LTB
LASERTECHNIK BERLIN

Interface Manual

MNL 100

For MNL100 firmware version 2.61

released 01.07.2009

Oct 2014

LTB Lasertechnik Berlin GmbH • Am Studio 2c • D-12489 Berlin
Tel.: +49 (0)30 91 20 75-100 • Fax: +49 (0)30 91 20 75-199
E-mail: info@ltb-berlin.de • http://www.ltb-berlin.de

1 Interface description for MNL 100 lasers

This manual gives an overview about the two possibilities for using an interface to integrate the laser into other computer-run systems.

One possibility is to use the interface software DLL for Windows and the other one is the use of the serial bus protocol of the laser's internal controller.

The software is also used for different types of lasers (MSG, MINEX, LTX, OPTEX, MNL). Therefore, there are functions, which just refer to a certain laser type (and don’t make sense in your application)

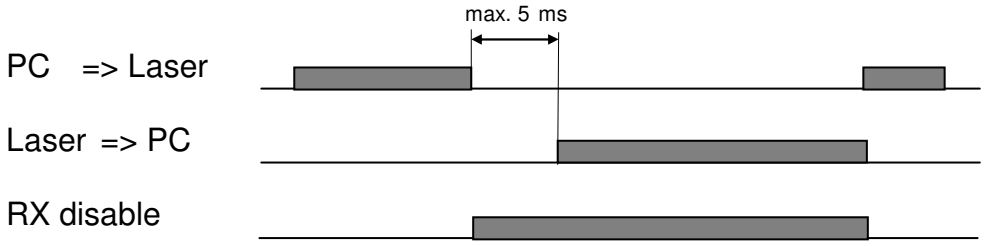
2 The bus protocol

By using the bus interface, the protocol-timing and the firmware-program routines need to be taken into account.

The interface is always ready for receiving data. The communication is delivered by the control computer, both, a transmitting and a receiving buffer are available.

After receiving a telegram, the receiving buffer is blocked and will be released after a response telegram was sent.
If the communication stops between the computer and the laser, the receiver is set back and the telegram in the receiving buffer will be deleted after one second. If there is an overflow in the receiving buffer, an error telegram will be sent and the receiving buffer will be deleted.
After a telegram was sent from the computer, the response telegram is sent back after five milliseconds.

RS232 Timing



After receiving a telegram, the telegram will be checked and forwarded for execution or an error telegram will be sent.
The process can just happen, if the laser is in a program loop, which accepts the command. False or invalid commands are rejected.
If there is no communication between the laser and the computer for more than 30 seconds, the laser will be switched into the standby mode.

The transmission rate comprises of 9600 baud, 1 Start-bit, 1 Stop-bit and no parities.

There are 4 different types of telegram: A ”Call-in” telegram, a ”Reply” telegram, an ”Error” telegram and an ”Acknowledge” telegram. All data are transmitted as ASCII characters.
Single characters are sent one after the other. If there is a break of more than one second between two characters, the receiver is set back and the receiving buffer is deleted. The receiving buffer at the laser side has a size of 14 bytes. It is filled with characters when the SD1 and DA is properly recognised. In case of too many dates, the receiver is set back and an error telegram will be sent.

The ”Call-in” telegram

SD1	DA	SA	ReqDataUnit	FSC	ED

e.g. : LASOn ‘#!@gEB(cr)’ HEX:(23H,21H,40H,67H,45H,42H,0DH)

The ”Reply” telegram

SD2	DA	SA	ResDataUnit	FSC	ED

e.g.:reply of GetStat8 : ‘<@!UU0000D91E21000000000000006467(cr)’

The ”Acknowledge” telegram ACK

ED

’(cr)’ HEX:(0DH)

ACK is transmitted in reply to the Call-in telegram, when there are no further data requests. Its value is 0DH (CR).

The ”Error” telegram

ESC	ESC	Errortype	FSC	ED

Errortype specifies the type of error:

'1' - Checksum error
'2' - incorrect format
'3' - incorrect parameter
'4' - forbidden error
'5' - busy error (preceding command is still processed)
'6' - TX-Buffer full

e.g.: ’(esc),(esc)46A(cr)’ HEX:(1BH,1BH,34H,36H,41H,0DH)

The Error telegram is sent as a reply, if the call-in telegram is recognised as being defective or if the laser determines that at present it cannot carry out the programmed function correctly.

e.g.: Set ’REPETITION ON’ during laser is off

SD1 : *Startdelimiter (1 byte) for the Call-in telegram*
The Startdelimiter1 characterises the start of the Call-in telegram.
Its value is'#' or 23H.

SD2 : *Startdelimiter (1 byte) for the Reply telegram*
The Startdelimiter2 characterises the start of the Reply telegram.
Its value is a '<' or 3CH.

DA : *Destination- Address (1 byte)*
The Destination-Address DA characterises the address of the communication partner, to whom the data is transferred or from whom the data is requested. The Destination-Address can assume a value between 20H and 0FFH. For the operation of a single laser a value of '!' or 21H is designated.

SA : Source-*Address (1 byte)*
The Source-Address SA characterises the address of the communication partner from whom the data is sent. The Source-Address can assume a value between 20H and 0FFH. For the operation of a single laser a value of '@' or 40H is designated.

ReqDataUnit : *Request-Data-Unit (RDU) (1..8 bytes)*
The Request-Data-Unit characterises a data field in the Call-in telegram, where the data for the communications partner is included with the address DA.

ResDataUnit : *Response-Data-Unit (RDU) (1..52 bytes)*
The Response-Data-Unit characterises a data field in the Reply telegram, where the data for the called-up communications partner is included with the address SA.

FCS : *Frame-Check-Sequence (2 bytes*)
The Frame-Check-Sequence FCS characterises the Check-sum of the telegram. It is the sum of the individual bytes (ASCII-Values) in the telegram modulo 256. It is made up of the Startdelimiter, the Destination-Address, the Source-Address and the Data Unit. The data is shown as a 2-digit ASCII-String (ASCII '00'..'FF').
Note: The FCS is displayed differently in the standard telegram.

