Operating Instructions

Spectroradiometer

Firmware

Version:3.x (specbos 1211)

Version:2.x (specbos 1201)

Table of Content

1. Explanation of Commands

The command syntax of JETI instruments is SCPI compatible.

Therefore, JETI firmware commands consist of a main category and a more specific part, both connected by a colon. Some very special commands contain only one piece.

A technical note (https://www.jeti.com/files/content/support/downloads/technical_notes/tn_18.pdf) describes the design of a program using firmware commands and the interaction between the commands more in detail.

This document lists the firmware commands summarized in logical categories.

2. Remarks

2.1. General Remarks

Communication with the device can be provided using special commands via USB interface. This interface is designed as a virtual COM port, so it can be handled similarly to a serial port with the settings 8n1/ no protocol. The allowed transfer rates are 38 400, 115 200 and 921 600 Bd.

Remark!	The Baudrate of 921 600 Bd is not allowed for specbos 1211-RS and specbos 1211-BT.

Default baudrate	**Device**
115 200 Bd	specbos 1211-RS specbos 1211-BT
921 600 Bd	specbos 1201 specbos 1211

2.2. Command Categories

The following list shows the available command categories. They match the SCPI standard regulations, begin with * (except ESC) and involve the following keywords:

Commands	
*PARAmeter	Get and set general parameters
*CONfigure	Get and set configuration data
*INITiate	Start a configured measurement
*FETCH	Get data from previous measurement
*READ	Start a configured measurement and get the data (combination of *INIT and *FETCH)
*MEASure	Configure, start the measurement and get the data (combination of *CONF, *INIT and *FETCH)
*CALCulate	Calculate data from the previous measurement
*STATus	Information about error and configuration status
*HELP	Output of help information

These keywords can be followed by one or two additional words, separated by colons, and by arguments. It is only necessary to use the indicated capital letters, other letters can be omitted.

2.3. Commands with Arguments

Several commands can be extended by arguments. The meanings of the arguments are as follows:

tint	Integration time in ms, range from: specbos 1201 -> 1 … 64999 ms specbos 1211 (FW <= 3.2.x) -> 1 … 64999 ms specbos 1211 (FW >= 3.4.x) -> 0.10 … 64 999.99 ms **Remark: If tint=0 the device will perform adaption automatically.**
av	Average counts for measurement (1 … 10 000)
format	Output format of spectral data (for a description of the data stream see '10') 0 no output (affects all data, also calculated values) 1 L/H binary output without length and checksum 2 ASCII output, space separated (only for testing purposes, influences the *MEAS :ALLVA output too) 3 L/H binary output with length and checksum 4 ASCII output, separated by <CR> 5 H/L binary output without length and checksum 6 H/L binary output with length and checksum 7 Raw data with wavelength 9 Interpolated ASCII output without wavelengths 10 Interpolated ASCII output with wavelengths 11 H/L binary integer interpolated output 12 H/L binary float interpolated output (only for raw data!) 13 ASCII integer interpolated output
function	Selection of output function (valid for pre configuration and the following commands *INIT, *MEAS, *READ and *FETCH) 0 None 1 Light (exposed) spectrum (with shutter opened/lamp on) 2 Dark spectra 3 Reference spectra (difference of exposed spectrum and dark spectrum) 4 Transmission spectra, unit: ‰ 5 Absorption spectra, unit: AU 6 Radiometric spectra, (unit: depending from selected calibration file) Radiance $W/(m^2 \cdot sr \cdot nm)$ Irradiance $W/(m^2 \cdot nm)$ Radiant flux W/nm Radiant intensity $W/(sr \cdot nm)$ 7 Radiometric value, (unit: dependent from selected calibration file) Radiance $W/(m^2 \cdot sr)$ Irradiance W/m^2 Radiant flux W

	Radiant intensity W/sr 8 Photometric value, (unit: dependent from selected calibration file) Luminance cd/m^2 Illuminance lx Luminous flux lm Luminous intensity cd 9 Chromaticity xy 10 Chromaticity u´v´ 11 Dominant wavelength and color purity 12 Radiometric, photometric, xy and u´v´ values, dominant wavelength and color purity 13 Correlated color temperature (CCT) K 14 CRI values
wbeg	Start of wavelength range, in nm (200 … 1099 nm)
wend	End of wavelength range, in nm (201 … 1100 nm)
wstp	Wavelength step (1 … 10 nm (integer)) (for radiometric calculation will be used 1 nm step)
temp	Color temperature of reference source for CRI calculation
filenr	Number of calibration file
arg	Other arguments, described in text

If commands with get and set options are used with ?, the stored value(s) will be given. If an argument is used, this argument will be set.

A space sign between command and argument is necessary (not in case of ?).

If a command which accepts arguments is used without arguments, then the configured arguments will be used (exception: tint – see *CONF:EXPO).

If a command was successfully proceeded it will be answered by an Acknowledge sign (ACK, 06 hex), otherwise the error message “Not acknowledged” (NAK, 15 hex) will be returned. The reason of an error can be read by the command *STATUS:ERR<CR>.

Several commands can be written successively in one line separated by semicolons.

2.4. Overview of Instrument Answers to the Firmware Commands

Command category	Answer
*PARA *CONF *CONTR	*with setting of value and ** *ACK (06 hex),* if value is accepted *NAK (15 hex),* if value is not accepted *with ? (data request):* *data sequence <CR>*
*INIT	*ACK (06 hex) immediately after command input* *BELL (07 hex) after finishing the measurement*
*READ *MEAS	*ACK (06 hex) immediately after command input* *BELL (07 hex) after finishing the measurement* *data sequence* *<CR> <CR> in case of spectral data output* *<CR> in case of single value output*
ESC	*NAK (15 hex) (Error 147 – breakscan)* [1]
*FETCH *CALC	*data sequence* *<CR> <CR> in case of spectral data output* *<CR> in case of single value output*
*STAT	*Status information <CR>*
*HELP	*List with commands and description <CR>*
*CALIB	with get: *data sequence* *<CR> <CR>* with state: *value* *<CR>*

2.5. Meaning of Error Codes

Error codes:

