EKSPLA

USB-CAN-LAN-RS232

CONVERTER module

User's Manual

Rev. 1805

2018

Lithuania

Contents

LIST OF FIGURES

LIST OF TABLES

Chapter 1 GENERAL INFORMATION

This document describes remote control of EKSPLA products using the USB-CAN-LAN-RS232 CONVERTER Module.

The Module is installed in EKSPLA products either as an optional add-on or as a standard feature. If the Module is provided as an optional add-on, its interfaces will function in parallel with the native USB/RS232 interfaces of the laser. Native USB/RS232 interfaces use different drivers and protocols and should be not confused with Module interfaces.

This document covers interfaces associated with the Converter module only.

1.1. Short Description

The Module acts as a communication bridge allowing control of various Ekspla products over a number of hardware interfaces:

- RS232
- USB
- Local Area Network (LAN)
- Wireless LAN (WLAN)

In addition to communication functions, the Module can log operational parameters to an SD card or USB stick.

The Module is a single printed circuit board (PCB), mounted either into the laser power supply unit (PSU), or into a separate enclosure:

– Internally mounted modules are installed into many types of laser PSUs, e.g. PS8000, PS8600, PS6000. The Module is distinguished from other interfaces by connectors named USB HOST, USB DEVICE, LAN and RS232.

– The standalone Converter module type is named ‘USB-CAN-LAN-RS232 CONVERTER’.

A major advantage of using the Converter is that it allows controlling and diagnosing the laser from virtually any device: a microprocessor, PC, a smartphone, running any operating system (Windows, iOS, Linux etc.) No software driver installation is necessary in most cases, except USB connection to Windows versions XP…8 (some exceptions apply).

No special application is needed to control the laser. Because the communication is text-based, standard applications, such as ‘terminal’ for USB/RS232 or a web browser for LAN, are sufficient.

The Converter module can bridge the following hardware interfaces:

- USB virtual serial port – EKSPLA CAN bus
- RS232 serial port – EKSPLA CAN bus

- 10BASE-T/100BASE-TX based LAN – EKSPLA CAN bus

The EKSPLA CAN bus is an internal bus that interconnects all laser drivers and sensors. The laser system can be controlled and diagnosed by sending messages via this bus. Although direct bridges work only in direction from/to the EKSPLA CAN bus, the messages on the bus are visible from all interfaces. This is useful for communication debugging, because it allows bus sniffing via a different interface.

1.2. Communication Protocols

1.2.1. Ethernet (LAN, Wireless LAN (WLAN))

- REST API running over HTTP protocol (HTTP GET command).

  HTML is used for messages; therefore any web browser may be used for communication testing.

- CAN messages tunneling over TCP/IP

  Closed protocol. *REMOTECONTROL.dll* is used on the PC side to communicate with the laser. The *CAN Browser* application uses this mode of communication.

1.2.2. USB (USB DEVICE connector)

- Protocol based on ASCII readable strings

  Plain text messages are used for messaging; therefore, any terminal application is suitable for communication testing.

- CAN messages tunneling over a USB virtual serial port

  Closed protocol. *REMOTECONTROL.dll* is used on the PC side to communicate with the laser.

1.2.3. RS232 (RS232 connector)

- Protocol based on ASCII readable strings

  Plain text messages are used for messaging; therefore, any terminal application is suitable for communication testing.

- CAN messages tunneling over a USB virtual serial port

  Closed protocol. *REMOTECONTROL.dll* is used on the PC side to communicate with the laser.

USB and RS232 share the same protocols.

Note:
Due to limited support for MS Windows 7 and earlier versions, driver installation and proper operation is guaranteed to work only with MS Windows 10.

No special installation procedures are needed for:

- LAN, Wireless LAN and RS232 – on all systems.
- USB – on MS Windows 10, Linux.

USB on Windows XP, Windows Vista, Windows 7 and Windows 8 requires installation of a USB-serial driver. This happens automatically for some computers connected to the Internet, as soon as the USB cable from the Module is connected to the PC. In case this automatic installation fails, a manual procedure will have to be performed.

First of all, download the required file; click the following link, select 'Save as', save the file to disk:

https://www.kernel.org/doc/Documentation/usb/linux-cdc-acm.inf

It is also possible to enter the link into a web browser, copy the contents of the displayed page to a blank *Notepad* document and save it as *linux-cdc-acm.inf* (make sure to save it in the *.inf* format).

2.1. Windows 7/8

1. Open the *Start menu* and:
   a. Windows 7: open *Control Panel* → *System and Security* → *System* → *Device Manager*.
   b. Windows 8: point to *Settings*, select *Control Panel,* select *System*. Then, on the left, click *Device Manager*.
2. In *Ports (COM & LPT)*, you should see an open port *Gadget Serial V2.4*.

   Please note the number of this port – it is relevant when connecting the *CAN Browser* application to your product.
3. Right-click *Gadget Serial V2.4*, choose the *Update Driver Software* option.
4. Select *Browse my computer for driver software*.
5. Navigate to the location of the downloaded file by selecting the *Let me pick from a list of device drivers on my computer* option, then *Show All Devices*, then *Have disk…*
6. Windows will display a message that the driver did not pass Windows Logo testing; ignore it, select *Continue anyway* and finish the driver installation.

2.2. Windows XP

1. After plugging in the module, the *Found New Hardware Wizard* starts up.
2. Select *Install from a list or specific location (Advanced)*.
3. Select *Include this location in the search* and enter the path or browse to the folder containing the file.
4. Windows will display a message that the driver did not pass Windows Logo testing; select *Continue anyway* and finish the driver installation.
5. Go to *Start Menu → Control panel → System → Hardware → Device Manager*.
6. In the *Device Manager*, expand *Ports (COM & LPT) – Gadget Serial* should be listed as the driver for one of the COM ports.

Once the drivers have been successfully installed, the USB to PC connection will appear as a COM port.

Chapter 3 REST API over LAN or WLAN Protocol

Use the LAN connector on the front panel of the Module to physically connect to the Module. The WiFi module is optional.

See Chapter 5 for setup instructions.

3.1. Communication Control

The communication is controlled by a LAN/WLAN/USB/RS232/CAN communication server. It links different buses by relaying messages, keeps the system configuration file(s), and stores messages in queues to enable data transfer between buses with different speeds and latency times. More complex systems may have multiple servers attached to several CAN bus segments.

Figure 1 illustrates how individual entities are addressed within a multi-server (left) and a stand-alone (right) system.