ED : *Enddelimiter (1 byte)*
The Enddelimiter characterises the end of a telegram.
Its value is a 'CR' or 0DH.

It is important that the individual characters of a telegram are transmitted directly one after the other. If there is a break of more than one second between two characters the receiver will be reset and the Receiver Buffer will be erased. The Receiver Buffer on the laser side is 14 bytes in size. This will be filled with characters after an acknowledgement of the SD1 and the correct DA. In the case of an overflow, the receiver will be erased and an Error telegram transmitted.

2.1 The use of the Commands

The Commands are transmitted in the Request-Data-Unit (RDU). The Commands are divided into two different types those with parameter response and those without. A Reply telegram is transmitted to those with parameter response. Those commands without parameter response receive either an ACK or an Error telegram in reply. The contents of the RDU are pure ASCII-characters. Values in the form of figures are transmitted as ASCII-HEX-BYTE or ASCII-HEX-WORD.

ASCII-HEX-BYTE *(2 ASCII Characters)*
Defines a decimal value between 0 and 255 which is shown in hexadecimal format. The two hex-numbers are transmitted as ASCII-characters (the highest first).
e.g.: The decimal figure 43 is equal to 2B Hexadecimal. This results in two ASCII- characters: ’2’,’B’ => 32H,42H

ASCII-HEX-WORD *(4 ASCII Characters)*
Defines a decimal value between 0 and 65535 which is shown in hexadecimal format. The four hex-numbers are transmitted as ASCII-characters (the highest first).
e.g.: The decimal figure 8619 is equal to 21AB Hexadecimal. This results in four ASCII-characters: ’2’,’1’,’A’,’B’ => 32H,31H,41H,42H

Note: It is important to pay special attention to the capital letters when using the commands.

2.1.1 Commands without parameter response

Command	RDU	Description	Example(ASCII)
LASOff (LASER OFF)	X	The laser will be deactivated. The Repetition, Quantity, External trigger mode will be interrupted.	#!@XDC(cr)
LASOn (STAND BY)	g	The laser is switched in STANDBY. Before, the laser must be in 'READY' status. If this command was transmitted, the laser does not accept any further commands for more then 10 seconds. (Warning lamp is flashing.) The laser is turned off, if the communication with the PC is broken more than 30 seconds.	#!@gEB(cr)
Repetition (REPETITION ON)	h	Switches the laser into 'REPETITION'- mode. Before, the Laser must be in STANDBY .	#!@hEC(cr)
Quantity (BURST)	j	Switches the laser into 'QUANTITY'- mode (Burst). Before, the laser must be in STANDBY.	#!@jEE(cr)
ExtTrigmode (EXT-TRIGGER)	u	Activates the External triggering mode. The laser expects external trigger pulses. Before, the laser must be in STANDBY.	#!@uF9(cr)
Off (STOP)	i	Interrupts the Repetition, Quantity, External trigger. The remains in STANDBY.	#!@iED(cr)
SetQuantity	l*nnnn*	Sets the Quantity to the value of *nnnn*. *nnnn* is a ASCII-HEX-WORD.	#!@l03E8D0 (cr) quantity = 1000

SetFreq (REP.RATE)	m*nn*	Sets the Frequency for the repetition operation with a value of *nn*. *nn* is a ASCII-HEX-BYTE. The value of *nn* is restricted to the maximum possible repetition rate that the laser can generate.	#!@m0A62 (cr) rep.rate = 10
SetHV	n*nn*	Sets the value of the HV-Voltage in % steps. A value between 0 and 100 is allowed	#!@n3257(cr) HV=50%
IncHV	o1	Increments the value of the HV-Voltage at 1%	#!@o124(cr)
DecHV	o0	Decrements the value of the HV-Voltage at 1%	#!@o023(cr)
SetShutter	z*n*	Opens the Shutter if *n*=1, else shutter will be closed. This function is only working after fLasReady - bit is set	#!@z12F(cr) #!@z02E(cr)
SetStepperPosition	O3*nnnn*	Set the stepper position to the new setpoint *nnnn*. Only values between 0 and 399 are reasonable.	#!@O30064D0 (cr)
SetTransmission	O4*nn*	Sets the transmission for the attenuation unit. *nn* is the set-point in 0.5% steps. Max. value for *nn* is 200. transmission = *nn* / 2 [%]. Not all values are possible due to the attenuator-glass properties	#!@O46471(cr) Tr=50%
SetAttenuationEnergy	O5*nnnn*	Sets the output energy for the attenuation unit. *nnnn* is the set-value for scaling see **GETVER3** actual: Energy[µJ]=*nnnn*/64000*250 Not all values are possible due to the attenuator-glass properties	#!@O53200CD(cr) E = 50µJ
InitAttenuator	O60000	Reinitialization of the attenuator. The attenuator is searching his indexpoint and then going to the setpoint.	#!@O60000C9(cr)

2.1.2 Commands with a parameter response

These commands are answered by a Reply telegram. They serve to determine the laser status and can also be transmitted in Off line. The Call-in telegram only has one byte (command byte) in the Request-Data-Unit (RDU). The Destination Address (DA) of the Reply telegram is filled with that of the sender from the Call-in telegram. Correspondingly the DA of the Call-in telegram can be found in the Source Address (SA) of the Reply telegram. In the Reply telegram, the first character in the Response-Data-Unit (RDU) is the command byte. On reception, this enables the simple identification of the individual messages. The incorrect receipt of a Call-in telegram is registered with an Error telegram.

