LANTRONIX®

Application Note

CANopen Gateway Functions for AVL Devices

Part Number APP-0048
Revision A April 2020

Intellectual Property

Lantronix is a registered trademark of Lantronix, Inc. in the United States and other countries.

Patented: http://patents.lantronix.com; additional patents pending.

All trademarks and trade names are the property of their respective holders.

Contacts

Lantronix, Inc.
7535 Irvine Center Drive, Suite 100
Irvine, CA 92618, USA
Toll Free: 800-526-8766
Phone: 949-453-3990
Fax: 949-453-3995

Technical Support
Online: www.lantronix.com/support

Sales Offices
For a current list of our domestic and international sales offices, go to the Lantronix web site at www.lantronix.com/about/contact

Disclaimer

All information contained herein is provided “AS IS.” Lantronix undertakes no obligation to update the information in this publication. Lantronix does not make, and specifically disclaims, all warranties of any kind (express, implied or otherwise) regarding title, non-infringement, fitness, quality, accuracy, completeness, usefulness, suitability or performance of the information provided herein. Lantronix shall have no liability whatsoever to any user for any damages, losses and causes of action (whether in contract or in tort or otherwise) in connection with the user’s access or usage of any of the information or content contained herein. The information and specifications contained in this document are subject to change without notice.

Revision History

Date	Rev.	Comments
Month Year	A	Initial document.

For the latest revision of this product document, please check our online documentation at www.lantronix.com/support/documentation.

Table of Contents

1. About this Document

This application note provides information about the CANopen gateway functions for AVL devices. CANopen gateway functions are supported in the AVL firmware 3.5.x and higher.

2. CANopen

1.1. About CANopen

CANopen is a CAN-based communication system, standardized by CAN in Automation (CiA). https://www.can-cia.org The document CiA® 309 describes the CANopen gateway functions.

1.2. AVL CANopen commands and settings

1.1.1. Enable and disable CANopen gateway functions

1.1.1.1. sys.can.CANopen.enable

Functionality:	Enable the CANopen interface
Example:	$pfal,sys.can.CANopen.enable $pfal,sys.canb.CANopen.enable $pfal,sys.can.CANopen.enable,0A
Syntax:	$pfal,sys.can.CANopen.enable,[<hex node id>]
Parameter:	- <node id>: Hexadecimal CANopen node id (12 ist the default value)

1.1.1.2. sys.can.CANopen.disable

Functionality:	Disable the CANopen interface
Example:	$pfal,sys.can.CANopen.disable
Syntax:	$pfal,sys.can.CANopen.disable

1.1.1.3. sys.can.CANopen.cmd

Functionality:	Execute CANopen command
Example:	$pfal,sys.can.CANopen.cmd,“[100] 4 r 0x1008 0 vs”
Syntax:	$pfal,sys.can.CANopen.cmd,<command string>
Parameter:	- <command string>: CANopen gateway command string according CiA 309-3 (enclosed in double quotation)
Answer:	$<SYS.CAN.CANopen.cmd> <CANopen gateway answer string> $SUCCESS

1.1.2. CANopen configuration settings

1.1.1.4. DEVICE.CANopen.STARTUP

Functionality:	Enable or disable the CANopen interface
Example:	$PFAL,CNF.Set,DEVICE.CANopen.STARTUP=on,0,5 $PFAL,CNF.Set,DEVICE.DTCO.D8=off
Syntax:	$PFAL,CNF.Set,DEVICE.CANopen.STARTUP=on,<can_device>,<node id> $PFAL,CNF.Set,DEVICE.CANopen.STARTUP=off
Parameter:	- <can_device>: CAN device 0-CAN 1-CANB - <node id>: hexadecimal value for the CANopen node address

1.1.3. Dynamic variables

1.1.1.5. CO.pdo

Functionality:	Display the CANopen PDO event string The event message strings are queued as character strings in the order in which they were received. The PDO received first is read from the queue with the variable CO.pdo. If the queue is empty, CO.pdo contains an empty character string.
Example:	$PFAL,Msg.Send.Serial0,0,“PDO: &(CO.pdo)”
Output:	PDO event string according CiA 309-3
Output example:	$PDO: pdo 1 3 0x01 0x02 0x0403*74

1.1.4. Events

1.1.1.6. SYS.eCO.PDO

Functionality:	The event SYS.eCO.PDO occurs when a PDO has been received
Example:	$PFAL,CNF.Set,AL20=SYS.eCO.PDO:Msg.Send.Serial0,0,“PDO: &(CO.pdo)”