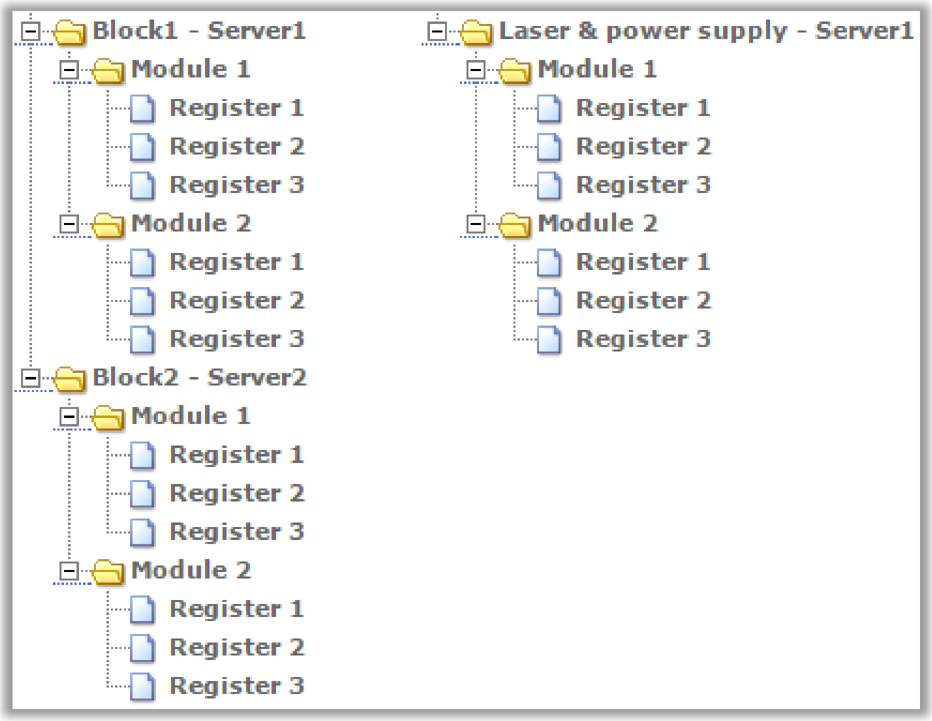


Figure 1 Laser/block hierarchies for multi-server (left) and stand-alone (rigth) systems

3.2. Addressing Conventions

Servers are addressed either by IP or by host name. REST requests are served by a special low-latency HTTP server on the 8080 port. Internally, the application software uses *Device ID* for block/laser identification.

Modules are addressed by name and ID in the form */(Module name)/(ID).*

Registers are addressed by name.

3.3. Modules and Registers

The internal block/laser hierarchy is stored in a special configuration file REMOTECONTROL.CSV. The CAN Browser service utility reads the hierarchy on the fly, the rest of the software uses the stored REMOTECONTROL.CSV file.

Table 1 describes the columns of the register list.

Table 2 lists the data types for the *Type* column of the register list.

Table 6 (see Chapter 8) is an example of a laser register list. The actual register list of your product can be found in the software CD folder *\CAN network\Remote Control test utility\REMOTECONTROL.csv* or can be downloaded from the server over the LAN Connection at http://your_host_or_ip/REMOTECONTROL.CSV.

Table 1 Columns of the register list for REST API

Column	***Description***
Name/ID	Each module has an ID and a name. ID is a number in the range 0…63. (decimal), but in the table, it is given in HEX form, e.g. $10 in HEX is 16 in DEC. In commands, the decimal value of ID is used, and it is tied to the name. If the product is comprised of a single module only, the command will always begin with the same symbols: /LDD1A/18.
Menu	Not used in remote control.
Type	Type of data in the register, see Table 2. Protocol operates in numbers converted to floating point and text representation; therefore the type may be ignored.
User rights	In remote control, value *ArUrSr* marks a read-only parameter. All other values are for writable parameters.
Non-volatile	The *NV* value indicates that a parameter may be written to non-volatile memory.
Min value / Max value	Parameter bounds. If exceeded, the communication controller will respond with an error message.
Short name	Not used.
Print format	Integer. *%u[x]*, where x is the dimension of parameter. Decimal. Because the register value is an integer, for display purposes it must be divided by a factor, e.g. *%.1f[x]* – by 10, *%.2f[x]* – by 100, etc. Float. *%f[x]* Set type [XXX, YYY, ...], with register type *u8*. The register value points to the actual (n-1)th element of the set. In ASCII remote control, actual set elements are sent, not the register value, e.g. if the print format is [OFF, ON], the reply to an inquiry will be ‘OFF’ or ‘ON’, not 0 or 1.
Name	Full register name, used to address the parameter. Copy this value to *Register name* argument when making function calls.
Value	Value of register.

Table 2 Register data types for *Type* column

Type	***Description***
u8	byte, 0..255, unsigned 8-bit
s8	shortint, -128..127, signed 8-bit
u16	word, 0..65535, unsigned 16-bit
s16	smallint, -32768..32767, signed 16-bit
u32	longword, 0..4294967295, unsigned 32-bit
s32	long integer, -2147483648..2147483647, signed 32-bit
float	single, 1.5 x 10^-45 .. 3.4 x 10^38, Significant digits 7-8
string8	array of bytes

3.4. Protocol Description

Commands and queries are sent by issuing HTTP GET requests, which are not authenticated.

The command line is an URL string.

The response body is an html document.

URL starts with the host name (IP address) and port, e.g. http://host:8080 or http://ip:8080. The host name is programmable. By default, it is the product serial number. Please note that not all networks resolve the host name properly. See Chapter 5 for details about host names and IP addresses.

All of the following examples are shown without the host address part, e.g. instead of http://192.168.1.192:8080/list(), you will see just /list().

Command/query groups:

– Server commands/queries - /ID(), /List(), /Logget(), /Logstart()

  These are not relayed to the CAN bus and work even with no modules live on the CAN bus.

– CAN module commands/queries - /(Module name)/(ID)/(Register name)

  Modules are commanded and queried by performing register read and write operations.

3.4.1. Commands and Queries

A short list of commands and queries is presented in Table 3.

See Table 8 in Chapter 8 for an expanded list of commands with their sample responses. See Chapter 6 Command Errors for a list of command errors.

Table 3 Commands and queries for REST API

Name	***Syntax***
Communication test	*/*
Device ID	*/id()*
Register list query	*/list()*
Read command	*/(Module name)/(ID)/(Register name)*
Write command	*/(Module name)/(ID)/(Register name)/(Value)*
Write to NVRAM command	*/(Module name)/(ID)/(Register name)/(Value)/NV*
Start/stop logging register updates	*/(Module name)/(ID)/(Register name)/logstart()/(buffer size in bytes)* Here: *(buffer size in bytes)* > 0 will turn on logging register changes *(buffer size in bytes)* = 0 will turn off logging register changes
Receive values with timestamp from the log	*/(Module name)/(ID)/(Register name)/logget()/(number of records)*

This chapter describes remote control of Ekspla products via RS232 or USB DEVICE ports.

The commands shown here are just examples. For relevant commands, see the REMOTECONTROL.CSV configuration file of your product.