2.1.2.1 GetShortStatus

(since Version 2.58)
RDU = 'W'

Reply:

RDU = W*aa*

aa - StatusFlagbyte see below

aa - StatusFlagbyte :

Bit	name	Meaning at 1
0	fLasOn	High voltage module is activated and will be switched on in conjunction with one of the following modes. (STAND BY)
1	fLasWorking	High voltage module is switched on (laser is working)
2		unused
3	fEE_Error	EEPROM error
4	fPemError	Energy monitor error occurred
5	fTempWarning	Temperature too high (> 48°C) warning
6	fStaticError	Static error
7	fOpError	Operation error: Laser must be switched off

2.1.2.2 GetStat 7

RDU = 'UT'

Reply:

RDU = UT*aabbooeeeeffiimmzz*

aabboo	- Flagbytes 1, 2, 3 see below
eeee	- represents the Quantity value pre-set in the laser
ff	- represents the Frequency value pre-set in the laser
ii	- represents regulated value of the High voltage within a certain operation range
mmmm	-unused
zzzz	- last measured energy value (single value)

e.g.:
HEX: 0x23,0x21,0x40,0x55,0x54,0x32,0x44,0x0D
ASCII: #!@UT2D(CR)
an answer from Laser could be:
<@!UT040003000A14320000000088 (CR)

2.1.2.3 GetStat 8

RDU = 'UU'

Reply:

RDU = UU*ppqqgghhnnkkkkxxxxyyyyyyyy*

ppqq	- Flagbytes 4,5 see below
gg	- represents the internal supply voltage (1LSB=0,11V)
hh	- represents the measured temperature value 2
nn	- represents the measured temperature value 1
kkkk	- represents energy value measured at the laser output (averaged value with τ = 20 shots) for scaling see **GETVER3**
xxxx	- Quantity counter value (2 bytes - > 4 ASCII figures)
yyyyyyyy	- Shot counter value (4 bytes - > 8 ASCII figures)

e.g.:
HEX: 0x23,0x21,0x40,0x55,0x55,0x32,0x45,0x0D
ASCII: #!@UU2E(CR)
an answer from Laser could be:
<@!UU00000022220000000000001154C4D (CR)

2.1.2.4 Meaning of the Flagbytes

aa - Flagbyte1:

Bit	Name	Meaning at 1
0	fShutter	Shutter is Open
1		unused
2	fLasReady	Laser is ready for operation. (High voltage module can be activated).
3	fLasOn	High voltage module is activated and will be switched on in conjunction with one of the following modes. (STAND BY)
4 - 7	fLaserMode	0000 – Off (switching off of all modes) 0001 - Repetition mode 0010 - Burst mode 0100 - External triggering mode

bb - Flagbyte 2:

Bit	Name	Meaning at 1
0 - 7		unused

oo - Flagbyte 3:

Bit	name	Meaning at 1
0	ftestmode	Service Mode activated
1	fFlowmode	unused (momentary always 1)
2	feregakt	unused
3	fRegEn	unused
4	fHeizfail	unused
5	fEE_Error	EEPROM error
6	fCPUError	Watchdog Reset occurred
7	fInspection	unused

pp - Flagbyte 4:

Bit	name	meaning at 1
0	fStaticError	Static error At least one error in bit 1 or 3 occurred
1	fOpen	Laser head chamber open (Enclosure)
2	fRemote	External interlock circuit open (Remote)
3	fTempLimit	Temperature too high (> 60°C) (internal interlock)
4	fTempWarning1	Temperature 1 too high (> 48°C) warning
5	fTemp Warning2	Temperature 2 too high (> 48°C) warning
6	fPemError	Energy monitor error occurred
7		unused

qq - Flagbyte 5:

Bit	name	meaning at 1
0	fOpError	Operation error: Laser must be switched off
1		unused
2		unused
3	fHVsupplyError	High voltage supply error or temperature error
4	fTempError1	Temperature error 1 (analogue sensor 1)
5	fTempError2	Temperature error 2 (analogue sensor 2)
6	fPowerSwitchError	Power switch is damaged
7	fPowersupplyWeak	Power supply is too weak

2.1.2.5 GetVer3

RDU = 'V3'

Reply:

RDU = V*uuvvääöö*wwwwwwwwwnnüü….üüü

This command delivers the data version of the laser software.

uu	represents the internal main revision byte. *(here 0xBD)*
vv	represents the Release byte. The PC software can therefore determine which control modules are supported.
ää	*Typebyte 1 (specifies certain functions of the laser)*
öö	*Typebyte 2 (specifies certain functions of the laser)*
wwwwwwww	program version as text (8 chars) e.g. 'RC002.50'
nn	*number for following chars*
üü….üüü	laser type as text e.g. MNL100 (nn chars)

GETVER
(older command, use GETVER3 instead)
GETVER2
(older command, use GETVER3 instead)
GETVERTXT
(older command, use GETVER3 instead)

vv - Release byte:

Bit	Name	Meaning at 1
0	fWithoutShutter	Shutter control is **not** supported
1	fAttenuation	Attenuation module is supported
2		unused
3	fhvst	High voltage control is supported
4-5	Laser type	00 - not in use 01 - MINex/LTX/OPTEX 10 - MSG **11 - MNL**
6	fEnMeas	Energy measuring is supported
7		unused

ää type 1

Bit	Name	Meaning
2/1/0	Pressure measuring range	unused
5/4/3	Energy measuring range	000 - Energy[mJ]=kkkk/10*2 001 - Energy[mJ]=kkkk/10 **100 - Energy[µJ]=kkkk / 64000 * 250 (MNL100)** 101 - Energy[µJ]=kkkk / 64000 * 500
6/7	Not in use	

öö type 2

Bit	Name	Meaning
2/1/0	Temperature measuring range	000 – temperature[°C]=(nn-92)/0.7599 001 - temperature[°C]=(nn-10)/0.8976 **010 - temperature[°C]= nn (MNL100)**
5/4/3	Not in use	
6	fAutoStart	Laser is switched in standby automatically
7	fAutoHvOn	HV is switched on automatically

2.1.2.6 GetSernum

RDU = 'US'

Reply:

RDU = USxxxxxxxxyyyy

This command delivers the serial numbers.

xxxxxxxx represents the laser serial number
yyyy represents the serial number for the internal energy monitor

2.1.2.7 GetAttenuatorStatus

RDU = 'UV'

Reply:

RDU = *UVaabbbbccccdd*

aa	Steppermode bit0 – Stepper is initialized bit1 – Stepper is in InitMode bit2 – higher current level is active bit7 – Stepper error (index not found)
bbbb	set point of stepper position
cccc	actual stepper position
dd	actual transmission in % = dd / 2 the output energy can be calculated with the attenuation and the energy value from GESTAT8

e.g.:

