CANopen
Gateway Server DS309-3
User Manual

© port GmbH, Halle 8.1.2014; CANopen Gateway Server Version 5.4.2
Disclaimer
All rights reserved

The programs, boards and documentations supplied by port GmbH are created with due diligence, checked carefully and tested on several applications.

Nevertheless, port GmbH can not take over no guarantee and no assume del credere liability that the program, the hardware board and the documentation are error-free respective are suitable to serve the special purpose.

In particular performance characteristics and technical data given in this document may not be constituted to be guaranteed product features in any legal sense.

For consequential damages, which are emerged on the strength of use the program and the hardware boards therefore, every legal responsibility or liability is excluded.

port has the right to modify the products described or their documentation at any time without prior warning, as long as these changes are made for reasons of reliability or technical improvement.

All rights of this documentation lie with port. The transfer of rights to third parties or duplication of this document in any form, whole or in part, is subject to written approval by port. Copies of this document may however be made exclusively for the use of the user and his engineers. The user is thereby responsible that third parties do not obtain access to these copies.

The soft- and hardware designations used are mostly registered and are subject to copyright.

CANopen®
is registered trademark, licensed by CiA - CAN in Automation e.V., Germany.

EtherCAT®
is registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.

We are thankful for hints of possible errors and may ask around for an information.

We will go all the way to verify such hints fastest

Copyright

© 2014 port GmbH
Regensburger Straße 7
D-06132 Halle
Tel. +49 345 - 777 55 0
Fax. +49 345 - 777 55 20
E-Mail service@port.de
Internet http://www.port.de
# Table of Contents

1. Overview .................................................. 7
   1.1. Features .............................................. 8
   1.2. Installation and Start .................................. 8
       1.2.1. CAN driver ....................................... 8
       1.2.2. Command line options ......................... 9
           1.2.2.1. Common options ............................... 9
           1.2.2.2. Options for Linux, EtherCAN, IGW900 .......... 9
           1.2.2.3. Options for EMS Wünsche CPC ................ 9
           1.2.2.4. Options for Peak ............................. 10
           1.2.2.5. Options for Kvaser ........................... 10
   2. Definitions ............................................ 11
       2.1. Commands .......................................... 12
           2.1.1. Command Request ............................ 12
           2.1.2. Command Response .......................... 12
           2.1.3. Event triggered messages ................... 13
   3. Network access command specification .................. 14
       3.1. SDO access commands ............................... 14
           3.1.1. Upload SDO .................................... 14
           3.1.2. Download SDO ................................. 14
           3.1.3. Configure SDO timeout ....................... 14
           3.1.4. SDO Block Event ............................. 14
       3.2. PDO access commands ............................... 15
           3.2.1. Configure RPDO command .................... 15
           3.2.2. Configure TPDO command .................... 15
           3.2.3. Read PDO data command ..................... 15
           3.2.4. Write PDO data command ..................... 16
           3.2.5. RPDO Event .................................. 16
       3.3. CANopen NMT commands ............................. 16
3.3.1. Start Node ........................................... 16
3.3.2. Stop node ........................................... 16
3.3.3. Set node Preoperational ......................... 17
3.3.4. Reset node ........................................... 17
3.3.5. Reset communication ............................. 17
3.3.6. Enable node guarding ............................ 17
3.3.7. Disable node guarding ............................ 18
3.3.8. Enable heartbeat ................................. 18
3.3.9. Disable heartbeat ................................. 18
3.3.10. Error control Event .............................. 18

3.4. Device Failure management ....................... 18
3.4.1. Read device error command ..................... 18
3.4.2. Emergency Event ................................. 19

3.5. CANopen interface configuration commands .... 19
3.5.1. Initialize gateway ............................... 19
3.5.2. Store configuration .............................. 19
3.5.3. Restore configuration ............................ 20
3.5.4. Set heartbeat producer ......................... 20
3.5.5. Set node id ........................................ 20
3.5.6. Start emergency consumer ...................... 20
3.5.7. Stop emergency consumer ...................... 20

3.6. Gateway management commands .................. 20
3.6.1. Set default network ............................. 20
3.6.2. Set default node id .............................. 20
3.6.3. Get version ....................................... 21

4. Command Extensions ..................................... 22
4.1. Status and Configuration ........................... 22
4.1.1. Set master ........................................ 22
4.1.2. Set verbose response ............................ 22
4.1.3. Get default network .............................. 22
4.1.4. Get default node ................................. 22
4.1.5. Get Build information ........................... 23
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1.6.</td>
<td>Get CANopen node Id of server</td>
<td>23</td>
</tr>
<tr>
<td>4.1.7.</td>
<td>Get CAN bitrate</td>
<td>23</td>
</tr>
<tr>
<td>4.1.8.</td>
<td>Get SDO timeout</td>
<td>23</td>
</tr>
<tr>
<td>4.1.9.</td>
<td>Get TPDO setup</td>
<td>23</td>
</tr>
<tr>
<td>4.1.10.</td>
<td>Get RPDO setup</td>
<td>24</td>
</tr>
<tr>
<td>4.2.</td>
<td>SDO</td>
<td>24</td>
</tr>
<tr>
<td>4.2.1.</td>
<td>Register SDO Server Write Indikation</td>
<td>24</td>
</tr>
<tr>
<td>4.2.2.</td>
<td>Unregister SDO Server Write Indikation</td>
<td>24</td>
</tr>
<tr>
<td>4.3.</td>
<td>PDO</td>
<td>24</td>
</tr>
<tr>
<td>4.3.1.</td>
<td>Register PDO</td>
<td>24</td>
</tr>
<tr>
<td>4.3.2.</td>
<td>Unregister PDO</td>
<td>24</td>
</tr>
<tr>
<td>4.4.</td>
<td>NMT-Master</td>
<td>25</td>
</tr>
<tr>
<td>4.4.1.</td>
<td>Register Nodeguarding</td>
<td>25</td>
</tr>
<tr>
<td>4.4.2.</td>
<td>Unregister Nodeguarding</td>
<td>25</td>
</tr>
<tr>
<td>4.4.3.</td>
<td>Register heartbeat</td>
<td>25</td>
</tr>
<tr>
<td>4.4.4.</td>
<td>Unregister heartbeat</td>
<td>25</td>
</tr>
<tr>
<td>4.4.5.</td>
<td>Register emergency</td>
<td>25</td>
</tr>
<tr>
<td>4.4.6.</td>
<td>Unregister emergency</td>
<td>26</td>
</tr>
<tr>
<td>4.4.7.</td>
<td>Enable Sync Producer</td>
<td>26</td>
</tr>
<tr>
<td>4.4.8.</td>
<td>Disable Sync Producer</td>
<td>26</td>
</tr>
<tr>
<td>4.5.</td>
<td>LSS Master</td>
<td>26</td>
</tr>
<tr>
<td>4.5.1.</td>
<td>Switch Selective</td>
<td>26</td>
</tr>
<tr>
<td>4.5.2.</td>
<td>Switch global</td>
<td>26</td>
</tr>
<tr>
<td>4.5.3.</td>
<td>Configuration of node id</td>
<td>27</td>
</tr>
<tr>
<td>4.5.4.</td>
<td>Request node</td>
<td>27</td>
</tr>
<tr>
<td>4.5.5.</td>
<td>Identify LSS slaves</td>
<td>27</td>
</tr>
<tr>
<td>4.5.6.</td>
<td>Bitrate Configuration</td>
<td>27</td>
</tr>
<tr>
<td>4.5.7.</td>
<td>Bitrate activation</td>
<td>28</td>
</tr>
<tr>
<td>4.5.8.</td>
<td>Store Configuration</td>
<td>28</td>
</tr>
<tr>
<td>4.5.9.</td>
<td>Identify unconfigured LSS slaves</td>
<td>28</td>
</tr>
<tr>
<td>4.5.10.</td>
<td>LSS Event</td>
<td>28</td>
</tr>
<tr>
<td>4.6.</td>
<td>Miscellaneous</td>
<td>28</td>
</tr>
</tbody>
</table>
1. Overview
The CANopen Gateway Server implements the protocol DS309-3. It can be accessed from a remote workstation or locally.
### 1.1. Features