4.1. Physical Connection

To connect, use the RS232 female type connector or USB DEVICE connector on the front panel of your product.

The parameters of RS232 port are as follows:

- Speed: 19200
- Data bits: 8
- Stop bits: 1
- Parity: none
- Flow control: none

Figure 2 shows the RS232 connector pinout between the PC and the controller. The cable is terminated by a male-female D-Sub 9 pin connector.

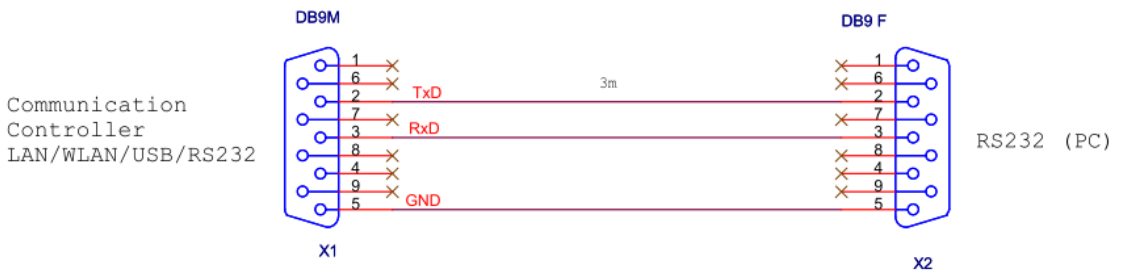


Figure 2 RS232 connector pinout

4.2. Registers

The internal block/laser hierarchy is stored in a special REMOTECONTROL.CSV configuration file. The CAN Browser service utility reads the hierarchy on the fly, the rest of the software uses the stored REMOTECONTROL.CSV file.

Table 3 is an example of a laser register list. The actual register list of your product can be found in the software CD folder *\CAN network\Remote Control test utility\REMOTECONTROL.csv* or can be downloaded from the server over the LAN Connection at http://your_host_or_ip/REMOTECONTROL.CSV.

Table 4 describes the register columns.

Table 7 (see Chapter 8) is an example of a laser register list

Table 4 Columns of the register list for ASCII

Column	***Description***
Module name / Module ID	Each module has an ID and a name. ID is a number in the range 0…63 (decimal). Module name and ID are tied together in commands, e.g. */SY3PL50M/32*.
Type	Type of data in the register, see Table 2. Protocol operates in numbers converted to floating point and text representation; therefore the type may be ignored.
User rights	In remote control, value *ArUrSr* marks a read-only parameter. All other values are for writable parameters.
Non-volatile	The *NV* value indicates that a parameter may be written to non-volatile memory.
Min value / Max value	Parameter bounds. If exceeded, the communication controller will respond with an error message.
Print format	Integer. *%u[x]*, where x is the dimension of parameter. Decimal. Because the register value is an integer, for display purposes it must be divided by a factor, e.g. *%.1f[x]* – by 10, *%.2f[x]* – by 100, etc. Float. *%f[x]* Set type [XXX, YYY, ...], with register type *u8*. The register value points to the actual (n-1)th element of the set. In ASCII remote control, actual set elements are sent, not the register value, e.g. if the print format is [OFF, ON], the reply to an inquiry will be 'OFF' or 'ON', not 0 or 1.
Register name	Full register name, used to address the parameter. Copy this value to *Register name* argument when making function calls.
Captured value	Value of register.
Comments	Comments.

4.3. Protocol Description

The command is an ASCII string.

Symbols used in communication:

- #13 – carriage return [CR] ('Enter' key)
- #10 – line feed
- #03 – end of text

The command is entered by a #13 [CR] symbol.

#13#10 signals the end of line.

#03 ends the message.

Three apostrophes in a row (' ' ') precede a command error message.

4.3.1. Commands and Queries

A short list of commands and queries is presented in Table 5. See Table 9 in Chapter 8 for an expanded list of commands with their sample responses. See Chapter 6 Command Errorsfor a list of command errors.

Table 5 Commands and queries for ASCII

Name	***Syntax***
Communication test	/
Device ID	*/id()*
Register list query	*/list()*
Read command	*/(Module name)/(ID)/(Register name)*
Write command	*/(Module name)/(ID)/(Register name)/(Value)*
Write to NVRAM command	*/(Module name)/(ID)/(Register name)/(Value)/NV*
Start/stop logging register updates	*/(Module name)/(ID)/(Register name)/logstart()/(buffer size in bytes)* Here: *(buffer size in bytes)* > 0 will turn on logging register changes *(buffer size in bytes)* = 0 will turn off logging register changes
Receive values with timestamp from the log	*/(Module name)/(ID)/(Register name)/logget()/(number of records)*

This page is intentionally left blank

Chapter 5 SETUP OF LASER CONTROL OVER LAN OR WLAN

This chapter describes the setup of laser control over LAN or Wireless LAN.

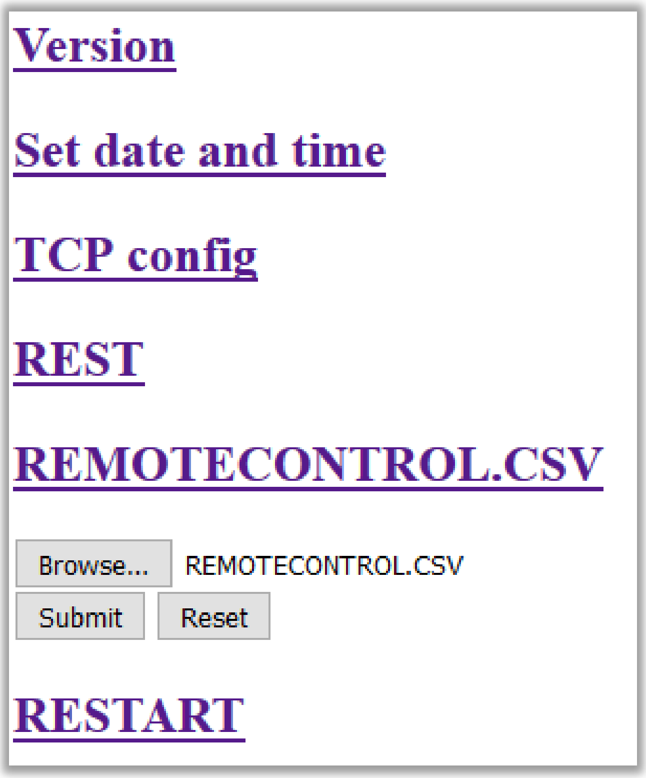
Use the LAN connector on the front panel.

The WiFi module is optional. If applicable, the WiFi antenna is attached to the SMA connector labelled 'ANT'.

5.1. LAN Configuration

The server is initially configured for dynamic IP addressing. First, it is necessary to connect the laser to a local network running a DHCP server. After this, it is possible to change the configuration, once the communication has been established. Please note that a direct connection to PC via a LAN cable will not work, unless the PC is running DHCP server software.