HEX: 0x23,0x21,0x40,0x55,0x56,0x32,0x46,0x0D
ASCII: #!@UV2F(CR)
an answer from Laser could be:
<@!UV0100000000018A(CR)

2.1.2.8 GetEnergyValues

(since Version 2.58)
RDU = 'P'

Reply:

RDU = *Paannxxxxyyyy….zzzz*

aa	count of energy values stored before read out
nn	count of following energy values
xxxx, yyyy, zzzz	energy values as 16 bit number. the oldest values will be transferred first The energy can be calculated as follow: E[µJ] = value * 250/64000 (MNL100)

The last measured Energy values are held in a FIFO - buffer with a size for 100 values. The buffer can be read out with this command. After read out, the read values will be removed from buffer. If the buffer is full, the oldest values will be overwritten. Not more than 35 values will be sent in one telegram. To prevent any loss of measured values, the buffer must be read cyclically. The parameter 'aa' can be used to calculate the exact point of time for measuring the first transferred value.

t = (actual time) – (transfer time) – (aa / pulse frequency)

3 The DLL Interface

This chapter gives some fundamental information about the interface software.

The MNL laser is controlled through an internal controller. An external controller (e.g. system controller, PC, Laptop or Notebook) is used for the input of instructions and the display of messages, values and operating states. The external controller and the internal controller communicate via the serial interface. More information about the communication between the external and internal controller are given in the bus protocol description. The interface software is used to create laser control programs (application programs) for a PC. It is stored on the PC and works between the application program and the serial interface to the MNL laser device. The interface software controls the communication via the interface. It checks the current laser state and provides this information on a certain storage area on the PC. Application programs, which communicate with the interface software, are able to retrieve the information and to send commands to the interface software. This user commands are translated into laser control commands and sent to the laser device via the serial interface.

The interface software can be run under

Windows 95, 98 and Windows NT/XP.

The interface software consists of two parts:

- **EXCIMER32.EXE**: a 32 bit program containing all functions to control the laser via the serial interface and report the laser device state continuously
- **EXCIM32.DLL**: a 32 bit DLL used for communication between the 32 bit applications and the EXCIMER32.EXE.

The applications transfer their command by calling the DLL routine “PUTCOMMAND”. The DLL temporarily stores this command and sends a message to the EXCIMER32.EXE (see Figure 1). EXCIMER32.EXE retrieves this command from the DLL and adds it to the command queue. This command is sent to the laser device as soon as the serial line to the laser is free. After adding the command to the queue and after receiving the acknowledgment from the laser device, EXCIMER32.EXE calls the DLL. The DLL sends a message to inform the calling application.

The laser status is independently requested by EXCIMER32.EXE via the RS232 interface. Every registered application is informed about the receipt of a new laser device state by a corresponding message.

3.1 The Interface structure

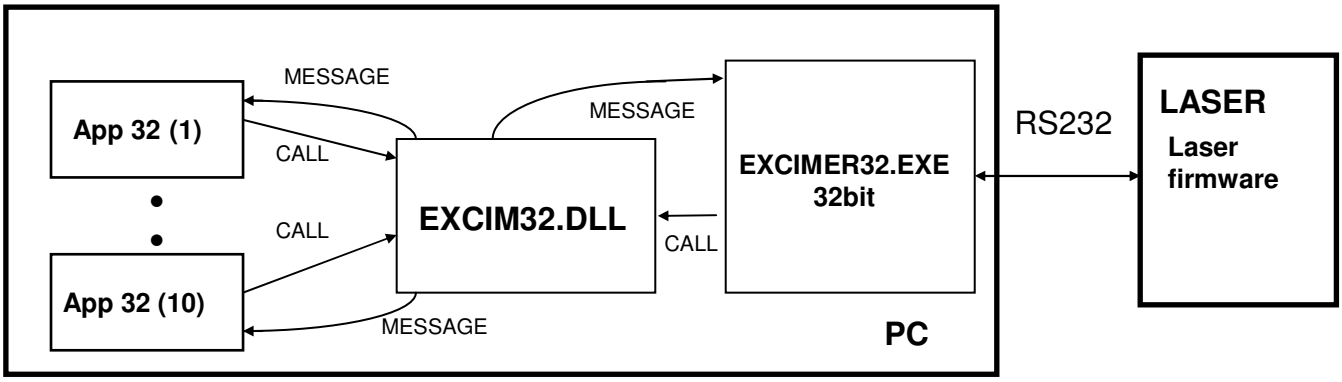


Figure 1: Command processing through the interface software DLL

3.2 The DLL-Functions

Using the interface software DLL, there are six call functions available:

– function registerapplication(han:thandle):longint;
– function releasedll(han:thandle):longint;
– function getmessagenumber:word;
– function getstatuspointer:pointer;
– function putcommand(prec:pointer):longint;

3.2.1 Registerapplication

Syntax: function registerapplication(han:thandle):longint;

Every application, which wants to communicate with the DLL, has to register with the DLL. Messages will only be sent to registered applications. The first ever call from Registerapplication automatically loads the EXCIMER.EXE program into the memory. The communication with the laser device via the serial interface is able to begin.
When calling the function, the application's Handle must be given to the DLL. The DLL then transmits the individual messages to this Handle in future.
THANDLE is a 32 bit value for the 32 bit DLL. In reply, the function returns a 32 bit value. If the function was successful, the value is 0. Otherwise, one of the error codes indicated below is returned:

Error	Errorcode(hex)	Description
err_appl_already_registered	$0101	the Handle given is already registered
err_appl_register_full	$0102	no further applications could be registered
err_dll_no_init	$0100	the DLL is not initialized

3.2.2 ReleaseDll

Syntax: function registerapplication(han:thandle):longint;

Using this function, an application is released from the DLL. When calling this function, the registered Handle (see above) has to be entered. After this call, no more messages will be sent to the Handle.
If there are no more registered applications, the EXCIMER.EXE program will be erased automatically from the memory and the serial interface will be enabled.
THANDLE is a 32 bit value for the 32 bit DLL. In reply, the function returns a 32 bit value. If the function was successful, the value is 0. Otherwise, one of the error codes indicated below is returned:

Error	Errorcode(hex)	Description
err_appl_register_wrongId	$0103	the Handle given is unknown
err_dll_no_init	$0100	the DLL is not initialized

3.2.3 Getmessagenumber

Syntax: function getmessagenumber:longint;

This function designates the type of a message sent from the DLL to the applications, as a 32 bit value. The number is valid for the total duration of the session, i.e. until the EXCIMER.EXE program is closed.