<table>
<thead>
<tr>
<th>Service</th>
<th>Instances</th>
<th>Full version</th>
<th>Starterkit</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDO-Client</td>
<td>127</td>
<td></td>
<td>2(*)</td>
</tr>
<tr>
<td>SDO-Server</td>
<td>1</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>PDO-Producer</td>
<td>512</td>
<td></td>
<td>1,2</td>
</tr>
<tr>
<td>PDO-Consumer</td>
<td>512</td>
<td></td>
<td>1,2</td>
</tr>
<tr>
<td>EMCY-Consumer</td>
<td>127</td>
<td></td>
<td>127</td>
</tr>
<tr>
<td>Heartbeat-Consumer</td>
<td>127</td>
<td></td>
<td>2 (*)</td>
</tr>
<tr>
<td>Time-Producer</td>
<td>x</td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>Time-Consumer</td>
<td>x</td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>NMT-Master</td>
<td>x</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>LSS-Master</td>
<td>x</td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>Domain Size</td>
<td>15 MByte</td>
<td></td>
<td>1 MByte (***)</td>
</tr>
<tr>
<td>TCP/Clients</td>
<td>10</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Other</td>
<td>-</td>
<td></td>
<td>automatic exit after 1 hour</td>
</tr>
</tbody>
</table>

(*) Node id 32 or 64 can be accessed.

(**) EtherCAN: 128 kByte, one domain transfer at a time

See appendix 1 for the complete object directory.

**Note:** The performance of the CANopen Gateway Server depends on the used CAN interface hardware. Especially at high bus load and high baud rates some CAN messages may be lost.

### 1.2. Installation and Start

#### 1.2.1. CAN driver

The CANopen Gateway Server accesses the CAN interface card by means of a layer 2 driver. This driver has to be installed before using the CANopen Server. For installation of the driver please refer to the delivered driver manual.

- **EMS Wünsche** (http://www.ems-wuensche.com)
- **I+ME Actia** (http://www.ime-actia.de)
- **Kvaser** (http://www.kvaser.com)
- **Sontheim** (http://www.sontheim-industrie-elektronik.de)
- **Janz** (http://www.janztec.com)
The most current list of supported hardware can be requested at service@port.de.

1.2.2. Command line options

1.2.2.1. Common options

<table>
<thead>
<tr>
<th>Option</th>
<th>Argument</th>
<th>Description</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p</td>
<td>port</td>
<td>- TCP/IP port</td>
<td>7234</td>
</tr>
<tr>
<td>-b</td>
<td>baud</td>
<td>- CAN Baudrate (*)</td>
<td>125</td>
</tr>
<tr>
<td>-n</td>
<td>id</td>
<td>- Node-ID of CANopen Gateway Server</td>
<td>6</td>
</tr>
<tr>
<td>-H</td>
<td>time</td>
<td>- start producing HB frames at start-up</td>
<td>0</td>
</tr>
<tr>
<td>-R</td>
<td></td>
<td>- Don’t send NMT PREOP ALL at exit</td>
<td></td>
</tr>
<tr>
<td>-D</td>
<td>device</td>
<td>- CAN device/channel, e.g. can1</td>
<td></td>
</tr>
<tr>
<td>-V</td>
<td></td>
<td>- Version</td>
<td></td>
</tr>
<tr>
<td>-S</td>
<td></td>
<td>- Server mode</td>
<td></td>
</tr>
<tr>
<td>-L</td>
<td>filename</td>
<td>- Path of licence file</td>
<td></td>
</tr>
</tbody>
</table>

(*) values for baudrate: 20, 50, 125, 250, 500, 1000

When the CANopen Gateway Server is started without option -S it runs in command line mode. Commands can be entered interactively from command line, then, and sequence numbers can be omitted. In command line mode the CANopen Gateway Server processes the initialization file \m4d\ini.cmd. This ASCII file is contains DS309-3 commands. Lines starting with "#" are comments and are not executed. It can be tailored to match the application needs.

1.2.2.2. Options for Linux, EtherCAN, IGW900

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-s</td>
<td>Timer resolution in ms</td>
</tr>
</tbody>
</table>

1.2.2.3. Options for EMS Wünsche CPC

Option -D specifies the channel given in the configuration file C:\Windows\cpccconf.ini which is located in the Windows system directory.
1.2.2.4. Options for Peak

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-i</td>
<td>Interface</td>
</tr>
<tr>
<td></td>
<td>1 - USB</td>
</tr>
<tr>
<td></td>
<td>2 - Parallel Dongle</td>
</tr>
<tr>
<td>-I</td>
<td>Interrupt for parallel dongle</td>
</tr>
<tr>
<td>-B</td>
<td>I/O base address for parallel dongle</td>
</tr>
</tbody>
</table>

1.2.2.5. Options for Kvaser
Option -D specifies the card channel. The card channel can be obtained with the "Kvaser Hardware Configuration" tool.
2. Definitions

Command

controls the CANopen Gateway Server and interacts with CANopen devices. It may have a long form and a short form. The short form is a one or two letter abbreviation of the long form. The long form is obtained by concatenating the short form and the string enclosed in brackets "[", "]".