0 : no error
4 : command error
7 : error password
8 : digit error
10 : error argument 1
11 : error argument 2
12 : error argument 3
13 : error argument 4
20 : error parameter argument
21 : error config argument
22 : error control argument
23 : error read argument
24 : error fetch argument

[1] Valid for all scan functions

25 : error measuring argument
26 : error calculation argument
27 : error calibration argument
101 : error parameter checksum
102 : error userfile checksum
103 : error userfile2 checksum
104 : error userfile2 argument
120 : error overexposure
121 : error underexposure
123 : error adaption integration time
130 : error shutter not exist
131 : error no dark measurement
132 : error no reference measurement
133 : error no transmission measurement
134 : error no radiometric calculation
135 : error no cct calculation
136 : error no cri calculation
137 : error no dark compensation
138 : error no light measurement
139 : error no peak calculation
140 : error calibration data
141 : error exceed calibration wavelength
147 : error scan break
160 : error timeout cycle optical trigger
161 : error divider cycle time
170 : error write parameter to flash
171 : error read parameter from flash
172 : error erase flash
180 : error no calib file
181 : error calib file header
182 : error write calib file
183 : error calib file values
184 : error calib file number
186 : error clear calib file
187 : error clear calib file argument
190 : error no lamp file
191 : error lamp file header
192 : error write lamp file
193 : error lamp file values
194 : error lamp file number
196 : error clear lamp file
197 : error clear lamp file argument
200 : error ram check
220 : error data output
225 : error insufcient ram
230 : error first memory allocation
231 : error second memory allocation
232 : error third memory allocation
251 : error wavelength range for radiometric calculation
280 : error jump boot by battery power
500 : error trigger configuration 1
501 : error trigger configuration 2

3. Request and Setting of Parameters

These commands allow to set and to read the basic settings of the instrument. They are set in factory and normally the user has no reason to change them.

The values can be checked using the appropriate command, followed by a question mark and a CR. They can be changed by specifying arguments.

Remark:	Keep in mind that changed parameters can cause erroneous measurement results.

There are two main categories of parameter commands:

*PARA commands are intended for permanent storage. Changes on one or more parameters have to be saved with the *PARA:SAVE command.

*CONFIG commands are intended for temporal storage. Sometimes there exists the same command in both categories.

Parameter and configuration commands specially related to particular kinds of measurements are listed in the appropriate chapters of section 4 4.

3.1. General Parameter Commands

The following commands can be used to identify an instrument after checking all virtual COM ports.

***IDN?<CR>**

Get device ID

Example of answer:

```
JETI_SB1211 (for specbos 1211)
SB05 (for specbos 1201)
JETI_VersaPIC_RU60 (for Versa PS2000)
```

***PARAmeter:SPNUMber arg<CR>**

Get/ Set spectrometer number

arg: some number with maximal 7 digits (because of float numbers precision)

Remark: Please, never change it as long as you are not an engineer of JETI!

Float; bytes 252–255 of the parameter block.

Example of answer for "*PARA:SPNUM?<CR>":

```
spectrometer number:2005184
```

***PARAmeter:SERNumber?<CR>**

Get device-ID (internal number)

Float; bytes 64–67 of the parameter block

Example of answer:

```
serial number:1072
```

3.2. Settings Concerning the Detector Array and the AD Converter

***PARAmeter:PIXel?<CR>**

Get pixel quantity

Maximal value: 2048
Float; bytes 72–75 of the parameter block.
Example of answer:
```
pixel:2048
```

***PARAmeter:SENSor?<CR>**

Get sensor type (see the table with sensor codes)

Float; bytes 84–87 of the parameter block
Example of answer:
```
Sensor:35
```

***PARAmeter:REFADC arg<CR>**

Get/ Set analog voltage of the reference source

Range: 1 < arg ≤ 5.5
Default values: 4.50
Float; bytes 794–797 of the parameter block
Example of answer for *PARA:REFADC?<CR>:
```
ADC reference:4.510000
```

***PARAmeter:ADCResolution arg<CR>**

Get/ Set ADC resolution

arg = 14 ... 16 [bit]

Remark: Use only 14 or 15 bits for radiometric applications!

Default value:
14 for specbos 1201,
15 for specbos 1211,
16 for Versa PS 2000

Float; bytes 148–151 of the parameter block.
Example of answer for "*PARA:ADCR?<CR>":
```
ADCResolution:15
```

***PARAmeter:ADCVoltage arg<CR>**

Get/ Set input voltage range of ADC.

arg = 0:4 V
arg = 1:2 V
One bit; bit #9 of a 32-bytes-field (see *0:CONFIG); bytes 144–147 of the parameter block.
Example of answer for "*PARA:ADCV ?<CR>":
```
ADC input range:1
```

***PARAmeter:FASTscan arg<CR>**

Get/ Set time to next fast cycle (in ms).

arg: 0 ... 65000
Float; bytes 164–167 of the parameter block.
Example of answer for "*PARA:FAST?<CR>":
```
Fastscan:50
```

***PARAmeter:OFFSet arg<CR>**

Get/ Set offset value.

arg: -300 ... +300 mV

Float; bytes 280–283 of the parameter block.

Example of answer for "*PARA:OFFS?<CR>":

```
Offset Channel 1:50
```

***PARAmeter:GAIN arg<CR>**

Get/ Set gain value.

arg: 1.0 ... 5.0

Default value:3 for specbos 1201,
1.2 for specbos 1211,
1 for Versa PS2000

Float; bytes 276–279 of the parameter block.

Example of answer for "*PARA:GAIN?<CR>":

```
Gain Channel 1:1.0
```

***PARAmeter:DIRECTion arg<CR>**

Change scan direction.

arg: 0 or 1

One bit; bit #31 of the 4 bytes field;

bytes 140–143 of the parameter block

Example of answer for *PARA:DIRECT?<CR>:

```
Scan direction:0
```

3.3. Settings Concerning the System and the Power Consumption

***PARAmeter:BAUD arg<CR>**

Get/ Set baudrate.

arg:

384 – 38 400 Bd

115 – 115 200 Bd

921 – 921 600 Bd

Default value: 921 600 Bd

Float; bytes 248–251 of the parameter block.

