laserDESK
laser processing software

Definition of the Remote Control Functionality

SCANLAB GmbH
Siemensstr. 2a
82178 Puchheim
Germany

Tel. +49 (89) 800 746-0
Fax: +49 (89) 800 746-199

info@scanlab.de
www.scanlab.de

(Doc. Rev. 2.4 e - April11, 2022)

Contents

1 Introduction

The remote control functionality enables a master control of laserDESK via the remote control interface. The purpose of this control is to execute and adjust the processing of laserDESK jobs. Especially, this includes the selection of the job to be processed, the definition of text contents and the start of job execution.

The remote control doesn't allow to create or modify graphic objects inside a job.

Generally, the remote control interface can be used for the processing tasks, but not for the design tasks of laserDESK.

Note:

The remote control functionality can only be executed if it is activated on the laserDESK USB dongle. The dongle must be configured for one of the two following laserDESK software packages:

– Standard + Remote Control
– Premium

If necessary, the dongle must be upgraded (please refer to the laserDESK Help, topics "Software Packages / Functional Range" and "Upgrading the laserDESK Software").

Note:

laserDESK jobs are executable only with an RTC5 or RTC6 PC interface board or an RTC6 Ethernet board from SCANLAB.

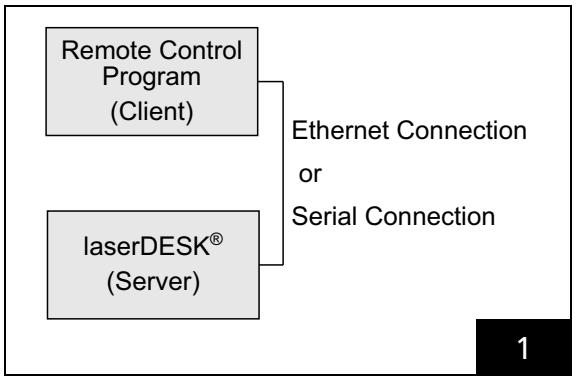
In the remainder of this manual, these boards will be referred to as "RTC board".

2 Configuration

There are two alternative options to connect the remote control program (client) to laserDESK (server):

- Ethernet connection with TCP/IP protocol
- Serial connection

The client can be installed on an external PC as well as on the server PC running laserDESK (internal).



Connection Remote Control Program – laserDESK

If the client is installed on the server PC, the connection can occur via virtual ethernet. Serial connection is not recommended (in that case two COM interfaces in the PC are required and must be connected by a 0-Modem wire).

Data transfer between client and server is realized via telegrams. Telegram and data format are the same for both connection types.

laserDESK can be operated by remote control under the following conditions:

- If the client is installed on an external PC, both PCs must be connected.
- laserDESK must be configured as described in chapter 2.1 and chapter 2.2.
- The laserDESK program must be started.
- laserDESK must not run in pilot laser mode.
- The access to laserDESK must not be blocked by another (remote control) program. Only one client can be connected at the same time.

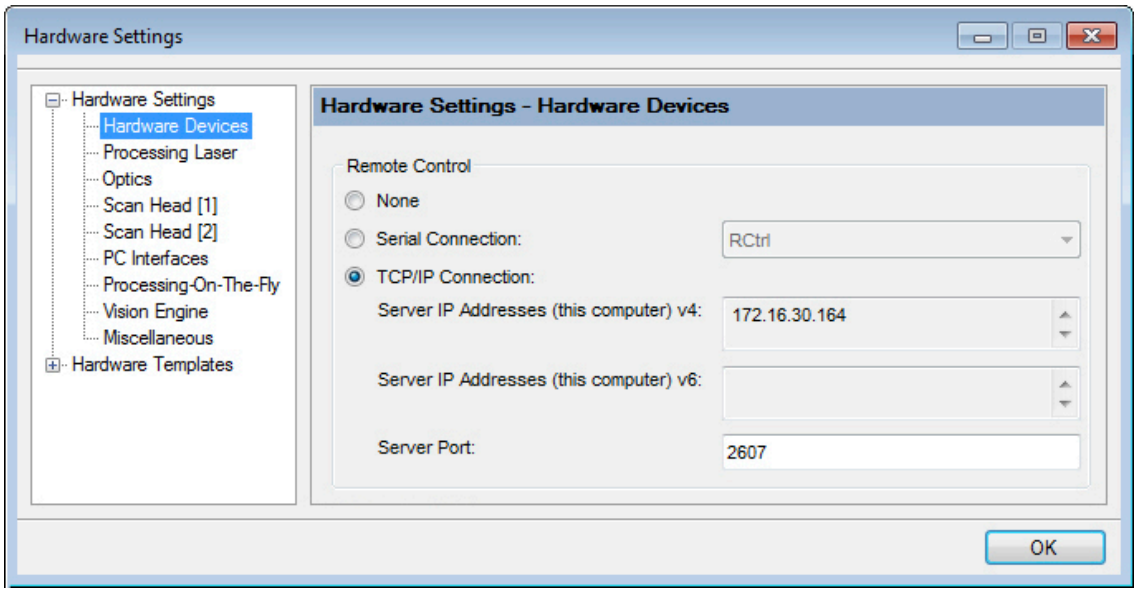
2.1 Ethernet Connection

The ethernet connection has following parameters:

Table 1

Parameter	Values	Remarks
Server IP-Address (laserDESK)	IP-Address of the server PC (PC running laserDESK)	Will be evaluated by the laserDESK program.
Client IP-Address (Remote Control Program)	Not to define	Will be evaluated by the laserDESK program when the client connects to the server (laserDESK) If the client is installed on the server PC, the IP-address "127.0.0.1" can be used.
Server Port	Port of the PC running laserDESK	Must be defined in 'Hardware Configuration' (see below)
Client Port	Not to define	Will be evaluated by the laserDESK program when the client connects to the server (laserDESK)

The ethernet connection parameters must be defined in the 'Hardware Configuration':



laserDESK 'Hardware Settings' Window – Remote Control Setting

- Select 'Open ▸ Hardware Configuration' in the 'File' menu.
- Select the 'Hardware Devices' section.
- Activate the 'TCP/IP' radio button.
- Enter the required value into the 'Server Port' input field.
- Confirm with {OK}.

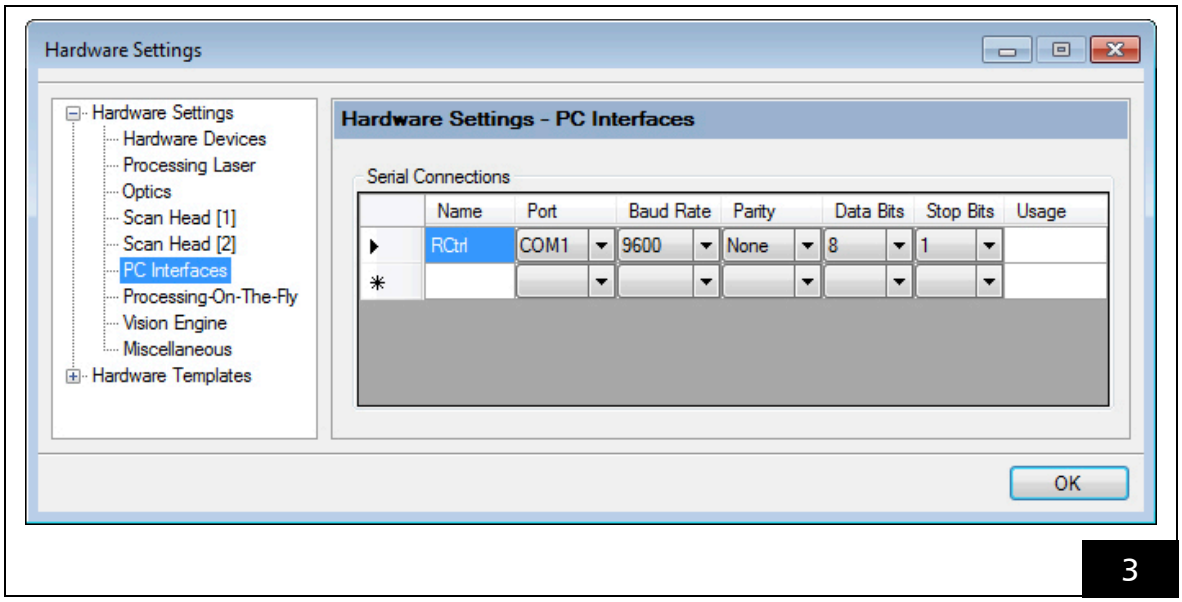
2.2 Serial Connection

The serial connection has following parameters:

Table 2

Parameter	Values	Remarks
Serial Port	COM 1, … (selectable)	Selection from the available COM-ports of the PC
Baud Rate	4800, 9600, 19200, 38400, 57600, 115200, 128000 or 256000 (selectable)	Default is “9600”
Parity	None, Even or Odd (selectable)	Default is “None”
Data Bits	5, 6, 7 or 8 (selectable)	Default is “8”
Stop Bits	1, 1.5 or 2 (selectable)	Default is “1”
Usage	“Remote Control”	Will be set automatically by the laserDESK program when selecting ‘Serial Connection’

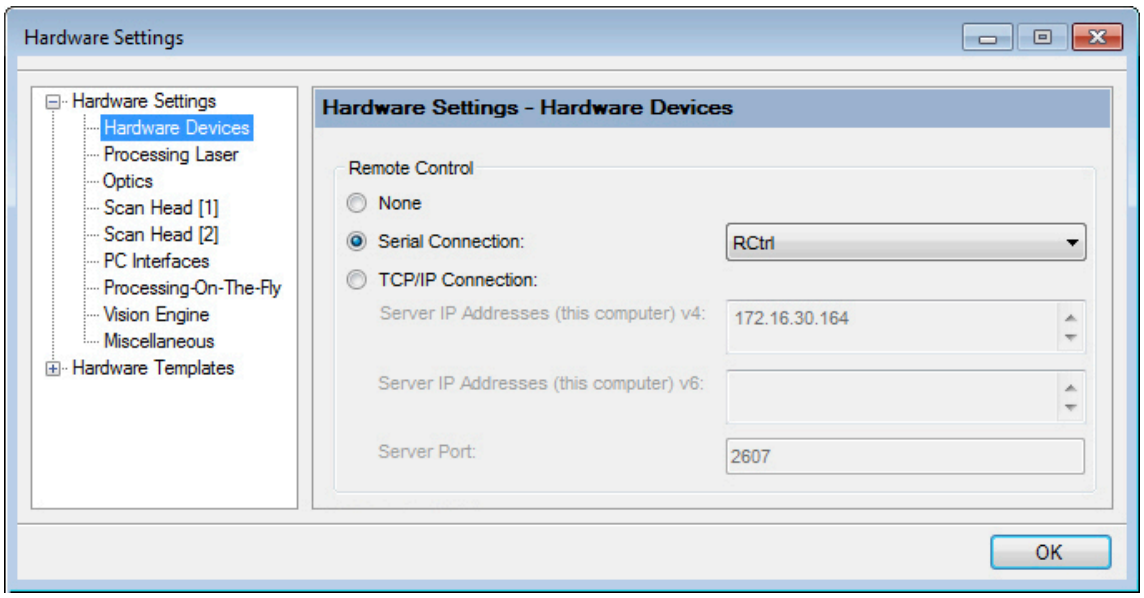
(1) The serial connection’s interface parameters must be defined in the ‘PC Interfaces’ section of the ‘Hardware Configuration’:



laserDESK ‘Hardware Settings’ Window – Serial Connections Setting

- Select ‘Open ▸ Hardware Configuration’ in the ‘File’ menu.
- Select the ‘PC Interfaces’ section.
- Select a new row.
- Enter a desired name in the column ‘Name’
- Click the {▾} buttons to select the required values in the selection fields (‘Port’ ... ‘Stop Bits’).
- Confirm with {OK}.

(2) The connection for the remote control must be defined in the 'Hardware Devices' section of the 'Hardware Configuration':



laserDESK 'Hardware Settings' Window – Remote Control Setting

- Select 'Open ▸ Hardware Configuration' in the 'File' menu.
- Select the 'Hardware Devices' section.
- Activate the 'Serial Connection' radio button.
- Click the {▾} button to select the required connection (set in the 'PC Interfaces' section before) in the corresponding selection field.
- Confirm with {OK}.

(After the connection has been defined, in the column 'Usage' in the list of the 'PC Interfaces' page (see figure 3 on page 7) "Remote Control" will be displayed.)

3 Telegram Syntax

Every message to and from laserDESK is sent as a block of data, called telegram. The telegram consists of a block frame (start and end character), the data and a checksum:

Start character	Data...

...Data	Checksum	End character

- The start character is: STX = 0x02
- The end character is: ETX = 0x03

The data part of each telegram defines a command together with the correspondent parameters. The maximum number of bytes in a telegram is 4095.

3.1 Checksum Calculation

The checksum is calculated as the sum of all byte values of the data bytes. Then the sum of the digits is calculated and the last digit is taken as the checksum. If no data was sent, the checksum will be 0.

Example of checksum calculation (in C#):

```
public static Byte GetChecksum(Byte[] arr, int len)
   {
      Byte        cs = 0;
      int         i, h = 0, z = 0;

      for (i = 0; i < len; i++)      // summarize every byte of array
      {
         h += arr[i];
      }

      do                             // calculate sum of digits
      {
         z += (h % 10);
         h /= 10;
      }
      while (h > 0);                 // every digit of byte

      return cs = (byte)(z % 10);    // take last digit
   }
```

3.2 DLE Correction

To distinguish between data bytes and block frame characters (STX, ETX), the following data bytes must be masked via the preceding mask character 0x10. This mask character is called DLE (data link escape) correction.

Table 3

Data Byte	Masked Byte in the Data Part of the Telegram
0x02 (STX)	0x10 0x02
0x03 (ETX)	0x10 0x03
0x10 (DLE)	0x10 0x10

If checksum = 0x02 or 0x03, it must be masked in the same manner (checksum is always < 0x10). Take care to appropriately mask or demask the telegrams when sending or receiving messages to or from laserDESK. The DLE correction has to be executed after checksum calculation and before sending as well as the first step after receiving.

3.3 Data Structure

The data part of the telegram is structured as follows:

Length of data byte array	Command number	

Parameter 1	...	Parameter n

The length value is of type integer (4 Bytes) and includes the command number and all parameters. It does not include the block frame characters, the checksum and their own length.

The command number is of type integer. It defines the command to laserDESK and thus the amount and the type(s) of the parameter(s) the command needs.

The data type of each parameter is exactly defined, thus no separators between parameters are needed. Only the strings have variable size. Strings must be null-terminated (C, C++ syntax) and all strings are represented by Unicode characters, thus the string end consists of 2 zero-Bytes (0x00 0x00).

For each character the byte order is little endian, thus the least significant byte must be sent first. E.g. “S” (Unicode 0053) must be sent as 0x53 0x00.

Unicode string example:

The byte representation of the string “SCANLAB” with the string end added is
0x53 0x00 0x43 0x00 0x41 0x00 0x4E 0x00 0x4C 0x00 0x41 0x00 0x42 0x00 0x00 0x00

All other parameters have their defined byte size:

- Integer = 4 Bytes
- Double = 8 Bytes

Command example ‘Status Query’:

Byte no.	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10	Byte 11
Meaning	STX	command length (integer)				command number (integer)				checksum	ETX
Byte value	0x02	0x04	0x00	0x00	0x00	0x04	0x00	0x00	0x00	0x08	0x03

3.4 Commands

The list shows all commands which are available for the client to control laserDESK:

Table 4

Command Number	Parameter	Command (Name)	since laserDESK Version
1	Client identification (string) = “1990”	Login of the remote control (client) at laserDESK	1.0.2
2	–	Logout	1.0.2
3	–	Query program version	1.0.2
4	–	Query program status	1.0.2
5	–	Remote mode on	1.0.2
6	–	Remote mode off	1.0.2
7*	1 = File name (string) 2 = Save Current Job (integer: 1 = Yes, 0 = No)	Open job file	1.0.2
8*	1 = File name (string) 2 = Overwrite existing file (integer: 1 = Yes, 0 = No)	Save job as...	1.0.2
9*	–	Switch laser on	1.0.2
10*	–	Switch laser off	1.0.2
11*	Parameter 1: Selection type (integer) Parameter 2 (dependent on selection type): 1: Name of the variant (string) 2: Bit pattern (string, “00001001”) 3: UID* (of the variant, string) 4: First variant (–) 5: Next variant (–) * for a description of “UID”, see chapter 5.1, "Unique Identifier (UID)"	Specify variant	1.0.2
12*	–	Start laser processing – job or specified variant	1.0.2
13*	–	Emergency stop of execution	1.0.2
14*	1 = UID (string) 2 = Text (string)	Define the text string of a text object (UID)	1.0.2
15*	1 = UID (string) 2 = Value of serial number (integer) 3 = Value selection (integer)	Define the start, actual or end value of a serial number	1.0.2, extended in 1.4.2.3
16*	1 = x-coordinate (double) (in [mm]) 2 = y-coordinate (double) (in [mm]) 3 = Rotation angle (double) (in [°]) optional: 4 = Scan head (integer)	Shifts and rotates the marking/coordinate system in the automatic mode Has a global effect on all variants or the job Is only valid during the automatic mode Will be reset when switching the automatic mode on Parameter 4 selects the scan head; possible values are 1, 2, and 3 for both scan heads used	1.0.2
17*	Target quantity (integer)	Define the target quantity of the specified variant or job	1.0.2
18*	–	Query of the number of executions of the specified variant or job	1.0.2
19*	1 = RTC interface (integer) 2 = Output value	Define the output lines of the RTC board	1.0.2