3.2.4 Getstatuspointer

Syntax: function getstatuspointer:pointer;

This function returns a 32 bit pointer to the STATUS data structure (Tstatusrec). This data structure is only readable. The pointer is valid for the total duration of the application.
In the 32 bit DLL, every application is using its own copy of the data structure (local memory). These local data structures are only updated when calling the Getstatuspointer function. In that way, parallel data access can be synchronized. Sequence, set up and contents of the data structure (Tstatusrec) are described below. The data structure is a packed structure. „packed“ means that the fields in the structure are not aligned on word or double-word boundaries.

Description of the Tstatusrec structure:

```
Tstatusrec  =     packed record
                    size              : word;
                    initstat          : byte;
                    queuefill         : byte;
                    anzapp16          : word; (not used)
                    anzapp32          : word;
                    anzdock16         : word; (not used)
                    anzdock32         : word;
                    dll16ver          : tchararray; (not used)
                    dll32ver          : tchararray;
                    VersionStr        : array[0..32] of char;;
                    Dummy1            : array[0..11] of byte;;
                    NewVersion        : single;
                    Dummy2            : byte;
                    excimer           : TExcimerStatus or array[0..144] of byte;;
                    ADL               : array[0..4488] of byte;;
                    InfoArray         : array[0..1023] of byte;
                    paf_LaserStatus2  : Pointer to LaserStatus2;
                end;
```

Field	Type	Size [byte]	Description
size	word	2	total size of the data structure in bytes
initstat	byte	1	0: EXCIMER.EXE not loaded unequal 0: EXCIMER.EXE loaded
queuefill	byte	1	number of commands in the command queue
anzapp16	word	2	(not used)
anzapp32	word	2	number of registered 32 bit applications
anzdock16	word	2	(not used)
anzdock32	word	2	number of applications connected to the 32 bit DLL
dll16ver	tchararray	50	(not used)
dll32ver	tchararray	50	character string: EXCIM32.DLL version as 0 terminated string, unused bytes are shown with a 0
VersionStr	array of char	33	character string: EXCIMER.EXE version as 0 terminated string, unused bytes are shown with a 0
Dummy1	array of char	12	unused
NewVersion	Single	4	Excimer.exe version as floating point number
Dummy2	char	1	unused
excimer	TExcimerStatus	145	Laser status for MNL200 and Optex lasers
ADL	TAdlStatus	4489	Status for ATM-unit (not used)
InfoArray	Array of byte	1024	Not used
paf_LaserStatus2	pointer	4	Pointer to Laserstatus2 (recommended for MNL100)

The field excimer (TExcimerStatus) contains the data to characterize the laser device status. This structure was used in prior laser versions. It can be used for MNL100-laser, but is impractical. **For MNL100, it is recommended to use the LaserStatus2 – structure**. The Pointer paf_LaserStatus2 points to this structure. Both structures are described on the next pages.
The ExcimerStatus and LaserStatus2 can only be used, if anzapp32 >1. A value of 0 or 1 for anzapp32 can occur, if the initialization of excimer.exe is not finished at the program start.

3.2.4.1 Structure LaserStatus2 (TLaserSatus2)

```
TLaserStatus2 = packed record
                //-----------------------------
                dummy0                  : array[1..10] of char;
                    fVersion            : single;
                    ac_VerText          : array[1..10] of char;
                    ul_SerialNumber     : longint;
                    ui_PEM_SerialNumber : word;
                //-----------------------------
                    ul_StatusFlags1     : longint;
                    ul_StatusFlags2     : longint;
                    ul_ErrorFlags1      : longint;
                    ul_ErrorFlags2      : longint;
                    uc_LaserType1       : byte;
                    uc_LaserType2       : byte;
                dummy1                  : array[1..8] of byte;
                //-----------------------------
                    ui_Frequency        : word;
                    ui_Quantity         : word;
                    ui_Burstcounter     : word;
                    ul_PulseCounter     : longint;
                dummy2                  : array[1..8] of byte;
                //-----------------------------
                    f_Energyreal        : single;
                    f_Energydirectreal  : single;
                    uc_HVperCent        : byte;
                dummy3                  : array[1..9] of byte;
                //-----------------------------
                    f_LaserVoltage      : single;
                    f_Temperature1      : single;
                    f_Temperature2      : single;
                dummy4                  : array[1..8] of byte;
                //-----------------------------
                    ui_AttenuatorSetPos : word;
                    ui_AttenuatorActPos : word;
                    f_AttenuatorRatio   : single;
                    uc_AttenuatorStatus : byte;
                dummy5                  : array[1..9] of byte;
                //-----------------------------
                    ac_LasTypeText      : array[1..34] of char;
end;
```