1.1.5. CANopen LUA events

1.1.1.7. event ALARM_SYS_CO_PDO_RECEIVED

Functionality:	ALARM_SYS_CO_PDO_RECEIVED occurs when a PDO has been received
Example:	if e.type == ALARM_SYS_CO_PDO_RECEIVED then os.trace(“CANopen PDO data "%s" (%d bytes) (%d ms)”, e.u_recvdata, e.u_recvlen, t)

Fields:	- e.u_recvdata: first PDO event string in the event FiFo - e.u_recvlen: length of PDO event string

1.3. Gateway Commands

1.1.6. Command syntax

The syntax is specified in CiA 309-3: [[net] node] <command> <parameter>

Parameter	Description
net: node: <command>: <parameter>:	Specifies the CANopen network for the command - always 1 Specifies the CANopen node for the command CANOPEN gateway command One or more parameters of the command

1.1.7. Upload SDO

Functionality:	Read SDO data from CANopen device
Example:	$pfal,sys.can.CANopen.cmd,“[1] 4 r 0x1008 0 vs”
Syntax:	[[net] node] r[ead] <multiplexor> <datatype>
Parameter:	- <multiplexor>: <index> <subindex> - <datatype>: b|i8|i16|i32|u8|u16|u32|r32|vs|os|us|t|td|d|i24|r64|i40| i48|i56|i64|u24|u40|u48|u56|u64

1.1.8. Download SDO

Functionality:	Write SDO data to CANopen device
Example:	$pfal,sys.can.CANopen.cmd,“[2] 3 w 0x1017 0 i16 1000”
Syntax:	[[net] node] w[rite] <multiplexor> <datatype> <data>
Parameter:	- <multiplexor>: <index> <subindex> - <datatype>: b|i8|i16|i32|u8|u16|u32|r32|vs|os|i64|u64

1.1.9. Start node

Functionality:	Start specified CANopen node
Example:	$pfal,sys.can.CANopen.cmd,“[3] 4 start”
Syntax:	[[net] node] start

1.1.10. Stop node

Functionality:	Stop specified CANopen node
Example:	$pfal,sys.can.CANopen.cmd,“[4] 4 stop”
Syntax:	[[net] node] stop

1.1.11. Set node to pre-operational

Functionality:	Set specified node to pre-operational mode
Example:	$pfal,sys.can.CANopen.cmd,“[5] 4 preop”
Syntax:	[[net] node] preop[erational]

1.1.12. Reset node

Functionality:	Reset specified CANopen node
Example:	$pfal,sys.can.CANopen.cmd,“[6] 4 reset node”
Syntax:	[[net] node] reset node

1.1.13. Reset communication

Functionality:	Reset communication of specified CANopen node
Example:	$pfal,sys.can.CANopen.cmd,“[6] 4 reset node”
Syntax:	[[net] node] reset node

1.1.14. Configure SDO timeout

Functionality:	Set the timeout abort error used by the gateway’s SDO client
Example:	$pfal,sys.can.CANopen.cmd,“[7] set sdo_timeout 1000”
Syntax:	[net] set sdo_timeout <ms>

1.1.15. Set default node-ID

Functionality:	Set the default node id of the CANopen gateway device
Example:	$pfal,sys.can.CANopen.cmd,“[8] set sdo_timeout 1000”
Syntax:	[net] set node <value>

1.1.16. Enable or disable block transfer

Functionality:	Enable or disable block transfer of the CANopen gateway device
Example:	$pfal,sys.can.CANopen.cmd,“[9]set sdo_block 1”
Syntax:	[net] set sdo_block 0|1

1.1.17. Configure RPDO

Functionality:	Configure RPDO
Example:	$pfal,sys.can.CANopen.cmd,“[10] set rpdo 1 0x80000000 event 0” ;delete rpdo 1 $pfal,sys.can.CANopen.cmd,“[11] set rpdo 1 0x181 event 3 u8 u8 u16”
Syntax:	[[net] node] set rpdo <nr> <COB> <tx-type> <nr-of-data> <map-obj1>[..<map-obj16>]
Parameter:	<nr>: Nr of the PDO configuration 1…16 <COB>: node address of the PDO to receive and flags; if COB=0x80000000 -> deactivate setting <tx-type>: “event” <nr-of-data>: Number of data objects in the PDO <map-objx>: data type of the object - b|i8|i16|i32|u8|u16|u32|r32|vs|os|i64|u64

1.1.18. Display RPDO configuration

Functionality:	Display RPDO configuration
Example:	$pfal,sys.can.CANopen.cmd,"[10] info rpdo 1
Syntax:	[[net] node] info rpdo <nr>
Parameter:	<nr>: Nr of the PDO configuration 1…16