Table 4

Command Number	Parameter	Command (Name)	since laserDESK Version
20*	1 = x-coordinate (double) (in [mm]) 2 = y-coordinate (double) (in [mm]) 3 = Rotation angle (double) (in [°]) optional: 4 = Scan head (integer)	Shifts and rotates the coordinate system in manual mode (rotation center is origin) Parameter 4 selects the scan head; possible values are 1, 2, and 3 for both scan heads used	1.0.2
21*	–	Switch Automatic mode on	1.0.2
22*	–	Switch Automatic mode off	1.0.2
23*	–	Request for releasing the RTC board	1.6.0.1
24*	–	Return the control of the last acquired RTC board to laserDESK	1.6.0.1
25*	RTC interface (integer): 1 = Output pins of Extension 1 connector 4 = Input pins of Extension 1 connector 5 = Input pins of the Laser connector	Reads the signals from a connection to the RTC board	1.0.2
26*	1 = x-coordinate (double) (in [mm]) 2 = y-coordinate (double) (in [mm]) 3 = z-coordinate (double) (in [mm])	Defines the positions of the galvanometer scanners (x, y) and the dynamic focusing system's optics (z)	1.0.2
27*	1 = Configuration name (string)	Selects a defined hardware configuration and initializes the system with it	1.0.2
28*	1 = m_{11} (double) 2 = m_{12} (double) 3 = m_{21} (double) 4 = m_{22} (double) 5 = permanent (integer)	Transforms the total job additional to the hardware configuration coordinate shift and rotation	1.0.2
29*	Select Scan Head (integer)	Define the scan head(s) to be used. Possible values are 1, 2 and 3 for both heads used.	1.0.2
30*	0 = Detect reference positions for ASC 1 = Execute ASC (Automatic Self-Calibration) 2 = Reset and switch off ASC 4 = Check for ASC available on the system	Executes Automatic Self-Calibration (ASC) or checks, if ASC is available on the system	1.0.7.1
31*	Set marking parameter set for job 1= parameter set name (string)	Exchanges the marking parameter set of a job node by a predefined one (either in the job or in the library)	1.0.8.2
32*	Load/Replace vector graphics file parameter: 1 = Vector graphics file name (string) 2 = x-position (double) (in [mm]) 3 = y-position (double) (in [mm]) 4 = Width (size in x) (double) (in [mm]) 5 = Height (size in y) (double) (in [mm]) 6 = UID* of the vector graphics group to be replaced (string) 7 = Flags for import options (integer) (see chapter 4.32, optional) * for a description of "UID", see chapter 5.1, "Unique Identifier (UID)"	Imports a new vector graphics file or replaces an existing group	1.2.0.0, extended in 1.6.0.1
33	Get execution time	Returns the last execution time as a double (in [ms])	1.2.0.0
34	Get list of available parameter set names – no parameter	Returns a list of all available marking parameter sets	1.2.0.0

Table 4

Command Number	Parameter	Command (Name)	since laserDESK Version
35*	Hide/Show program GUI (integer) 1 = Hide 0 = Show	Hides or shows the GUI	1.2.0.1
36*	Set power scaling factor 1 = Power scaling factor (double) 2 (integer): 1 = permanent, 0 = temporary	Changes the scaling factor of the laser power	1.2.0.2
37*	Set vision system parameter Parameter 1 = UID of the vision system element (string) 2 = Job name of the *SCAN*align job (string) 3 = x-offset of the vision job 4 = y-offset of the vision job	Defines new parameters for the vision system element	1.2.0.3
38*	Set new marking parameter set for objects of a specific layer Parameter 1 = Name of marking parameter set (string) 2 = Layer name (string)	Defines another marking parameter set for all objects of a layer	1.2.1.1
39*	Get list of all defined layers in the job – no parameter	Returns a list of all layer names defined in the job	1.2.1.1
40*	Motor axis control Parameter 1 = Axis name (string) 2 = Type of action (integer) 3 = Position (double) 4 = Type of reference run (integer)	Controls the defined axis correspondent to the parameters	1.4.0.2
41	Motor axis status query Parameter 1 = Type of selection (integer) 2 = Selection parameter If parameter 1 = 1: Selection by axis name (string) If parameter 1 = 2: Selection by index number of Axis as defined in the hardware settings (zero based, integer)	Request of motor state	1.4.0.2
42*	1 = x-offset (double) (in [mm]) 2 = y-offset (double) (in [mm]) 3 = Rotation (double) (in [°])	Sets the coordinate transformation for POF applications	1.4.0.2
43*	Parameter 1 = x Offset [mm] (double) 2 = y Offset [mm] (double) 3 = z Offset [mm] (double) 4 = M11 (double) (matrix coefficient) 5 = M12 (double) (matrix coefficient) 6 = M21 (double) (matrix coefficient) 7 = M22 (double) (matrix coefficient)	Sets the coordinate transformation for a 3D system	1.4.1.2

Table 4

Command Number	Parameter	Command (Name)	since laserDESK Version
44*	Parameter for Scan Head 1: 1 = x Offset (double) (in [mm]) 2 = y Offset (double) (in [mm]) 3 = M11 (double) (matrix coefficient) 4 = M12 (double) (matrix coefficient) 5 = M21 (double) (matrix coefficient) 6 = M22 (double) (matrix coefficient) for Scan Head 2: 7 = x Offset (double) (in [mm]) 8 = y Offset (double) (in [mm]) 9 = M11 (double) (matrix coefficient) 10 = M12 (double) (matrix coefficient) 11 = M21 (double) (matrix coefficient) 12 = M22 (double) (matrix coefficient)	Sets individual coordinate transformations for both scan heads	1.4.1.2
45*	1 = Card type (integer) 0 = Serial number of card (integer)	Selects a new RTC board to be used	1.4.2.3
46*	1 = UID (string) 2 = Point distance (double) (in [mm]) 3 = Marking time of a point (integer) (in[µs]) 4 = Equal spacing (integer)	Converts vectors into a list of points	1.4.2.3
47*	1 = Power (double) (in [%]) 2 = Frequency (double) (in [kHz]) 3 = Pulse length (integer) (in [µs]) 4 = Laser-on time (double) (in [ms])	Manual activation of the laser beam	1.4.3.1
50*	Parameter 1 = Scan head (1 or 2) (integer) 2 = Defines the returned value (integer); possible values for parameter 2: 1 = Head state (as defined in the manual) 2 = Actual position (in [mm]) 3 = Actual output stage current (in [mA]) 4 = Temperature (in [°C])	Returns scan head specific parameters	1.6.5.0
51*	Parameter 1 = UID of the element (string) 2 = Defines the specific parameter (integer)* 3 = New value for the selected parameter (double) 4 = Optional: new parameter set name (string) * see Table 15 – Table 17 on page 33 ff	Set a specific parameter value	1.6.5.0

Table 4

Command Number	Parameter	Command (Name)	since laserDESK Version
53*	Up to 12 parameters – parameters 1 to 8 mandatory 1 = File name (string) 2 = Length [mm] (double) 3 = Width [mm] (double) 4 = Height [mm] (double) 5 = Base center X (double) 6 = Base center Y (double) 7 = Base center Z (double) 8 = Slices (double) 9 = Option flags (uint) 10 = Filling offset (double) 11 = Signal output (integer) 12 = Signal input (integer)	Import and slicing of an STL file	1.6.5.0
54*	Parameter 1 = UID (string) 2 = Marking Count (integer)	Sets the marking count of a graphic element with given UID	1.6.6.0
55*	Parameter ||: 1 = ID of the transformation (integer) 2 = x Offset [mm] (double) 3 = y Offset [mm] (double) 4 = Rotation [°] (double) :|| ||: ... :|| this block may be repeated several times	Sets the coordinate transformation for all transformation control nodes in the job which are referenced by the given ID	1.6.11.0
* This command is only accepted, when the remote mode is on (see chapter 4.5). It will only be executed if the RM_STATE_LST_CALC state flag is reset.			

Notes:

- All strings are null-terminated Unicode strings and a character has 2 bytes (see chapter 3.3).
- The commands 23 and 24 are reserved for the SSEI vision system integration.
- Every command will be answered by laserDESK (see chapter 3.5).

The table below shows examples how the different parameter types are redout:

Table 5

Type	Example	Bytes
integer	100	0x64 0x00 0x00 0x00
double	100.0	0x00 0x00 0x00 0x00 0x00 0x00 0x59 0x40
string	“100”	0x31 0x00 0x30 0x00 0x30 0x00 0x00 0x00

3.5 Data Structure of the Answer Returned by laserDESK

The answer has the same telegram structure as the command. The returned data stream always includes the command number, to which the answer is given and possibly further values, the command has requested.

The answer data is structured as follows:

Length of data byte array	Command number	

Returned value(s)

The answer is sent immediately. From case to case status queries have to be used to check whether the requested action has been executed. This is especially necessary for longer actions like a marking process.

The table below shows the answers (from server to client) to the correspondent commands. If no query command was sent, a return value of “1” generally means that the command is accepted and executed. A value of “0” means, that the command is refused or the execution was not successful.

If an unknown command is received or the command syntax is wrong, “0” will be returned.

Table 6

Command Number	Answer	Meaning
1	Integer	1 = Login accepted 0 = Login refused
2	Integer	1 = Logout done
3	Unicode-string “a.b.c.d”	Program version of the laserDESK-assembly
4	Bit pattern (unsigned integer)	Program state as Bit pattern (see chapter 4.4)
5	Integer	1 = Remote mode on
6	Integer	1 = Remote mode off
7*	Integer	1 = Job file opened (1) 0 = Job file can’t be opened
8*	Integer	1 = Job is saved 0 = Job can’t be saved
9*	Integer	1 = Laser switched on 0 = Laser can’t be switched on
10*	Integer	1 = Laser switched off 0 = Laser can’t be switched off right now
11*	Integer	1 = Variant selection ok 0 = Variant could not be selected
12*	Integer	1 = Start laser processing 0 = Not possible to start laser processing
13*	Integer	Emergency stop of processing
14*	Parameter 1: integer Parameter 2: UID (string)	1 = String of text object is set 0 = String of text can’t be set
15*	Parameter 1: integer Parameter 2: UID (string)	1 = Value of serial number is set 0 = Value of serial number can’t be set
16*	Integer	1 = Coordinate transformation is applied 0 = Coordinate transformation can’t be set
17*	Integer	1 = Target quantity is set 0 = Target quantity can’t be set
18*	Number of executions (integer)	Number of executions of the selected variant or job
19*	Integer	1 = Output signal is set 0 = Output signal can’t be set
20*	Integer	1 = Coordinate shift and rotation is applied 0 = Coordinate shift and rotation can’t be set

Table 6

Command Number	Answer	Meaning
21*	Integer	1 = Switch automatic mode on 0 = Not possible to switch automatic mode on
22*	Integer	1 = Switch automatic mode off 0 = Not possible to switch automatic mode off
23*	Integer	1 = Release RTC 0 = Not possible to release RTC
24*	Integer	1 = Acquire RTC control 0 = Not possible to acquire RTC
25*	Return value: Input pins (uint)	Return value is the state of the input pins of the selected port
26*	Integer	1 = Position is set 0 = Position can't be set
27*	Integer	1 = Configuration has been changed 0 = Configuration can't be changed (name can't be found, in automatic mode ...)
28*	Integer	1 = Transformation is applied 0 = Not possible to apply matrix transformation
29*	Integer	1 = Set new active scan head(s) 0 = Not possible to change scan head
30*	Integer	1 = Success / Automatic Self-Calibration available 0 = Error / No Automatic Self-Calibration present
31*	Integer	1 = New parameter set is applied (2) 0 = Parameter set can't be changed
32*	Integer	1 = File is imported 0 = Command is not accepted
33	Double	Execution time (in [ms]) of last execution If value is 0.0 or negative an error has occurred
34	String	Comma separated list of parameter set names
35*	Integer	1 = Command accepted 0 = Command not accepted
36*	Integer	1 = Scaling factor adapted 0 = Command refused
37*	Integer	1 = Vision system parameters are set 0 = Command refused
38*	Integer	1 = Parameter set applied 0 = Command refused
39*	String	Comma separated list of layer names
40*, **	String	1 = Command accepted 0 = Command refused or syntax error
41**	Integer	0 or positive: Axis state –1 = Command refused, or syntax error
42*	Integer	1 = Command successful 0 = Command failed
43*	Integer	1 = Command successful 0 = Command failed
44*	Integer	1 = Command successful 0 = Command failed
45*	Integer	1 = Command accepted 0 = Command failed

Table 6

Command Number	Answer	Meaning
46*	Integer	1 = Command accepted 0 = Command failed
47*	Integer	1 = Command executed 0 = Command failed
50*	Integer Double Double	>0 = return values are valid, value is the head number 0 = Command failed Return value for Galvo 1, dependent on request Return value for Galvo 2, dependent on request
51*	Integer	1 = Success, parameter changed >0 = Command failed* * see Table 18 with return values on page 35
53*	Integer	1 = Command accepted and import started 0 = Command not accepted or syntax error
54*	Parameter 1: integer Parameter 2: UID (string)	Marking count of element (UID) set (Parameter 1 = 1) or could not be set (0)
55*	Integer	1 = Command successful 0 = Command failed

* This command is only accepted, when the remote mode is on (see chapter 4.5).
** This command is a non-blocking command. That means, the answer is returned immediately and the command is executed afterwards. The answer only signals that the command is received and will be executed. The execution process has to be checked by status queries.
(1) Changed 1/9/2013, Version 1.0.6 and higher: Answer returned immediately, process may fail!
(2) Available for laserDESK version 1.0.8.1 or higher

4 Command Description

4.1 Command #1 – Login

(from laserDESK version 1.0.8.1 or higher)

This command serves to log in the client at the laserDESK server. Only one client can be logged in at the same time. Every further login command will be refused until the client has logged out again. The login parameter is the string "1990" and serves to identify a legal login request.

4.2 Command #2 – Logout

This command serves to log out the client from laserDESK. Now the remote control is available for another client. No command parameter is needed.

4.3 Command #3 – Query for Program Version

No parameter is needed. The return value is a Unicode-string "a.b.c.d" with 2 zero bytes as the string delimiter. "a.b.c.d" represents the program version of laserDESK.

4.4 Command #4 – Query for Program State

No parameter is needed. The return value is an unsigned integer used as a bit field. Every bit represents a different state of laserDESK (see table below).

Table 7

Bit#	Bit	Meaning (if Bit is set)
0x00000000	RM_STATE_NO_INIT	Reset value
0x00000001	RM_STATE_WND_OPEN	Program runs
0x00000002	RM_STATE_RTC_INIT	RTC board initialized
0x00000004	RM_STATE_LAS_INIT	Laser system is initialized
0x00000008	RM_STATE_MOT_INIT	All external controls are initialized
0x0000000F	RM_STATE_ALL_INIT	All hardware components are initialized (sum of Bit 0-3)
0x00000010	RM_STATE_READY	RTC command thread runs (equals output pin 12)
0x00000020	RM_STATE_AUTOMODE	Automatic mode on (equals output pin 13)
0x00000040	RM_STATE_LST_EXEC	List is in execution (equals output pin 14)
0x00000080	RM_STATE_LST_EXE_ERR	Execution error (equals output pin 15)
0x00000100	RM_STATE_RM_MODE	Remote control mode on
0x00000200	RM_STATE_JOB_LOAD	Job is loaded, ready for execution
0x00000400	RM_STATE_LST_CALC	Execution list calculation in progress
0x00000800	RM_STATE_CMD_ERR	Command execution error occurred
0x00001000	RM_STATE_LAS_ERR	Laser system is in error state
0x00002000	RM_STATE_LAS_ON	Laser is switched on
0x00004000	RM_STATE_DEV_ERR	Error detected on a hardware device
0x00008000	RM_STATE_HEAD_OK	Scan head state is OK
0x00010000	RM_STATE_EXEC_DONE	Set, when start has been executed, reset by start command
0x00020000	RM_STATE_PILOT_MODE	Set, when pilot laser mode is active
0x00040000	RM_STATE_RTC_RELEASED	Set, if no RTC board is acquired
0x00080000	RM_STATE_JOB_ABORTED	Set if the last job execution was aborted
0x00100000	RM_STATE_SWITCH_AUTOMODE	Switching Automatic mode on in progress
0x00200000	RM_STATE_AXIS_EXECUTION	Set, if an axis movement is in execution
0x00400000	RM_STATE_DEV_RUNNING	Set, when an external device is running during a job execution or when a reference run for all axes is executed.

- The bits 0 to 3 show the initialization state of the connected hardware.
- The bits 4 to 11 show the execution state of laserDESK, where the bits 4 to 7 reflect the output pins 12–15 of the RTC Extension 1 connector.
  If bit 10 RM_STATE_LST_CALC is set, all commands which require the remote mode 'on' are refused.
  The RM_STATE_CMD_ERR flag will be set, if a remote command can't be executed without an error, e.g. a job cannot be loaded, because it doesn't exist. The flag will be reset after this status query.
- The bits 12 and 13 show the state of the connected hardware (bit 13 is always "1", if the laser is not controlled by laserDESK).
- Bit 14 shows the state of a connected *SCAN*align tool and/or motor control device.
  The RM_STATE_DEV_ERR flag will be set, if an error occurs during execution. The flag will be reset by the next execution start.
- Bit 15 shows the state of the scan head. This state is evaluated by the RTC command get_head_status (Bit 7 – power OK). Under certain configurations, this bit might not be detected correctly. Please refer to the scan head manual.
- Bit 16 is for checking, whether a start has been executed. The bit is set until a start request command (see "Command #12 – Start Laser Processing") has been received. Then the bit is reset and stays reset until the processing has been successfully finished. The bit is reset before the answer of the start command is sent to the client program. Thus, if the state query after a marking start returns this bit as set, the execution is already finished. This is important, if the processing time is in the range of the *Windows* cycle time. Then one may not receive a bit 16 not set at all.
- Bit 17 is for checking, whether the pilot laser mode is activated or deactivated. If set, switching into the remote mode isn't possible.
- If bit 18 is set, no RTC is acquired.
- The bit 19 shows the state of an aborted job. It is set, if the last job execution was aborted by following conditions:
  (1) User has pressed the {STOP} button
  (2) External stop signal has aborted the execution
  (3) Vision job node has cancelled the execution
  (4) Serial control node has cancelled the execution
  This signal is only valid after a job execution (RM_STATE_EXEC_DONE is set) and is reset by the next start or switching to automatic mode.
- Bit 20 is set during the process of switching on the automatic mode to check the proceeding of "Command #21 – Switch Automatic Mode On", reset when finished. If the execution was not successful, the RM_STATE_CMD_ERR flag will be set (see above).

