controller ID

Table 4

The request frame consists of the following elements:

Status Flag / Error Code — Contains the information related to the execution of the operation type of the request; Table 5 shows the coding information

Data 0 … Data 3 — Contain the data information returned to the panel in response to a Read request

Data Length and Job Number — It is the copy of the corresponding field of the request frame

Operation Type and Controller ID — It is the copy of the corresponding field of the request frame

	Status Flag / Error Code	
Operation Type in the Request Frame	**No Errors**	**Error**
Read	0x01	0x81
Write	0x02	0x82

Table 5

Protocol Editor Settings

Add (+) a driver in the Protocol editor and select the protocol called “CANopen HMI” from the list of available protocols.

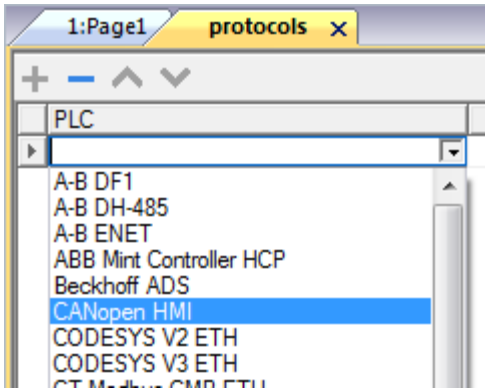

Figure 1

The driver configuration dialog is shown in figure.

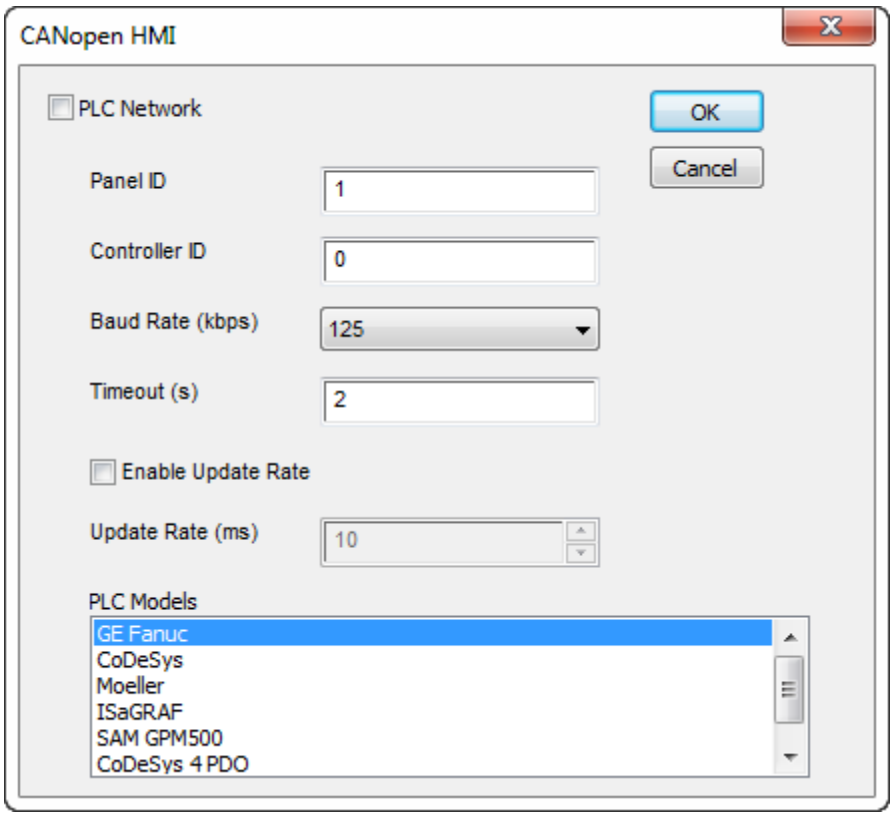

Figure 2

Panel ID CANopen node ID assigned to the HMI

Controller ID CANopen Node ID assigned to the CAN controller device

Baud Rate (kbps) Speed of the CANopen network

Timeout (s) Maximum allowed time the driver will wait for a response from the PLC before reporting a communication error

Enable Update Rate Use this option to enable a wait time between two communication requests

Update Rate (ms) Minimum interval time between two requests; it can be useful when the bus load needs to be properly controller and limited

PLC Models The list allows selecting the controller model you are going to connect to. The selection will influence the data range offset per each data type according to the specific controller memory resources

PLC Network The protocol allows the connection of multiple controllers to one operator panel. To set-up multiple connections, check “PLC network” checkbox and enter the node ID per each slave you need to access.