Example of answer for "*PARA:BAUD ?<CR>":

```
Baud:921
```

***CONFigure:SYSTEM arg<CR>**

Set MCU clock frequency

arg = 0:3.75 MHz
arg = 1:64 MHz (default PLLCLK = 0)
arg = 2:72 MHz (default PLLCLK = 1)
arg = 3:80 MHz

Example of answer for "*CONF:SYSTEM ?<CR>":

```
SYSCLC – processor clock rate [Hz]; PBCLK – peripheral bus
clock rate [Hz].
SYSCLK:72000000
```

```
PBCLK:36000000
U4BRG:77
```

***CONFigure:ENCHARge?<CR>**

Get/ Set enable charge flag.

arg: 1 – enabled, 0 – disabled)

Example of answer for "*CONF:ENCHAR?<CR>":

```
Charge enable:1
```

***PARAmeter:ADPWdown arg<CR>**

Get/ set no power down flag. Will be deactivated automatically with the next command.

arg: 0 – off; 1 – on

Default value: 1

One bit; bit #15 of a 32-bytes-field (see *0:CONFIG); bytes 144–147 of the parameter block.

Example of answer for "*PARA:ADPW ?<CR>":

```
power down ADC (1-on, 0-off):0
```

3.4. Settings for Peripheral Components

The instruments contain several peripheral components or external components can be connected with them. The following commands allow the control of these components.

3.4.1. Lamps/ Shutter

***PARAmeter:SHUTter arg<CR>**

Get/ set external shutter availability.

arg:
0 – not available (only dark compensation possible)
1 – available (dark measurement with shutter possible)

Remark: This setting has to be 1 to ensure that the dark measurements with shutter will work correctly. Shutter is necessary for radiometric measurements.

Default value: 1

One bit; bit #16 of a 32-bytes-field (see *0:CONFIG); bytes 144–147 of the parameter block.

Example of answer for "*PARA:SHUT ?<CR>":

```
Shutter:1
```

These commands are used to control the peripheral elements lamp/ shutter, laser, Hall sensors, optical trigger and battery.

***CONTRol:LAMP arg<CR>**

Get/ Set the shutter (or lamp) status.

arg:
0 – shutter closed/ lamp off
1 – shutter opened/ lamp on

Example of answer:

```
lamp:1
```

***PARAmeter:SDELay arg<CR>**

Get/ Set scan delay (time difference between initiating a measurement and its real start, in ms), is necessary for full shutter opening (or lamp stability), has to be set to values between 200 and 400 ms for the inbuilt shutter of specbos 1201/ 1211.

arg: 0 ... 60 000
Float; bytes 124–127 of the parameter block.
Example of answer for "*PARA:SDEL ?<CR>":
`ScanDelay:400`

*The following commands are used to set the status of the external lamp/ shutter control (see Mini DIN connector pin 3 by *PARA:TRSL).*

***PARAmeter:LAMPEnable arg<CR>**

Get/ set enable state of external lamp/shutter control output.

arg: 0 – disabled, 1 – enabled
Default value:1
One bit; bit #2 of a 32-bytes-field (see *0:CONFIG); bytes 144–147 of the parameter block.
Example of answer for "*PARAmeter:LAMPEnable?<CR>":
`Lamp enable:1`

***PARAmeter:LAMPPolarity arg<CR>**

Get/ set polarity of external lamp/shutter control.

arg: 0 = high, 1 = low
Default value: 1
One bit; bit #3 of a 32-bytes-field (see *PARA:CONFIG); bytes 144–147 of the parameter block.
Example of answer for "*PARA:LAMPP ?<CR>":
`Lamp low:1`

3.4.2. Target Laser

***CONTRol:LASER arg<CR>**

Get/ Set target laser status.

(1 – laser on, 0 – laser off)
Example of answer:
`laser:0`

***PARAmeter:LASMODulation arg_1 arg_2<CR>**

Get/ Set laser modus (control the brightness of the laser).

arg_1: intensity, 1 ... 1000 [‰]

arg_2:

0 – laser off

1 – PWM 7 Hz

2 – PWM 28 Hz

3 – PWM 255 Hz

Default values:

arg_1 = 1000,

arg_2 = 0 for specbos 1201 and 1 for other devices

arg1: unsigned short; bytes 834–835 of the parameter block

arg2: unsigned short; bytes 832–833 of the parameter block

Example of answer for "*PARA:LASMOD ?<CR>":

```
Laser parameters:1000 3
```

3.4.3. Hall Sensors in Measuring Head to Detect Attached Accessories

***CONTRol:MHEAD?<CR>**

Get measuring head configuration "ab" (signal of hall sensors)

Sensor signal 00: uses calibration file 1
Sensor signal 01: uses calibration file 2
Sensor signal 10: uses calibration file 3
Sensor signal 11: uses calibration file 4

Example of answer:

```
mhead:0
```

3.5. Wavelength Settings

3.5.1. Definition of Wavelength Ranges

An overview of the definitions of different wavelength ranges under '9'.

***PARAmeter:WAVBEGin wbeg<CR>**

Get/ Set predefined begin of wavelength range for interpolation (see '9' block_WLrangeDefinitions).

Remark: Not valid for measurement commands, which deal with not-interpolated wavelengths.

Range: LimWavMin ≤ wend < LimWavMax (set by *PARA:WAVLIMIT)
Default value: 380 for specbos 1201; for others – 350
Signed short; bytes 128–129 of the parameter block

Example of answer:

```
Predefined start wave:350
```

***PARAmeter:WAVEND wend<CR>**

Get/ Set predefined end of wavelength range for interpolation (see '9' block_WLrangeDefinitions).

Remark: Not valid for measurement commands, which deal with not-interpolated wavelengths.

Range: LimWavMin < wend ≤ LimWavMax (set by *PARA:WAVLIMIT)
Default value: 780 for specbos 1201; for others – 1000
Signed short; bytes 130–131 of the parameter block

Example of answer:

```
Predefined end wave:900
```

***PARAmeter:WAVSTEP wstp<CR>**

Get/ Set predefined wavelength step for interpolation (see '9')

Remark: Not valid for measurement commands, which deal with not-interpolated wavelengths.

wstp: 1 ... 10

Default value: 1
Signed short; bytes 132–133 of the parameter block
Example of answer for *PARAmeter:WAVSTEP ?<CR>:
```
Predefined wave step:1
```

***PARAmeter:WAVLIMIT min max<CR>**

Get/ Set predefined wavelength limit (see '9').