Note:

The status query is allowed during the execution of the remote "Command #7 – Open Job File" and "Command #21 – Switch Automatic Mode On" to check the state of these actions by the RM_STATE_JOB_LOAD flag and the RM_STATE_SWITCH_AUTOMODE flag respectively.

- Bit 21 is set if an axis movement is in execution, either a reference run or a normal movement.
- Bit 22 is set when an external device is running (e.g. serial communication, motor axis movement, vision system, ...) during a job execution or when a reference run for all axes is executed (see second parameter 5 in "Command #40 – Motor Axis Control").

4.5 Command #5 – Remote Mode On

This command switches to remote mode. The command has no parameters. In remote mode, the GUI is blocked and manual commanding isn't possible. The client has full access to laserDESK. Only in this mode, most of the commands are available (see chapter 3.4, "Commands"). A blocking dialog will appear on the display. For a user with administrator or supervisor access authorization it is possible to finish the remote mode locally.

4.6 Command #6 – Remote Mode Off

This command switches off the remote mode. Only a few commands are allowed in that state, especially the status query. The blocking dialog vanishes, when set in the GUI settings (to open via the 'Edit\Options' menu; then select the 'General' section and activate the 'Close dialog after remote control is off' check box in the field 'Remote Control Mode'.)

4.7 Command #7 – Open Job File

The command opens an existing job. Parameter 1 (string) has to be the full qualified path and name of the file. Parameter 2 (integer) defines, whether the already opened job should be stored (= 1) or not (= 0).
The answer, whether the command is received, will be returned immediately.

Notes:

- The DLL "SLLDRemoteControl.dll" was adapted to the modified remote control. The new function "OpenJobNoWait()", which executes the action asynchronously, can only be used with the laserDESK Software Release 1.0.6 (or higher).
  The function "OpenJob()" can be used with all laserDESK versions. In this case, the answer is send only after the action's completion.
- While this command is executed only status queries are allowed.
  All other remote commands are refused.

4.8 Command #8 – Save Job As ...

This command saves the actual job with a new name. Parameter 1 (string) is the new file name (with full qualified path) and parameter 2 (integer) defines, whether an already existing file should be overwritten (= 1) or not (= 0).

4.9 Command #9 – Switch Laser On

This command switches a specified laser on. It has no effect, if a 'General Type' laser is selected. The command has no parameters.

4.10 Command #10 – Switch Laser Off

This command switches off a specified laser. It has no effect, if a 'General Type' laser is selected. The command has no parameters.

4.11 Command #11 – Select Variant

This command selects a variant. For a job which includes variants this command must be send prior to any execution. If no variants are defined in the job, "0" will be returned. In that case it is not necessary to select the job node (UID = "G#1"). It is preselected.

All following commands which are related to a specified variant, will be linked to the variant defined by this command. These are "Command #12 – Start Laser Processing", "Command #17 – Define the Target Quantity", "Command #18 – Query the Number of Executions", and "Command #20 – Set Coordinate Transformation in Manual Mode".

The command has 2 parameters. Parameter 1 (integer) defines the kind of selection and the type of parameter 2 (see table below).

Table 8

Parameter 1 (Integer)	Meaning	Parameter 2 (Type)
1	Name of the variant	String
2	Bit pattern (e.g. "00101001")	String
3	UID	String
4	Selects first variant	Empty string (" ")
5	Selects next variant	Empty string (" ")

If parameter 2 equals the property of a variant in the job, this variant is selected.

The variants are always handled internally by their bit pattern. Even if you select the variant by UID or name, you have to define a unique bit pattern in the job (also refer to the "Graphic Parameters – Job" topic in the laserDESK Help).

4.12 Command #12 – Start Laser Processing

This command starts the laser processing of the selected variant. Prior to this command a valid variant has to be selected (see "Command #11 – Select Variant"). If no variant is present, the job is executed (no selection needed). The command starts the marking independent of the actual mode (see "Command #21 – Switch Automatic Mode On" and "Command #22 – Switch Automatic Mode Off").

The command has no parameter.

4.13 Command #13 – Emergency Stop

This command stops the execution of the processing immediately. It will not finish the automatic mode, if it is switched on. The command has no parameter.

4.14 Command #14 – Set Text String

This command sets the string of a text object (font text, vector text, barcode) inside the opened job. To allow the changing of the text by this remote command, the property 'Variable' of the text object has to be set to "True". The actual string of the text object at processing start will be used. (For further 'Variable Text' details, see chapter 5.2.)

During calculation (state bit 10 is set), this command will be refused.

The command has 2 parameters. The first (string) defines the UID of the text object, the second parameter (string) defines the new string.

4.15 Command #15 – Define the Serial Number Start, Actual or End Value

This command sets the start, actual or end value of a serial number. The start value of the serial number is used, when the job switches to automatic mode and the serial number has to be reset ('Reset Start Value' = true). Thus, this command is only allowed in manual mode (state bit 5 = "0").

The command can have 2 or 3 parameters:

(1) The first parameter (string) defines the UID of the serial number.

(2) The second parameter (integer) defines the new value dependent on the third parameter.

(3) Extension (since version 1.4.2.4):
The command accepts an optional third parameter 'Value selection' (integer) to select whether the start value (Selection = 1) the actual value (Selection = 2) or the end value (Selection = 3) is changed.
If the Selection parameter is omitted, the start value will be changed (downward compatibility).

If the actual value is outside the defined range, it will be set to the limit value (no error is returned).

4.16 Command #16 – Set Coordinate Transformation in Automatic Mode

This command allows to rotate and shift the marking in the automatic mode temporarily. This command defines a transformation for all executions and is not specific to a selected variant.

The command will be refused in manual mode (see "Command #22 – Switch Automatic Mode Off").

First the rotation is executed, then the shifting. The rotation axis is the origin, because one needs a well defined point in laserDESK, which is also known in the controlling master system.

The transformation is additional to the settings in the 'Hardware Configuration'.

The command has 3 parameters (doubles) and an optional 4th parameter (integer). The first defines the shift in x direction, the second the shift in y direction (both in [mm]). The third parameter defines the rotation (in [°], positive values rotate counter-clockwise).

The 4th parameter is optional and defines the scan head where the transformation is applied. The values 1 or 2 define the respective scan head, value 3 both scan heads and corresponds to the old command syntax where the last parameter is omitted. If the activation of the second scan head (Use Second Scan Head) in the 'Hardware Configuration' is not selected, the 4th parameter is omitted.

The command will be applied for the next markings (and not for the current proceeded marking) as long it will not be changed by an new command. Switching the automatic mode off, the transformation defined by this command will be canceled.

If transformation control nodes are used in the job, the x, y-offset and the rotation will be reset at the end of each job execution. In that case, this part of the transformation defined here is only valid for the next execution and only until the first transformation control node is executed.
You should consider to use the "Command #55 – Set Transformation for Control Node" instead.

4.17 Command #17 – Define the Target Quantity

This command sets the target quantity of the selected variant (see "Command #11 – Select Variant") or of the job. Because the target quantity is used by the automatic mode, the changing is only allowed in manual mode (state bit 5 = "0").

The command has one integer parameter, which defines the target quantity.

4.18 Command #18 – Query the Number of Executions

This command asks for the number of executions of the selected variant (see "Command #11 – Select Variant") or of the job. The command has no parameters. The number of executions is returned as an integer.

4.19 Command #19 – Set Output Signals of the RTC

This command allows to set the output lines of the RTC board. In principle, all lines could be set. But several lines are predefined and used by laserDESK. Thus the pins 8–15 of the Extension 1 connector are excluded at all. The other pins may not be used too, if they are handled inside laserDESK. Especially if a specified laser type is used, several output lines are reserved for this purpose. In that case, refer to the laserDESK Help or contact SCANLAB about usable lines.

The command has 2 parameters. Parameter 1 (integer) defines the output port:

Table 9

Parameter 1	Meaning
1	Extension 1 connector (output lines 0–7)
2	Extension 2 connector (output lines 0–7)
3	Laser connector (output lines 0 and 1)

In parameter 2 (string) every character defines an output line:

Table 10

Parameter 2	Meaning
0	Pin is used and reset to this value
1	Pin is used and set to this value
X	Pin is masked and remains unchanged

The first character of the string corresponds to the digital output line 7.

Example:

"01XXX0X1" defines output lines 7–0 of the Extension 1 connector.

Char 8 = 1	Char 7 = X	Char 6 = 0	Char 5 = X
Dig Out 0	Dig Out 1	Dig Out 2	Dig Out 3
Ext1, Pin1	Ext1, Pin3	Ext1, Pin5	Ext1, Pin7

Char 4 = X	Char 3= X	Char 2= 1	Char 1= 0
Dig Out 4	Dig Out 5	Dig Out 6	Dig Out 7
Ext1, Pin9	Ext1, Pin11	Ext1, Pin13	Ext1, Pin15

Digital out 0 and 6 are set, digital out 2 and 7 are reset and all other lines remain unchanged.

4.20 Command #20 – Set Coordinate Transformation in Manual Mode

This command allows to rotate and shift the marking in the manual mode. If variants are defined inside the job and a variant is selected, this command sets the transformation of the selected variant (see "Command #11 – Select Variant"), else it sets the transformation of the job node.

The command will be refused in automatic mode (see "Command #21 – Switch Automatic Mode On"), because there all calculations are already done and a recalculation can't be processed. In that case use "Command #16 – Set Coordinate Transformation in Automatic Mode".

The command has 3 parameters (doubles) and an optional 4th parameter (integer). The first defines the shift in x direction, the second the shift in y direction (both in [mm]). The third parameter defines the rotation (in [°], positive values rotate counter-clockwise).

First the rotation is executed, then the shifting. The rotation axis is the origin, because one needs a well defined point in laserDESK, which is also known in the controlling master system. A relative reference point (e.g. the center of a variant) is not suitable, because the master system doesn't know this point or needs to get this information for every defined variant.

The transformation is additional to the settings in the 'Hardware Configuration'. If the optional 4th parameter is not used, the command sets the object parameters 'Mark Translation' and 'Mark Rotation' (corresponds to previous versions of the remote interface). It replaces the previously defined transformation parameters of the variant or job node. If the job is saved, the transformation defined by this command remains valid. It will be used when the job is reloaded and also in automatic mode.

The optional 4th parameter is for selecting the scan head the transformation will be applied to. In that case the object parameters are **not** set, because object parameters can't be scan head selective! Instead temporary program settings are used and applied.
Parameter value 1 or 2 selects the appropriate scan head, value 3 applies the transformation to both scan heads. In that case, for each scan head its additional individual hardware configuration setting is used.
This transformation remains valid until the command is executed again (new settings), the automatic mode is switched on (reset of this transformation) or the remote mode is switched off (defined transformation will no more be applied, but the settings remain valid).

4.21 Command #21 – Switch Automatic Mode On

This command switches laserDESK in automatic mode. There all static objects are precalculated and downloaded to the RTC board. The system reacts on external start signals (laser connector pin 3, see RTC5 or RTC6 manual) without any delay, because the start signal is directly sent to the RTC5 board without any interference of the operating system of the PC. laserDESK detects the start not until the RTC processor has already started the list execution. The variable part of the processing is calculated during the execution of the static part. Thus it is an advantage to define the static objects in a job or variant first and the variable objects behind them. This will increase the time for the calculation of the variable objects. (For further details of the automatic mode, see chapter 5.3.)

In automatic mode there exist 2 methods to start a marking:

(1) Using an external start signal to the RTC start input (Laser connector pin 3).

(2) Using the remote "Command #12 – Start Laser Processing".

Method 1 needs an additional IO-connection which the master program has to switch, but it is faster than method 2.

The command has no parameters.

The answer, whether the command is received, will be returned immediately.

Notes:

- The DLL "SLLDRemoteControl.dll" was adapted to the modified remote control. The new function "SwitchAutomaticModeOnNoWait()", which executes the action asynchronously, can only be used with the laserDESK Software Release 1.0.6 (or higher).
  The function "SwitchAutomaticMode()" can be used with all laserDESK versions. In this case, the answer is send only after the action's completion.
- While this command is executed only status queries are allowed.
  All other remote commands are refused.

4.22 Command #22 – Switch Automatic Mode Off

This command switches laserDESK from automatic mode to manual mode. There, the external start signals are disabled. Only manual starts via the remote "Command #12 – Start Laser Processing" are possible. In manual mode, "Command #20 – Set Coordinate Transformation in Manual Mode" can be executed. In that case a new calculation of all objects is necessary. Therefore the automatic mode would have no advantages regarding the execution time.

4.23 Command #23 – Release RTC

The RTC board will be released. No further control is possible. All hardware access is stopped, because the IOs of the RTC board are not available anymore.
This command sets the RM_STATE_RTC_RELEASED state flag. The state flag is reset when the connection shuts down, the client logs out or a new RTC board will be acquired.

4.24 Command #24 – Acquire RTC

The control of the last acquired RTC board is returned to laserDESK. The RTC board will be initialized again, because all settings could be changed.
This command resets the RM_STATE_RTC_RELEASED state flag.

4.25 Command #25 – Read the RTC Input Lines

The command can be used to get the state of the input lines of the RTC board. The returned answer contains this state. The command parameter (integer) defines the connection to be read:

Table 10

Parameter 1	Meaning
1	Extension 1 connector (output lines 0–15)
4	Extension 1 connector (input lines 0–15)
5	Laser connector (input lines 0 and 1)

The return value is an unsigned integer, where the individual bits represent the state of the input lines of the selected connection. Bit 0 represents line 0 of the connection.

Example:

Returned input value is 13339 = 0x341B = 0011 0100 0001 1011.
That means that the input lines 0, 1, 3, 4, 10, 12 and 13 are set.
(Refer to the RTC5 or RTC6 manual, to see which line corresponds to which connector pin.)

Note:

The string representation is reverse to the bit representation of the lines.

- String representation: Lowest line comes first (= first character, left side)
- Bit representation: Highest line comes first (= left digit).

4.26 Command #26 – Set Galvanometer Scanners to a specific Position

The command positions the galvanometer scanners and – if available – the dynamic focusing system's optics to specific positions. This command is e.g. suited to adjust the camera position.

This command is only executable, if no marking is executed (status: RM_STATE_EXEC_DONE = 1).

The command has 3 parameters (doubles), the first defines position regarding the x direction, the second in y direction and the third in z direction (all in [mm]).

4.27 Command #27 – Change the Hardware Configuration

The command replaces the actual hardware configuration. The new configuration is defined by parameter 1 (configuration name). The configuration must be defined manually prior to this usage.

This command acts like the manual selection of a configuration in the hardware settings dialog box. This change is temporarily valid and discarded when the program closes.

4.28 Command #28 – Execute global matrix transformation

The command uses the parameters m_{11}, m_{12}, m_{21} and m_{22} as the matrix elements of the transformation to be applied. The parameter *permanent* defines the time of duration. Allowed values are:

permanent = 0:
In manual mode the transformation is only applied for the next marking. If a marking process is executed, the command will be refused. In automatic mode it is applied until the automatic mode is switched off.

permanent = 1:
The transformation is applied until this function is used again or the hardware configuration is changed (new RTC initialization).

If this command is sent during an execution in automatic mode, the transformation is applied not until the next execution.

Note:

This transformation is additional to the transformation defined by "Command #16 – Set Coordinate Transformation in Automatic Mode" or "Command #20 – Set Coordinate Transformation in Manual Mode".
This command enables scaling, skewing and rotation. An offset has to be executed by the commands number 16 or number 20.

4.29 Command #29 – Set active Scan Head

This command corresponds to the 'Head' selection field in the Laser Control window. It allows to change the active head. Allowed integer parameter values are 1 for the first scan head, 2 for the second scan head and 3 for using both scan heads.

If the RTC board's 'Second Scan Head Control' option is not activated or if the second scan head is not enabled in the 'Hardware Configuration', a changing command is refused.

This command can only be executed in manual mode and when no job is running.

4.30 Command #30 – Execute Automatic Self-Calibration

This command lets you execute the Automatic Self-Calibration (ASC). The ASC will be executed for the selected scan head (by command *SetScanHead*). If both scan heads are selected (see Command #29 above), only the first scan head will execute the ASC.

The execution can last a few seconds and thus the answer returned from laserDESK (see also chapter "Automatic Self-Calibration" in the RTC5 or RTC6 manual).

4.31 Command #31 – Set Marking Parameter Set for Job

This command replaces the marking parameter set of the job node. All objects inside the job, which inherit the marking parameter set from the job node, will use the new marking parameters. The parameter set may be defined either in the job itself or in the marking library. This exchange will be temporary unless the job will be saved.

The command parameter is the name of the parameter set (string).