Datatypes

<table>
<thead>
<tr>
<th>Syntax</th>
<th>CANopen Type</th>
<th>Supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>Bool</td>
<td>✓</td>
</tr>
<tr>
<td>u8</td>
<td>Unsigned8</td>
<td>✓</td>
</tr>
<tr>
<td>u16</td>
<td>Unsigned16</td>
<td>✓</td>
</tr>
<tr>
<td>u24</td>
<td>Unsigned24</td>
<td>✓</td>
</tr>
<tr>
<td>u32</td>
<td>Unsigned32</td>
<td>✓</td>
</tr>
<tr>
<td>u24</td>
<td>Unsigned40</td>
<td>✓</td>
</tr>
<tr>
<td>u48</td>
<td>Unsigned48</td>
<td>✓</td>
</tr>
<tr>
<td>u56</td>
<td>Unsigned56</td>
<td>✓</td>
</tr>
<tr>
<td>u64</td>
<td>Unsigned64</td>
<td>✓</td>
</tr>
<tr>
<td>i8</td>
<td>Integer8</td>
<td>✓</td>
</tr>
<tr>
<td>i16</td>
<td>Integer16</td>
<td>✓</td>
</tr>
<tr>
<td>i24</td>
<td>Integer24</td>
<td>✓</td>
</tr>
<tr>
<td>i32</td>
<td>Integer32</td>
<td>✓</td>
</tr>
<tr>
<td>i24</td>
<td>Integer40</td>
<td>✓</td>
</tr>
<tr>
<td>i48</td>
<td>Integer48</td>
<td>✓</td>
</tr>
<tr>
<td>i56</td>
<td>Integer56</td>
<td>✓</td>
</tr>
<tr>
<td>i64</td>
<td>Integer64</td>
<td>✓</td>
</tr>
<tr>
<td>r32</td>
<td>Real32</td>
<td>✓</td>
</tr>
<tr>
<td>r64</td>
<td>Real64</td>
<td>✓</td>
</tr>
<tr>
<td>t</td>
<td>Time of day (two arguments: day ms)</td>
<td>✓</td>
</tr>
<tr>
<td>td</td>
<td>Time difference</td>
<td>-</td>
</tr>
<tr>
<td>vs</td>
<td>Visible string</td>
<td>✓</td>
</tr>
<tr>
<td>os</td>
<td>Octet string</td>
<td>✓</td>
</tr>
<tr>
<td>us</td>
<td>Unicode string</td>
<td>✓</td>
</tr>
<tr>
<td>d</td>
<td>Domain</td>
<td>✓</td>
</tr>
</tbody>
</table>

Visible strings can have a maximum length of 127 bytes. If the string contains "whitespace" it has to be enclosed in double quotes.

Octet and unicode strings and domain data are base64 encoded. The encoded octet and unicode string can have a maximum length of 127 bytes. Domain data up to 1 mega byte is supported.
2.1. Commands

Commands are communicated as case insensitive ASCII strings. Numbers are represented in manner of the C programming language:

- decimal, starting with a number
- hexadecimal, starting with the string 0x
- float
- float

2.1.1. Command Request

A command is composed of tokens separated by whitespaces and closed with a CRLF. All commands are confirmed. Each command starts with a 4 byte sequence number which is enclosed by square brackets. The sequence number is followed by net and node of the CANopen device. Both numbers are optional (see Set default network/node).

In BNF notation a command defines as follows:

```
<command-request> ::= "["<sequence>"]" [[<net>] <node>] <command>
<sequence> ::= UNSIGNED32
<net> ::= UNSIGNED8
<brand> ::= UNSIGNED8
<command> ::= <command-specifier> | <compound-command>
<compound-command> ::= <command-specifier> <parameter>
<parameter> ::= <value> | <command-parameter>
<compound-parameter> ::= <value> <parameter>
```

Net and node numbers are starting with 1. The value 0 is used to address all nodes and all networks.

2.1.2. Command Response

The CANopen Gateway Server responds with the same sequence number at the first position as given by the request. This number shall be given in decimal format.

```
<command-response> ::= "["<sequence>"]" <response>
<response> ::= <value> | <error-string> | <emcy-list> | "OK"
<error-string> ::= "Error:" <error code>
<error-code> ::= <internal-error-code> | <sdo-abort-code>
<emcy-list> ::= ["<emcy1>" "..<emcy254>"
<emcyN> ::= UNSIGNED32
```

Allowed internal-error-codes (IEC) are listed below:

<table>
<thead>
<tr>
<th>IEC</th>
<th>Message text</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>request not supported</td>
</tr>
<tr>
<td>101</td>
<td>syntax error</td>
</tr>
<tr>
<td>102</td>
<td>Request not processed due to internal state</td>
</tr>
<tr>
<td>IEC</td>
<td>Message text</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------</td>
</tr>
<tr>
<td>200</td>
<td>lost Guarding Msg</td>
</tr>
<tr>
<td>201</td>
<td>lost connection</td>
</tr>
<tr>
<td>202</td>
<td>Heartbeat started</td>
</tr>
<tr>
<td>203</td>
<td>Heartbeat lost</td>
</tr>
<tr>
<td>204</td>
<td>new NMT state</td>
</tr>
<tr>
<td>205</td>
<td>Boot up</td>
</tr>
<tr>
<td>300</td>
<td>CAN passive</td>
</tr>
<tr>
<td>301</td>
<td>CAN bus off</td>
</tr>
<tr>
<td>302</td>
<td>CAN overflow</td>
</tr>
<tr>
<td>303</td>
<td>CAN buffer overflow</td>
</tr>
<tr>
<td>304</td>
<td>CAN init</td>
</tr>
<tr>
<td>305</td>
<td>CAN active</td>
</tr>
<tr>
<td>400</td>
<td>PDO already used</td>
</tr>
<tr>
<td>401</td>
<td>PDO length exceeded</td>
</tr>
</tbody>
</table>

2.1.3. Event triggered messages

Messages due to errors in the CANopen network or the occurrence of communication objects using the producer-consumer principle (PDO, EMCY) shall not use a sequence number.

<event-triggered-message> ::= [[net] node] <event-specifier> <parameter>
<event-specifier> ::= "EMCY" | "ERROR" | "SYNC" | <pdo-event> | "USER" <user-event>
<pdo-event> ::= [net] "PDO" <parameter>
$user-event$ ::= <lss-event> <block-event>
<lss-event> ::= "LSS" <parameter>
<block-event> ::= "BLOCK" <parameter>

The content of event-triggered messages is described within the command description that enables the specific service.
3. Network access command specification

3.1. SDO access commands

3.1.1. Upload SDO

Command syntax

```plaintext
[[net] node] r[ead] <index> <subindex> <datatype>
```

Accesses the remote device via SDO read messages.
Domain data have to be passed base64 encoded. See also "SDO Block event" and Appendix: Examples.

3.1.2. Download SDO

Command syntax

```plaintext
[[net] node] w[rite] <index> <subindex> <datatype> <value>
```

Accesses the remote device via SDO write messages.

3.1.3. Configure SDO timeout

Command syntax

```plaintext
[[net] node] set sdo_timeout <ms>
```

The SDO timeout time is used to abort a SDO transfer when there is no device with the given node id. SDO timeout times are node specific. In order to set one SDO timeout for all nodes node id 0 can be used.

A timeout is measured between each SDO request and confirmation.

3.1.4. SDO Block Event

Command syntax

```plaintext
[[net] [node] USER BLOCK <current> <maximum>
```

With SDO domain down/upload the server indicates progress of the current transmission with the SDO Block Event. On an SDO Upload it is required that the SDO server provides correct size information. The current argument is the number of blocks that have been transmitted/received. The maximum argument is the number of total blocks that have to be transmitted or received.
3.2. PDO access commands

3.2.1. Configure RPDO command

Command syntax

```
[net] set RPDO <nr> <COB-ID> <tx-type> <nr-of-data> <datatype1>..<datatype64>
```

```
<tx-type> ::= "rtr" | "event" | "sync<0..240>"
```

Setup a PDO for receiving data. The datatype maps to an object in the object directory. On reception of an RPDO the RPDO Event is triggered. The receive PDO is seen from the side of the CANopen Gateway Server.

A PDO can be deactivated by writing using value 0x80000000 for the cob.

TCP/IP clients do not have separate namespaces for PDO, i.e. if client 1 has setup PDO 1 and 2 and a client 2 connects and also sets up PDO 1 and 2 it alters the PDO definition of client 1.

It is allowed to set all possible COB-Id. No extra checking is carried out. This means that if the same COB-ID is used for RPDO and TPDO it may happen that no PDO is received.

