CANopen Connector

This driver is still in QA validation and will be available soon. If you have any questions, please contact Tatsoft support.

Reads and writes CANopen devices (I/O modules, drives, gateways) over a CAN bus, per CiA 301.

- Name: **CANopen**
- Version: 1.0.0.2
- Protocol: CANopen (CiA 301 v4.2.0)
- Interface: CAN bus (Linux SocketCAN, Windows PEAK PCAN)
- Runtime: .NET 2.0 (Multiplatform)
- Configuration:
  - Devices / Protocols

Overview
Channel Configuration
Nodes Configuration
Points Configuration
Driver Revision History

Overview

Communication Driver Information	
Driver name	CANopen
Assembly Name	T.ProtocolDriver.CANopen
Assembly Version	1.0.0.2
Available for Linux	True (x64, ARM64 and 32-bit ARM, for example the Eaton XV HMI panels)
Devices supported	CANopen devices (CiA 301): I/O modules, drives, gateways
Manufacturer	N/A (Not Applicable)
Protocol	CANopen
Interface	CAN bus
PC Requirements	Linux: a CAN interface with a SocketCAN driver (can0). Windows: a PEAK CAN adapter (USB, PCI or LAN) with the PEAK device driver, which installs PCANBasic.dll.

FrameworX acts as the CANopen client and, by default, as the NMT master of the network. The connector supports:

- SDO read and write (expedited and segmented) of any object dictionary entry.
- TPDO1 to TPDO4 received from the device and RPDO1 to RPDO4 sent to the device, with the COB-IDs of the pre-defined connection set.
- Emergency messages (EMCY).
- NMT commands, with node supervision by heartbeat or node guarding.
- An optional SYNC producer.

The connector uses no third-party library. On Linux the bit rate is set on the CAN interface, not in FrameworX, for example: `ip link set can0 up type can bitrate 250000 restart-ms 100`.

Channel Configuration

Protocol options, in order: **Adapter;Interface;BitrateKbps;NmtMaster;SyncPeriodMs;SdoTimeoutMs;AutoRestart**. Example: `Auto;can0;250;true;0;1000;true`.

- ***<Adapter>***: Auto, SocketCAN or PCAN. Auto selects PCAN when Interface is a PCAN channel name, SocketCAN otherwise. SocketCAN is Linux only; PCAN is Windows only.
- ***<Interface>***: CAN port. Linux: the network interface name (can0, can1). Windows: PCAN_USBBUS1 to PCAN_USBBUS16, PCAN_PCIBUS1 to PCAN_PCIBUS16 or PCAN_LANBUS1 to PCAN_LANBUS16.
- ***<BitrateKbps>***: bus bit rate in kbit/s (10 to 1000), the same on every device of the network. Used with PCAN; on Linux the interface's own bit rate applies.
- ***<NmtMaster>***: FrameworX is the NMT master. It starts each node when the bus opens and when a node boots, and runs node guarding. Clear it when another master, for example a CODESYS CANopen Manager, owns the network. FrameworX then still uses each node's default SDO channel, which that master also uses: prefer PDO points, and keep SDO points out of the other master's start-up configuration.
- ***<SyncPeriodMs>***: SYNC producer period in ms, 10 or more; 0 (default) sends no SYNC. Needed by synchronous PDOs when no other device produces SYNC.
- ***<SdoTimeoutMs>***: time-out of one SDO answer in ms (default 1000).
- ***<AutoRestart>***: with NmtMaster, FrameworX keeps each supervised node in a desired state: Operational, or the last Start, Stop or Pre-operational written to its NMT:State point. When the node reports another state, is lost while its heartbeat time is configured, or its boot-up after a reset never comes, FrameworX sends the command again, at most every 5 s. Clear it when a node that the device stops by itself must stay stopped; bus open and boot-up then always send Start. On by default.

Channels that use the same CAN port share the first channel's settings. Only one FrameworX process can open a CAN port.

Nodes Configuration

Station Configuration

Station, in order: **NodeId;Monitoring;MonitorTimeMs;HeartbeatProducerMs**. Example: `2;Heartbeat;1500;1000`.

- ***<NodeId>***: CANopen node-ID of the device, 1 to 127.
- ***<Monitoring>***: how FrameworX knows the node is alive. **Heartbeat** consumes the node's heartbeat. **NodeGuarding** polls the node and needs NmtMaster on the channel. **None**: no supervision.
- ***<MonitorTimeMs>***: Heartbeat: time-out in ms, above the heartbeat period (for example 1.5 times). NodeGuarding: guard time in ms; the node is lost after 3 guard times without a reply.
- ***<HeartbeatProducerMs>***: Heartbeat only. Heartbeat period in ms that FrameworX writes to the device's object 1017h after every boot-up and when it repairs a lost node (NmtMaster on). A reset returns 1017h to its stored value, usually 0, so without this setting the node is reported lost after a power cycle. 0 leaves the device's own setting; then store the period in the device.

Points Configuration

The syntax for the CANopen communication points is ***<Object>:<Location>:<DataType>***.

Object	Location	Description
SDO	index.sub-index in hexadecimal, optionally .bit for a BOOL	Reads or writes one object dictionary entry. E.g. SDO:1018.01:U32, SDO:6200.01.3:BOOL.
TPDO1 to TPDO4	byte offset 0-7, optionally .bit for a BOOL	Last process data the node sent (COB-ID 180h/280h/380h/480h + node-ID). Read only. Bad quality until a PDO arrives, and while the node is lost or not Operational. E.g. TPDO1:0:U16.
RPDO1 to RPDO4	byte offset 0-7, optionally .bit for a BOOL	Process data sent to the node (COB-ID 200h/300h/400h/500h + node-ID). Before the first write, FrameworX reads the RPDO mapping (1600h-1603h) and the current value of every mapped object from the device; each write sends the whole PDO with those values. While the node is lost or not Operational, an RPDO write fails and changes nothing, and RPDO points read Bad. E.g. RPDO1:0:I16.
EMCY	Code, Register, Data or Count	Last emergency message of the node: error code, error register, manufacturer data, and the number of messages received.
NMT	State or Online	State: the node's NMT state; write 1 (Start), 2 (Stop), 128 (Pre-operational), 129 (Reset node) or 130 (Reset communication) to send the command. Online: the node is supervised and alive (needs Heartbeat, or NodeGuarding with NmtMaster).

Data types: BOOL, I8, I16, I24, I32, I64, U8, U16, U24, U32, U64, R32, R64, VS (VISIBLE_STRING) and OS (OCTET_STRING). EMCY and NMT points take no data type. Use the .bit form for bits; the Bit modifier is not supported.

The Designer **Test** button reads and writes points without sending NMT commands, SYNC or node guarding, so it does not change the state of the network.

Driver Revision History

CANopen Revision History	
Version	**Notes**
1.0.0.2	Initial release, FrameworX 10.1.7.

In this section...