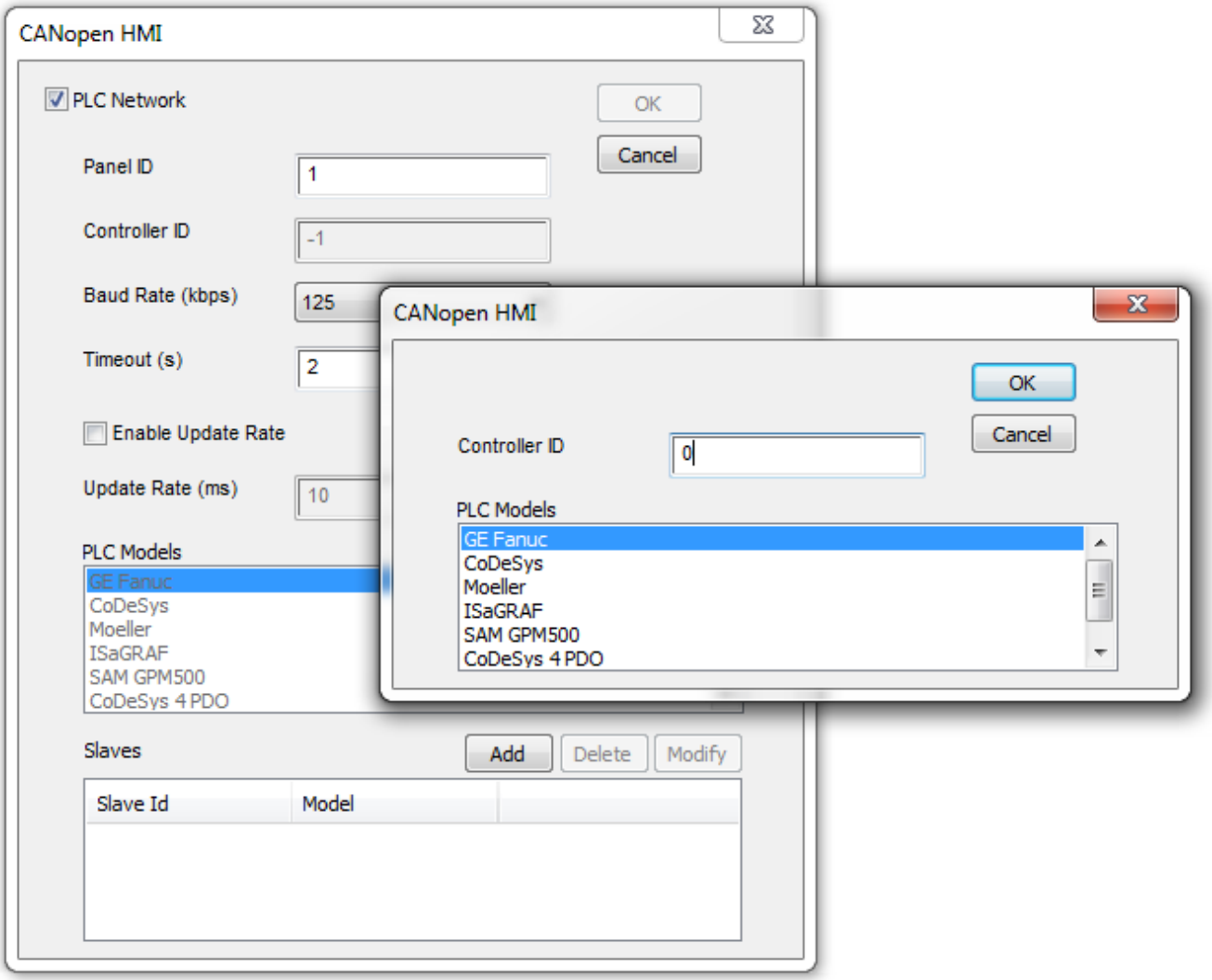

Figure 3

Connecting UniOP to CODESYS V2 Controllers

This chapter describes all the steps you have to follow in order to establish a successful connection between UniOP and CODESYS CANopen master controller.
The PLC support program has been developed with CODESYS programming software version 2.

PLC Library Call

The server function running in the PLC program has been designed in the form of Library called “Exor_Canh”, written using the “ST” programming language. Proper working example is available on demand.