3.2.2. Configure TPDO command

Command syntax

```
[net] set tpdo <nr> <COB> <tx-type> <nr-of-data> <datatype1>..<datatype64>
```

Setup a PDO for sending data. The datatype maps to an object in the object directory. The transmit PDO is seen from the side of the CANopen Gateway Server.

A PDO can be deactivated by writing using value 0x80000000 for the cob.

TCP/IP clients do not have separate namespaces for PDO, i.e. if client 1 has setup PDO 1 and 2 and a client 2 connects and also sets up PDO 1 and 2 it alters the PDO definition of client 1.

It is allowed to set all possible COB-Id. No extra checking is carried out. This means that if the same COB-ID is used for RPDO and TPDO it may happen that no PDO is received.

3.2.3. Read PDO data command
3.2.4. Write PDO data command

**Command syntax**

```
[net] write pdo <nr> <nr-of-data> <value1>[..<value64>]
```

Sends given data with a PDO. The PDO has to be setup before.

3.2.5. RPDO Event

**Command syntax**

```
[net] PDO <nr> <nr-of-data> <value1>[..<value64>]
```

A PDO was received. The mapped data is given as arguments formatted as decimal values. The PDO has to be setup, previously.

3.3. CANopen NMT commands

The CANopen Gateway Server has link any other CANopen device a node id itself (see command line parameters). Node id 0 and 0x80 have special meanings. A NMT command to node id 0 adresses all CANopen nodes in the network including the server. A NMT command to node id 0x80 adresses all CANopen nodes in the network excluding the server. The server remains in the NMT state that was issued before.

3.3.1. Start Node

**Command syntax**

```
[net] node start
```

Sends the OPERATIONAL command.

3.3.2. Stop node

**Command syntax**

```
[net] node stop
```

Sends the STOPPED command.
3.3.3. Set node Preoperational

Command syntax

```
[[net] node] preoperational
```

Sends the PREOPEATIONAL command.

3.3.4. Reset node

Command syntax

```
[[net] node] reset node
```

Sends the RESET NODE command.

In case a SDO transfer is currently in progress that was started by another TCP client then "Reset node" is not carried out. If the SDO transfer was started by the TCP client that also sent the "Reset node" command the SDO transfer is aborted and the reset is carried out.

**Note:** With node number 0 the gateway is reset, too, i.e. the PDO configuration is lost.

3.3.5. Reset communication

Command syntax

```
[[net] node] reset communication
```

Sends the RESET COMMUNICATION command.

See also Reset Node Siehe auch Knoten zurücksetzen

3.3.6. Enable node guarding

Command syntax

```
[[net] node] enable guarding <guardingtime> <lifetimefactor>
```

Activates the node guarding error control mechanism. If the master doesn’t receive a response of the slave in the given time it sends the event triggered message ERROR 200 or ERROR 201.

Accuracy of the Guarding depends on the timer resolution. On Linux platforms it can be adjusted with the command line parameter '-s'. On Windows it is fixed.
3.3.7. Disable node guarding

Command syntax

```
[[net] node] disable guarding
```

Deactivates the node guarding error control mechanism.

3.3.8. Enable heartbeat

Command syntax

```
[[net] node] enable heartbeat <heartbeattime>
```

Start heartbeat detection on the CANopen Gateway Server. It is checked that heartbeat messages are received in the given time. If a heartbeat message is not received an event triggered ERROR message is sent. The time is given in milliseconds.

The heartbeat time of the producer has to be set with a separate SDO Download Command.

Accuracy of the Guarding depends on the timer resolution. On Linux platforms it can be adjusted with the command line parameter '-s'. On Windows it is fixed.

3.3.9. Disable heartbeat

Command syntax

```
[[net] node] disable heartbeat
```

Stop heartbeat detection on the CANopen Gateway Server.

3.3.10. Error control Event

Command syntax

```
[[net] node] ERROR <internal-error-code>
```

On detection of NMT errors or CAN errors this message is sent to a connected client.

3.4. Device Failure management

3.4.1. Read device error command

NOT implemented
3.4.2. Emergency Event

*Command syntax*

```
[[net] node] EMCY <emcy-code> <error-register> <m-error-code>
```


<m-error-code> ::= <UNSIGNED8> <UNSIGNED8> <UNSIGNED8> <UNSIGNED8> <UNSIGNED8>

On reception of a CANopen EMCY message this message is sent.

**Note:** Reception has to be enabled. See command extension.

3.5. CANopen interface configuration commands

3.5.1. Initialize gateway

*Command syntax*

```
[net] init <bitrate>
```

Sets the bitrate the CANopen Gateway Server is working with. The bitrate can be given as value or table index of the default CANopen table.

<table>
<thead>
<tr>
<th>Bitrate</th>
<th>Table index</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>0</td>
</tr>
<tr>
<td>800</td>
<td>1</td>
</tr>
<tr>
<td>500</td>
<td>2</td>
</tr>
<tr>
<td>250</td>
<td>3</td>
</tr>
<tr>
<td>125</td>
<td>4</td>
</tr>
<tr>
<td>reserved</td>
<td>5</td>
</tr>
<tr>
<td>50</td>
<td>6</td>
</tr>
<tr>
<td>20</td>
<td>7</td>
</tr>
<tr>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>Autobaud</td>
<td>9</td>
</tr>
</tbody>
</table>

Autobaud is not supported.

3.5.2. Store configuration

*Command syntax*

```
[net] store <specifier>
```

NOT implemented.
3.5.3. Restore configuration

Command syntax

```
[net] restore <specifier>
```

NOT implemented.

3.5.4. Set heartbeat producer

Command syntax

```
[net] set heartbeat <ms>
```

Start sending heartbeat messages of the CANopen Gateway Server. The time is given in milliseconds.

3.5.5. Set node id

Command syntax

```
[net] set id <value>
```

NOT supported. Can only be set at start. See commandline options.

3.5.6. Start emergency consumer

See port extension.

3.5.7. Stop emergency consumer

See command extension.

3.6. Gateway management commands

3.6.1. Set default network

Command syntax

```
[net] set network <value>
```

Set default network. When set the network parameter can be omitted at following commands.

3.6.2. Set default node id
Command syntax

[net] set node <value>

Set default remote node id. When set the node parameter can be omitted at following commands.

3.6.3. Get version

Command syntax

info version

Request version of the CANopen Gateway Server. The version is given in the format:

/vendor-id/ <product-code>
/version-high/>. <version-low> <serial-number>
/network-class> <protocol-version>
/implementation-class>
4. Command Extensions
The standard DS309-3 allows user specific commands and event triggered messages. All extensions provided by the port CANopen Gateway Server are prepended with the prefix: _port_. Exception are the info commands since it is already available.

The CANopen Gateway Server allows multiple TCP/IP connections. This can be problematic if two clients don’t know from each other and send NMT commands like start and preop. To handle this situation one client can set a master flag and thus restrict others that way that they can not send NMT or LSS commands. However, to be compliant with the standard this only works with port specific commands. The "normal" DS309-3 commands do not know of the master flag.

Especially for multiclient environments the CANopen Gateway Server provides register commands to hook up on a running server and register for already configured PDO, heartbeat, nodeguarding, EMCY and TIME services.