Remark: Not valid for raw data measurement commands.

Maximal range: 190 ... 2500
Default values: min = 200, max = 1000
min: unsigned short; bytes 884–885 of the parameter block
max: unsigned short; bytes 886–887 of the parameter block
Example of answer for "*PARA:WAVLIMIT ?<CR>":
```
Wave limit:200 1000
```

***CONFigure:WRANge wbeg wend wstp<CR>**

Get/ Set wavelength range and step for wavelength-interpolation (see arguments and '9'). Irrelevant for not interpolated wavelengths (e.g. format 7).

Remark: Works the same way as *CONF:BEG, *CONF:END and *CONF:WSTP called separately.

wbeg and wend must be within the range set by *PARA:WAVLIMIT , and wend > wbeg
1 nm ≤ wstp ≤ 20 nm, integer; if not integer, will be rounded
Default values (those which are set by calling *CONF:DEF) are given by following parameters:* PARA:WAVBEG , *PARA:WAVEND, and *PARA:WAVSTEP. Note that setting any of the mentioned parameters also changes the setting of *CONF:WRAN .
Example of answer for *CONF:WRAN ?<CR>:
```
Wave begin:380
Wave end:800
Wave step:1
```

***CONFigure:BEGin wbeg<CR>**

Get/ Set wavelength begin. Overwrites the corresponding argument of *CONF:WRAN .
Example of answer:
```
Wave beg:350
```

***CONFigure:END wend<CR>**

Get/ Set wavelength end. Overwrites the corresponding argument of *CONF:WRAN .
Example of answer:
```
Wave end:900
```

***CONFigure:WSTP wstp<CR>**

Get/ Set wavelength step width (1 to 10 nm). Overwrites the corresponding argument of *CONF:WRAN .
Example of answer:
```
Wave step:1
```

3.5.2. Pixel-Wavelength Relation

The correlation between the pixel numbers of the detector array and the corresponding wavelength is done with a fit polynomial of the 4th order.

***PARAmeter:FITn fit_n<CR>**

Get/ Set wavelength fit parameters

$\lambda(p) = fit_0 + fit_1 \cdot p + fit_2 \cdot p^2 + fit_3 \cdot p^3 + fit_4 \cdot p^4$

p = pixel number; n = 0 ... 4

fit_n: any legal float number

fit_0 ... fit_4 are stored as an array of 5 floats; bytes 256–275 of the parameter block.

Example of answer for "*PARA:FIT0?<CR>":

```
Fit0 Channel 1:2.729578e+02
```

3.5.3. Wavelength Table

The fit polynomial is used once to create a pixel-wavelength table. This table is used as a look up table to get faster access to the wavelengths. The following command allows the output of this table.

***PARAmeter:STLINTP?<CR>**

Get an internal table of interpolated wavelengths. The table is generated automatically for the wavelength range defined by *PARA:WAVLIMIT

Example of answer:

WL	Pix	Near	Divident	Divisor	Faktor	PixWav-	+PixWav+
200	624	625	11918	16291	7.315696e-01	199.330032	200.245377
201	625	626	13424	16283	8.244181e-01	200.245377	201.160263
202	626	627	14938	16275	9.178495e-01	201.160263	202.074707
203	628	628	203	16257	1.248693e-02	202.988586	203.902145
204	629	629	1741	16252	1.071253e-01	203.902145	204.815262
205	630	630	3286	16244	2.022901e-01	204.815262	205.727936
206	631	631	4840	16236	2.981030e-01	205.727936	206.640167
207	632	632	6401	16228	3.944417e-01	206.640167	207.551987
208	633	633	7970	16221	4.913384e-01	207.551987	208.463333

etc.

3.6. Time Settings

3.6.1. General Time Settings

***PARAmeter:TINT tint<CR>**

***CONFigure:TINT tint<CR>**

Get/ Set default integration time (preset value: 100 ms)

tint (specbos 1201, 1211 with FW <= 3.2.x): 1 ... 64 999 [ms]
tint (specbos 1211 with FW >= 3.4.x): 0.10 … 64999.99 [ms]
Float; bytes 76–79 of the parameter block

After switching device on (before any measurements are done)
Previous tint = Configured tint.

Example of answer for "*PARA:TINT?<CR>":

```
Tint:100.00
```

Example of answer for “*CONF:TINT?<CR>”:

```
Previous tint:2.00
Configured tint:3.00
```

***CONFigure:MINTINt?<CR>**

Get the shortest possible integration time in ms. It depends from the type and version of instrument.
Example of answer:

```
Shortest integration time:5.00
```

***PARAmeter:MAXTINt maxtint<CR>**

***CONFigure:MAXTINt maxtint<CR>**

Get/ Set maximum integration time used by adaption.

Note, this parameter has nothing to do with TINT set with *PARA:TINT or *CONF:TINT. Its meaning is the upper border by adaption, and it is relevant *only* for adaption algorithm. With other words, it is possible that TINT (if set explicitly) > MAXTINT .
maxtint: 1000 ... 60 000 [ms]
Unsigned short; bytes 900–901 of the parameter block
Example of answer for "*PARA:MAXTIN ?<CR>" and “*CONF:MAXTIN<CR>":

```
Max_Tint:4000
```

***PARAmeter:SCALTI arg<CR>**

Get/ Set scaled integration time factor. The set integration time is multiplied with 10^{-arg}.

arg: 0 ... 3
Default value: 0
Unsigned short; bytes 880–881 of the parameter block
Example of answer for "*PARA:SCALTI?<CR>":

```
Scal faktor tint:0
```

3.6.2. Split of Integration Time

Some detector arrays show a large dark current in case of long integration times preventing measurements with high dynamics. Therefore, the concept of splitting the integration time into smaller parts was introduced.