The Function Block parameters are the following:

MasterID	CANopen Master Node number;
MinBound	Lower limit of the PLC memory addressable (visible) by UniOP
MaxBound	Upper limit of the PLC memory addressable (visible) by UniOP
HHIr	Offset in the PLC memory where the PDO message received from the panel is mapped
HMIt	Offset in the PLC memory where the PDO message to be sent to the panel is mapped
MemPt	Offset in the PLC memory where the data is received
Status	Status

The PLC Function block support the use of more than one panel simply repeating the call of the same function for all the additional units specifying before each call the proper calling parameters.

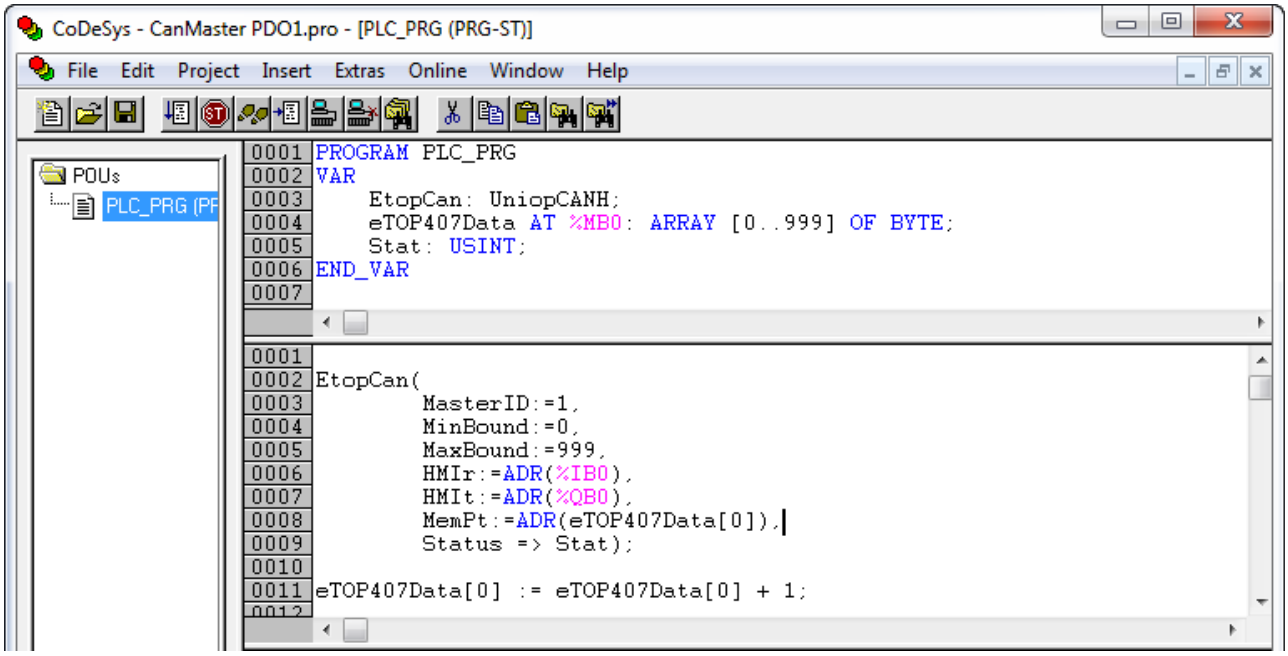

Figure 4

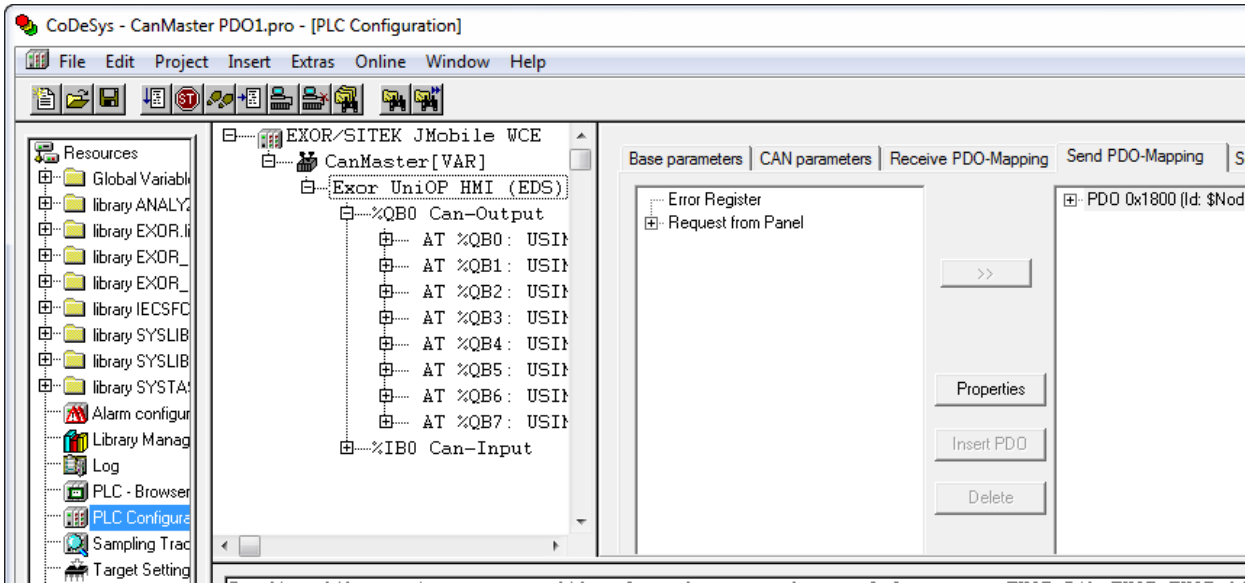

Figure 5

CODESYS V2 4PDO

In some cases it is useful to choose the model “CODESYS 4 PDO” where 4 PDO objects are used for transmission and 4 for reception. This solution may provide higher communication speed between the two devices.
To operate with 4 PDO the correct model should be set in HMI project and the PDOs for receive and transmit slots.

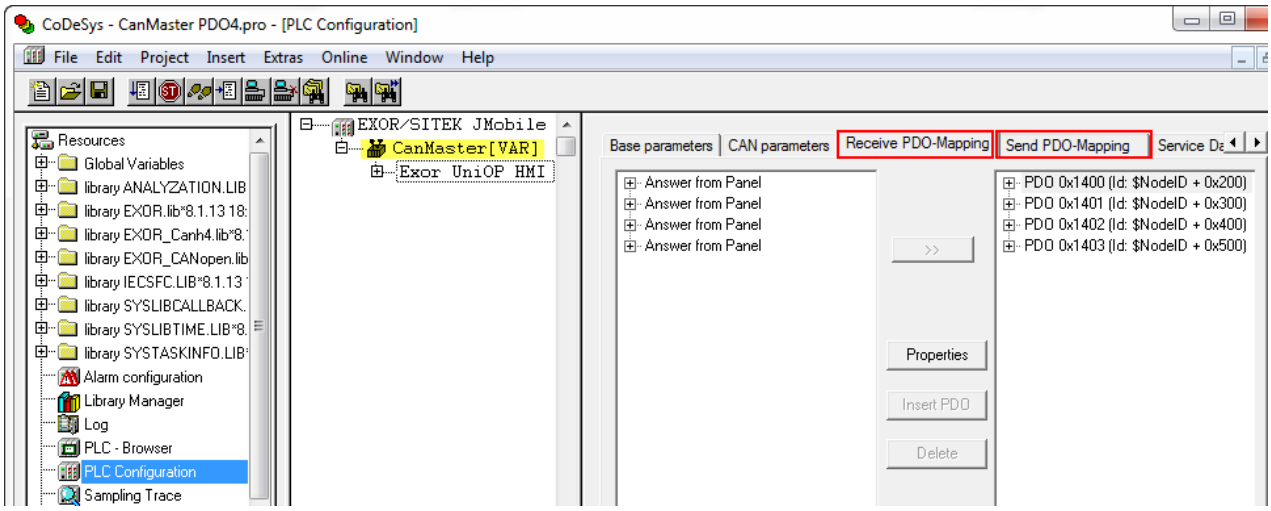

Figure 6

Note: CANopen Master PLC Configuration must be configured properly. In case of “CODESYS 4 PDO”.

Communication Status

The current communication status can be displayed using the dedicated system variables. Please refer to the User Manual for further information about available system variables and their use.

The codes supported for this communication driver are:

Error	Notes
NAK	Controller replies with a not acknowledge.
Timeout	Request is not replied within the specified timeout period; ensure the controller is connected and properly configured for network access
Line Error	Returned when an error on the communication parameter setup is detected (baud rate); ensure the communication parameter settings of the controller is compatible with panel communication setup
Invalid response	The panel did receive from the controller a response, but its format or its contents or its length is not as expected; ensure the data programmed in the project are consistent with the controller resources.
CAN port not found	Make sure option module is correctly plugged
CAN port in use	Make sure option module is not already in use
General error	Error cannot be identified; should never be reported; contact technical support