4.32 Command #32 – Load/Replace Vector Graphics

This command replaces a (vector graphics import) group in a job by another one or inserts a new import group. It is possible to define the size and position of the graphics. All other parameters (marking and hatching parameters) will be transferred from the replaced group or inherited from the job node when inserted.

Using the remote "Command #31 – Set Marking Parameter Set for Job" enables to exchange the job marking parameter and there-fore the vector graphics marking parameters, too.

The command needs following parameters:

(1) Vector graphics file name, full path (string)

(2) x position (double)

(3) y position (double)

(4) Width, size in x (double)

(5) Height, size in y (double)

(6) ID of the vector graphics group to be replaced (string)

(7) Flags parameter for import options, may be omitted (integer)

If 'width' and 'height' are positive values, the graphics may be distorted. If one value is zero or negative, it will be adapted proportional to the other. If both values are zero or negative, the file values will be used with the unit [mm].

If the UID is an empty string, the graphic file will be imported as a new group getting the job parameters as individual (not inherited) parameter sets. Therefore they will not be changed by "Command #31 – Set Marking Parameter Set for Job" later on. Else the file will replace the existing group, take over the group parameters (marking and hatching) and keep the inheritance.

If the UID is not empty but not defined inside the job, the command execution will cancelled.

Possible settings which are present in the file import dialog will be available by the flags parameter. The flags are:

Table 11

Flag	Value	Meaning
enNone	0x00	All flags are cleared
enConnect	0x01	Lines are connected. The connection tolerance is: default graphics resolution / 10 (can't be defined by this command)
enArea	0x02	Closed curves are always imported as 'Areas' and can be filled
enCollectPath	0x04	All objects are collected into a graphics paths group (import is faster) with the same parameter sets
enSplineAsBezier	0x08	Only dxf: SPLINES entities are imported as Bezier curves
enValues1to1	0x10	Width, height and position parameters are ignored. The file values are used as absolute coordinates in [mm]
enScaleFactors*	0x20	The width and height parameters will be used as scale factors in x and y, respectively. If the flag 'enValue1to1' is set, this flag will be ignored.
enKeepPosition*	0x40	If this flag is set, the position values of the command will be ignored and the (scaled) file values are valid. If the flag 'enValue1to1' is set, this flag is internally set.
* since laserDESK version 1.6.0.1		

If the flags parameter is omitted, the default value is:
– enArea is set;
– all other flags are cleared.

If the flag 'enScaleFactors' is set, first the vector file will be imported using the unit information which is included in the file.

Hint: The default unit for plt-files is [0,025 mm] for all other file types [1 mm]. This unit is used for the scaled import if no unit information is present. To import a plt-file which has [mm] units you can either use the 'enValues1to1' flag or use scaled import with scaling factor 40. Only the latter allows to set a different origin.

After the import the whole graphics will be scaled by the given factors. This is different to the 'enValues1to1' flag where no unit information is used but the file values are taken as [mm] values.

The 'enScaleFactors'-flag allows to import a vector graphic with known measure and unknown size to be imported with the correct size. Negative scale factors are ignored.

If the 'enValues1to1' flag is set, 'enScaleFactors' will be ignored.

Symbols are always imported and actualized.

The answer is returned immediately to show the execution start of this command. The state and success of this command execution can be detected by status queries using the RM_STATE_JOB_LOAD flag. While importing the vector graphics this flag is reset. After the command execution, this flag is set again. If an error occurs or the command is cancelled, the RM_STATE_CMD_ERR error flag will be set together with RM_STATE_JOB_LOAD. The error flag will be cleared again by a status query.

While this command is executed only status queries are allowed. All other remote commands are refused.

Hint: To define special parameter sets for the imported graphic, you may define an empty group with individual parameter sets. Then replace this group by the vector graphic which will use these parameter sets.

4.33 Command #33 – Get Execution Time

This command returns the execution time of the last marking in [ms]. The return value is a double. The command has no parameters.

4.34 Command #34 – Get Marking Parameter Sets

This command returns the names of all available marking parameter sets as a list of comma separated names. The sets can either be in the job or the program library. The names are separated by a comma and a space character.

Every name can be used to select the actual job parameters (see "Command #31 – Set Marking Parameter Set for Job").

4.35 Command #35 – Hide or Show GUI of laserDESK

This command forces laserDESK to hide or show the GUI (parameter = 1: hide; parameter = 0: show). When hiding, an icon in the taskbar is displayed.

4.36 Command #36 – Set Power Scaling Factor

This command changes the scaling factor of the laser power.

If the scaling of the laser power should be **permanent** (parameter 2 = 1), the laser calibration factor of power in the hardware configuration is changed. This setting will also be written in the hardware configuration file (active hardware configuration) and therefore be used even after a program restart.

If the laser power should only be scaled **temporarily** (parameter 2 = 0), it will be reset to the hardware configuration value when the RTC is initialized again (on program start or configuration change) or when the remote mode is finished (to be consistent with the GUI).

The permanent setting is an absolute value, the temporary setting is a relative scaling factor. During execution both factors are multiplied. Therefore, it is possible to define a temporary factor larger than 1.0 to increase the power level if the permanent factor is below 1.0.

For the execution it is checked that the total scaling factor is between 1.0 and 0.0.

4.37 Command #37 – Set Vision System Parameter

This command defines the parameters *Camera Job* and *Translation* of the vision system element with the defined UID. The vision job has to exist in *SCAN*align and is not checked by this command.

The offset parameters are not checked, too. It belongs to the user to verify the correct execution. Not adapted offset parameters may lead to scan head positions outside of the working area.

Because this command changes individual execution parameters it requires a new calculation of the job. Therefore it is only allowed in manual mode when no execution takes place.

This command can only be executed by laserDESK version 1.2.0.3 and higher.

4.38 Command #38 – Set Marking Parameter Set for all Objects of a specific Layer

This command defines a marking parameter set which will be applied for all objects in the specified layer.

The marking parameter set has to be already defined either in the job or the program library. If applicable the parameter set is copied into the job library.

The layer has to be defined in the job. The command evaluates all objects of the specified layer and assigns the marking parameter set to them.

Because this command changes individual execution parameters it requires a new calculation of the job. Therefore it is only allowed in manual mode when no execution takes place.

This command can only be executed by laserDESK version 1.2.0.6 and higher.

4.39 Command #39 – Get Layer Names of Job

This command returns the names of all defined layers inside the job as a list of comma separated names. The names are separated by a comma and a space character.

Every name can be used for command #38 (see above) to select the layer which objects should get another marking parameter set.

4.40 Command #40 – Motor Axis Control

This command controls a defined axis. The first parameter represents the axis name defined in the hardware configuration. If the axis doesn't exist the command will not be executed and the remote state RM_STATE_CMD_ERR will be set.
To do a reference run for all enabled axes (same as pressing the {Reference Run} button), set this parameter to 'ALLAXIS' (or use an empty string).

The second parameter defines the action which the axis controller should do. Possible actions are:

Table 12

Integer Number	Kind of Action
0	Initialize the axis
1	Reference run; the 4th parameter is needed
2	Absolute movement; the 3rd parameter is needed
3	Relative movement; the 3rd parameter is needed
4	Emergency stop
5	Reference run for all selected axes

Dependent on the selected action further parameters are evaluated. For action number 2 and 3 a position as a double is needed and has to be provided as a 3rd parameter. The unit is defined by the calibration factor of the axis in the hardware configuration.

For action number 1 the position parameter is not used. But the 4th parameter defines the kind of reference run. It depends on the specific axis controller which kinds of reference runs are available. Following reference run values are possible:

Table 13

Integer Number	Type of Reference Run
0	Actual position is new reference
1	Running back to lower end switch
2	Running forth to upper end switch
3	Running back to lower end switch and then move to reference point
4	Running forth to upper end switch and then move to reference point
5	Move back to reference point
6	Move forth to reference point

If the selected axis can't execute the defined reference run, the command will not be executed and the remote state RM_STATE_CMD_ERR will be set.

Note:

For *ISEL* and *Aerotech Ensemble* controllers the reference drive is defined in the controller. For these controllers the 'Type of Reference Run' parameter is not evaluated.

For action number 0 (initialization) the further parameters are not evaluated.

For action number 5 no further parameter is evaluated. This action corresponds to the {Reference Run} button in the status bar. All axes which are selected in the hardware configuration by activating the 'Automatic Reference Run' checkbox are commanded to do the reference run. The values of the hardware configuration are used.
To check the execution of this command, use "Command #4 – Query for Program State". During execution, the RM_STATE_DEV_RUNNING flag (= 0x00400000) will be set.
To check the individual success, use "Command #41 – Motor Axis Status Query" for each axis.

Even if some parameters are not used, all have to be provided by the remote command.

If the command is executed but will fail due to some hardware problems, the remote state RM_STATE_DEV_ERR will be set.

The status bit RM_STATE_AXIS_EXECUTION is provided for checking the execution. It will be set during an axis movement, either a reference run or a normal movement.

Note:

If an axis error state is signaled during a movement, the 'Axis movement active' flag may remain set.
To reset that flag, use the 'emergency stop' command to be able to reinitialize the controller.

4.41 Command #41 – Motor Axis Status Query

This command returns the state of the selected motor axis. The first parameter defines the type of the selection:

- "1" defines the selection by name.
  The parameter 2 is a string representing the axis name as defined in the laserDESK hardware settings.
- "2" defines the selection by number.
  The parameter 2 is an integer representing the (zero based) axis number as defined in the table of axis in the laserDESK Hardware Configuration.

The return value represents the laser state or "-1", if the command was not successful. This is the case, if the selected axis is not defined in the actual hardware configuration. Because "0" is a valid axis state, the error value has to be "-1". If the return value is "0", the axis is defined but not initialized.

The motor communication error may be reset by this command, if the error was temporary present.

The state is given as a bit field:

Table 14

Bit	Meaning (if bit is set)
0x00000001	Motor control active
0x00000002	Motor axis initialized, connection established
0x00000004	Reference run executed, reference detected
0x00000008	Motor axis error
0x00000010	Lower end switch signals
0x00000020	Upper end switch signals
0x00000040	Emergency break signals
0x00000080	Motor communication error
0x00010000	Reference run active
0x00020000	Axis movement active
0x00040000	Position acknowledge

Notes:

- These status flags are only valid for laserDESK version 1.4.8.x and higher.
- Not all state information is available for each axis type.
  Please refer to the manual of the used axis type to get more information.

4.42 Command #42 – Set POF Field Transformation

This command sets a transformation for the virtual image field of the scan head (see chapter "Virtual Image Field" in the RTC5 or RTC6 manual). This Processing-on-the-fly (POF) transformation is only applied, if the POF function is activated (in the job properties, the 'Use POF' variable has to be set to "True").

The transformation will be applied on the next execution. If an execution is just running, it will be finished first.

There is no GUI element where you can change this transformation. It is only changeable by this remote command. This POF transformation will only be reset in following situations:

- When you send this command with parameters (0.0, 0.0, 0.0).
- When the RTC board is initialized again (e.g. when changing the hardware configuration).
- When the automatic mode is switched off.

The POF transformation is an additional transformation, applied prior to the POF correction (see chapter "Processing-on-the-fly" in the RTC5 or RTC6 manual).

It is independent from and additional applied together with the hardware configuration settings and possible transformations set by the remote commands #16 and #20 (see page 22 and page 23).

4.43 Command #43 – Set Transformation for a 3D System

This command sets a combined transformation (x, y and z-offsets and a 2D-transformation matrix) for a 3D system. It can only be used in automatic mode. It is a combination of the commands #16 and #28 and an additional z-offset.

The transformation is valid until this command is executed again or parts of the transformation may be changed by their correspondent command:

- "Command #16 – Set Coordinate Transformation in Automatic Mode" will replace the x, y-offset.
- "Command #28 – Execute global matrix transformation" will replace the matrix part.

The z-offset is only changeable with this command.

When the automatic mode is switched off, the transformation is reset except of the z-offset. This value remains valid for further manual starts until the hardware configuration is new initialized or the auto-

matic mode is switched on again. Thus, it is recommended to reset the z-offset prior to switch off the automatic mode.

The transformation will be applied on the next execution. If an execution is just running, it will be finished first.

The transformation is additional applied together with the hardware configuration settings. The z-offset is an absolute value.It will overwrite the hardware configuration z-offset temporarily (until hardware reset).

If transformation control nodes are used in the job, the x, y-offset and the rotation will be reset at the end of each job execution. In that case, this part of the transformation defined here is only valid for the next execution and only until the first transformation control node is executed.
You should consider to use the "Command #55 – Set Transformation for Control Node" instead.

4.44 Command #44 – Set Transformation for a 2 Scan Head System

This command sets a combined transformation (x, y-offsets and a 2D-transformation matrix) for a second scan head system. It can only be used in automatic mode. It is a combination of the commands #16 and #28 and is individual for each scan head.

The transformation is valid until this command is executed again or parts of the transformation may be changed by their correspondent command:

- "Command #16 – Set Coordinate Transformation in Automatic Mode" will replace the offset.
- "Command #28 – Execute global matrix transformation" will replace the matrix part.

When the automatic mode is switched off, the transformation is reset.

The transformation will be applied on the next execution. If an execution is just running, it will be finished first.

The transformation is additional applied together with the hardware configuration settings.

If transformation control nodes are used in the job, the x, y-offset and the rotation will be reset at the end of each job execution. In that case, this part of the transformation defined here is only valid for the next execution and only until the first transformation control node is executed.
You should consider to use the "Command #55 – Set Transformation for Control Node" instead.

4.45 Command #45 – Select RTC Board

This command selects an RTC board which will be acquired and used.

The type parameter is defined as follows:
1 = RTC5
2 = RTC6
3 = RTC6 Ethernet
Other values are not valid.

The serial number of the RTC board is the selection parameter. If a RTC board with that serial number and the defined type is present in the system, it will be acquired and initialized.

This is a non-blocking command and the answer is returned immediately to inform the client that the command will be executed. The successful execution can only be checked by status queries. If some error occurs, the RM_STATE_CMD_ERR flag will be set (especially if the serial number does not fit). Else the RM_STATE_RTC_INIT flag will show the successful acquire and initialization of the selected RTC board.

Note:

After this command, it is strongly recommended to adapt the Hardware Configuration using "Command #27 – Change the Hardware Configuration". Else the selected board may use wrong settings.

4.46 Command #46 – Convert Vectors into Points

This command converts a specified group or object into a list of points with a defined distance. In addition, the time duration the laser will be switched on to mark each point, will be set.

The command's parameters are:

(1) UID (string):
The UID of the element which should be converted

(2) Point distance (double):
The distance between the points in [mm]

(3) Marking time of a point (integer):
The time duration the laser will be switched on at each point in [µs]

(4) Equal spacing (integer):
If "0", the defined distance will not be adapted. In that case, the last point will normally have a different distance (remainder) than all other points.
If not "0", the points on the polygon edges will be

shifted to have equal distances. For interpolated curves this calculation is not exact, due to rounding errors during interpolation. This calculation is only executed for vectors longer than 5 times the point distance to keep the distance variation lower than 20%.

The point line is integrated in a graphics paths group. This group replaces the selected vector element or group.

The command is not applicable for the job node, variants or bitmaps.

This is a non-blocking operation. The return value of "1" shows only, that the command is accepted and will be executed. During the execution, the RM_STATE_JOB_LOAD status flag is reset. If an error occurs, the RM_STATE_CMD_ERR flag will be set.

4.47 Command #47 – Activate Laser Beam

This command activates the laser for a certain time with defined power, frequency and pulse length.

The command's parameters are:

(1) Power (double) – The used laser power (in [%])

(2) Frequency (double) – The used frequency of the laser (in [kHz]); ignored for CO_2 lasers (laser mode 0)

(3) Pulse length (integer) – The pulse length (in [μs])

(4) Laser-on time (double) – The time duration the laser will be switched on (in [ms])

The laser system has already been switched on (RM_STATE_LAS_ON is set), else this command will be refused.

The laser-on time parameter defines the behavior of this command:

- If the laser-on time is positive, the laser beam will be switched on for this defined time. During this time an RTC list is running, meanwhile the RM_STATE_EXEC_DONE status flag is reset. The list execution can only be stopped by the 'Emergency Stop' command (see "Command #13 – Emergency Stop") – it cannot be stopped by using this command again with laser-on time set to zero (see below).
- If the laser-on time is negative, the laser beam will be switched on and never switched off again. In that case, you have to send this command again with laser-on time equal to "0.0" or use the 'Emergency Stop' command (see "Command #13 – Emergency Stop") to switch off the laser beam.
- If the laser-on time is "0.0", the command serves only to switch off the laser beam when switched on before by this command with negative laser-on time (see above).
  Else this command has no effect.

Note:

Be careful when using this command, because the laser activation occurs without warning to people at the setup.

4.48 Command #48

Reserved

4.49 Command #49

Reserved

4.50 Command #50 – Get Scan Head specific Parameter

This command requests specific scan head parameters from a system connected to the active board. Only if an i*DRIVE* system is connected valid values can be returned.

It can only be used if no RTC list execution is present, the automatic mode is switched off and no other remote command is currently being executed.

The first parameter (integer, 1 or 2) selects the scan head, the second parameter (integer) defines the requested value (see Table 4 on page 11).

Three values (integer, double, double) are returned. If the command fails, the answer returns 0 for the integer value. If the state of the scan head is requested, it will be returned in the first value as an integer, else the integer value corresponds to the scan head number and the requested values are returned as doubles. The units are [mm] for positions, [mA] for current and [°C] for the temperature.

4.51 Command #51 – Set specific Parameter Value

This command replaces a specific parameter of a marking object or parameter set. It may create either an individual parameter set or a named parameter set, which is then present in the job library. There are several possibilities:

(1) If the job has already a parameter set with given name it will be replaced, else it will be created.