1. Connect LAN connector to the Ethernet switch or router.
2. Connect PC to the same network.
3. Turn the laser ON.
4. Enter http://Serial_Number_of_Your_Product (e.g. http://PGL426) into the browser address bar.
5. The Main setup page is displayed (Figure 3).



Version

Set date and time

TCP config

REST

REMOTECONTROL.CSV

RESTART

Figure 3 Main setup page

6. Click *TCP config* to open the Network setup page.

7. Use the Network setup page to enable/disable LAN and WLAN interfaces and configure connection parameters.

8. Click *Submit* to save settings.

   Done. Restarting… will appear on the screen.

   After some time (usually tens of seconds), revisit the Main setup page by entering the URL discussed above.

   If the wireless connection is used, add 'W' to the end of your URL, e.g. http://PGL426W.

5.2. Important Notes

- LAN connections are not password-protected.
- Any connection to the laser via LAN or WLAN in order to control it remotely must be done on a dedicated, limited access network (without access to the Internet). Connecting to a general office network will introduce hazards and risks to the device and personnel. This is especially important for wireless connections, as there is no easy way to quickly terminate communication (like unplugging a cable).
- The wireless module is radio equipment. Often there are import restrictions on radio equipment; therefore, the server is normally shipped without the wireless module installed.

"(-1) Log FIFO overrun"

"(-1) 3rd argument is missing (/%s/%s/???)"

"(-1) 2nd and 3rd arguments are missing (/%s/???/???)"

"(1) End of registers list or enumerated values list"

"(2) No config file found"

"(3) wrong CFG file"

"(4) Application provided return buffer is too short"

"(5) No such device name"

"(6) No such register name"

"(7) Can't connect"

"(8) Timeout waiting for device answer"

"(9) Register is read only"

"(10) Register is not NV capable"

"(11) Violating top value limit"

"(12) Violating bottom value limit"

"(13) Wrong value, not included in allowed values list"

"(14) Register is not being logged"

"(15) Not enough memory"

"(16) Queue is empty"

"(17) Already connected, please disconnect first"

"(18) Not connected, please connect first"

This page is intentionally left blank

Chapter 7 COMMUNICATION TROUBLESHOOTING GUIDE

This chapter describes establishing, maintaining and troubleshooting communication with Ekspla lasers fitted with the Module.

It is assumed that installation and connecting procedures are accomplished.

7.1. Physical Connection

The Converter Module provides the following physical connectors on the front panel:

- USB DEVICE – the PC uses the USB communication devices class (CDC) driver included in Windows.
- RS232 – DCE, only RxD, TxD and ground are used.
- LAN connector – RJ45, 100Mb Ethernet.

At least one connection should be used, but pairing USB+LAN or RS232+LAN is recommended. In case LAN is not used, use USB, RS232, or both.

7.2. Testing and Troubleshooting Process Flow

1. Examination of the status of interfaces using the Windows Management Instrumentation (WMI) tool.
2. Probing of RS232 or virtual COM through USB connection.
3. Reading of data.

7.3. Communication Testing Utility

All three steps listed above are performed by the dedicated *CommTest.exe* utility. It uses the WMI service to discover COM ports, identifies the laser USB virtual COM port, probes the RS232 connection, scans the LAN for Ekspla lasers, etc.

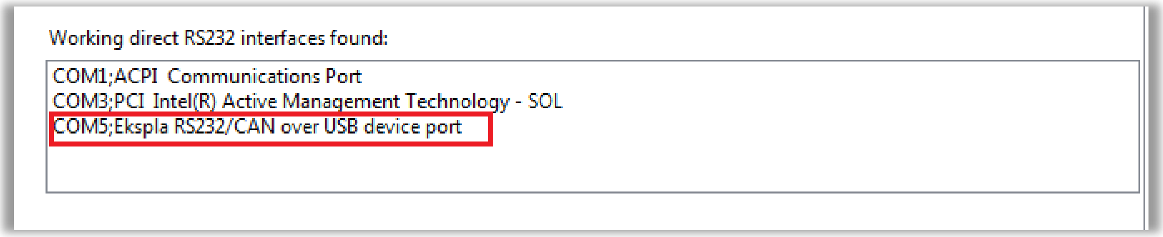
The utility is located in the \Tools folder. It should be copied to your local hard disk drive and run from there.

After the utility is closed, a *log.txt* file is created in the same folder. The log file contains responses from WMI queries. The log file should be sent to Ekspla together with questions if support is needed.

7.4. Step-by-step Instructions

1. Connect the USB, LAN cables.
2. Run the *CommTest.exe*.

3. Press the *Start inspection of hardware interfaces* button. If recognized the COM port will appear in *Working direct RS232 interfaces found* list.



Working direct RS232 interfaces found:

COM1;ACPI Communications Port
COM3;PCI Intel(R) Active Management Technology - SOL
COM5;Ekspla RS232/CAN over USB device port

Figure 4 *Working direct RS232 interfaces found* list

Please note that the entry *Ekspla RS232/CAN over USB device port* in the list is just a preliminary guess by the software. There is no reliable way to determine this using only WMI tools. Further steps describe the probing of ports.

If the serial driver is not loaded, a *Faulty interfaces found* message will appear.

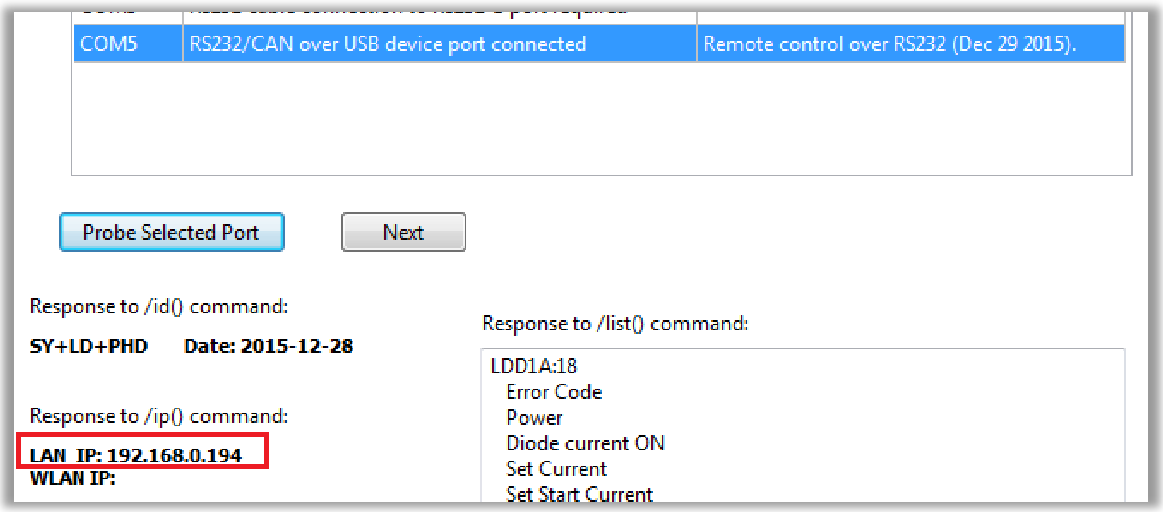
When using Windows 10, the procedure should complete regardless of the serial driver loading status. For earlier Windows versions, loading of the serial driver depends on the success of the compatible driver search procedure. This procedure requires an internet connection and a fully updated Windows PC, with no updates/restarting pending or deferred.If the search procedure fails, install the *linux-cdc-acm.inf* file from

https://www.kernel.org/doc/Documentation/usb/linux-cdc-acm.inf

The .inf file instructs Windows to explicitly use *usbserial.sys* for the virtual COM port.