***PARAmeter:SPLITTime arg<CR>**

Get/ Set splitting interval for the integration time.
Special solution to avoid a high dark signal of the detector at long tint (see ‘9’ block_SplitTimeConcept).

arg: 400 ... 6000 ms, for S9840 and ELIS array (0 = no split)

Default value:
5000 for specbos 1201,
1000 for other devices

Unsigned short; bytes 228–229 of the parameter block.
Example of answer for "*PARA:SPLITT ?<CR>":

```
Split time:1000
```

***CONFigure:STABILity arg<CR>**

Get/ Set stability level in % a scan with tint > splitt will be aborted, if the deviation of an individual scan exceeds the set stability level compared to the first scan or if it produces overexposure (see ‘9’)

Example of answer:
```
StabValue:20
```

***PARAmeter:SPLITDARK arg<CR>**

Get/ Set splitted dark measurement flag

arg: 0 – off; 1 – on

One bit; bit #22 of a 32-bytes-field (see *0:CONFIG); bytes 144–147 of the parameter block.

Example of answer for "*PARAmeter:SPLITDARK?<CR>":
```
splitted dark:1
```

3.7. Measurement Preconfigurations

The following commands allow to preconfigure a measurement.

***PARAmeter:FORMat format<CR> and**

***CONFigure:FORMat format<CR>**

Get/set predefined output format (see the list of arguments in format ‘5’)
Default value: 7

Unsigned char; byte 232 of the parameter block.

Example of answer:for "*PARA:FORM ?<CR>":
```
Predefined format:7 #
```
Example of answer for “*CONF:FORM ?<CR>”:
```
Previous format:2
Configured format:4
```

***PARAmeter:FUNCtion function<CR> and**

***CONFigure:FUNCtion function<CR>**

Get/ set predefined measurement function (see the list of arguments in function ‘5’)

Default value:
1 for Versa PS 2000,
12 for specbos 1201/1211

Unsigned char; byte 233 of the parameter block.

Example of answer for "*PARA:FUNC?<CR>":
```
Predefined function:1
```
Example of answer:
```
Previous function:1
Configured function:7
```

***CONFigure:ALL arg<CR>**

Get/ Set the measurement parameters tint, av, format and function

Example of answer:
```
Configured tint:100
Configured average:1
Configured format:4
Configured function:1
```

3.8. Additional Settings

3.8.1. Smoothing

***PARAmeter:BOXCAr arg<CR>**

Get/ Set boxcar mode

arg: running average of pixels, odd number 1, 3, 5 ... 25. 1 – no boxcar integration

Default value: 1
Unsigned short; bytes 226–227 of the parameter block.
Example of answer for "*PARA:BOXCA?<CR>":

```
Boxcar count:7
```

***CONFigure:AVERage av<CR>**

Get/ Set the number of measurement scans for average calculation

av: 1 ... 10 000
Example of answer:

```
Average:5
```

3.8.2. Settings for Bluetooth Units (specbos 1211-BT)

***PARAmeter:PONTIMe arg<CR>**

Get/ Set on-time till device is switched off due to inactivity.

arg: 5 ... 30000 [min]

Unsigned long = arg×60000 (the time is stored internally in ms); bytes 892–895 of the parameter block
Example of answer for *PARA:PONTIM?<CR>:

```
Pon time:15
```

***PARAmeter:PONMODe arg<CR>**

Get/ Set power on mode if no battery detected.

Default value: 0
One bit; bit #10 of a 32-bytes-field (see *PARA:CONFIG); bytes 144–147 of the parameter block.
Example of answer for "*PARA:PONMOD?<CR>":

```
power mode:0
```

***PARAmeter:UMINTIme arg<CR>**

Get/ Set scan switch off time at voltage under limit in min.

arg: 2 ... 30000 [min]

Unsigned long = arg×60000 (the time is stored internally in ms); bytes 896–899 of the parameter block
Example of answer for *PARA:UMINTI?<CR>:

```
Umin time:5
```

***PARAmeter:ENCHARge?<CR>**

Get/ Set enable charge flag.

Range: 0 or 1
One bit; bit #30 of the 4-bytes-field; bytes 140–143 of the parameter block
Example of answer:

```
Charge enable:1
```

***CONTRol:INPUT?<CR>**

Get input power voltage

battery powered: Min Voltage 3.45 V
USB or extern powered: Min Voltage 4.50 V
Example of answer:

```
Input voltage:4.03
```

***CONTRol:CHARGE?<CR>**

Get charge state
(1 – charge on, 0 – charge off)
Example of answer:

```
charge:0
```

***CONTRol:ENCHAR arg<CR>**

Get/ Set charge enable

arg: 0 – charge disabled, 1 – charge enabled
Example of answer:

```
enable charge:0
```

***CONTRol:CURRENT?<CR>**

Get charge current
Example of answer:

```
Charge current[mA]:1.95
```

***PARAmeter:SPIENable arg<CR>**

Get/ Set SPI interface enable flag

arg: 0 – disabled, 1 – enabled

Default value: 0

One bit; bit #1 of a 32-bytes field

(see *PARA:CONFIG); bytes 144–147 of the parameter block.
Example of answer for "*PARA:SPIEN?<CR>":

```
SPI enable (1 on, 0 off):0
```

3.8.3. Digital in and outputs

***CONTRol:DIGINPut?<CR>**

Get input state
Example of answer:

```
DigInp:7
```

***CONTRol:DIGOUTput arg<CR>**

Get/ Set digital output
Example of answer:

```
DigOut:94
```

***CONTRol:DIGPIN nr arg<CR>**

Set single digital output
Example for setting output 1 to 1:

```
*contr:digpin 1 1<CR>
```

3.8.4. Storage of Parameter and User Data

It is possible to read and store the parameter block with one command. Furthermore, it is possible to use a user data block.

***RDPARA<CR>**

Read parameter field (1024 bytes)

The last 2 bytes checksum over first 1022 bytes

***WRPARA<CR>**

Write parameter field (1024 bytes)

The last 2 bytes must be the checksum over the first 1022 bytes

***RDUSR2 beg end<CR>**

Read user data beg to end
(end - beg + 1)*1024 bytes + 2 bytes checksum

conditions:
beg < 64 , beg ≤ end, end < 64

***WRUSR2 blk<CR>**

Write user data block blk
1024 bytes + 2 bytes checksum
conditions:
blk < 64

***PARAmeter:CONFIG?<CR>**

Gets a 32-bytes field of configuration parameters in HEX-form.