4.1. Status and Configuration
4.1.1. Set master
Command syntax

```
_port_set master <0|1>
```

Activate master mode for TCP/IP client. The master flag can only be set from one client.

4.1.2. Set verbose response
Command syntax

```
_port_set verbose 0xff
```

Show verbose error messages.

4.1.3. Get default network
Command syntax

```
info net[work]
```

Shows default network used for commands.

4.1.4. Get default node
Command syntax

info node

Shows default node used for commands.

4.1.5. Get Build information

Command syntax

info build

Returns a string that contains the build date, version and additional information.

Example

5.3.10, Nov 13 2012, CPC-PCI can4linux,

4.1.6. Get CANopen node Id of server

Command syntax

info id

Returns CANopen node Id of the server. See also command line parameters.

4.1.7. Get CAN bitrate

Command syntax

info bitrate

Returns CAN bitrate of the server.

4.1.8. Get SDO timeout

Command syntax

[node] info sdo_timeout

Shows the node specific SDO timeout. A value of 0 for node is not allowed.

4.1.9. Get TPDO setup

Command syntax

info tpdo

Shows the TPDO configuration in form of a comma separated list. `<pdoNr> <cobId> <nrOfElements> <datatype1> .. <datatypeN>[,<pdoNr> <cobId> <nrOfElements> <datatype1> .. <datatypeN>]`
4.1.10. Get RPDO setup

Command syntax

```
info rpdo
```

Shows the RPDO configuration in form of a comma separated list. 

```
<pdoNr> <cobId> <nrOfElements> <datatype1> .. <datatypeN>[,<pdoNr> <cobId> <nrOfElements> <datatype1> .. <datatypeN>]
```

4.2. SDO

4.2.1. Register SDO Server Write Indikation

Command syntax

```
_port_reg sdoserverind
```

Register for receiving of SDO Server Write Indikation

The indication is called for all objects at manufacturer area.

Numeric values are indicated as:

```
SDOSERV <index>:<subindex> size: <len> data: <value>
```

Non-Numeric values are indicated as:

```
SDOSERV <index>:<subindex> b64size: <len> data: <data>
```

<data> are codes as b64.

4.2.2. Unregister SDO Server Write Indikation

Command syntax

```
_port_unreg sdoserverind
```

Unregister for receiving of SDO Server Write Indikation

4.3. PDO

4.3.1. Register PDO

Command syntax

```
_port_reg rpdo <pdoNr>
```

Register for receiving an RPDO.

4.3.2. Unregister PDO
Command syntax

```plaintext
_port_unreg rpdo <pdoNr>
```

Unregister an RPDO.

### 4.4. NMT-Master

#### 4.4.1. Register Nodeguarding

Command syntax

```plaintext
[net] [node] _port_reg guarding
```

Register for receiving Nodeguarding events. Events are only sent when a protocol violation has occurred.

#### 4.4.2. Unregister Nodeguarding

Command syntax

```plaintext
[net] [node] _port_unreg guarding
```

Stop receiving nodeguarding events.

#### 4.4.3. Register heartbeat

Command syntax

```plaintext
[net] [node] _port_reg heartbeat
```

Register for receiving heartbeat events. Events are only sent when a protocol violation has occurred or a new heartbeat start was detected.

#### 4.4.4. Unregister heartbeat

Command syntax

```plaintext
[net] [node] _port_unreg heartbeat
```

Stop receiving heartbeat events.

#### 4.4.5. Register emergency

Command syntax

```plaintext
[net] [node] _port_reg emcy
```

Register for receiving an emergency events.
4.4.6. Unregister emergency

Command syntax

```
[net] [node] _port_unreg emcy
```

Stop receiving emergency events.

4.4.7. Enable Sync Producer

Command syntax

```
_port_enable sync <cobid> <sync-cycle_in_us> [sync_counter]
```

Start sending SYNC messages. The master flag has to be set for this command. SYNC cycle time is specified in micro seconds.

4.4.8. Disable Sync Producer

Command syntax

```
_port_disable sync
```

Stop sending SYNC messages. The master flag has to be set for this command.

4.5. LSS Master

The CANopen Gateway Server can configure LSS slaves with the LSS commands. For all commands the master flag has to be set. The CANopen Gateway Server own identity object can be read with the `info version` command.

It is recommended that LSS services should be carried out in STOPPED state.

4.5.1. Switch Selective

Command syntax

```
[net] _port_lss switch_sel <vendorId> <product> <revision> <serialNo>
```

Set single LSS slave in CONFIGURATION state.

4.5.2. Switch global

Command syntax

```
[net] _port_lss switch_glob <0|1>
```

Set complete network in CONFIGURATION or OPERATION state.
4.5.3. Configuration of node id

Command syntax

```
[net] _port_lss set_node <nodeId>
```

Set the node id of an LSS slave.

4.5.4. Request node

Command syntax

```
[net] _port_lss get_node
```

Get the node id of an LSS slave.

4.5.5. Identify LSS slaves

Command syntax

```
[net] _port_lss identity <vendorId> <product> <rev lo> <rev hi>
    <serial low> <serial hi>
```

Scans the network for nodes that are in the given address range.

4.5.6. Bitrate Configuration

Command syntax

```
[net] _port_lss set_bitrate <sl_table_sel> <sl_table_idx>
    [<gw_table_sel> <gw_table_idx>]
```

Set the new bitrate of an LSS slave. The LSS slave has to be in state CONFIGURATION.

The first two parameter define the bitrate of the LSS slave. The last two parameter define the bitrate of the CANopen Gateway Server. They are used when autobaud is to be set at the LSS slaves.

<table>
<thead>
<tr>
<th>Bitrate</th>
<th>Table index</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>0</td>
</tr>
<tr>
<td>800</td>
<td>1</td>
</tr>
<tr>
<td>500</td>
<td>2</td>
</tr>
<tr>
<td>250</td>
<td>3</td>
</tr>
<tr>
<td>125</td>
<td>4</td>
</tr>
<tr>
<td>reserved</td>
<td>5</td>
</tr>
<tr>
<td>50</td>
<td>6</td>
</tr>
<tr>
<td>20</td>
<td>7</td>
</tr>
</tbody>
</table>
Only table 0, the standard CANopen table, is supported by the CANopen Gateway Server.

4.5.7. Bitrate activation

*Command syntax*

```
[net] _port_lss activate_bitrate <time>
```

Activates the bitrate. The CANopen Gateway Server responds after 2 * time is elapsed. The time is given in milli seconds.

4.5.8. Store Configuration

*Command syntax*

```
[net] _port_lss store
```

On reception of this command the LSS slave saves the bitrate and node id. The LSS slave has to be in CONFIGURATION state.

4.5.9. Identify unconfigured LSS slaves

*Command syntax*

```
[net] _port_lss identity_non_cfg
```

Invokes all unconfigured LSS slaves to send a message. On reception of this message the CANopen Gateway Server will send the LSS event.

4.5.10. LSS Event

*Command syntax*

```
USER LSS <no>
```

On detection of an unconfigured LSS slave the CANopen Gateway Server sends the LSS Event.