4. Press *Next*. Working serial interfaces are listed. Choose the port or use the suggested one.

5. Press P*robe Selected Port* to check communication. This sends /id(), /list() and /ip() commands to the selected serial port. The responses are displayed.



COM5	RS232/CAN over USB device port connected	Remote control over RS232 (Dec 29 2015).

Response to /id() command:
SY+LD+PHD Date: 2015-12-28

Response to /ip() command:
LAN IP: 192.168.0.194
WLAN IP:

Response to /list() command:
LDD1A:18
Error Code
Power
Diode current ON
Set Current
Set Start Current

Figure 5 Successful connection response

If there is no response to the /ip() command, try another COM port: select another one and press *Probe Selected Port* again. Try disconnecting and connecting the USB again.

For RS232 testing, check electrical wiring first. See connector pin signals on the laser side.

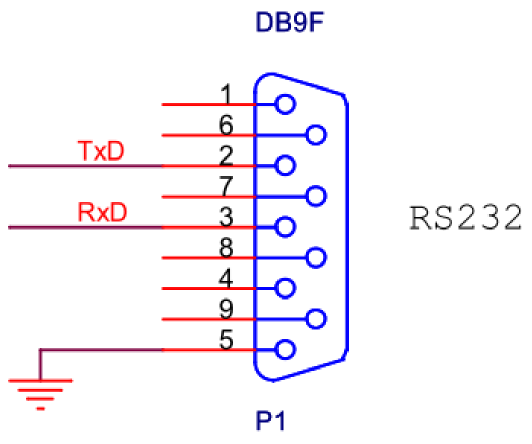
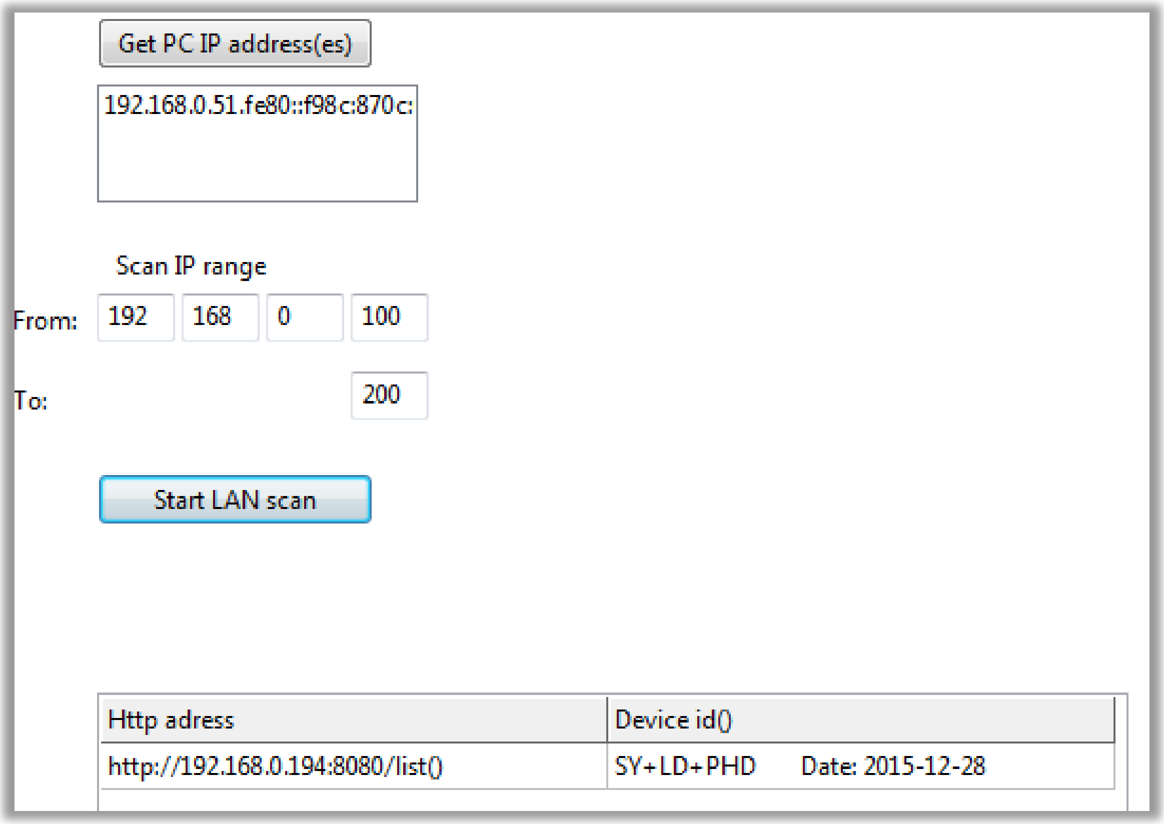


Figure 6 RS232 connector pinout on the laser side.

7.5. Testing LAN without an USB/RS232 Connection

1. Go to the *LAN* tab.
2. Press the *Get PC IP address(es)* button. This queries your PC's IP and fills the list. It is also possible to input IP addresses by hand.
3. Press the *Start LAN scan* button. It will attempt to connect to http://xxx.xxx.xxx.xxx:8080/id() URLs throughout all the IP range.

   The connection timeout is relatively short (100 ms) for the scan to be quick. This is usually enough for discovery in the LAN.