Field	Type	Size [byte]	Description
dummy0	array[1..10] of byte	10	unused
fVersion	single	4	Laser firmware version as floating point
ac_VerText	array[1..10] of char	10	Laser firmware version as string
ul_SerialNumber	longint	4	Laser serial number
ui_PEM_SerialNumber	word	2	Serial number of energy monitor
ul_StatusFlags1	longint	4	**Flag 1**, meaning at 1: bit 0: laser ready for operation bit 1: power supply is switched on (standby) bit 2: laser operates bit 3: laser operates in burst mode bit 4: laser operates in repetition mode bit 5: laser operates in external triggered mode --- bit 12: shutter is open --- bit 16: HV control supported bit 17: energy measurement supported bit 18: shutter is supported bit 19: attenuator is supported --- bit 28: laser is offline
ul_StatusFlags2	longint	4	unused
ul_ErrorFlags1	longint	4	**Error-flag 1**, meaning at 1: bit 0: statical error bit 1: enclosure error bit 2: interlock error (remote) bit 3: temperature limiter error --- bit 8: operation error bit 9: HV supply error bit 10: power supply error 1 bit 11: power supply error 2 bit 12: over temperature error sensor 1 bit 13: over temperature error sensor 2 bit 14: over temperature warning sensor 1 bit 15: over temperature warning sensor 2 --- bit 16: EEPROM error bit 17: energy monitor error bit 18: CPU error (watchdog-reset occurred) --- bit 22: Stepper couldn't properly initialized
ul_ErrorFlags2	longint	4	unused
uc_LaserType1	byte	1	original typebyte 1 from bus- protocol
uc_LaserType2	byte	1	original typebyte 2 from bus- protocol
dummy1	array[1..8] of byte	8	unused
ui_Frequency	word	2	set point for pulse frequency
ui_Quantity	word	2	set point for quantity (number of pulses in burst mode)
ui_Burstcounter	word	2	number of remaining pulses in burst mode
ul_PulseCounter	longint	4	pulse counter
dummy2	array[1..8] of byte	8	unused
f_Energyreal	single	4	measured energy value as a floating point value (moving average value)
f_Energydirectreal	single	4	last measured energy value (unfiltered)

uc_HVperCent	byte	1	HV tuning value between 0 and 100
dummy3	array[1..9] of byte	9	unused
f_LaserVoltage	single	4	power supply voltage [volt]
f_Temperature1	single	4	temperature on sensor1 [°C]
f_Temperature2	single	4	temperature on sensor2 [°C]
dummy4	array[1..8] of byte	8	unused
ui_AttenuatorSetPos	word	2	set point for attenuator position
ui_AttenuatorActPos	word	2	actual value of attenuator position
f_AttenuatorRatio	single	4	attenuator ratio
uc_AttenuatorStatus	byte	1	Status of attenuator bit 0 – stepper is initialized bit 1 – stepper is in InitMode bit 2 – higher current level is active bit 7 – stepper error (index not found)
dummy5	array[1..9] of byte	9	unused
ac_LasTypeText	array[1..34] of char	34	laser type as string

3.2.4.2 Structure Excimer (TExcimerStatus)

(only for information)

For the MNL100, it is recommended to use the Laserstatus2. The only value of interest in TExcimerStatus is the value of COM-port, which represents the number of the serial interface.

```
TExcimerStatus  =     packed record
                            FLAG1           : byte;
                            FLAG2           : byte;
                            FLAG3           : byte;
                            FLAG4           : byte;
                            FLAG5           : byte;
                            FLAG6           : byte;
                            FLAG7           : byte;
                            FLAG8           : byte;
                            Frequency       : word;
                            QUANTITY        : word;
                            Counter         :longint;
                            Watchcounter    :word;
                            Pressure        : byte;
                            PressDig        : byte;
                            Pressreal       : single;
                            HVvalue         : byte;
                            HVvaluereal     : single;
                            Energybyte      : byte;
                            Energyreal      : single;
                            EnergySoll      : byte;
                            Energysollreal  : single;
                            FLAG9           : byte;
                            Energydirect    : byte;
                            Energycorrfact  : single;
                            CRC16           : word;
                            Temperature     : single;
                            Energydirectreal        : single;
                            NewFillCycles   : byte;
```

```
        internal1       : array[1..47] of byte;
        SerialNumber    : longint;
        Reserve7        : word;
        CRC16Ref        : word;
        comport         : byte;
        Time            : byte;
        Version         : byte;
        releasebyte     : byte;
        lastype1        : byte;
        lastype2        : byte;
        Release         : array[1..9] of char;
        LaserType       : array[1..10] of char;
        GasType         : array[1..9] of char;
end;
```

Data types:

byte:	8 bit Integer-number without sign
char:	8 bit ASCII-character
word:	16 bit Integer-number without sign
smallint:	16 bit Integer-number with sign
single:	4 byte floating point number

3.2.5 Putcommand

Syntax: function putcommand(prec:pointer):longint;

Using this function, a command is transmitted to the DLL and added to their queue. The DLL then sends a message to the EXCIMER.EXE program. This clears the DLL command queue and adds the commands to the EXCIMER command queue. The calling application will be informed of this with an event_command_queued-message.
From the EXCIMER queue, the commands are transferred to the COM-interface driver and finally transmitted to the laser device. If a command cannot be transmitted an event_error_command_fail-message will be sent to the calling application. (e.g. transmission of a „Watchmode“ command without preceding “LASOn” command). The laser device acknowledges every command either with an OK or error. This message (event_Rx_OK-message or event_Rx_Error-message) is sent to the calling application too.
The queue can receive up to 100 commands. When the filling level (STATUS.QUEUEFILL) reaches 80, no further commands are accepted and an event_error_command_queue message will be sent to the calling application instead. Commands that are not to be transmitted to the laser (system commands) are not added to the queue. These commands are carried out immediately and an event_syscommand_OK-message to the calling application confirms its effect.
In reply, the Putcommand function returns a 32 bit value. If the function was successful, the value is 0. Otherwise, one of the error codes indicated below is returned:

Error	**Errorcode(hex)**	**Description**
err_dll_no_init	$0100	EXCIMER.EXE not loaded
err_command_queue_full	$0104	command queue full

Together with the Putcommand function, a 32 bit pointer, which points to command message structure (see below) is to be transferred to the DLL:

```
TCommandMsg     =     packed record
                            ApId            : longint;
                            commandnr       : word;
                            command         : tcommand;
                      end;
```

with:

Field	Type	Size [byte]	Description
ApId	Longint	4	application's Handle (the DLL will send the reply message to this Handle)
commandnr	Word	2	identification number, which will be sent back in the reply messages (may be determined individually)
command	Tcommand	–	command structure containing the command to be executed (see below)

```
tcommand    =   packed record
                    commandID   : word;
                    param       : tparam;
                end;
```

with:

Field	Type	Size [byte]	Description
commandID	Word	2	command (see below)
param	Tparam	4	parameter

3.3 The DLL Commands

The following commands (commandIDs) can be sent to the DLL using the PutCommand function:

CommandID	hex-digits
excom_Set_comport	$1000
excom_LASER_ON	$1012
excom_LASER_OFF	$1013
excom_BURST	$1014
excom_REPETITION	$1015
excom_EXTTRIG	$1022
excom_OFF	$1016
excom_SetFrequency	$1017
excom_SetQuantity	$1018
excom_INC_HV	$1033
excom_DEC_HV	$1034
excom_SET_HV	$1035
excom_RESET_PEM_ERROR	$1032
excom_Energy_Calwindow	$1039
excom_CloseShutter	$103D
excom_OpenShutter	$103C
excom_AttenuatorInit	$1855
excom_AttenuatorSetRatio	$1851
excom_AttenuatorSetEnergy	$1852
excom_AttenuatorSetPos	$1850

These commands are explained in the following section.