<table>
<thead>
<tr>
<th>no</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>unconfigured device detected</td>
</tr>
</tbody>
</table>

4.6. Miscellaneous

Manchmal ergibt sich die Notwendigkeit ohne Bezug auf CANopen, ohne ein PDO zu konfigurieren, einen beliebigen CAN Frame zu senden. Dazu kann das Kommando
_port_wr benutzt werden. **Command syntax**

```
_port_wr <cob-id> <type> <length> <data0> ... <data7>
```

<table>
<thead>
<tr>
<th>length</th>
<th>number of bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>Frame specifier, 2 characters</td>
</tr>
<tr>
<td></td>
<td>first character, frame format</td>
</tr>
<tr>
<td>[sS]</td>
<td>standard or base frame format</td>
</tr>
<tr>
<td>[XX]</td>
<td>extended base frame format</td>
</tr>
<tr>
<td>second character, frame type</td>
<td></td>
</tr>
<tr>
<td>[dD]</td>
<td>data frame</td>
</tr>
<tr>
<td>[rR]</td>
<td>RTR frame</td>
</tr>
</tbody>
</table>
5. Appendix

5.1. Object Directory Overview

<table>
<thead>
<tr>
<th>Index (hex)</th>
<th>Object</th>
<th>Name</th>
<th>Type</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>VAR</td>
<td>Device Type</td>
<td>UNSIGNED32</td>
<td>const</td>
</tr>
<tr>
<td>1001</td>
<td>VAR</td>
<td>Error Register</td>
<td>UNSIGNED8</td>
<td>ro</td>
</tr>
<tr>
<td>1018</td>
<td>VAR</td>
<td>Identity Object</td>
<td>IDENTITY</td>
<td>const</td>
</tr>
<tr>
<td>1002</td>
<td>VAR</td>
<td>Manufacturer Status Register</td>
<td>UNSIGNED32</td>
<td>ro</td>
</tr>
<tr>
<td>1003</td>
<td>ARRAY</td>
<td>Pre-defined Error Field</td>
<td>UNSIGNED32</td>
<td>ro</td>
</tr>
<tr>
<td>1005</td>
<td>VAR</td>
<td>COB-ID SYNC</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>1006</td>
<td>VAR</td>
<td>Communication Cycle Period</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>1007</td>
<td>VAR</td>
<td>Synchronous Window Length</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>1008</td>
<td>VAR</td>
<td>Manufacturer Device Name</td>
<td>VISIBLE_STRING</td>
<td>const</td>
</tr>
<tr>
<td>1009</td>
<td>VAR</td>
<td>Manufacturer Hardware Version</td>
<td>VISIBLE_STRING</td>
<td>const</td>
</tr>
<tr>
<td>100A</td>
<td>VAR</td>
<td>Manufacturer Software Version</td>
<td>VISIBLE_STRING</td>
<td>const</td>
</tr>
<tr>
<td>100C</td>
<td>VAR</td>
<td>Guard Time</td>
<td>UNSIGNED16</td>
<td>rw</td>
</tr>
<tr>
<td>100D</td>
<td>VAR</td>
<td>Life Time Factor</td>
<td>UNSIGNED8</td>
<td>rw</td>
</tr>
<tr>
<td>1010</td>
<td>ARRAY</td>
<td>Store Parameter Field</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>1011</td>
<td>ARRAY</td>
<td>Restore Default Parameters</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>1012</td>
<td>VAR</td>
<td>COB-ID Time Stamp</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>1013</td>
<td>VAR</td>
<td>High Resolution Time Stamp</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>1016</td>
<td>ARRAY</td>
<td>Heartbeat Consumer Entries</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>1017</td>
<td>VAR</td>
<td>Producer Heartbeat Time</td>
<td>UNSIGNED16</td>
<td>rw</td>
</tr>
<tr>
<td>1019</td>
<td>VAR</td>
<td>Synchronous counter overflow value</td>
<td>UNSIGNED8</td>
<td>rw</td>
</tr>
<tr>
<td>1028</td>
<td>ARRAY</td>
<td>Emergency Consumer</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>1200</td>
<td>ARRAY</td>
<td>Server SDO Parameter</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>Index (hex)</td>
<td>Object</td>
<td>Name</td>
<td>Type</td>
<td>Access</td>
</tr>
<tr>
<td>------------</td>
<td>--------</td>
<td>------</td>
<td>------</td>
<td>--------</td>
</tr>
<tr>
<td>1201</td>
<td>ARRAY</td>
<td>Server SDO Parameter</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>1280 - 12FF</td>
<td>ARRAY</td>
<td>Client SDO Parameter</td>
<td>SDO_PARAMETER</td>
<td>rw</td>
</tr>
<tr>
<td>1400 - 143F</td>
<td>ARRAY</td>
<td>Receive PDO Communication Parameter 1</td>
<td>PDO_COMM_PARA</td>
<td>rw</td>
</tr>
<tr>
<td>1600 - 163F</td>
<td>ARRAY</td>
<td>Receive PDO Mapping Parameter 1</td>
<td>PDO_MAPPING</td>
<td>rw</td>
</tr>
<tr>
<td>1800 - 183F</td>
<td>ARRAY</td>
<td>Transmit PDO Communication Parameter 1</td>
<td>PDO_COMM_PARA</td>
<td>rw</td>
</tr>
<tr>
<td>1A00 - 1A3F</td>
<td>ARRAY</td>
<td>Transmit PDO Mapping Parameter 1</td>
<td>PDO_MAPPING</td>
<td>rw</td>
</tr>
<tr>
<td>1F50</td>
<td>ARRAY</td>
<td>Download Program Data</td>
<td>DOMAIN</td>
<td>rw</td>
</tr>
<tr>
<td>1F51</td>
<td>ARRAY</td>
<td>Program Control</td>
<td>UNSIGNED8</td>
<td>rw</td>
</tr>
<tr>
<td>1F80</td>
<td>VAR</td>
<td>NMT Startup</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>1F81</td>
<td>ARRAY</td>
<td>Slave Assignment</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>1F82</td>
<td>ARRAY</td>
<td>Request NMT</td>
<td>UNSIGNED8</td>
<td>rw</td>
</tr>
<tr>
<td>1F83</td>
<td>ARRAY</td>
<td>Request Guarding</td>
<td>UNSIGNED8</td>
<td>rw</td>
</tr>
<tr>
<td>1F84</td>
<td>ARRAY</td>
<td>Device Type Identification</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>1F85</td>
<td>ARRAY</td>
<td>Vendor Identification</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>1F86</td>
<td>ARRAY</td>
<td>Product Code</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>1F87</td>
<td>ARRAY</td>
<td>Revision Number</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>1F88</td>
<td>ARRAY</td>
<td>Serial Number</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>1F89</td>
<td>VAR</td>
<td>Boot Time</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>2000</td>
<td>ARRAY</td>
<td>UNSIGNED8 PDOs</td>
<td>UNSIGNED8</td>
<td>rw</td>
</tr>
<tr>
<td>2001</td>
<td>ARRAY</td>
<td>UNSIGNED16 PDOs</td>
<td>UNSIGNED16</td>
<td>rw</td>
</tr>
<tr>
<td>2002</td>
<td>ARRAY</td>
<td>UNSIGNED32 PDOs</td>
<td>UNSIGNED32</td>
<td>rw</td>
</tr>
<tr>
<td>2003</td>
<td>ARRAY</td>
<td>INTEGER8 PDOs</td>
<td>INTEGER8</td>
<td>rw</td>
</tr>
<tr>
<td>2004</td>
<td>ARRAY</td>
<td>INTEGER16 PDOs</td>
<td>INTEGER16</td>
<td>rw</td>
</tr>
<tr>
<td>2005</td>
<td>ARRAY</td>
<td>INTEGER32 PDOs</td>
<td>INTEGER32</td>
<td>rw</td>
</tr>
<tr>
<td>2006</td>
<td>ARRAY</td>
<td>REAL32 PDOs</td>
<td>REAL32</td>
<td>rw</td>
</tr>
<tr>
<td>2007</td>
<td>ARRAY</td>
<td>UNSIGNED24 PDOs</td>
<td>UNSIGNED24</td>
<td>rw</td>
</tr>
<tr>
<td>Index (hex)</td>
<td>Object</td>
<td>Name</td>
<td>Type</td>
<td>Access</td>
</tr>
<tr>
<td>------------</td>
<td>--------</td>
<td>--------------------</td>
<td>------------</td>
<td>--------</td>
</tr>
<tr>
<td>2008</td>
<td>ARRAY</td>
<td>UNSIGNED40 PDOs</td>
<td>UNSIGNED40</td>
<td>rw</td>
</tr>
<tr>
<td>2009</td>
<td>ARRAY</td>
<td>UNSIGNED48 PDOs</td>
<td>UNSIGNED48</td>
<td>rw</td>
</tr>
<tr>
<td>200A</td>
<td>ARRAY</td>
<td>UNSIGNED56 PDOs</td>
<td>UNSIGNED56</td>
<td>rw</td>
</tr>
<tr>
<td>200B</td>
<td>ARRAY</td>
<td>UNSIGNED64 PDOs</td>
<td>UNSIGNED64</td>
<td>rw</td>
</tr>
<tr>
<td>2100</td>
<td>ARRAY</td>
<td>Short Message Ser-</td>
<td>VISIBLE_STRING</td>
<td>rw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>vice Object</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2200</td>
<td>VAR</td>
<td>ostring</td>
<td>OCTET_STRING</td>
<td>wo</td>
</tr>
</tbody>
</table>
5.2. Examples
All examples have been carried out in commandline mode.