Bit numbers and their meanings are:

Bit 0:enable parallel interface (only for Multichannel spec)
Bit 1:enable SPI interface (see *PARA:SPIEN)
Bit 2:enable lamp (see *PARA:LAMPE)
Bit 3:lamp low active (see *PARA:LAMPP)

Bit 4:enable interrupt1 (see *PARA:TRIG)
Bit 5:enable trigger enquire function (see *PARA:TRIG)
Bit 6:rising edge of trigger active (see *PARA:TRSL)
Bit 7:low gain PDA-sensor (Hamamatsu) (see *PARA:PDAG)

Bit 8:PLL active or high clock frequency (see *PARA:PLLC)
Bit 9:input range ADC AD9826 0–4 V; 1–2 V (see *PARA:ADCV)
Bit 10:power mode if no battery (SB1211) / enable SPI-break (see *PARA:PONMOD)
Bit 11:enable interrupt2

Bit 12:use power function for SC30 calculation
Bit 13:select triggered measurement / enable BMG trigger (see *PARA:BMGTRIG)
Bit 14:enable auto increment for row scan (S9840) (see *PARA:ENAROW98)
Bit 15:no power down ADC AD9840 (see *PARA:ADPW)

Bit 16:shutter available (see *PARA:SHUT)
Bit 17:optical trigger available (see *PARA:OPTIC)
Bit 18:enable flash modus (see *PARA:FLMOD)
Bit 19:select frame scan mode (ELIS) (see *PARA:FRAME)

Bit 20:select pixelsum for LIS_512 and LIS_256 (see *PARA:SUMLI)
Bit 21:select NDRO-Mode ELIS (not used!)
Bit 22:select splitted dark measurement (see *PARA:SPLITDARK)
Bit 23:select trigger output with line interpolation (FOS) (see *PARA:TRMODE)

Bit 24:not used
Bit 25:not used
Bit 26:not used
Bit 27:not used

Bit 28:not used
Bit 29:not used
Bit 30:not used
Bit 31:not used

The field is stored in bytes 144–147 of the parameter block
Example of answer for *PARAmeter:CONFIG?<CR>:

```
config:005c8194
```

Remark:	so, the 4 means, that the bit 2 is set; the 9 means, that bits 4 and 7 are set etc.

*PARAmeter:BASIC?<CR>

Get configured basic parameters
Example of answer:

```
For specbos 1211:
COMMAND:A(ask)
SB1211_M30626_SPEC_RAD VERSION 3.0.0 250413 sn:1262
CHANNELS            :    1
PIXEL PER LINE      : 2048
LAMP  PREHEAT TIME[ms]:  200
INTEGRATION TIME[ms]:  100
CHANNEL 1 FITx^0    :-5.542372E+02
CHANNEL 1 FITx^1    :1.689806E+00
CHANNEL 1 FITx^2    :-1.138420E-03
CHANNEL 1 FITx^3    :6.840287E-07
CHANNEL 1 FITx^4    :-1.569578E-10

For specbos 1201:
COMMAND:A(ask)
SB05S_1201_VIS_vP VERSION 2.0.0 250413 S/N 0934
CHANNELS            :    1
PIXEL PER LINE      :  512
LAMP PREHEAT TIME[ms]:  400
INTEGRATION TIME[ms]:  100
CHANNEL 0 FITx^0    :2.728866E+02
CHANNEL 0 FITx^1    :1.299100E+00
CHANNEL 0 FITx^2    :-4.534631E-04
CHANNEL 0 FITx^3    :-1.802557E-08
CHANNEL 0 FITx^4    :3.571662E-11
```

*PARAmeter:EXTENDed?<CR>

Get configured extended parameters
Example of answer:

```
fastscan time:50 ms
Image sensor:S9840-1024-OS4
gain value:1.00
offset value:250 mV
lamp enable
lamp low active
```

```
flash intervall:20 ms
flash length:20 µs
```

***PARAmeter:ALLPARA?<CR>**

Get a list of all parameters

Example of answer:

```
Firmware: SB1211_M30626_SPEC_RAD VERSION 3.0.0 250413
Spectrometer number: 2013547
Serial number: 1262
Channel count:  1
Pixel count:  2048
Image sensor nr.:  34
PDA gain (1-low, 0-high): 1
Frame mode ELIS (1-on, 0-off):0
PLL multiplayer (1-on, 0-off):1
ADC type (0-AD9826, 1-HT82V26, 2-HT82V36): 65535
ADC resolution:15
ADC range (1-2V, 0-4V):0
ADC Power down (1-on, 0-off):0
Baudrate: 921

Integration time[ms]:   100
Split tint [ms]:1000
Scan delay[ms]:   200
Border min [%]:   65
Border max [%]:   90
Fastscan [ms]:   50
Flash mode (0-off 1-on): 0
Flashlight intervall [ms]:    20
Flashlight length   [µs]:    20

Lamp enable:1
Lamp low:1
Trigger (0-off, 1-on, 2-enq):1
Trigger slope (1-falling, 0-raising):0
Shutter (1-exist, 0-not available):1
SPI enable (1-on, 0-off): 0

Offset value [mV]:   55
Gain   value:1.20
Channel 0 FITx^0    :-5.542372E+02
Channel 0 FITx^1    :1.689806E+00
Channel 0 FITx^2    :-1.138420E-03
Channel 0 FITx^3    :6.840287E-07
Channel 0 FITx^4    :-1.569578E-10
Calib number: 0
Measurement distance:2345
Scan direction (1-backw, 0-forw):0

Predefined function: 12
Predefined format:  7
Predefined exposition mode: 0
Predefined adaption mode: 1
Boxcar count: 1
Wave min [nm]: -554
Wave max [nm]: 1246
```

```
Wave beg limit [nm]:200
Wave end limit [nm]:1000
Wave beg defined [nm]:350
Wave end defined [nm]:1000
Wave step defined [nm]:1

Pon time:15
Umin time:5
ADC reference:4.510000
Cycle time:1000
Optical trigger:1
Pixelsum (1-on, 0-off):0
measurement distance:2345
config:0043819c

Fault pixel:

Firmware: PS2000_RU60 PMP VERSION 2.3.4 10.10.13
Spectrometer number: 6013002
Serial number: 1036
Channel count:  1
Pixel count:  2048
Image sensor nr.: 107
PDA gain (1-low, 0-high): 1
Frame mode ELIS (1-on, 0-off):0
ADC type (0-AD9826, 1-HT82V26, 2-HT82V36): 32639
ADC resolution:16
ADC range (1-2V, 0-4V):0
ADC Power down (1-on, 0-off):0
Baudrate: 921

Integration time[ms]:     2
Split tint [ms]:1000
Scan delay[ms]:     1
Border min [%]:   65
Border max [%]:   90
Fastscan [ms]:    0
Flash mode (0-off 1-on): 0
Flashlight intervall [ms]:    20
Flashlight length   [µs]:   100
Lamp enable:1
Lamp low:1
Trigger (1-on, 0-off):1
Trigger slope (1-falling, 0-raising):1
Shutter (1-exist, 0-not available):0
SPI enable (1-on, 0-off): 0

Offset value [mV]:  200
Gain   value:1.00
Channel 0 FITx^0    :1.346545E+02
Channel 0 FITx^1    :4.661252E-01
Channel 0 FITx^2    :-5.588027E-06
Channel 0 FITx^3    :-2.631100E-09
Channel 0 FITx^4    :-9.252857E-14
Measurement distance:32000
Scan direction (1-backw, 0-forw):0
```

```
Predefined function:  1
Predefined format:  7
Predefined exposition mode: 0
Predefined adaption mode: 0
Boxcar count: 1
Wave min [nm]:  135
Wave max [nm]: 1041
Wave beg limit [nm]:200
Wave end limit [nm]:1000
Wave beg defined [nm]:200
Wave end defined [nm]:955
Wave step defined [nm]:1

Enable row mode S9840 (1-on, 0-off):0
config:00b080dc
Fault pixel:
```