3.3.1 Communication

Commands are sent to the DLL using the function Putcommand described above.

3.3.1.1 SetComport (1000H)

This command selects the COM-port for the communication with the laser device. In response, two messages (both event_set_comport and event_get_system_status) will be sent to all registered applications. The EXCIMER.INI file pre-sets the COM-port.
The current COM-port is shown with STATUS.EXCIMER.COMPORT. The command message structure is shown below:

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	00	word	2	excom_set_comport
	10			
param	Comnr	byte	1	COM-number, e.g. 1 with COM1, 0 with no COM

3.3.2 Laser Operating Commands

3.3.2.1 Laser_On (1012H)

This command switches the HV power supply of the laser device on, to enable laser operation and set in Stand by.
If the laser is ready for operation, the command will be transmitted to the laser. After transmitting this command, the laser device is unable to receive any command for 5 sec.

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	12	word	2	excom_LASER_ON
	10			

3.3.2.2 Laser_Off (1013H)

This command switches the HV power supply of the laser device off and therefore laser operation.

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	12	word	2	excom_LASER_OFF
	10			

3.3.2.3 Burst (1014H)

This command starts laser operation in Burst mode with a preset pulse number (QUANTITY) and frequency (repetition rate). It is added to the command queue.
Only if the HV power supply is switched on, the command will be transmitted to the laser.

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	14	word	2	excom_BURST
	10			

3.3.2.4 Repetition (1015H)

This command starts laser operation in the Repetition mode with a preset frequency (repetition rate).
Only if the HV power supply is switched on, the command will be transmitted to the laser.

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	15	word	2	excom_REPETITION
	10			

3.3.2.5 External_Trigger (1022H)

This command starts laser operation in External Trigger mode. In this mode, laser pulses are emitted in accordance with signals received from an external trigger generator via the optical trigger gate.
Only if the HV power supply is switched on, the command will be transmitted to the laser.

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	22	word	2	excom_EXTTRIG
	10			

3.3.2.6 Off (1016H)

This command disables the HV power supply to interrupt laser operation in Burst, Repetition or External Trigger mode (STOP). The command message structure is shown below:

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	16	word	2	excom_OFF
	10			

3.3.3 Laser Operating Parameters

3.3.3.1 SetFrequency (1017H)

This command enters the repetition rate for laser operation. Only values of up to the maximum repetition rate are accepted.

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	17	word	2	excom_SetFrequency
	10			
frequency	value	word	2	set point for the repetition rate

3.3.3.2 SetQuantity (1018H)

This command enters the Quantity value (max. 65,000 pulses). In Burst mode, this set number of laser pulses will be emitted.
Every generated laser pulse is registered and the counter will count down to zero. The counter is decremented with each laser pulse.

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	18	word	2	excom_SetQuantity
	10			
param	Quantity	word	2	set point for the number of pulses to be emitted

3.3.3.3 IncHV (1033H)

This command increases the HV set value by 1 %
NOTE: This command is only valid, if the HV control is supported by the laser firmware.

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	33	word	2	excom_INC_HV
	10			

3.3.3.4 DecHV (1034H)

This command decreases the HV set value by 1 %.
NOTE: This command is only valid, if the HV control is supported by the laser firmware.

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
Commandnr	number	word	2	identification number, which will be sent back in the reply messages
commanded	34	word	2	excom_DEC_HV
	10			

3.3.3.5 SetHV (1035H)

This command enters the HV set value directly. The set value is tunable from 0 % (min. permissible energy value for laser operation) to 100 % (max. permissible energy value for laser operation) in 1 % steps.
If the laser is ready for operation, the command will be transmitted to the laser.
NOTE: This command is only valid, if the HV control is supported by the laser firmware.

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	35	word	2	excom_SET_HV
	10			
param	HV	word	2	HV set point in %

3.3.3.6 Energy_Calwindow (1039H) (optional)

This command opens the energy monitor adjustment window

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	39	word	2	excom_Energy_Calwindow
	10			

3.3.3.7 ResetPemError (1032H) (optional)

This command resets an energy measuring error. An energy measuring error occurs when no laser pulse is detected by the energy monitor after triggering. An energy measuring error is always indicated when the internal energy monitor has been removed or disconnected.

Field	**Value**	**Type**	**Size [byte]**	**Description**
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	32	word	2	excom_RESET_PEM_ERROR
	10			

3.3.3.8 OpenShutter (103CH) (optional)

This command opens the shutter.
If the HV power supply is switched on, the command will be transmitted to the laser.
NOTE: This command is only valid, if the shutter is supported by the laser firmware.

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	3C	word	2	excom_OpenShutter
	10			

3.3.3.9 CloseShutter (103DH) (optional)

This command closes the shutter.
NOTE: This command is only valid, if the shutter is supported by the laser firmware.

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	3D	word	2	excom_CloseShutter
	10			

3.3.3.10 AttenuatorInit (1855H) (optional)

This command starts the initialization for the attenuator unit. The attenuator is searching his index point and then going to the set point. This command is automatically started by power on.
NOTE: This command is only valid, if the attenuator unit is supported by the laser firmware.