SDO Access
Access several indices of a CANopen node.
COM Shell > 32 r 0x1018 0 u32
COM Shell > [0] 0x4

COM Shell > 32 r 0x3000 0 u8
COM Shell > [0] 0x0

COM Shell > 32 w 0x3000 0 u8 12
COM Shell > [0] OK

COM Shell > 32 r 0x3000 0 u8
COM Shell > [0] 0xc

Set default node id. Access numerical values and string values.
COM Shell > set node 32
[0] OK

COM Shell > r 0x3000 0 u8
COM Shell > [0] 0xc

COM Shell > w 0x3010 0 i8 0xff
COM Shell > [0] OK

COM Shell > r 0x3010 0 i8
COM Shell > [0] -1

COM Shell > w 0x3010 0 i8 0xf9
COM Shell > [0] OK

COM Shell > r 0x3010 0 i8
COM Shell > [0] -7

COM Shell > r 0x1008 0 vs
COM Shell > [0] S16-LINUX

COM Shell > r 0x1008 0 os
COM Shell > [0] UzE2L0xJTlVYAAAAAAAAAAAAAAAAA=

Read a domain value.
COM Shell > w 0x2002 0 "test1.dat"
[0] ERROR 101 Syntax Error - Bad Para Count
Write a domain value.
The raw data has to be converted to base64 format. The Linux-Tool base64 is used for this.

COM Shell > w 0x2002 0 vs test1.dat
COM Shell > [0] OK
> echo 'Hello World!' | base64
SGVsbG8gV29ybGQhCg==
COM Shell > w 0x2000 0 d SGVsbG8gV29ybGQhCg==
[0] OK
COM Shell > r 0x2000 0 d
COM Shell > 32 USER BLOCK 1 1
[0] MTIzMTIzMTIzMTIzMTIzMTIzMTIzMTIzMTIzMTIzMQ==

PDO Konfiguration

Initialize PDO 1 and deinitialize PDO 1. Initialize PDO 2 and start network.

COM Shell > set rpdo 1 0x181 event 1 u8
[0] OK

COM Shell > set rpdo 1 0x80000181 event 1 u8
[0] OK

COM Shell > set rpdo 2 0x181 event 1 u8
[0] OK

COM Shell > 0 start
[0] OK


COM Shell > PDO 2 1 0xaa

COM Shell > _port_set verbose 1
[0] OK

COM Shell > set rpdo 3 0x182 event 2 u8
[0] ERROR 100

COM Shell > _port_set verbose 0xff
[0] OK

COM Shell > set rpdo 3 0x182 event 2 u8
[0] ERROR 100 Syntax Error - Invalid data
COM Shell > set rpdo 3 0x182 event 2 u8 u16
[0] OK

NMT Error control
Error control with Heartbeat. Node is switched off and started again (Boot up). Node has change NMT state.
COM Shell > 32 enable heartbeat 1200
[0] OK

COM Shell > 32 ERROR 203 Heartbeat lost
32 ERROR 205 Boot up
32 ERROR 202 Heartbeat started

COM Shell > 32 ERROR 204 new NMT state 127

Emergency
Switch on reception of emergency messages.
COM Shell > 32 _port_reg emcy
[0] OK

32 EMCY 0xff00 0x81 0xaa 0x0 0x12 0xaf 0x0

LSS node configuration
Switch on verbose error messages. Set master flag. LSS Event received. Change to NMT state STOPPED (recommended). Start identifying LSS slave. Switch LSS slave into configuration mode
COM Shell > _port_set verbose 0xff
[0] OK

COM Shell > _port_set master 1
[0] OK

COM Shell > USER LSS 1 unconfigured device(s) detected

COM Shell > 0 stop
[0] OK

COM Shell > _port_lss identity 0x34 12345 0 2 0 2
[0] OK

COM Shell > _port_lss identity 0x34 12345 0 1 0 1
[0] OK

COM Shell > _port_lss identity 0x34 12345 1 1 1 1
Request node id. Set node id and check it. Switch to operation mode. Change to NMT state peroperational. Access node with SDO commands.

Gateway management

Setup PDO and request configuration. Request version information. Request default node. Request SDO timeout from default node. Request SDO timeout from node 32.
[0] 2 0x181 event 1 u8

COM Shell > set tpdo 1 0x200 event 1 u8
[0] OK

COM Shell > set tpdo 2 0x300 event 2 u32 u32
[0] OK

COM Shell > info tpdo
[0] 1 0x200 event 1 u8, 2 0x300 event 2 u32 u32

COM Shell > info version
[0] 52 410640 4.2 0 3 1.00 0.0

COM Shell > info node
[0] 127

COM Shell > info sdo_timeout
[0] 1000

COM Shell > 32 set sdo_timeout 1500
[0] OK

COM Shell > 32 info sdo_timeout
[0] 1500

**Transmission of layer-2 CAN frames**

The example shows how the command `port_wr` can be used to send raw CAN frames.