192.168.0.51.fe80::f98c:870c:

Scan IP range

From: 192 168 0 100

To: 200

Http adress	Device id()
http://192.168.0.194:8080/list()	SY+LD+PHD Date: 2015-12-28

Figure 7 LAN scan results

8.1. REST API over LAN or WLAN Register List Example

Table 6 REST API over LAN or WLAN register list example

Name	*ID*	*Reg ID*	*Menu*	*Type*	*User rights*	*Non-volatile*	*Min value*	*Max value*	*Short name*	*Print format*	*Name*	*Value*
LDD1A	$12	$10	$0	u8	AUS	NV	0	1	State	[OFF,ON,FAULT]	Power	FAULT
LDD1A	$12	$12	$2	u16	AUrS	NV	0	2500	Iset	%.3fA	Set Current	0.850
LDD1A	$12	$47	$2	u16	AUrS	NV	0	2500	Istart	%.3fA	Set Start Current	0.940
LDD1A	$12	$13	$3	u16	ArUrSr		0	1000	Idisp	%.3fA	Display Current	0.021
LDD1A	$12	$15	$2	u16	AUrS	NV	0	1060	Tstat	%.1fC	Thermostat	100.0
LDD1A	$12	$19	$9	u8	ArUrSr		0	3	Fault	[NONE, DRIVER, COOLING, INTERLOCK]	Fault source	COOLING
LDD1A	$12	$1A	$A	u16	ArUrSr		0	65535	Fcode	%xHEX	Fault code	400
LDD1A	$12	$1B	$B	u32	AUrS	NV	0	4294967295	Time	%us	Work seconds	22812090
LDD1A	$12	$28	$23	u8	ArUrSr		0	2	IntlED	[OK,NOT OK,Defeated]	External interlock state	Defeated
LDD1A	$12	$29	$24	u8	ArUrSr		0	2	IntlCD	[OK,NOT OK,Defeated]	Cooling interlock state	NOT OK
LDD1A	$12	$2A	$31	u16	ArUrSr		0	1023	PhdOut	%u	Feedback photodiode output	20
LDD1A	$12	$32	$41	s16	ArUrSr		-2744	5000	Tdisp	%.2fC	Display temperature	47.38
LDD1A	$12	$34	$41	u8	ArUrSr		0	1	Stable	[Not Stable,Stable]	Stable	Not Stable
LDD1A	$12	$9	$F	u32	ArUrSr		0	4294967295	Version	%06u	Firmware	LDD1A; 131112
LDD1A	$12	$2	$0	u16	ArUrSr		0	65535	ErrCode	%04x	Error Code	

8.2. ASCII Serial Protocol Register List Example

Table 7 ASCII serial protocol register list example

Module name	***Module ID***	***Type***	***User rights***	***Non-volatile***	***Min value***	***Max value***	***Print format***	***Register name***	***Captured value***	***Comments***
PHD1K000	48	u16	ArUrSr		0	65535	%u	Data	31956	1064nm output energy sensor, raw ADC data. Sensor is located before attenuator
PHD1K000	48	float	ArUrSr		0	3,40 E+52	%f	Mean	100.997	1064nm energy sensor, calibrated and averaged for 1sec
SY3PL50M	32	u8	AUS	NV	0	1	[OFF,ON, Failure]	State	ON	Laser state. OFF - laser stopped, ON - laser is firing. 'RUN' and 'STOP' buttons on the control pad change this register
SY3PL50M	32	u8	AS		0	2	[Continuous, Burst,Trigger]	Continuous / Burst mode / Trigger burst	Continuous	Burst mode control and trigger the burst
SY3PL50M	32	u32	AUS	NV	2	524287	%u 1/OptClk	PRE-T delay	2	Allows preset the **Pre-T OUT** pulse delay
SY3PL50M	32	u16	AUS	NV	10	65535	%.1fns	OUT3 delay	14.Rgs	**TRIG1 OUT** pulse delay
SY3PL50M	32	u16	AS		1	50000	%u	Burst length, pulses	1	Allows setting the number of pulses in the burst to be emitted when operating laser in burst mode
SY3PL50M	32	u8	AUrS	NV	0	1	[Internal, External]	Synchronization Mode	Internal	When **External** mode is set, laser is triggered from external signal, supplied to SYNC IN
SY3PL50M	32	u8	AUS	NV	0	2	[OFF, Adjustment, Maximum]	Energy level	OFF	Operating mode. See 'Parameters Description' in 5 Chapter of Manual for description

Module name	***Module ID***	***Type***	***User rights***	***Non-volatile***	***Min value***	***Max value***	***Print format***	***Register name***	***Captured value***	***Comments***
SY3PL50M	32	u16	AS	NV	1	5000	%u	Frequency divider	1	Laser pulse repetition rate is divided by a factor set in this menu. Pumping frequency remains unchanged
SY3PL50M	32	u16	AUS	NV	50	5000	%uus	Pump delay, adj. level	348	Delay of regen pump in adjustment mode. Allows tweak pulse energy in adjustment mode
SY3PL50M	32	u32	ArUrSr		0	4294967295	%uHz	Optical Clock	87551104	Oscillator frequency,
SY3PL50M	32	u16	ArUrSr		0	1000	%.1fHz	External SyncIn frequency	1000.1	Repetition rate, measured
SM5	61	s32	ArUrSr		-2,00 E+09	2147483647	%d	Current position	261	Attenuator stepper motor actual position
SM5	61	s32	AUS	NV	-2,00 E+09	2147483647	%d	Target position	261	Attenuator stepper motor target position
SM5	61	u8	AUS	NV	0	1	[POSITIONING, VELOCITY]	Mode	POSITIONING	Attenuator stepper motor driver mode
CPU8000	17	u16	ArUrSr		0	1300	%.1fA	Display Current	0.4	Laser diagnostic register
HV40W	40	u16	ArUrSr		0	4000	%uV	HV voltage	11	Laser diagnostic register
LDCO48BP	28	s16	ArUrSr		-200	4600	%.2fC	Display temperature	28.64	Laser diagnostic register
LDM6A	16	u16	ArUrSr		0	1000	%.2fA	Display Current	1.97	Laser diagnostic register
LDM6A	16	s16	ArUrSr		-2744	5000	%.2fC	Display temperature	30.98	Laser diagnostic register

8.3. List of Commands for REST over API Protocol

Table 8 List of commands for REST over API protocol

Communication test	
Command	/ [slash symbol]