(2) All parameters are valid. A parameter set is created or replaced in the job library and assigned to the defined element.

(3) The UID parameter is invalid. A parameter set is created or replaced in the job library and no assignment is changed.

(4) The 4th value is omitted. An individual parameter set for the selected element is created.

If no valid element and no existing parameter set are defined, an error '-4' will be returned (for return values, see Table 18 on page 35).

Until no further error happens, any new created or replaced parameter set will get the new value.

The command can only be used if no RTC list execution is present, the automatic mode is switched off and no other remote command is currently being executed.

- Parameter 1 is the UID of the element, where the parameter should be replaced. It may be an empty string but may not be null.
- Parameter 2 defines the kind of parameter. The highest byte defines the parameter set, where 0x10000000 is for a marking parameter, 0x20000000 is for a filling parameter and 0x40000000 is for a bitmap parameter.
- Parameter 3 is the value type and is always a double. It may be interpreted as an integer, boolean or enumeration (see column 'Value Range' in the tables below).
- Parameter 4 defines the name of the changed parameter set. It may be omitted or null. Then an anonymous individual parameter set is created which is only assigned to the selected element.

The parameter identifiers are defined in following tables:

Marking Parameters

Table 15

Parameter ID	Parameter Name	Value Range
0x10000001	Power	[0 – 100]
0x10000002	Frequency	[0 ...]
0x10000003	Pulse Width	[0 – 1/frequency]
0x10000004	First Pulse Suppression	[0 ...]
0x10000005	Mark Speed	[>0]
0x10000006	Jump Speed	[>0]
0x10000007	LaserOn Delay	
0x10000008	LaserOff Delay	[0 ...]
0x10000009	Jump Delay	[0 ...]
0x1000000a	Mark Delay	[0 ...]
0x1000000b	Polygon Delay	[0 ...]
0x1000000c	Wobble Mode	1 = elliptic 2 = lying 8 3 = standing 8 else 'off'
0x1000000d	Wobble Frequency	
0x1000000e	Longitudinal Wobble Amplitude	[0 ...]
0x1000000f	Transversal Wobble Amplitude	[0 ...]
0x10000010	Skywriting Mode	1 = Mode 1 2 = Mode 2 3 = Mode 3 4 = 'default mode' else 'Off'

Table 15

Parameter ID	Parameter Name	Value Range
0x10000011	Timelag	
0x10000012	LaserOn Shift	
0x10000013	Forerun	[0 ...]
0x10000014	Overrun	[0 ...]
0x10000015	Angle Limit	
0x10000016	excelli*SCAN* LasOn Delay	
0x10000017	excelli*SCAN* LasOff Delay	
0x10000018	excelli*SCAN* Acceleration Scale	[0 ...]
0x10000019	excelli*SCAN* Corner Scale	[0 ...]
0x1000001a	excelli*SCAN* End Scale	[0 ...]

Filling Parameters

Table 16

Parameter ID	Parameter Name	Value Range
0x20000001	Filling Type	1 = Hatch 2 = Path else 'Path with Hatch'
0x20000002	Filling Line Distance	[>0]
0x20000003	Line Reduction	[0 ...]
0x20000004	Outline Reduction	[0 ...]
0x20000005	Beam Compensation	[0 ...]
0x20000006	Number of Loops	[0 ...]
0x20000007	Join First Lines Around	>0 = true
0x20000008	Line Join Type	1 = Miter 2 = Bevel else 'Round'
0x20000009	Innermost First	>0 = true
0x2000000a	Hatch Angle	1 = absolute 2 = aligned 3 = relative
0x2000000b	Number of Hatch Levels	[1 ...]
0x2000000c	Hatch Angle Start	
0x2000000d	Angle Step	
0x2000000e	Line Jump	[1 ...]
0x2000000f	Sorting	>0 = true
0x20000010	Search Radius	[>0]
0x20000011	Jump Reduction	>0 = true
0x20000012	Minimal Jump	[>0]
0x20000013	Bidirectional Hatching	>0 = true

Bitmap Parameters

Table 17

Parameter ID	Parameter Name	Value Range
0x40000001	Pixel Distance	[>0]
0x40000002	Dithering	1 = Floyd Steinberg 2 = Jarvice, Judice, Ninke else 'None'
0x40000003	BitmapMode	2 = UFPM (Ext1) 3 = UFPM (Ext21 4 = UFPM pulse width 5 = Microvectors else 'Standard'
0x40000004	Calibration Table	Not supported
0x40000005	BlackValue	>127.5 => 255, else 0
0x40000006	Bidirectional	>0 = true
0x40000007	Row Synchronization [mm]	
0x40000008	Pulses per Pixel	[> 0], only used for Microvectors mode
0x40000009	Fore- / Overrun [µs]	≥0
0x40000011	Pixel Distance (in X)	[>0]
0x40000012	Pixel Distance (in Y)	[>0]

Examples:

To select the marking parameter 'Frequency' set the parameter ID to 0x10000002.

To select Floyd-Steinberg Dithering set the parameter ID to 0x30000002 and the parameter value to 1.

The return value gives some additional information:

Table 18

Return Value	Meaning
1	Parameter successfully replaced
0	Parameter doesn't belong to the object or a prerequisite is missing
-1	Undefined kind of parameter set is selected (i.e. no marking, filling or bitmap parameter set)
-2	Undefined parameter is selected
-3	Value is out of range
-4	UID doesn't belong to an element and no valid parameter set is defined
-5	For any other error detection (e.g. a wrong parameter type)

4.52 Command #52

Reserved

4.53 Command #53 – STL File Import and Additional Slicing

This command imports an STL file (Standard Tesselation Language) and cuts it into parallel slices for 2,5D processes (e.g. deep engraving or additive manufacturing). Several options define the resulting laserDESK job(s).

The table below shows the parameters of this command. Parameters [1] to [8] are mandatory. You may omit parameters at the end if not needed.

Table_19

Parameter	Meaning
[1] File name (string)	Full path to the stl file
[2]Length (double)	Length [mm]
[3] Width (double)	Width [mm]
[4] Height (double)	Height [mm]
[5] Base center X (double)	X position of the figure center
[6] Base center Y (double)	Y position of the figure center
[7] Z-Position (double)	Z position of the first slice (bottom for ascending order, top for descending)
[8] Slizes (double)	Number of slices / distance between slices (dependent on distance flag, see Table_20 below)
[9] Option flags (uint)	Option flags (definition in Table_20 below)
[10] Filling offset (double)	Filling activation by flag; parameter selection by command #51, see Table 16 on page 34
[11] Signal output (integer)	Number of Digital Output of extension 1
[12] Signal input (integer)	Number of Digital Input of extension 1

Parameter [9] defines the option flags which control the resulting laserDESK job(s). These flags can be combined. The option flags are defined in the table below:

Table_20

Flag	Meaning
0x00000001	Apply filling
0x00000002	Calculate slices descendent
0x00000010	Insert z-focus control nodes
0x00000020	Create layers
0x00000040	Create job for each slice
0x00000080	Insert signal exchange
0x00000100	Distance flag; defines slices parameter [8] (0 = number of slices; 1 = distance in [mm])

- If flag **0x0001** (Apply filling) is set, the filling parameters of the job node are used and the angle offset defined by parameter 10 is applied. You may use remote command #51 to define appropriate filling parameters prior to the slicing.

- If flag **0x002** (Calculate slices descendent) is set, the slices are calculated from top to bottom. This is especially for deep engraving applications.
- If flag **0x0010** (Calculate slices descendent) is set, z-focus control nodes are inserted into the job instead of setting the z-value of each polygon point.
- If flag **0x0020** (Create layers) is set, the polygons of each slice are inserted in an own layer labelled "Layer ***n***". The z-values will be a layer property and not a polygon point value.
  If additional flag 0x0010 is set, the z-value is defined by the z-focus node and not by the layer.
- If flag **0x0040** (Create job for each slice) is set, each slice is saved in an own job. The job name consists of the actual job name with the extension "_Level***n***.sld" where ***n*** is the number of the slice.
- If flag **0x0080** (Insert signal exchange) is set, all parameters (especially 11 and 12) are required.
- If flag **0x0100** (Distance flag) is set, the parameter [8] will be taken as the distance of the slices, not as the amount of the calculated slices.

Each slice is saved in a group where the group UID is "Slice***#n***". Each calculated polygon gets the UID "Slice***#n***_Polygon***#k***" where ***n*** is the slice number and ***k*** is the polygon number in the slice. This naming convention allows identifying the individual objects by other remote commands.
If any UID is already existent in the job the values may change and cannot be exactly predicted anymore. So we recommend starting with an empty new job if the UID is needed for further remote commands.

The command is executed as an asynchronous operation. The answer is returned immediately and the successful execution has to be detected by status queries. During the calculation the status flag RM_STATE_JOB_LOAD will be reset. It will be set again when the slicing calculation has finished.
If an error occurs the RM_STATE_CMD_ERR flag will be set.

4.54 Command #54 – Set Marking Count of an Element

This command sets the marking count of the element with the given UID.
Parameter 1 defines the element, parameter 2 is the new marking count value.
Only positive values and zero are allowed.

4.55 Command #55 – Set Transformation for Control Node

This command sets the transformation(s) for the transformation control nodes in a job. The transformation is additional to the transformation defined in the Hardware Configuration and can be head specific. The head to which the transformation should be applied can be defined in the control node and is not changeable by the remote command.

The transformation data block (parameter 1 to 4) can be repeated several times in the command parameter list to change all used transformation control nodes with one remote command. Only fully defined blocks are allowed.

Each control node references one transformation by the ID (**not** the UID). Therefore, it is possible to change all transformation control nodes of the same ID by one transformation data block. In addition, a transformation can be used several times in a job, allowing to switch back and forth between transformations.

- Function in **automatic mode**:
  The transformation(s) of the command will be applied on the following executions instead of the parameters of the node(s). If an execution is just running, it will be finished first. When the automatic mode ends, all transformations are cleared and the node parameters are valid again.
- Function in **manual mode**:
  The parameters of the control node(s) that refer to an ID of this command will be replaced. The next execution will use these parameters during list calculation. These values remain temporarily valid until they are changed again or the job is closed. They will also be used when switching to automatic mode. If the job is saved, the new parameters will be permanently set in the job.

The command parameters are:

(1) ID of the transformation (int):
Several Transformation Control Nodes can use the same ID to allow the remote changing all together.

(2) x Offset [mm] (double)

(3) y Offset [mm] (double)

(4) Rotation [°] (double):
Rotation around the origin.

You may use several blocks of parameters 1 to 4 to set several Transformation Control Nodes with one remote command.

Example (without DLE correction):

Command #55 – Set Transformations (1,-10,0,0) (2,0,0,10)
0x02, 0x3c, 0x00, 0x00, 0x00, 0x37, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x24, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x02, 0x00, 0x24, 0x40, 0x04, 0x03

This command sets the transformation
– for ID = 1 to x-shift of –10 mm and
– for ID = 2 to a rotation of 10°.

Answer:
If at least one transformation block is valid, the answer is 1 (success). If further transformation blocks are invalid, they are simply ignored.

5 Remarks

5.1 Unique Identifier (UID)

laserDESK assigns an unique identifier to every marking object inside one job. These UIDs allow the remote commands to define the target object of the command exactly. The command number defines the action, which will be executed with the target object.

5.2 Variable Text

The string of every text object can be changed by "Command #14 – Set Text String". All other text object parameters can't be changed by a remote command. The text object is selected by its unique identifier (UID). It can be changed at any time.

At execution start, the actual text string will be taken to evaluate the marking vectors. Therefore, this command will be refused during this vector calculation.

To allow the changing of text by the remote command, the text object's graphic parameter 'Variable' has to be set to "True".

The objects Serial Number, Date/Time and Import Text can't be changed by this command, because they will be actualized by their special functions.

5.3 Automatic Mode

When switching to automatic mode, several settings are done:

- The external start directly to the RTC board is enabled.
- The process data of the constant marking objects is calculated.
- The RTC board's list memory is initialized and loaded with constant data.
- The serial numbers are reset to their start values (if enabled).
- The current index of a text import is reset to the start index (if enabled).

During the executions in automatic mode the actual number of executions of the variants or job is updated. Additional the target quantity is checked and if this value is reached, no further execution is possible. If the target quantities of all variants are reached, the program will switch back to manual mode.

Additional the actual value of serial numbers and import text objects are updated. To keep these values, it is possible to save the job file automatically, when switching off the automatic mode. This can be set in the GUI settings (to open via the 'Edit\Options' menu; then select the 'General' section and activate the 'Safe file after marking' check box in the field 'Automatic Mode'.)

It is possible to define the start value of a serial number by "Command #15 – Define the Serial Number Start, Actual or End Value". The start value will only be used when switching the automatic mode on. Therefore the definition of the start value can only be done in manual mode.

Note:

If a coordinate transformation via remote control and also a serial number (or import text) is needed, the parameter 'Reset Start Value' has to be set to 'False', else the serial number (or text string) will be reset at every start.

If one needs to change the actual value, the command 15 can be used, too. The current value will be set to the start value, if the current value is out of range. If you create a job with a serial number start value equal to zero, then you can send command 15 to set the start value to the desired one and execute the first marking with this value. If you have finished, don't save the job to be able to use this mechanism again.

5.4 List of Requirements to Execute the Commands

Table 21

Command Number	Command (Name)	Requirements
1	Login of the remote control (client) at laserDESK	Only possible, if no client is already logged in
2	Logout	Permitted, if client is logged in
3	Query program version	Permitted, if client is logged in
4	Query program status	Permitted, if client is logged in
5	Remote mode on	Permitted, if client is logged in, sets RM_STATE_RM_MODE state flag
6	Remote mode off	Permitted, if RM_STATE_RM_MODE is set
7*	Open job file	Permitted, if RM_STATE_RM_MODE is set and RM_STATE_AUTOMODE and RM_STATE_LST_CALC are reset
8*	Save job as...	Permitted, if RM_STATE_RM_MODE is set and RM_STATE_AUTOMODE and RM_STATE_LST_CALC are reset
9*	Switch laser on	Permitted, if RM_STATE_RM_MODE is set
10*	Switch laser off	Permitted, if RM_STATE_RM_MODE is set and RM_STATE_AUTOMODE and RM_STATE_LST_EXEC are reset
11*	Specify variant	Permitted, if RM_STATE_RM_MODE is set
12*	Start laser processing, job or specified variant	Permitted, if RM_STATE_RM_MODE and RM_STATE_LAS_ON are set and RM_STATE_LST_EXEC is reset
13*	Emergency stop of execution	Permitted, if RM_STATE_RM_MODE is set.
14*	Define the text string of a text object (UID)	Permitted, if RM_STATE_RM_MODE is set and RM_STATE_LST_CALC is reset
15*	Define the start, actual or end value of a serial number	Permitted, if RM_STATE_RM_MODE is set and RM_STATE_AUTOMODE and RM_STATE_LST_CALC are reset
16*	Shift and rotate the marking/coordinate system	Permitted, if RM_STATE_RM_MODE and RM_STATE_AUTOMODE are set
17*	Define the target quantity of the specified variant or job	Permitted, if RM_STATE_RM_MODE is set and RM_STATE_AUTOMODE and RM_STATE_LST_CALC are reset
18*	Query of the number of executions of the specified variant or job	Permitted, if RM_STATE_RM_MODE is set
19*	Define the output lines of the RTC board	Permitted, if RM_STATE_RM_MODE is set
20*	Shift and rotate the coordinate system	Permitted, if RM_STATE_RM_MODE is set and RM_STATE_AUTOMODE and RM_STATE_LST_CALC are reset
21*	Automatic mode on	Permitted, if RM_STATE_RM_MODE is set and RM_STATE_AUTOMODE and RM_STATE_LST_EXEC are reset

Table 21

Command Number	Command (Name)	Requirements
22*	Automatic mode off	Permitted, if RM_STATE_RM_MODE and RM_STATE_AUTOMODE are set and RM_STATE_LST_CALC is reset
23*	Release RTC sets RM_STATE_RTC_RELEASED	Permitted if RM_STATE_RM_MODE is set and RM_STATE_AUTOMODE and RM_STATE_LST_CALC are reset
24*	Acquire RTC resets RM_STATE_RTC_RELEASED	Permitted if RM_STATE_RM_MODE and RM_STATE_RTC_RELEASED are set
25*	Read the signals from a connection to the RTC board	Permitted, if RM_STATE_RM_MODE is set
26*	Define the positions of the x and y galvanometer scanners and the z optics	Permitted, if RM_STATE_RM_MODE and RM_STATE_EXEC_DONE are set
27*	Change the hardware configuration	Permitted, if RM_STATE_RM_MODE and RM_STATE_EXEC_DONE are set and RM_STATE_AUTOMODE is reset
28*	Set the global matrix transformation	Permitted, if RM_STATE_RM_MODE is set Refused, if RM_STATE_AUTOMODE and RM_STATE_EXEC_DONE are reset
29*	Select the scan head (integer)	Permitted, if RM_STATE_RM_MODE and RM_STATE_EXEC_DONE are set and RM_STATE_AUTOMODE is reset
30*	Execute ASC / Check, if ASC is available on the system	Permitted, if RM_STATE_RM_MODE and RM_STATE_EXEC_DONE are set and RM_STATE_AUTOMODE is reset
31*	Set marking parameter set for job	Permitted, if RM_STATE_RM_MODE and RM_STATE_EXEC_DONE are set and RM_STATE_AUTOMODE is reset
32*	Load/Replace vector graphics	Permitted, if RM_STATE_RM_MODE and RM_STATE_EXEC_DONE are set and RM_STATE_AUTOMODE is reset
33	Get marking time	Permitted, if client is logged in
34	Get marking parameter sets	Permitted, if client is logged in
35*	Hide/Show program GUI	Permitted if RM_STATE_RM_MODE is set
36*	Set power calibration factor	Permitted if RM_STATE_RM_MODE and RM_STATE_EXEC_DONE are set and RM_STATE_AUTOMODE is reset
37*	Set vision system parameter	Permitted if RM_STATE_RM_MODE and RM_STATE_EXEC_DONE are set and RM_STATE_AUTOMODE is reset
38*	Set new marking parameter set for objects of a specific layer	Permitted if RM_STATE_RM_MODE and RM_STATE_EXEC_DONE are set and RM_STATE_AUTOMODE is reset
39*	Get layer names of job	Permitted, if client is logged in
40*	Motor axis control	Permitted if RM_STATE_RM_MODE and RM_STATE_EXEC_DONE are set and RM_STATE_AUTOMODE is reset
41	Motor axis status query	Permitted, if client is logged in