COM Shell > `port_wr` 0x1111 xd 4 1 2 3 4
[0] OK

COM Shell > `port_wr` 0x1111 sr 5
[0] OK

COM Shell > `port_wr` 0x111 sd 3 10 16 Oxaa
[0] OK
6. Index

- A -

access, Examples SDO 17
activate_bitrate, LSS 15

- B -

base64 9
Bitrate get 14
Block, SDO 11
Bool, Datatypes 9
Build get 14

- C -

CAN
   driver 8
   frame transmission, Examples 15, 19
CANopen
   Node Id get 14
   Object directory 16
Command line options 8
command request 9
Commands 9
commands, extension 14
Commands, NMT 11
commands, PDO 11
communication, NMT reset 12
Configuration 12–13
configuration, Examples
   Examples LSS node 18
   Examples PDO 17
Evaluation
   Heatbeat producer 13
   Init 12
   Restore 13
   Set id 13
   Start emergency consumer 13
   Stop emergency consumer 13
   Store 13
configuration, Examples
   Examples LSS node 18
   Examples PDO 17
consumer, Configuration
   Configuration Start emergency 13
   Configuration Stop emergency 13
control, Examples NMT Error 18

- D -

d, Datatypes 9
Datatypes 9
   Bool 9
domain 9
   d 9
   INTEGER8 9
   INTEGER16 9
   INTEGER24 9
   INTEGER32 9
   INTEGER40 9
   INTEGER48 9
   INTEGER56 9
   INTEGER64 9
   INTEGER8 9
   I8 9
   octet string 9
   os 9
   t 9
   Time difference 9
   Time of day 9
   U16 9
   U24 9
   U32 9
   U40 9
   U48 9
   U56 9
   U64 9
   U8 9
   unicode string 9
   UNSIGNED16 9
   UNSIGNED24 9
   UNSIGNED32 9
   UNSIGNED40 9
   UNSIGNED48 9
   UNSIGNED56 9
   UNSIGNED64 9
   UNSIGNED8 9
UNSIGNED48 9
UNSIGNED56 9
UNSIGNED64 9
UNSIGNED8 9
us 9
visible string 9
vs 9
day, Datatypes Time of 9
default network 14
Default network, Gateway management 13
default node 14
Default node, Gateway management 13
device error, Device failure 12
Device
failure 12
failure device error 12
Failure EMCY Event 12
Failure Emergency Event 12
difference, Datatypes Time 9
directory, CANopen Object 16
disable
guarding, NMT 12
heartbeat, NMT 12
SYNC 14
domain, Datatypes 9
download, SDO 11
driver, CAN 8

- E -
EMCY Event, Device Failure 12
emergency consumer
Configuration Start 13
Configuration Stop 13
Emergency
Event, Device Failure 12
Examples 18
register 14
unregister 14
enable
guarding, NMT 12
heartbeat, NMT 12
SYNC 14

- F -
failure
Device 12
device error, Device 12
Failure
EMCY Event, Device 12
Emergency Event, Device 12
file, initialization 8
frame transmission, Examples CAN 15, 19

- G -
Gateway management 13
Default network 13
Default node 13
Examples 19
Version 13
get
Bitrate 14
Build 14
CANopen Node Id 14
SDO Timeout 14
get_node, LSS 15
Guarding, NMT
  NMT disable 12
  NMT enable 12

- H -
Heartbeat, NMT
  NMT disable 12
  NMT enable 12
Heartbeat
  register 14
  unregister 14
Heatbeat producer, Configuration 13

- I -
I16, Datatypes 9
I24, Datatypes 9
I32, Datatypes 9
I40, Datatypes 9
I48, Datatypes 9
I56, Datatypes 9
I64, Datatypes 9
I8, Datatypes 9
id, Configuration Set 13
Id get, CANopen Node 14
identify_non_cfg, LSS 15
identity, LSS 15
Init, Configuration 12
initialization file 8
INTEGER16, Datatypes 9
INTEGER24, Datatypes 9
INTEGER32, Datatypes 9
INTEGER40, Datatypes 9
INTEGER48, Datatypes 9
INTEGER56, Datatypes 9
INTEGER64, Datatypes 9
INTEGER8, Datatypes 9

- L -
Layer Setting Services 14
line options, Command 8
LSS 14–15
  activate_bitrate 15
  Event 15
  get_node 15
  identify_non_cfg 15
  identity 15
  node configuration, Examples 18
  set_bitrate 15
  set_node 15
  store 15
  switch_glob 15
  switch_sel 15

- M -
management
  Default network, Gateway 13
  Default node, Gateway 13
  Examples Gateway 19
  Gateway 13
  Version, Gateway 13
  master 14
  masterflag 14
  messages, Event triggerd 9

- N -
network
  default 14
  Gateway management Default 13
NMT 12
  Commands 11
  disable guarding 12
  disable heartbeat 12
  enable guarding 12
  enable heartbeat 12
  Error control, Examples 18
  Error Event 12
  preoperational 12
  reset communication 12
  reset node 12
  start 12
stop 12
NMT-Master 14
node
configuration, Examples LSS 18
default 14
Gateway management Default 13
Node Id get, CANopen 14
node, NMT reset 12
Nodeguarding
register 14
unregister 14
number, sequence 9

-O-
Object directory, CANopen 16
octet string, Datatypes 9
of day, Datatypes Time 9
options, Command line 8
os, Datatypes 9

-P-
PDO 11
commands 11
configuration, Examples 17
Event 11
read 11
register 14
RPDO 11
TPDO 11
unregister 14
write 11
preoperational, NMT 12
producer, Configuration Heatbeat 13

-R-
read, PDO 11
register
Emergency 14
Heartbeat 14
Nodeguarding 14
PDO 14
SDO 14
request, command 9
reset
communication, NMT 12
node, NMT 12
Restore, Configuration 13
RPDO
PDO 11
setup 14

-S-
SDO 11
access, Examples 17
Block 11
download 11
register 14
timeout 11
Timeout get 14
unregister 14
upload 11
sequence number 9
Services, Layer Setting 14
Set id, Configuration 13
set_bitrate, LSS 15
set_node, LSS 15
Setting Services, Layer 14
setup
RPDO 14
TPDO 14
Start emergency consumer, Configuration 13
start, NMT 12
Status 14
Stop emergency consumer, Configuration 13
stop, NMT 12
Store, Configuration 13
store, LSS 15
string, Datatypes
Datatypes octet 9
Datatypes unicode 9
Datatypes visible 9
switch_glob, LSS 15
switch_sel, LSS 15
SYNC
disable 14
  enable 14
-T-
t, Datatypes 9
td, Datatypes 9
Time
difference, Datatypes 9
  of day, Datatypes 9
Timeout get, SDO 14
timeout, SDO 11
TPDO
  PDO 11
  setup 14
transmission, Examples CAN frame 15,
  19
triggered messages, Event 9
-U-
U16, Datatypes 9
U24, Datatypes 9
U32, Datatypes 9
U40, Datatypes 9
U48, Datatypes 9
U56, Datatypes 9
U64, Datatypes 9
U8, Datatypes 9
unicode string, Datatypes 9
unregister
  Emergency 14
  Heartbeat 14
  Nodeguarding 14
  PDO 14
  SDO 14
UNIGNED16, Datatypes 9
UNIGNED24, Datatypes 9
UNIGNED32, Datatypes 9
UNIGNED40, Datatypes 9
UNIGNED48, Datatypes 9
-V-
verbose 14
Version, Gateway management 13
visible string, Datatypes 9
vs, Datatypes 9
-W-
write, PDO 11