***CONFigure:DEFault<CR>**

Set the following measurement parameters to their default (i.e. predefined, set with *PARA–commands by the user or by the manufacturer) values (tint, av, format, function, wbeg, wend, expo, adaption, splitt, cyctim, cycmod, cycdiv, maxtint, darkmode (if *PARA:SHUT = 1)

3.8.5. Miscellaneous Parameters and Commands

***RST<CR>**

Software reset

***VERS?<CR>**

Get firmware version

64-bytes field of char (plain text ASCII); bytes 0–63 of the parameter block
Example of answer:

```
SB1211_M30626_SPEC_RAD VERSION 3.0.0 250413 (for specbos
1211)
SB05S_1201_VIS_vP VERSION 2.0.0 250413 (for specbos 1201)
PS2000_RU60 PMP VERSION 2.3.4 10.10.13 (for Versa PS2000)
```

ESC

Abort a running measurement

After any change of parameters this change has to be saved to make it permanent using the following command.

***PARAmeter:SAVE<CR>**

Write parameters to flash ROM
Example of answer:

```
ACK (06 hex)
```

The following commands are only valid for multichannel units:

***PARAmeter:CHANnels?<CR>**

Get number of channels

Default value: 1
Float; bytes 68–71 of the parameter block

Example of answer:
```
channel count:1
```

***PARAmeter:CHECKpar arg1 arg2<CR>**

Get/ Set threshold counts and integration time for check channel in multichannel system

arg1: threshold (lowest level)
arg2: check tint

arg1: unsigned short (bytes 122–123 in the parameter block)
arg2: unsigned short (bytes 120–121 in the parameter block)
Example of answer for *PARA:CHECK?<CR>
```
lowest level:4000
check tint:50
```

3.8.6. Correction Parameters

This command can be used to mark faulty pixel of a sensor array. The replacement data value will be calculated by the average of the two neighbouring pixel.

***PARAmeter:FAULTPIxel arg1 arg2 .... arg8<CR>**

Get/ Set faulty pixels

arg_n: 1 ... TotalPixels (see *PARA:PIX).

Note:	no combination of arguments can be consecutive numbers, e.g. *PARA:FAULTPI 10 20 778<CR> is legal, but *PARA:FAULTPI 100 35 99<CR> is not.

Array of 19 unsigned shorts; bytes 842–879 of the parameter block
Example of answer for "*PARA:FAULTPI ?<CR>":
```
Fault pixel:153 179
```

The following commands allow an improvement of the measuring results especially in the UV.

***PARAmeter:CORRENAble arg<CR>**

Get/ Set UV-correction enable flag.

arg:
0 – no correction
1 – if UV-correction polynomial start and end pixels set by *PARA:CORRRANGE are equal (beg = end), only offset correction is made. If beg ≠ end, both offset and polynomial corrections are made.
One bit; bit #29 of the 4-bytes-field; bytes 140–143 of the parameter block
Example of answer for *PARAmeter:CORRENAble?<CR>:
```
Correcture enable:1
```

Remark: Not for specbos 1201.

***PARA:CORRCOEFF n arg<CR>**

Get/ Set UV-correction polynomial coefficients k_n

n: 0 ... 3
arg: any valid float number

Array of 4 floats; bytes 576–591 of the parameter block.

Example of answer for "*PARA:CORRCOEFF?<CR>":

```
k0:-4.09834e-05
k1:1.58104e-07
k2:-1.97507e-10
k3:8.13864e-14
```

Remark: Not for specbos 1201.

*PARA:CORRRANGE beg end<CR>

Get/ Set UV-correction polynomial start and end pixels

Remark: Firmware version > 1.71.

beg: 1 ... TotalPixels (set by *PARA:PIX); beg < end
end: 2 ... TotalPixels; end > beg

If beg = end, UV-correction is disabled.

Remark: So, to make it work, one must set non-trivial combination of beg ≠ end and enable the correction with *PARA:CORRENA.

beg: float; bytes 604–607 of the parameter block
end: float; bytes 608–611 of the parameter block

Example of answer for *PARA:CORRRANGE?<CR>:

```
corrng:600 900
```

Remark: Not for specbos 1201.

*PARAmeter:CUTOFFENAble arg<CR>

Get/ Set correction of negative values flag.

arg:
0 – no correction
1 – All negative values in spectrum that occur after subtracting of the dark current from a measured spectrum are compensated.

Example of answer for *PARA:CUTOFFENA?<CR>:

```
Cutoff enable : 1
```

4. Measurements

All commands of the *MEAS category provide a measurement scan and the following output of data. Alternatively, it is possible to make a measurement without output and to output the data later.