Table 21

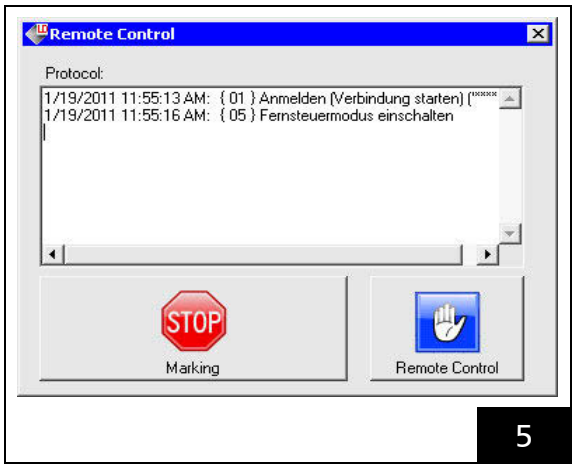
Command Number	Command (Name)	Requirements
42*	Set coordinate transformation for POF application (transformation of the whole virtual image field)	Permitted if RM_STATE_RM_MODE is set
43*	Set total transformation for a 3D system	Permitted, if RM_STATE_RM_MODE and RM_STATE_AUTOMODE are set
44*	Set total transformation individual for each scan head of a second scan head system	Permitted, if RM_STATE_RM_MODE and RM_STATE_AUTOMODE are set
45*	Select RTC board	Permitted if RM_STATE_RM_MODE and RM_STATE_EXEC_DONE are set and RM_STATE_AUTOMODE is reset
46*	Converts vectors to points	Permitted if RM_STATE_RM_MODE and RM_STATE_EXEC_DONE are set and RM_STATE_AUTOMODE is reset
47*	Activates the laser beam	Permitted if RM_STATE_RM_MODE and RM_STATE_LAS_ON and RM_STATE_EXEC_DONE are set and RM_STATE_AUTOMODE is reset
50*	Get scan head specific parameter	Permitted if RM_STATE_RM_MODE and RM_STATE_EXEC_DONE are set and RM_STATE_AUTOMODE is reset
51*	Set specific parameter	Permitted if RM_STATE_RM_MODE and RM_STATE_EXEC_DONE are set and RM_STATE_AUTOMODE is reset
53*	Import and slice an STL file	Permitted if RM_STATE_RM_MODE and RM_STATE_EXEC_DONE are set and RM_STATE_AUTOMODE is reset or if no RTC board is acquired
54*	Set marking count	Permitted if RM_STATE_RM_MODE is set and RM_STATE_AUTOMODE and RM_STATE_LST_CALC are reset
55*	Set coordinate transformation for the transformation control nodes in the job	Permitted if RM_STATE_RM_MODE is set

* This command is only accepted, when the remote mode is on (see chapter 4.5). It will only be executed if the RM_STATE_LST_CALC state flag is reset.

6 Executing the Remote Control

After laserDESK has been started (and if neither the pilot laser mode nor another program blocks the access to laserDESK), the remote control of laserDESK can be used.

- Log in to laserDESK.
  - The access to laserDESK is blocked for every further program.
  - In this state you can query the program version and program status.
- Switch on the remote mode.
  - A blocking dialog appears on the laserDESK program window (see figure below). Any user interaction is blocked. Only persons with laserDESK access authorization “Administrator” or “Supervisor” can cancel the block access via the {Remote Control} button.



Connection Remote Control Program – laserDESK

- Execute the required commands.

7 SLLDRemoteControl.dll

(Version 2.6.1)

7.1 Description of the Functions

The *SLLDRemoteControl.dll* enables a programmer to easily integrate the remote control of laserDESK into his own program without any knowledge of *Windows* sockets and TCP/IP connections. The whole connection is handled by the dll and one just needs to call simple functions for executing tasks in laserDESK.

The dll is written in C# with .NET and therefore easily to integrate in other .NET programs. Just add a reference into your project and use the SLLD_Functions class (see example in chapter 7.4 on page 54).

This dll can't be used to use the remote control interface of laserDESK with other systems (PLC …). In this case the standard and more general TCP/IP or RS232 interface must be used. Then all programming languages on different platforms with TCP/IP or RS232 support can be used for the remote control of laserDESK.

The *SLLDRemoteControl.dll* file is included in the installation package and is stored in the 'Remote' program subdirectory.

7.2 Function Overview

public int OpenConnection(byte[] ipAdr, int Port)

Parameters: byte[] ipAdr is the IP address (ipv4) of the laserDESK server, most significant byte first (e.g. 127.0.0.1 corresponds to byte[0] = 127, byte[1] = 0, byte[2] = 0, byte[3] = 1).
Port is the port number of the connection. It has to be defined in the laserDESK 'Hardware Configuration' page too.
Function: The method opens a TCP/IP connection to the laserDESK server and sends the login command. This command has to be executed prior to all other commands.
Return value: 1 = success, 0 = error.
Remark: If the connection was interrupted but you were already logged in, you need to send this command again to re-establish the connection. You may get 0 back because the log in command stays valid and the laserDESK server may not detect this connection breakdown.

public int CloseConnection()

Parameter: None.
Function: Closes the connection.
Return value: 1 = success, 0 = error.
Remark: Even if 0=error is returned, this command stops the network connection thread and you can't send a command any more. Only the OpenConnection() method is working.

public string GetVersion()

Parameter: None.
Function: Returns the program version of laserDESK. The version allows to check whether laserDESK supports the command or not (see last column in Table 4 on page 11)
Remark: If the laserDESK version is lower than 1.0.5.8, the functions OpenJobNoWait and SwitchAutomaticModeOnNoWait can't be executed successfully.

public uint GetState()

Parameter: None.
Function: Returns the laserDESK state as a bit field. For details, refer to "Command #4 – Query for Program State".

public int GetStateAsInt()

Parameter: None.
This is the same function as GetState(), but the return value is an integer.

public int SwitchRemoteMode(bool on)

Parameter: on = true switches the remote mode on.
Return value: 1 = success, 0 = error.
Remark: All methods described below can only be executed successfully with the remote mode switched on. This command blocks any local user interaction of the laserDESK GUI.

public int OpenJob(string jobname, bool saveOld)

Parameters:
jobname: The full qualified path to the job file
saveOld = true saves the old job.
Function: Opens a new laserDESK job and closes the old one. The answer is returned when the remote command has been executed (synchronous operation).
Return value: 1 = success, 0 = error.

public int OpenJobNoWait(string jobname, bool saveOld)

Parameters:
jobname: The full qualified path to the job file
saveOld = true saves the old job.
Function: Opens a new laserDESK job and closes the old one. The answer is returned immediately (asynchronous operation). The success of this command has to be detected by status queries.
Return value: 1 = success, 0 = error.
Remarks: This command can only be executed by laserDESK version 1.0.5.8 and higher. Lower versions return 0.
While this command is executed, only status queries are allowed. All other commands will be refused.

public int SaveJob(string jobname, bool overwrite)

Parameters:
jobname: The full qualified path to the job file where it has to be saved.
overwrite = true saves the job even if it already exists.
Return value: 1 = success, 0 = error.

public int SwitchLaser(bool on)

Parameter: on = true switches the laser on.
Return value: 1 = success, 0 = error.
Remark: This command is only necessary for specified laser types not for the GeneralType.

public int SelectVariant(string UID)

Parameter: UID is the identifier of the variant to be selected.
Return value: 1 = success, 0 = error.
All further commands referring to a variant need this command in advance. For details, refer to "Command #11 – Select Variant" on page 21.

public int StartMarking()

Parameter: None
Function: Starts a marking process in automatic or manual mode.
Return value: 1 = success, 0 = error.
Remark: If the job has variants a variant has to be selected prior to this command.

public int EmergencyStop()

Parameter: None.
Function: Stops a marking process immediately.
Return value: 1 = success, 0 = error.

public int SetText(string UID, string text)

Parameters:
UID is the identifier of the text object to be changed.
text is the new content of the text object.
Function: Replaces the text string of the selected text object.
Return value: 1 = success, 0 = error.
Remark: The text object needs the property variable = true to be set. For details, refer to "Command #14 – Set Text String" on page 22.

public int SetSerialNo(string UID, int value, int selection)

Parameters:
UID is the identifier of the serial number object.
pValue is the new value to be set.
selection defines the kind of the serial value which is set:

- selection = 1: Start value is set (default)
- selection = 2: Current value is set
- selection = 3: End value is set (default)

Function: Sets the selected value for the serial number. If the current value is outside of the limits it will be clipped to this limit.
Return value: 1 = success, 0 = error.

public int SetTotalExecutions(int executions)

Parameter: executions is the new value for the target quantity property.
Function: Sets the 'Target quantity' property for the job or selected variant.
Return value: 1 = success, 0 = error.

public int GetActualExecutions()

Parameter: None
Function: Returns the 'Number of executions' property for the job or selected variant.

public int SetOutputPins(int connector, uint mask, uint value)

Parameters:
connector: 1 = Extension 1, 2 = Extension 2, 3 = Laser connector.
mask: The set bits (= 1) are used, the unset bits (=0) are not changed.
value: The bit pattern to be set (e.g. line 0 and 2 set corresponds to value 5).
Function: Sets the output lines of the RTC connectors. The real output sets only the unmasked lines as (mask & value).
Return value: 1 = success, 0 = error.
For details, refer to "Command #19 – Set Output Signals of the RTC" on page 23.

public int SetOutputPinsByInt(int connector, int mask, int value)

Same function as SetOutputPins(…) but with integer parameter type.

public int SetTransformation(double xshift, double yshift, double rotation)

Parameters:
xshift, yshift and rotation define the transformation of the whole job.
Return value: 1 = success, 0 = error.
Remarks: This method uses the necessary command #16 or #20 dependent on the previous used method SwitchAutomaticMode(). It can be applied in automatic and manual mode. For details, refer to "Command #16 – Set Coordinate Transformation in Automatic Mode" on page 22 and "Command #20 – Set Coordinate Transformation in Manual Mode" on page 23.

public int SetTransformationEx(double xshift, double yshift, double rotation, int ScanHead)

Parameters:
xshift, yshift and rotation define the transformation of the whole job.
ScanHead defines the scan head on which the transformation should be applied. Possible values are 1, 2 for the specific scan head or 3 for both scan heads.
Return value: 1 = success, 0 = error.
Remarks: This method is available for laserDESK version 1.0.7.3 or higher only.
It uses the necessary command #16 or #20 dependent on the previous used method SwitchAutomaticMode() and can be applied in automatic and manual mode. For details, refer to "Command #14 – Set Text String" on page 22 and "Command #20 – Set Coordinate Transformation in Manual Mode" on page 23.
In manual mode 'SetTransformationEx' with parameter ScanHead = 3 is different than 'SetTransformation'. Here the transformation is always temporary, in contrast 'SetTransformation' will define the object properties 'Mark Translation' and 'Mark Rotation' (not scan head specific) which may persist and can be stored.

public int SwitchAutomaticMode(bool on)

Parameter: on = true switches the automatic mode on
Return value: 1 = success, 0 = error.
Switches the automatic mode. The answer is returned when the remote command has been executed (synchronous operation).
For details, refer to "Command #21 – Switch Automatic Mode On" on page 24 and "Command #22 – Switch Automatic Mode Off" on page 24.

public int SwitchAutomaticModeOnNoWait()

Parameter: None.
Return value: 1 = success, 0 = error.
Switches the automatic mode on. The answer is returned immediately (asynchronous operation). The success of this command has to be detected by status queries.
For details, refer to "Command #21 – Switch Automatic Mode On" on page 24 and "Command #22 – Switch Automatic Mode Off" on page 24.
Remarks: This command can only be executed by laserDESK version 1.0.5.8 and higher. Lower versions return 0.
While this command is executed, only status queries are allowed. All other commands will be refused.

public int ReleaseRTC();

Parameter: None.
This command releases the selected RTC.

public int AcquireRTC();

Parameter: None.
This command acquires the last used RTC of the last selected type.

public uint GetInputPins(int connector)

Parameters:
connector: 1 = Extension 1 output, 4 = Extension 1 input, 5 = Laser connector input.
Function: Returns the line states of the RTC connectors.
Return value: Selected input lines. For details, refer to "Command #25 – Read the RTC Input Lines" on page 25.

public int GetInputPinsAsInt(int connector)

Same function as GetInputPins(int connector), but returns an integer.

public int SetScannerPosition(double xpos, double ypos, double zpos)

Parameters: xpos, ypos and zpos define the new position of the scanner.
Return value: 1 = success, 0 = error.

public int ChangeConfiguration(string configName)

Parameter: configName is the name of the hardware configuration parameter set defined in laserDESK.

Return value: 1 = success, 0 = error.

Remark: Changes temporarily the applied hardware configuration. The selected configuration must be predefined in laserDESK. For details, refer to "Command #27 – Change the Hardware Configuration" on page 25.

public int SetGlobalTransformationMatrix (double M11, double M12, double M21, double M22, int permanent)

Parameter:
M11, M12, M21, M22 are the matrix elements of the transformation to be applied. permanent defines the time of duration. Allowed values are:

permanent = 0: In manual mode the transformation is only applied for the next marking. If a marking process is executed, the command will be refused. In automatic mode it is applied until the automatic mode is switched off.

permanent = 1: The transformation is applied until this function is used again or the hardware configuration is changed (new RTC initialization).

If in automatic mode this command is sent during an execution, the transformation is applied not until the next execution.

Remark: This transformation is additional to the transformation defined by the SetTransformation() function. This function enables scaling, skewing and rotation. An offset has to be executed by SetTransformation().

public int SetScanHead(int selectedhead)

Parameters: selectedhead: 1 = Scan head 1, 2 = Scan Head 2, 3 = both Scan Heads.

Function: Defines the active scan head(s).

Return value: 1 = success, 0 = error.

Remarks: The RTC option 'Second Scan Head' must be activated and the second scan head has to be enabled and defined in the laserDESK hardware configuration.
The command will be refused if the automatic mode is switched on or a job execution is in progress.

public int ExecuteASC(int typeNr)

Parameters:
typeNr:0 = Detect reference positions for ASC
1 = execute ASC (Automatic Self Calibration)
2 = Reset and Switch off ASC
4 = Check for ASC available on the system

Return value: 1 = success / ASC available, 0 = error / no ASC present.
The ASC is executed for the selected scan head (by command SetScanHead). If both scan heads are selected only the first one will execute the ASC.

public int SetMarkingParameter(string parameterName)

Parameter:
parameterName is the name of the marking parameter set defined either in the actual loaded laserDESK job (local) or in the marking library (global lib).

Return value: 1 = success, 0 = error.

Remarks: The job marking parameters are exchanged by the defined parameter set. All objects inside the job which inherit the job parameter will use this parameter set, too. This change is temporary until the job is saved.
This command can only be executed by laserDESK version 1.0.8.1 and higher.

public int ImportVectorGraphic(string fileName, double xPos, double yPos, double width, double height, string UID, int optionFlags)

Parameters:
filename: The full qualified path to the vector graphic file.
xPos: x position of the origin (center) of the graphics in [mm].
yPos: y position of the origin (center) of the graphics in [mm].
width: size of the graphics in x in [mm].
height: size of the graphics in y in [mm].
UID: Object UID in the laserDESK job of the group which should be replaced.
optionFlags: Possible flag values are shown in Table 11 on page 27.

Function: Imports a vector graphic file into the job. The answer is returned when the remote command has been executed (synchronous operation).

Return value: 1 = success, 0 = error.

Remarks: The command imports a vector graphic file as a new group if the UID is an empty string or replaces an existing group when the UID matches. If a group is replaced, the marking and hatching parameter are taken over. A new group will get the job parameter set but as an individual one and not inherited. Therefore it can't be changed later on by the remote function SetMarkingParameter.