Sample response	Remote control REST app (Nov 10 2015).
HTML format	<html><TABLE> <tr><th>Remote control REST app (Nov 10 2015). </th></tr> </TABLE></html>
Device ID query	
Command	/id() Sends the first line from REMOTECONTROL.CSV file.
Sample response	Device ID SY320100 Date: 2015.10.29 Here, ‘SY320100’ is the device type, ‘Date’ is the configuration file creation date.
HTML format	<html><TABLE> <tr><th>Device ID</th></tr> <tr><td id=DN>SY320100 Date: 2015.10.29 </td></tr> </TABLE></html>
Register list query	
Command	/list() Sends list of registers from REMOTECONTROL.CSV file, together with their respective URLs.
Sample response	IO:15 Error Code RA0,pin22,AN0 RA0 direction RA1,pin23,AN1 !SY3PL50M:32 Error Code State Synchronization Mode External SyncIn frequency Photodiode bias Optical Clock
HTML format	<TABLE> <tr><th>IO:15</th></tr> <tr><td><A HREF="http://192.168.0.94:8080/IO/15/Error Code">Error Code</A></td></tr> <tr><td><A HREF="http://192.168.0.94:8080/IO/15/RA0,pin22,AN0">RA0,pin22,AN0</A></td></tr> <tr><td><A HREF="http://192.168.0.94:8080/IO/15/RA0 direction">RA0 direction</A></td></tr> <tr></tr> <tr><th>SY3PL50M:32</th></tr> <tr><td><A HREF="http://192.168.0.94:8080/SY3PL50M/32/Error Code">Error Code</A></td></tr> <tr><td><A HREF="http://192.168.0.94:8080/SY3PL50M/32/State">State</A></td></tr> <tr><td><A HREF="http://192.168.0.94:8080/SY3PL50M/32/Synchronization Mode">Synchronization Mode</A></td></tr> <tr><td><A HREF="http://192.168.0.94:8080/SY3PL50M/32/External SyncIn frequency">External SyncIn frequency</A></td></tr> <tr><td><A HREF="http://192.168.0.94:8080/SY3PL50M/32/Photodiode bias">Photodiode bias</A></td></tr> <tr><td><A HREF="http://192.168.0.94:8080/SY3PL50M/32/Optical Clock">Optical Clock</A></td></tr> <tr></tr> </TABLE>

Read command (numerical type variable)	
Command	/(Module name)/(ID)/(Register name) E.g. /SY3PL50M/32/Oscillator clock
Sample response	Get register Device SY320100:32 Register Optical Clock Min. value 0 Max. value 4.29497e+09 RW No NV No Format %uHz Error (0) Success, no error Value 0Hz

HTML format	`<html><body><TABLE>`
	`<tr><th>Get register</th></tr>`
	`<tr><th></th><th></th></tr>`
	`<tr><td>Device</td><td id=D1>SY320100:32</td></tr>`
	`<tr><td>Register</td><td id=R1>Optical Clock</td></tr>`
	`<tr><td>Min. value</td><td id=minV>0</td></tr>`
	`<tr><td>Max. value</td><td id=maxV>4.29497e+09</td></tr>`
	`<tr><td>RW</td><td id=RW>No</td></tr>`
	`<tr><td>NV</td><td id=NV>No</td></tr>`
	`<tr><td>Format</td><td id=FMT>%uHz</td></tr>`
	`<tr> <td>Error</td> <td id=E1>(0) Success, no error</td> </tr>`
	`<tr><td>Value</td><td id=V1>0Hz</td></tr></TABLE></body></html>`

Read command (set type variable)	
Command	/(Module name)/(ID)/(Register name) E.g. /SY3PL50M/32/State

Sample response	Get register Device SY320100:32 Register Command Min. value 0 Max. value 4 RW Yes NV Yes Format [POWEROFF,SLEEP,STOP,PAUSE,RUN,FAULT] Error (0) Success, no error Value FAULT
HTML format	<html><body><TABLE><tr><th>Get register</th></tr> <tr><th></th><th></th></tr> <tr><td>Device</td><td id=D1>SY320100:32</td></tr> <tr><td>Register</td><td id=R1>Command</td></tr> <tr><td>Min. value</td><td id=minV>0</td></tr> <tr><td>Max. value</td><td id=maxV>4</td></tr> <tr><td>RW</td><td id=RW>Yes</td></tr> <tr><td>NV</td><td id=NV>Yes</td></tr> <tr><td>Format</td><td id=FMT>[POWEROFF,SLEEP,STOP,PAUSE,RUN,FAULT]</td></tr> <tr> <td>Error</td> <td id=E1>(0) Success, no error</td> </tr> <tr><td>Value</td><td id=V1>FAULT</td></tr></TABLE></body></html>
Write command	
Command	(Module name)/(ID)/(Register name)/(Value) E.g. /SY320100/32/Repetition rate/500 This example command will set the oscillator clock to 500 kHz.
Sample response, accepted command	Set register to Device SY320100:32 Register Repetition rate Value 500 Error (0) Success, no error
HTML format, accepted command	<body><TABLE><tr><th>Set register to</th></tr> <tr><th></th><th></th></tr><tr><td>Device</td><td id=D1>SY320100:32</td> <tr><td>Register</td><td id=R1>Repetition rate</td> <tr><td>Value</td><td id=V1>500</td> <tr> <td>Error</td> <td id=E1>(0) Success, no error</td> </tr></TABLE></body>

Sample response, rejected command	Set register to Device SY320100:32 Register Repetition rate Value 1005 Error (11) Violating top value limit
HTML format, rejected command	<body><TABLE> <tr><th>Set register to</th></tr> <tr><th></th><th></th></tr><tr><td>Device</td><td id=D1>SY320100:32</td> <tr><td>Register</td><td id=R1>Repetition rate</td><tr><td>Value</td><td id=V1>1005</td> <tr> <td>Error</td> <td id=E1>(11) Violating top value limit</td> </tr></TABLE></body>
Write to NVRAM command format	
Command	/(Module name)/(ID)/(Register name)/(Value)/NV E.g. /SY320100/32/Repetition rate/500/NV
Sample response	Set NV register to Device SY320100:32 Register Repetition rate Value 500 Error (0) Success, no error
HTML format	<body><TABLE><tr><th>Set NV register to</th></tr> <tr><th></th><th></th></tr> <tr><td>Device</td><td id=D1>SY320100:32</td> <tr><td>Register</td><td id=R1>Repetition rate</td> <tr><td>Value</td><td id=V1>500</td> <tr> <td>Error</td> <td id=E1>(0) Success, no error</td> </tr></TABLE></body>
Start/stop logging register updates	
Command	/(Module name)/(ID)/(Register name)/logstart()/(buffer size in bytes) E.g. /SY320100/32/Photodiode%20bias/logstart()/100 (buffer size in bytes) > 0 will turn on logging of register changes (buffer size in bytes) = 0 will turn off logging of register changes The sample command will turn on logging of register ‘Photodiode bias’.
Sample response, accepted command	Start register log Device SY320100:32 Register Photodiode bias Log memory size 100 Error (0) Success, no error