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	55	word	2	excom_AttenuatorInit
	18			

3.3.3.11 AttenuatorSetRatio (1851H) (optional)

This command enters the transmission ratio of the attenuator unit. Only values between 0 and 100 are permitted.
NOTE: This command is only valid, if the attenuator unit is supported by the laser firmware.

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	51	word	2	excom_AttenuatorSetRatio
	18			
param	Ratio	single	4	Attenuator ratio in % (0..100)

3.3.3.12 AttenuatorSetEnergy (1852H) (optional)

This command enters the desired output energy of the laser. From this the laser firmware is calculating and adjusting a corresponding ratio of the attenuator unit.
NOTE: This command is only valid, if the attenuator unit is supported by the laser firmware.

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	52	word	2	excom_AttenuatorSetEnergy
	18			
param	energy	single	4	laser output energy in µJ

3.3.3.13 AttenuatorSetPos (1850H) (optional)

This command enters directly the set point for the stepper motor position of the attenuator unit. Only values between 0 and 400 are permitted.
NOTE: This command is only valid, if the attenuator unit is supported by the laser firmware.

Field	Value	Type	Size [byte]	Description
ApID	Handle	longint	4	application's Handle
commandnr	number	word	2	identification number, which will be sent back in the reply messages
commandID	50	word	2	excom_AttenuatorSetPos
	18			
param	postion	word	2	stepper motor position

3.4 Messages

The EXCIMER DLL's send their messages to the registered applications via the Windows message system. The receivers are addressed by using the Handle given by calling the function *Registerapplication*.
The individual messages are indicated using numbers. The corresponding number of the messages sent from the DLL to the applications can be retrieved using the function *Getmessagenumber*. This number is valid for the total running time of Windows.
A Windows message has two parameters, LPARAM and WPARAM. These parameters further specify the message.
The commandnr given with the function *Putcommand* is stored in WPARAM.
LPARAM is a 32 bit number. The lower 8 bits contain the sub-message. The next 8 bits contain the byte parameter PAR1. The top 16 bits of LPARAM are called COMMRETURN. COMMRETURN contains the command that initiated the message. If the message was initiated by the DLL itself, the value of COMMRETURN is zero. The sub-messages are indicated below:

Sub-message	**Value (hex)**	**Description**
event_blink1	30H	This message is sent periodically to all registered applications. The content of PAR1 alternates between 0 and 1 with every message. COMRETURN and WPARAM have the value 0. The standard cycle time of 400 ms can be changed by using the command Set_timer1.
event_blink2	31H	This message is sent periodically to all registered applications. The content of PAR1 alternates between 0 and 1 with every message. COMRETURN and WPARAM have the value 0. The standard cycle time of 1 sec can be changed by using the command Set_timer2.
event_getWUtime	32H	This value indicates the progress of the Warm-Up period from 0 % to 100 %. During the Warm-Up period, this event occurs every 2 sec. It signalizes a change of the STATUS.EXCIMER.TIME value.
event_getRelease	33H	This message indicates an incoming Release information from the laser. The information causes changes of the contents of STATUS.EXCIMER.VERSION, RELEASEBYTE, RELEASE and LASERTYPE.
event_getVersion	34H	This message indicates an incoming version information from the laser. The message is requested by the RX_online event or after calling the function Registerapplication. The information causes changes of the content of STATUS.EXCIMER.VERSION.
event_Get_Excimer_status	35H	This message indicates an incoming status information from the laser. The message is requested continuously (normally approx. every 110 ms) as long as no other command is to be transmitted to the laser.
event_get_system_status	36H	This message indicates changes of the system configuration caused by a command. WPARAM contains the commandnr and COMMRETURN the initiating command.
event_getSerNo	37H	The serial number of the laser software (CLS) has been received.
event_TX_Flash_on	40H	This message will always be sent when a command is transmitted to the laser.
event_TX_Flash_off	41H	This message is sent 55 ms after event_TX_Flash_on.
event_RX_Flash_on	42H	This message will always be sent when an information has been received at the serial interface.

event_RX_Flash_off	43H	This message is sent 55 ms after event_RX_Flash_on.
event_RX_Timeout_on	44H	This message is sent alternately (500 ms) with event_RX_Timeout_off if the communication to the laser has been interrupted.
event_RX_Timeout_off	45H	This message is sent alternately (500 ms) with event_RX_Timeout_on if the communication to the laser has been interrupted. When the connection to the laser is re-established, the message will be sent at last.
event_RX_offline	50H	This message is sent if the communication to the laser has been interrupted.
event_RX_online	51H	This message is sent if the connection to the laser is re-established.
event_set_comport	52H	This message is sent if the setting of the COM-port has been changed.
event_RX_OK	60H	This message is sent to the calling application if a command has been successfully processed by the laser. WPARAM contains the commandnr and COMRETURN the initiating command. The sub-command sent to the laser is stored in PAR1.
event_syscommand_OK	65H	This message is sent to the calling application if a command for changing the system setting has been processed. WPARAM contains the commandnr and COMRETURN the initiating command.
event_command_queued	66H	This message is sent to the calling application if a command has been processed. WPARAM contains the commandnr and COMRETURN the initiating command.
event_excimer_closed	70H	This message will be sent if the EXCIMER.EXE program is closed.
event_RX_Error	80H	This message is sent to the calling application if an incorrect command has been rejected by the laser.
event_error_getstatus	86H	This message is sent if a transmission error is indicated by the laser during a state request.
event_error_nosupport	81H	This message is sent if the connected laser is not supported by the software interface.
event_error_command_fail	82H	This message is sent to the calling application if a command cannot be processed by the laser. WPARAM contains the commandnr and COMRETURN the initiating command.
event_error_unknow_command	83H	This message is sent to the calling application if a command is unknown. WPARAM contains the commandnr and COMRETURN the initiating command.
event_error_command_busy	84H	This message is sent to the calling application if a command is repeatedly not accepted by COM-interface driver. WPARAM contains the commandnr and COMRETURN the initiating command.
event_error_command_queue	85H	This message is sent to the calling application if the command queue is full. WPARAM contains the commandnr and COMRETURN the initiating command.

On request you can obtain a **sample and test program** (Borland-Delphi source code) that runs independently.