If the width or height is zero or negative they are adapted proportional to the other value. If both are zero the file values will be used as [mm]. If additional the position (center) should not be set by this command use flag 'enValues1to1' = 0x10.
If the flag 'enScaleFactors' is set, first the vector file will be imported using the unit information which is included in the file. Hint: The default unit for plt-files is [0.025 mm] for all other file types [1 mm]. This unit is used for the scaled import if no unit information is present. To import a plt-file which has [mm] units you can either use the 'enValues1to1' flag or use scaled import with scaling factor 40. Only the latter allows to set a different origin.
After the import the whole graphics will be scaled by the given factors. This is different to the 'enValues1to1' flag where no unit information is used but the file values are taken as [mm] values.
The 'enScaleFactors'-flag allows to import a vector graphic with known measure and unknown size to be imported with the correct size. Negative scale factors are ignored.
If the 'enValues1to1' flag is set, 'enScaleFactors' will be ignored.
This command can only be executed by laserDESK version 1.2.0.1 and higher. Lower versions return 0.

public int ImportVectorGraphicNoWait (string fileName, double xPos, double yPos, double width, double height, string UID, int optionFlags)

That's the same function than 'ImportVectorGraphic' except it is executed as an asynchronous operation. The answer is returned immediately. The success of this command has to be detected by status queries.
Return value: 1 = command in execution, 0 = command refused.
Remarks: While this command is executed, only status queries are allowed. All other commands will be refused.
This command can only be executed by laserDESK version 1.2.0.1 and higher. Lower versions return 0.

public double GetExecutionTime()

Parameter: None.
Function: Returns the execution time of the last execution in [ms].
Remark: This command can only be executed by laserDESK version 1.2.0.1 and higher.

public string GetMarkingParameterSets()

Parameter: None.
Function: Returns the names of all available marking parameter set names. They may either be defined in the actual loaded job or in the program library. The names are separated by a comma and a space character.
Remark: This command can only be executed by laserDESK version 1.2.0.1 and higher.
Hint: You can use each of the set names as a parameter for the method SetMarkingParameter(string parameterName).

public int HideProgramGUI(int hide)

Parameter: hide:
Value 1 (or positive) hides the laserDESK program GUI and sets a notification icon in the system tray (task bar).
Value 0 (or negative) shows the program GUI (if hidden) and removes the notification icon of the system tray.
Remark: This command can only be executed by laserDESK version 1.2.0.1 and higher.

public int SetPowerScaleFactor(double factor, int permanent)

Parameters:
factor: Scaling factor of the laser power control.
permanent:
Value = 1 (or positive): factor is permanent set and stored in the hardware configuration (page Processing Laser). The allowed range is $0 < \text{factor} \leq 1$.
Value = 0 (or negative): factor is temporary set and will be reset when the remote mode is switched off or the hardware configuration is changed. The allowed range is $0 < \text{factor} \leq 1/(\text{permanent factor})$.
Function: This command changes the calibration factor of the laser power. The temporary settings are additional to the permanent settings. During execution a range checking will be applied. The resulting scale factor will not be larger than the maximum power scaling factor defined in the laser definition file. In case it will be clipped.
Remark: This command can only be executed by laserDESK version 1.2.0.2 and higher.

public int SetVisionJobParameter(string UID, string VisionJobName, double xOffset, double yOffset);

Parameters:
UID: UID of the vision system element in the laserDESK job.
VisionJobName: Name of the SCANalign vision job to be executed. This job has to be created in SCANalign first.
xOffset, yOffset: Translation of the vision job in x and y.
Return value: 1 = success, 0 = error.
Function: This command defines the parameters Camera Job and Translation of the vision system element with the defined UID. The vision job has to exist in *SCAN*align and is not checked by this command.
Remark: This command can only be executed by laserDESK version 1.2.0.3 and higher.

public int SetMarkingParameterOfLayer(string pParameterName, string pLayerName)

Parameters:
pParameterName: Name of the marking parameter set. The parameter set has to exist either in the job or the program library.
pLayerName: Name of the selected layer. The layer has to be defined in the job.
Return value: 1 = success, 0 = error.
Function: All objects which belong to the selected layer get the defined marking parameter set. If it is a parameter set of the program library it will be copied into the job. It doesn't matter which parameter set the object had (individual or inherited) it will be replaced. If either the parameter set or the layer doesn't exist the return value is 0.
Remark: This command can only be executed by laserDESK version 1.2.0.6 and higher.

public string GetLayerNamesOfJob()

Parameter: None.
Function: Returns the names of all layers defined in the job. The names are separated by a comma and a space character. If the command fails an empty string is returned.
Remark: This command can only be executed by laserDESK version 1.2.0.6 and higher.
Hint: You can use each of the layer names as a parameter for the method SetMarkingParameterOfLayer(string pParameterName, string pLayerName).

public int AxisControllerCommand(string AxisName, int CommandType, double Position, int RefRunType)

Parameters:
AxisName: Name of the motor axis as defined in the Hardware Configuration of laserDESK.
For command type number 5 use an empty string or "ALLAXIS".
CommandType: Type of the axis command to be executed are shown in Table 12 on page 29.
The command type 5 corresponds to the function of the reference run button in the status bar. All axes which are selected in the Hardware Configuration by checking the 'automatic reference run' checkbox are commanded to do the reference run. The values of the Hardware Configuration are used.
This function is available for versions 1.4.0.6 and higher.
Position: If the CommandType is 2 or 3 this parameter defines the new position in units of the calibration factor of the axis defined in laserDESK. Else this parameter is ignored.
RefRunType: Defines the kind of reference run to be executed. Please check the axis controller manual which type the axis can really execute. This parameter is only used if CommandType is 1. The types which are possible are shown in Table 12 on page 29.
Return value: 1 = success, 0 = error.
Function: The selected axis executes the defined action. The answer is returned immediately and the return value signals only the acceptance of the command. The successful execution has to be checked by status queries. If the command can't be executed (e.g. because the axis is not defined) the remote state error flag is set. If the motor itself can't execute the command the axis state returns the error.
Remark: This command can only be executed by laserDESK version 1.4.0.2 and higher.

public int GetAxisState(string AxisName)

Parameter:
AxisName: Name of the motor axis as defined in the Hardware Configuration of laserDESK.
Function: Returns the axis state as a bit field. If the command fails -1 is returned. The state is given by a bit field shown in Table 14 on page 30.
Not all axis controller can return all status information listed here. Please check the axis controller manual which status information is possible. In case of a stepper motor controlled by the RTC only one end switch state is available and the bit states 0x08, 0x40 and 0x40000 are not available.
Remarks: This command can only be executed by laserDESK version 1.4.0.2 and higher.

public int SetPOFTransformation(double xshift, double yshift, double rotation)

Parameters:
xshift, yshift and rotation define the transformation of the whole virtual image field.
Return value: 1 = success, 0 = error.
Remarks: This method sets a global transformation and is applied only during a job execution with activated (Processing-on-the-fly) POF. This transformation is additional to the transformations defined by the remote commands SetGlobalTransformationMatrix and SetTransformation(Ex).
For details, refer to "Command #42 – Set POF Field Transformation".
This command can only be executed by laserDESK version 1.4.0.2 and higher.

public int Set3DTransformation(double xshift, double yshift, double zshift, double M11, double M12, double M21, double M22)

Parameters:
xshift, yshift and zshift define the 3D offset of the marking, the Mij values are the matrix coefficients for the get_matrix command of the RTC.
Return value: 1 = success, 0 = error.
Function: This command applies a 2D transformation (rotation and scaling) and an additional 3D shift of the whole job.
Remarks: This command is only applicable in automatic mode. It combines the functions SetGlobalTransformationMatrix with parameter permanent set to false and SetTransformation with parameter rotation set to 0°. The rotation can be defined by appropriate matrix coefficients.
The transformation parameters of the three commands SetTransformation, SetGlobalTransformationMatrix and Set3DTransformation share the same values and each will update them.
The transformation will be applied on the next start and remains valid until the automatic mode is switched off. Only the z-offset remains valid for further manual starts and has to be reset either by this command or by an RTC reinstallation.
This command is intended for a system with a vario*SCAN* and needs the 3D option of the RTC board.
This command can only be executed by laserDESK version 1.4.1.0 and higher.

public int Set2ScanHeadTransformation(double xshift1, double yshift1, double M11_1, double M12_1, double M21_1, double M22_1, double xshift2, double yshift2, double M11_2, double M12_2, double M21_2, double M22_2)

Parameters:
Xshift and yshift define the offset of the marking, the Mij values are the matrix coefficients for the get_matrix command of the RTC. The trailing number (1 or 2) defines the scan head for which the parameter is valid.
Return value: 1 = success, 0 = error.
Function: This command applies an individual 2D transformation and x,y - offset for each scan head.
Remarks: This command is only applicable in automatic mode. It combines the functions **SetGlobalTransformationMatrix** with parameter *permanent* set to false and **SetTransformation** with parameter *rotation* set to 0°. The rotation can be defined by appropriate matrix coefficients. The transformation will be applied on the next start and remains valid until the automatic mode is switched off.
The transformation parameters of the three commands **SetTransformation**, **SetGlobalTransformationMatrix** and **Set2ScanHeadTransformation** share the same values and each will update them.
This command is intended for a 2 scan head system and needs the second scan head option of the RTC board.
This command can only be executed by laserDESK version 1.4.1.0 and higher.

public int SelectRTCCard(int CardType, int SerialNr)

Parameters:
CardType defines the type of the RTC which should be acquired. Possible values are:
1 = RTC5
2 = RTC6
3 = RTC6 Ethernet
SerialNr is the serial number of the selected card.
Return value: 1 = ok, 0 = error.
Function: This command acquires and initializes the defined card and uses this card for all following actions.
It's a non-blocking command. That means the return value is 1 (= ok), if the command is fine and will be executed. The return value 0 means, the command can't be executed.
During the initialization of the selected card the flag RM_STATE_RTC_RELEASED will be set.

If the defined card is not available this function will return 1 (= ok), but the status flag RM_STATE_CMD_ERR will be set. That is especially true if the serial number does not match. Therefore it is strongly recommended to check the successful execution by status queries.
After the successful execution the flag RM_STATE_RTC_RELEASED will be reset and no error flag RM_STATE_CMD_ERR is set.
The command will not initialize and adapt the whole hardware. Therefore it is strongly recommended to use the ChangeConfiguration function to set the appropriate hardware configuration.

public int ConvertVectorToPoints(string UID, double Distance, int MarkingTime, bool EqualDistance)

Parameters:
UID: The Unique Identifier (UID) of the element which should be converted.
Distance: The distance in [mm] between the points
Marking Time: The time duration the laser is switched on for each point.
Equal Distance: If false, the defined distance will not be adapted. In that case normally the last point will have a different distance (remainder) than all the other points.
If true, the points on the polygon edges will be shifted to have equal distances. For interpolated curves this calculation is not exact because of rounding errors during interpolation. This calculation is only executed for vectors longer than 5 times the point distance to keep the distance variation lower than 20%.
Return value: 1 = ok, 0 = error.
Function: This command converts a specified group or object into a list of points with a defined distance. In addition, the time duration the laser will be switched on to mark each point will be set.
It's a non-blocking command. That means the return value is 1 (= ok), if the command is fine and will be executed. The return value 0 means, the command can't be executed. During the execution of this command the flag RM_STATE_JOB_LOAD will be reset. If an error occurs, the flag RM_STATE_CMD_ERR will be set.
The point list is integrated in a graphics paths group. This group replaces the selected vector element or group.
Remark: The command is not applicable for the job node, variants or bitmaps.

public int SwitchLaserBeam(double Power, double Frequency, int PulseLength, double OnTime)

Parameters:
Power: The laser power in [%].
Frequency: The laser frequency in [kHz]; ignored for CO_2 lasers (laser mode 0).
PulseLength: Length of the RTC Laser1 signal.
OnTime: Time span the laser beam will be switched on.
Function: This command will switch the laser beam on for the defined time using the given laser parameters. During this time an RTC list is running and can only be stopped using "Command #13 – Emergency Stop".
If the *OnTime* is negative, the laser beam is switched on infinitely. In that case no list is running and you can use this command with *OnTime* set to zero to switch the laser beam off again.

public int RequestHeadParameter(int HeadNr, int selection, ref double par1, ref double par2)

Parameters:
HeadNr: Head selection (1 or 2)
Selection: Requested parameter type. Possible values are.
1 = Status (return value)
2 = Position (x in par1, y in par2)
3 = Current of galvos (in [mA])
4 = Temperature of galvos (in [C°])
Function: This command returns an actual scan head parameter to check his state. All requests except the scan head state will be returned as two double values for both galvos of the scan head. For the position request the first double is the x position the second the y position.
The state is returned as a bitfield as defined in the RTC manual for the control_command.
The return value 0 is due to an error, maybe no i*DRIVE* system is connected.

public int ReplaceSpecificParameter(string UID, int ParameterID, double Value, string ParameterSetName)

Parameters:
UID: Defines the element UID in the job where to replace the parameter.
ParameterID: Specifies the parameter, which should be replaced (see Table 15 – Table 17 on page 33 ff).
Value: New value of the parameter (see Table 15 – Table 17 on page 33 ff).
ParameterSetName: If this value is null or an empty string, the parameter set of the object becomes individual and then the parameter gets the new value. Else a new parameter set is created and stored in the

program library. Additional the new set is assigned to the element. If the name corresponds to an existing set in the job library it will be overwritten.

public int SliceSTLFile(string filename, double length, double width, double height, double xCenterPos, double yCenterPos, double zBasePos, double slices, int options, double angleOffset, int digOut, int digIn)

Parameters:
filename: Full path to the stl file.
length, width, height: Dimension of the body given by the stl file (in [mm]). If all three parameters are larger than 0, the body will be stretched to fit all dimensions. Else the body will be scaled to the first value larger than 0.
xCenterPos, yCenterPos: Defines the center position of the imported body.
zBasePos: Defines the z-position of the first layer (bottom for ascending order, top for descending).
slices: Dependent on the distance flag it is the total number of the slices (flag cleared) or the distance between two slices (flag set).
options: Flags, which control the import behavior:

Table 22

Flag	Meaning
0x00000001	Apply filling
0x00000002	Calculate slices descendent
0x00000010	Insert z-focus control nodes
0x00000020	Create layers
0x00000040	Create job for each slice
0x00000080	Upper end switch signals
0x00000100	Emergency break signals

angleOffset: If the 'Apply filling' - flag is set the hatching of each slice will be rotated by this value.
digOut: If the signal exchange is inserted, this value defines the digital output line on Extension1 of the RTC. Allowed values are 0–7.
digIn: If the signal exchange is inserted, this value defines the digital input line on Extension1 of the RTC. Allowed values are 0–7.
Function: This command imports an stl-file and cuts it into parallel slices for 2.5D processes (e.g. deep engraving or additive manufacturing).
Several options define the resulting laserDESK job(s). The option flags can be combined.

- If flag 0x0001 (Apply filling) is set, the filling parameters of the job node are used and the angleOffset parameter is applied. You may use the remote function R**eplaceSpecificParameter** (see function before) to define appropriate filling parameters prior to the slicing.
- If flag 0x002 (Calculate slices descendent) is set, the slices are calculated from top to bottom. This is especially useful for deep engraving applications.
- If flag 0x0010 (Insert z-focus control nodes) is set, z-focus control nodes are inserted into the job instead of setting the z-value of each polygon point.
- If flag 0x0020 (Create layers) is set, the polygons of each slice are inserted in an own layer labelled "Layer ***n***". The z-values will be a layer property and not a polygon point value.
  If additional flag 0x0010 is set, the z-value is defined by the z-focus node and not by the layer.
- If flag 0x0040 (Create job for each slice) is set, each slice is saved in an own job. The job name consists of the actual job name with the extension "_Leveln.sld" where n is the number of the slice.
- If flag 0x0080 (Insert signal exchange) is set, a digital input and output line of the RTC Extension1 connector is used. Before each slice an output signal is set to allow external handling. Then the program waits for an input signal to go on with the next slice.
- If flag 0x0100 (Distance flag) is set, the slices parameter will be taken as the distance of the slices, not as the amount of the calculated slices.

Each slice is saved in a group where the group UID is "Slice***#n***". Each calculated polygon gets the UID "Slice***#n***_Polygon***#k***" where n is the slice number and ***k*** is the polygon number in the slice. This naming convention allows identifying the individual objects by other remote commands.
If any UID is already existent in the job the values may change and cannot be exactly predicted anymore. So we recommend starting with an empty new job if the UID is needed for further remote commands.
The command is executed as an asynchronous operation. The answer is returned immediately and the successful execution has to be detected by status queries. During the calculation the status flag RM_STATE_JOB_LOAD will be reset. It will be set again when the slicing calculation has finished.
If an error occurs the RM_STATE_CMD_ERR flag will be set.

public int SetMarkingCount(string UID, int MarkingCount)

Parameters:
UID: UID of the graphic element, where the marking count should be set.
MarkingCount: The new value of the marking count. Only positive values and zero are allowed.

public int SetControlNodesTransformation(int[] ID, double[] xshift, double[] yshift, double[] rotation)

Parameters:
ID: Array of the IDs of the transformation control nodes in the job which should be changed.
xShift: Array of the x-offsets of the transformations.
yShift: Array of the y-offsets of the transformations.
rotation: Array of the rotation angles of the transformations.

Function: This command replaces the transformation parameters of the transformation control nodes. It may have several transformation blocks to change the correspondent transformation control nodes together.
A transformation block contains an ID, an x, y-offset and a rotation and is defined by the array index. The ID defines the transformation control node(s) to which the transformation should be applied. Therefore, all arrays must have the same length.

public int SetTimeout(int timeout)

Parameter:
Timeout: Waiting time in [ms] to receive an answer. If no answer is received in this time the methods will return 0 and the timeout error flag will be set (see method GetTransmissionError). The default is 5 s.

public int GetTransmissionError()

Parameter: None
Function: Returns the transmission error that occurred due to timeout or command mismatch. Timeout error is indicated with 0x01 and command mismatch error is indicated with 0x02. The error is automatically cleaned once the user uses this command to check the transmission error. If a second error occurs before the user reads the error, both errors are displayed (for ex. 0x03 if both time-out and command mismatch errors have occurred).

public int SetExecutionTimeout(int timeout)

This timeout is only applied for the long running commands OpenJob, SwitchAutomaticMode and ImportVectorGraphic (see pages 44, 46, 47).
Parameter:
Timeout: Waiting time in [ms] to allow the command to finish successfully. If no answer is received within this time, the methods will return 0 and the timeout error flag will be set (see method GetTransmissionError on (see pages 53). The default is 30 s.
If this time is set too short, it may happen that the command execution inside laserDESK continues. This command will not cancel it. Further commands of the remote dll can only be sent if no blocking occurs within the dll.