HTML format, accepted command	<body><TABLE><tr><th>Start register log</th></tr> <tr><th></th><th></th></tr> <tr><td>Device</td><td id=D1>SY320100:32</td> <tr><td>Register</td><td id=R1>Photodiode bias</td> <tr><td>Log memory size</td><td id=V1>100</td> <tr> <td>Error</td> <td id=E1>(0) Success, no error</td> </tr></TABLE></body>
	Receive values with timestamp from the log
Command	/(Module name)/(ID)/(Register name)/logget()/(number of records) E.g. /SY3PL50M/32/Photodiode bias/logget()/5
Sample response, buffer overrun error	Get register log Device SY320100:32 Register Photodiode bias Error Log overflow detected Data Timestamp [ms] 5.93V 530169612 5.93V 530169633 5.93V 530169653 5.93V 530169674 5.93V 530169695
HTML format, buffer overrun error	<body><TABLE><tr> <th>Get register log</th> </tr> <tr> <th></th> <th></th> </tr> <tr> <td>Device</td> <td id=D1>SY320100:32</td> </tr> <tr> <td>Register</td> <td id=R1>Photodiode bias</td> </tr> <tr> <td>Error</td> <td id=E1>Log overflow detected</td> </tr> <tr> </tr></TABLE><TABLE id=L1><tr> <th>Data</th> <th>Timestamp [ms]</th> </tr> <tr> <td>5.93V</td> <td>530169612</td> </tr> <tr> <td>5.93V</td> <td>530169633</td> </tr> <tr> <td>5.93V</td> <td>530169653</td> </tr> <tr> <td>5.93V</td> <td>530169674</td> </tr> <tr> <td>5.93V</td> <td>530169695</td> </tr> <tr> <td></td> <td></td> </tr><tr> <td></td> <td></td> </tr> <tr> <td></td> <td></td> </tr><tr> <td></td> <td></td> </tr> <tr> <td></td> <td></td> </tr></TABLE></body>
Command	/(Module name)/(ID)/(register name)/logget()/(number of records) E.g. /CAMERA/57/Mean/logget()/10

Sample response, no errors	Get register log	
	Device	CAMERA:57
	Register	Mean
	Error	(0) Success, no error
	Data	Timestamp [ms]
	384.000000	535959608
	384.000000	535960119
	384.000000	535960630
	384.000000	535961142
	384.000000	535961653
HTML format, no errors	`<TABLE><tr> <th>Get register log</th> </tr>`	
	`<tr> <th></th> <th></th> </tr><tr> <td>Device</td>   <td id=D1>CAMERA:57</td> </tr>`	
	`<tr> <td>Register</td> <td id=R1>Mean</td> </tr>`	
	`<tr> <td>Error</td> <td id=E1>(0) Success, no error</td> </tr>`	
	`<tr>  </tr></TABLE><TABLE id=L1><tr> <th>Data</th> <th>Timestamp [ms]</th> </tr>`	
	`<tr> <td>384.000000</td> <td>535959608</td> </tr>`	
	`<tr> <td>384.000000</td> <td>535960119</td> </tr>`	
	`<tr> <td>384.000000</td> <td>535960630</td> </tr>`	
	`<tr> <td>384.000000</td> <td>535961142</td> </tr>`	
	`<tr> <td>384.000000</td> <td>535961653</td> </tr>`	
	`<tr> <td></td> <td></td> </tr><tr> <td></td> <td></td> </tr>`	
	`<tr> <td></td> <td></td> </tr><tr> <td></td> <td></td> </tr>`	
	`<tr> <td></td> <td></td> </tr></TABLE>`	

8.4. List of Commands for ASCII Serial Protocol

Table 9 List of commands for ASCII serial protocol

	Communication test
Command	[CR]
Sample response	Remote control over RS232 (Jun 18 2015) Here, ‘Jun 18 2015’ is the version date of the RS232 interpreter.
	Device ID query
Command	/id() [CR]

Sample response	Device: DNL207 Date: 17/09/2015 Here, ‘DNL207’ is the device serial number, ‘Date’ is the configuration file creation date.
Register list query	
Command	/list() [CR]

Sample response	LDM6A:16
	  Error Code
	  Display Current
	  Display temperature
	UCP:4
	  Error Code
	  Firmware UCPv3
	HV40W:40
	  Error Code
	  HV voltage
	TK6:44
	  Error Code
	  Display temperature
	SM5:61
	  Error Code
	  Current position
	  Target position
	  Mode
	CPU8000:17
	  Error Code
	  Display Current
	LDCO48BP:28
	  Error Code
	  Display temperature
	SY3PL50M:32
	  Error Code
	  State
	  Continuous / Burst mode / Trigger burst
	  PRE-T clock divider
	  PRE-T delay
	  OUT3 delay
	  Burst length, pulses
	  Synchronization Mode
	  Energy level
	  Frequency divider
	  Output redirection
	  Pump delay, adj. level
	  Optical Clock
	  External SyncIn frequency
	PHD1K000:48
	  Error Code
	  Data
	  Mean
	  Batch size
	PS5050:22
	  Error Code
	  Channel 1 state

Read command	
Command	/(Module name)/(ID)/(Register name) [CR] E.g. /SY3PL50M/32/State
Sample response	ON#13#10#03
Write command	
Command	(Module name)/(ID)/(Register name)/(Value) [CR] E.g. /SY3PL50M/32/State/ON This example command will turn on the laser.
Sample response, accepted command	#13#10#03

Sample response, rejected command	"'Error: (13) Wrong value, not included in allowed values list#13#10#03
Write to NVRAM command format	
Command	/(Module name)/(ID)/(Register name)/(Value)/NV [CR] E.g. /SY3PL50M/32/State/ON/NV This example command will turn the laser on and will make the ON value non-volatile, i.e. at the next power-on, the laser will try to run automatically.
Sample response, accepted command	#13#10#03
Sample response, rejected command	"'Error: (13) Wrong value, not included in allowed values list#13#10#03
Start/stop logging register updates	
Command	/(Module name)/(ID)/(Register name)/logstart()/(buffer size in bytes) [CR] E.g. /CPU8000/17/Display Current/logstart()/2000 (buffer size in bytes) > 0 will turn on logging of register changes (buffer size in bytes) = 0 will turn off logging of register changes The sample command will turn on logging of register ‘Display Current’.
Sample response, accepted command	#13#10#03
Receive values with timestamp from the log	
Command	/(Module name)/(ID)/(Register name)/logget()/(number of records) [CR] E.g. /CPU8000/17/Display Current/logget()/100
Sample response, no errors, messages present in the buffer	(value1) (time stamp1)#13#10 (value2) (time stamp2)#13#10 (value3) (time stamp3)#13#10 (value4) (time stamp4)#13#10 #03 0.020A 634698780#13#10 0.020A 634698799#13#10 0.020A 634698821#13#10 0.020A 634698840#13#10

Sample response, empty buffer	#13#10#03
Sample response, buffer overrun	0.020A 634698780#13#10 0.020A 634698799#13#10 0.020A 634698821#13#10 0.020A 634698840#13#10 0.020A 634698862#13#10 0.020A 634698881#13#10 '"Error: (-1) Log FIFO overrun#13#10#03
Sample response, rejected command	'"Error: (14) Register is not being logged#13#10#03
Device timer value query	
Command	/timestamp [CR]
Sample response	Timer value#13#10#03 Here, ‘Timer value’ is an actual value from the timer, e.g. 88479777.