This value is only an approximation of the [ms], because it depends strongly on the network traffic. You should define a value long enough for a normal import needed for the graphics used.

7.3 Status Flag Values

The table below shows the status flag values. These values are defined as 'public const uint' in the TelegramBuilder class. For the usage, see the "Programming Example in C#" on page 54.

Table 23

RM_STATE_NO_INIT	0x00000000
RM_STATE_WND_OPEN	0x00000001
RM_STATE_RTC_INIT	0x00000002
RM_STATE_LAS_INIT	0x00000004
RM_STATE_MOT_INIT	0x00000008
RM_STATE_ALL_INIT	0x0000000F
RM_STATE_READY	0x00000010
RM_STATE_AUTOMODE	0x00000020
RM_STATE_LST_EXEC	0x00000040
RM_STATE_LST_EXE_ERR	0x00000080
RM_STATE_RM_MODE	0x00000100
RM_STATE_JOB_LOAD	0x00000200
RM_STATE_LST_CALC	0x00000400
RM_STATE_CMD_ERR	0x00000800
RM_STATE_LAS_ERR	0x00001000
RM_STATE_LAS_ON	0x00002000
RM_STATE_DEV_ERR	0x00004000
RM_STATE_HEAD_OK	0x00008000
RM_STATE_EXEC_DONE	0x00010000
RM_STATE_PILOT_MODE	0x00020000
RM_STATE_RTC_RELEASED	0x00040000
RM_STATE_JOB_ABORTED	0x00080000
RM_STATE_SWITCH_AUTOMODE	0x00100000
RM_STATE_AXIS_EXECUTION	0x00200000
RM_STATE_DEV_RUNNING	0x00400000

7.4 Programming Examples

Programming Example in C#

```
// define class variable
public SLLDRemoteControl.SLLD_Functions LDRe-
moteFunc;
// Create class instance
LDRemoteFunc = new SLLD_Functions();
// open connection to laserDESK (here: using
loopback address)
byte[]  cadr = new Byte[4];
cadr[0] = 127;
cadr[1] = 0;
cadr[2] = 0;
cadr[3] = 1;
int i = LDRemoteFunc.OpenConnection(cadr,3000);
// execute remote functions (open and start job)
int ret;
ret = LDRemoteFunc.SwitchRemoteMode(true);
ret = LDRemoteFunc.OpenJob(@"C:\jobs\Test.sld",
false);
ret = LDRemoteFunc.StartMarking();
while ((LDRemoteFunc.GetState() & Telegram-
Builder.RM_STATE_EXEC_DONE) == 0)
{ System.Threading.Thread.Sleep(10); }  // wait
until execution is finished
ret = LDRemoteFunc.SwitchRemoteMode(false);
// close connection
LDRemoteFunc.CloseConnection();
```

Programming example in C++

(using SLLDRemoteControl as a COM component)

```
// ################ COM-class has to be registered (using Regasm.exe)   ##########
// Regasm.exe creates the tlb file (SLLDRemoteControl.tlb) needed for the import into
// your C++ project.
// During the tlb-Import one defines the class name of the C++ class and selects the
// interface (here _SLLD_Functions). There may be more than one interface.
// Inside the slldremotecontrol.tlh the original C# class name of the COM interface is
// available (in case of SLLDRemoteControl.dll it is SLLD_Functions) and denotes the
// needed CLSID
HRESULT hres;// = CoInitialize(NULL); should already be done
intret;
byte *ip =new byte[4];
ip[0] = 127;// loopback address
ip[1] = 0;
ip[2] = 0;
ip[3] = 1;

long port = 3000;// default port of laserDESK
// Create SafeArray for OpenConnection()
SAFEARRAY** psa;
VARTYPE vt = VT_UI1;
SAFEARRAY* sa = SafeArrayCreateVector(vt, 0, 4);
psa = &sa;
BYTE *pData;
hres = SafeArrayAccessData(sa,reinterpret_cast<void**>(&pData));
memcpy(pData, ip, 4);

// Create class instance of the COM interface class
CLSID clsid = __uuidof(SLLD_Functions);// class ID, see .tlh file
REFIID refiid = __uuidof(_SLLD_Functions);// interface ID, see .tlh file
_SLLD_Functions *fptr; // pointer to the interface
hres = CoCreateInstance(clsid, NULL,CLSCTX_INPROC_SERVER, refiid,(LPVOID *)&fptr);

// use the interface functions
long result = fptr->OpenConnection(psa, port);
unsigned int state = fptr->GetState();
bstr_t version = fptr->GetVersion();
WCHAR *lVersion = version.GetBSTR();
int ok = m_fptr->SwitchRemoteMode(true);

// add here your specific code to control laserDESK remotely
fptr->SwitchRemoteMode(false);
fptr->CloseConnection();

// clean up and free resources
    SafeArrayUnaccessData(sa);
SafeArrayDestroy(sa);
sa = NULL;

}
```

8 Revision History

This list ignores spelling/grammar corrections, syntax reformulations or additions (such as new cross-references) that don't affect the technical meaning. The supplied page numbers refer to this document.

Changes from revision 1.1 to revision 1.2

Page	Name of section / command	Change
12	chapter 3.4, "Commands"	Added: Command number 27
17	chapter 3.5, "Data Structure of the Answer Returned by laserDESK"	Added: Command number 27
23	chapter 4.19, "Command #19 – Set Output Signals of the RTC"	Corrected: Example / table
25	chapter 4.27, "Command #27 – Change the Hardware Configuration"	Chapter added
41	chapter 5.4, "List of Requirements to Execute the Commands"	Added: Command number 27

Changes from revision 1.2 to revision 1.3

Page	Name of section / command	Change
12	chapter 3.4, "Commands"	Added: Command number 28
17	chapter 3.5, "Data Structure of the Answer Returned by laserDESK"	Added: Command number 28
23	chapter 4.19, "Command #19 – Set Output Signals of the RTC"	Corrected: "The first character of the string corresponds to the digital output line **7**".
25	chapter 4.28, "Command #28 – Execute global matrix transformation"	Chapter added
41	chapter 5.4, "List of Requirements to Execute the Commands"	Added: Command number 28

Change from revision 1.3 to revision 1.4

Page	Name of section / command	Change
19	chapter 4.4, "Command #4 – Query for Program State"	Added: Description of bit 14

Change from revision 1.4 to revision 1.5

Page	Name of section / command	Change
12	chapter 3.4, "Commands"	Added: Command number 29
17	chapter 3.5, "Data Structure of the Answer Returned by laserDESK"	Added: Command number 29
26	chapter 4.29, "Command #29 – Set active Scan Head"	Chapter added
41	chapter 5.4, "List of Requirements to Execute the Commands"	Added: Command number 29

Change from revision 1.5 to revision 1.6

Page	Name of section / command	Change
5	chapter 1, "Introduction"	Added: Note
19	chapter 4.4, "Command #4 – Query for Program State"	Added: bit 20 and description of bit 20
21	chapter 4.7, "Command #7 – Open Job File"	Added: Notes
24	chapter 4.21, "Command #21 – Switch Automatic Mode On"	Added: Notes

Change from revision 1.6 to revision 1.7

Page	Name of section / command	Change
12	chapter 3.4, "Commands"	Added: Command number 30
17	chapter 3.5, "Data Structure of the Answer Returned by laserDESK"	Added: Command number 30
26	chapter 4.30, "Command #30 – Execute Automatic Self-Calibration"	Chapter added
41	chapter 5.4, "List of Requirements to Execute the Commands"	Added: Command number 30

Change from revision 1.7 to revision 1.8

Page	Name of section / command	Change
11	chapter 3.4, "Commands"	Modified: Command number 16
16	chapter 3.4, "Commands"	Modified: Command number 20
22	chapter 4.16, "Command #16 – Set Coordinate Transformation in Automatic Mode"	Chapter modified
23	chapter 4.20, "Command #20 – Set Coordinate Transformation in Manual Mode"	Chapter modified

Change from revision 1.8 to revision 1.9

Page	Name of section / command	Change
12	chapter 3.4, "Commands"	Added: Commands number 31, 32, 33, 34
17	chapter 3.5, "Data Structure of the Answer Returned by laserDESK"	Added: Commands number 31, 32, 33, 34
26	chapter 4.31, "Command #31 – Set Marking Parameter Set for Job"	Chapter added
26	chapter 4.32, "Command #32 – Load/Replace Vector Graphics"	Chapter added
27	chapter 4.33, "Command #33 – Get Execution Time"	Chapter added
28	chapter 4.34, "Command #34 – Get Marking Parameter Sets"	Chapter added
41	chapter 5.4, "List of Requirements to Execute the Commands"	Added: Commands number 31, 32, 33, 34

Change from revision 1.9 to revision 1.10

Page	Name of section / command	Change
13	chapter 3.4, "Commands"	Added: Commands number 35, 36
17	chapter 3.5, "Data Structure of the Answer Returned by laserDESK"	Added: Commands number 35, 36
28	chapter 4.35, "Command #35 – Hide or Show GUI of laserDESK"	Chapter added
28	chapter 4.36, "Command #36 – Set Power Scaling Factor"	Chapter added
41	chapter 5.4, "List of Requirements to Execute the Commands"	Added: Command number 35, 36

Change from revision 1.10 to revision 1.11

Page	Name of section / command	Change
13	chapter 3.4, "Commands"	Added: Commands number 37
17	chapter 3.5, "Data Structure of the Answer Returned by laserDESK"	Added: Commands number 37
28	chapter 4.37, "Command #37 – Set Vision System Parameter"	Chapter added
41	chapter 5.4, "List of Requirements to Execute the Commands"	Added: Command number 37

Change from revision 1.11 to revision 1.12

Page	Name of section / command	Change
13	chapter 3.4, "Commands"	Added: Commands number 38, 39
17	chapter 3.5, "Data Structure of the Answer Returned by laserDESK"	Added: Commands number 38, 39
19	chapter 4.4, "Command #4 – Query for Program State"	Added: Bit 0x00080000
28	chapter 4.38, "Command #38 – Set Marking Parameter Set for all Objects of a specific Layer"	Chapter added
28	chapter 4.39, "Command #39 – Get Layer Names of Job"	Chapter added
41	chapter 5.4, "List of Requirements to Execute the Commands"	Added: Commands number 38, 39

Change from revision 1.12 to revision 1.13

Page	Name of section / command	Change
13	chapter 3.4, "Commands"	Added: Commands number 40, 41
17	chapter 3.5, "Data Structure of the Answer Returned by laserDESK"	Added: Commands number 40, 41
29	chapter 4.40, "Command #40 – Motor Axis Control"	Chapter added
30	chapter 4.41, "Command #41 – Motor Axis Status Query"	Chapter added
41	chapter 5.4, "List of Requirements to Execute the Commands"	Added: Commands number 40, 41

Change from revision 1.13 to revision 1.14

Page	Name of section / command	Change
13	chapter 3.4, "Commands"	Added: Command number 42
17	chapter 3.5, "Data Structure of the Answer Returned by laserDESK"	Added: Command number 42
30	chapter 4.42, "Command #42 – Set POF Field Transformation"	Chapter added
42	chapter 5.4, "List of Requirements to Execute the Commands"	Added: Command number 42

Change from revision 1.14 to revision 1.15

Page	Name of section / command	Change
13	chapter 3.4, "Commands"	Added: Command number 43 and 44
17	chapter 3.5, "Data Structure of the Answer Returned by laserDESK"	Added: Command number 43 and 44
19	chapter 4.4, "Command #4 – Query for Program State"	Added in table: Bit 0x00400000
29	chapter 4.40, "Command #40 – Motor Axis Control"	Chapter modified; action 5 added
30	chapter 4.43, "Command #43 – Set Transformation for a 3D System"	Chapter added
31	chapter 4.44, "Command #44 – Set Transformation for a 2 Scan Head System"	Chapter added
42	chapter 5.4, "List of Requirements to Execute the Commands"	Added: Command number 43 and 44

Change from revision 1.15 to revision 1.16

Page	Name of section / command	Change
11	chapter 3.4, "Commands"	Extended: Command number 15
14	chapter 3.4, "Commands"	Added: Command number 45 and 46
16	chapter 3.5, "Data Structure of the Answer Returned by laserDESK"	Modified: Command number 15
17	chapter 3.5, "Data Structure of the Answer Returned by laserDESK"	Added: Command number 45 and 46
22	chapter 4.15, "Command #15 – Define the Serial Number Start, Actual or End Value"	Chapter extended
31	chapter 4.45, "Command #45 – Select RTC Board"	Chapter added
31	chapter 4.46, "Command #46 – Convert Vectors into Points"	Chapter added
40	chapter 5.4, "List of Requirements to Execute the Commands"	Modified: Command number 15
42	chapter 5.4, "List of Requirements to Execute the Commands"	Added: Command number 45 and 46

Change from revision 1.16 to revision 1.17

Page	Name of section / command	Change
14	chapter 3.4, "Commands"	Added: Command number 47
18	chapter 3.5, "Data Structure of the Answer Returned by laserDESK"	Added: Command number 47
32	chapter 4.47, "Command #47 – Activate Laser Beam"	Chapter added
42	chapter 5.4, "List of Requirements to Execute the Commands"	Added: Command number 47

Change from revision 1.17 to revision 1.18

Page	Name of section / command	Change
30	chapter 4.41, "Command #41 – Motor Axis Status Query"	Status bits changed and table with bit fields modified
40	chapter 5.4, "List of Requirements to Execute the Commands"	Requirements of command 21* modified

Change from revision 1.18 to revision 1.19

Page	Name of section / command	Change
5	chapter 1, "Introduction"	Added: Note on naming the RTC boards
12	chapter 3.4, "Commands"	Added: Command number 23 and 24
17	chapter 3.5, "Data Structure of the Answer Returned by laserDESK"	Added: Command number 23 and 24
24	chapter 4.23, "Command #23– Release RTC"	Chapter added
24	chapter 4.24, "Command #24– Acquire RTC"	Chapter added
26	chapter 4.32, "Command #32 – Load/Replace Vector Graphics"	Added in table and text: 'enScaleFactors' flag
41	chapter 5.4, "List of Requirements to Execute the Commands"	Added: Command number 23 and 24

Change from revision 1.19 to revision 2.0

Page	Name of section / command	Change
11	chapter 3.4, "Commands"	Added in table: Column with laserDESK version
19	chapter 4.4, "Command #4 – Query for Program State"	Added in table: Bit 0x00000000 Bit 0x00040000 Bit 0x00200000
26	chapter 4.32, "Command #32 – Load/Replace Vector Graphics"	Added in table: Flag 0x40
44	chapter 7, "SLLDRemoteControl.dll"	Chapter added

Change from revision 2.0 to revision 2.1

Page	Name of section / command	Change
14	chapter 3.4, "Commands"	Added: Commands number 50, 51, 53, 54
18	chapter 3.5, "Data Structure of the Answer Returned by laserDESK"	Added: Commands number 50, 51, 53, 54
32	chapter 4.50, "Command #50 – Get Scan Head specific Parameter"	Chapter added
33	chapter 4.51, "Command #51 – Set specific Parameter Value"	Chapter added
36	chapter 4.53, "Command #53 – STL File Import and Additional Slicing"	Chapter added
37	chapter 4.54, "Command #54 – Set Marking Count of an Element"	Chapter added
42	chapter 5.4, "List of Requirements to Execute the Commands"	Added: Commands number 50, 51, 53, 54
51	section "public int RequestHeadParameter(int HeadNr, int selection, ref double par1, ref double par2)"	Section added
51	section "public int ReplaceSpecificParameter(string UID, int ParameterID, double Value, string ParameterSetName)"	Section added
52	section "public int SliceSTLFile(string filename, double length, double width, double height, double xCenterPos, double yCenterPos, double zBasePos, double slices, int options, double angleOffset, int digOut, int digIn)"	Section added
52	section "public int SetMarkingCount(string UID, int MarkingCount)"	Section added
53	section "public int SetExecutionTimeout(int timeout)"	Section added

Change from revision 2.1 to revision 2.2

Page	Name of section / command	Change
36	chapter 4.53, "Command #53 – STL File Import and Additional Slicing"	Parameter [7] modified
52	section "public int SliceSTLFile(string filename, double length, double width, double height, double xCenterPos, double yCenterPos, double zBasePos, double slices, int options, double angleOffset, int digOut, int digIn)"	Parameter *zBasePos* modified

Change from revision 2.2 to revision 2.3

Page	Name of section / command	Change
15	chapter 3.4, "Commands"	Added: Command number 55
18	chapter 3.5, "Data Structure of the Answer Returned by laserDESK"	Added: Command number 55
22	chapter 4.16, "Command #16 – Set Coordinate Transformation in Automatic Mode"	Chapter extended
30	chapter 4.43, "Command #43 – Set Transformation for a 3D System"	Chapter extended
31	chapter 4.44, "Command #44 – Set Transformation for a 2 Scan Head System"	Chapter extended
37	chapter 4.55, "Command #55 – Set Transformation for Control Node"	Chapter added
42	chapter 5.4, "List of Requirements to Execute the Commands"	Added: Command number 55
53	section "public int SetControlNodesTransformation(int[] ID, double[] xshift, double[] yshift, double[] rotation)"	Section added

Change from revision 2.3 to revision 2.4

Page	Name of section / command	Change
34	chapter 4.51, "Command #51 – Set specific Parameter Value"	Table 16 and 17 extended