4.1. Raw Data Measurements

Normally each light measurement will be combined with an individual dark measurement. This measure will suppress base line drift effects.

***MEASure:DARKspectra tint av format<CR>**

tint ≠ 0: Run dark measurement
tint = 0: last dark integration time will be used data storage in dark buffer

The user has to ensure that the optical input of the unit is darkened during the *MEAS :COMPDARK and *MEAS:DARK measurements (is closed with an external shutter or the lamp is switched off)

Example of answer for *MEAS :DARK 100 1 4<CR>:

```
ACK (06 hex)
BEL (07 hex)
552
551
544
549
...
...
```

Example of answer for *MEAS :DARK 100 1 2<CR>:

```
ACK (06 hex)
BEL (07 hex)
551 553 544 552 540 549 544 544 547 552 549 549 545 552 545
550 544 553 546 555 546 554 550 553 550 556 550 553 547 555
552 552 549 551 547 555 548 553 549 555 551 ...
```

Afterwards it is possible to execute light measurements:

***MEASure:LIGHT tint av format<CR>**

Run light measurement (exposed spectrum – opened external shutter or lamp switched on), data storage in light buffer, data output according to the selected format.

Afterwards it is possible to execute reference measurements:

***MEASure:REFERence tint av format<CR>**

Run reference measurement (difference between light measurement and dark spectrum obtained before), same integration time as for dark scan is obligatory. Data storage in reference buffer. Data output according to the selected format.

4.2. Predefined Measurements

It is possible to define a measurement in advance (see '10') and to execute it with one command. There are two possibilities to start a predefined measurement:

***READ format<CR>**

Initiate a measurement configured before and output of data

***INITiate<CR>**

Run a measurement configured before (without data output)

The following command makes a measurement with individual arguments, where only the measuring function was predefined.

***MEASure tint av format<CR>**

Run measurement with parameters and output of data
Raw data: ADC counts as unsigned integer word (2 bytes, MSB first)
Example of answer:

```
4714 4744 4669 4787 4652 4826 4744 4780 4745 4846 4724 4829
4739 4835 4708 4704 4546 4817 4652 4819 4748 4757 4711 4749
4696 4730 4708 4782 4740 4818 4655 4791 4687 4737...
```

4.3. Measurements Without Individual Darkscan

It is possible to avoid the individual dark scan for every light measurement. This technique reduces the measuring time, but it is necessary to proceed a dark measurement before a series of light measurements and the dark measurement has to be repeated from time to time to avoid drift effects.

***CONFigure:DARKMode arg<CR>**

Can be used to switch off the dark measurement after each light measuring scan

arg:
0 – dark measurement only before a set of measurements
(* only in combination with *MEAS :COMPDARK)
1 – dark measurement proceeded during every scan (normal mode)
Example of answer:

```
Auto dark:1
```

***MEASure:COMPDARK<CR>**

Run dark compensation (in connection with *CONF:DARKMode 0), data storage in dark buffer
Example of answer:

```
ACK (06 hex)
BEL (07 hex)
```

Afterwards normal light measurements with arbitrary integration time can be made.

4.4. Measurement of Reflexion and Transmission

There is only one command for execution of both transmission and reflexion measurements. Its meaning depends from the individual measuring set up.

It is necessary to proceed a dark and a light or reference scan (100 % line) before executing a transmission measurement.

***MEASure:TRANSmission format<CR>**

Run light measurement and calculate the ratio to the actual reference spectrum (both dark signal subtracted, uses same integration time).

Data storage in a temporary buffer (only for 14- or 15-bit setting).

4.5. Measurement of Radiometric and Colorimetric Data

4.5.1. Selection of Calibration File

***PARAmeter:CALIBNumber arg<CR>**

Get/ Set calibration file to be used for radiometric calculations

arg:
0 – accept signal of hall sensors "ab" (for sensor signal see *CONTR :MHEAD)

Sensor signal 00:uses calibration file 1
Sensor signal 01:uses calibration file 2
Sensor signal 10:uses calibration file 3
Sensor signal 11:uses calibration file 4

1 ... 12 – use the selected calibration file

Default value: 0

Float, bytes 160–163 of the parameter block.
Example of answer for "*PARA:CALIBN ?<CR>":

```
calib number:0
```

4.5.2. Adaption of Integration time

***PARAmeter:EXPOsure arg<CR>**

***CONFigure:EXPOsure arg<CR>**

Get/set predefined exposure mode (handling of integration time tint)

arg:
0 – uses previous tint (default value)
1 – always adaption of tint
2 – uses configured tint (see *CONF:TINT)

Unsigned char; byte 230 of the parameter block.
Example of answer for "*PARA:EXPO ?<CR>":

```
Predefined exposure mode:0
```

***PARAmeter:ADAPtion arg<CR>**

***CONFigure:ADAPtion arg<CR>**

Get/set predefined adaption mode

arg:
0 – no adaption if under or overexposition
1 – new adaption if overexposition
2 – new adaption if under- or overexposition
(the limits for over- and underexposure are set with *PARA:BORD)

Remark: If exposure is set to 1, the setting of adaption is without effect.

Default value: 1
Unsigned char; byte 231 of the parameter block.
Example of answer:

```
Predefined adaption mode:1
```

***PARAmeter:BORDer lowlimit hilimit<CR>**

Get/ Set low and high limits for the adaption of integration time (percent of

fullscale). It is used by *CONF:ADAP

lowlimit: 10 ... 97
hilimit: 11 ... 98

Default values:
for specbos 1201 – lowlimit = 70, hilimit = 98
for other devices – lowlimit = 65, hilimit = 90

hilimit:float, bytes 152–155 of the parameter block;
lowlimit:float, bytes 156–159 of the parameter block.

Example of answer for "*PARA:BORD?<CR>":

```
border:70 98
```

For testing purposes:

***CONFigure:LEVEL?<CR>**

Get the maximum of the previously measured spectrum in counts and %

Example of answer:

```
Level/cnt:15463
Level/%:0064
```

***MEASure:TIADAPT<CR>**

Adaption of integration time and output of tint and counts

Available from firmware version 3 (specbos 1211) or version 2 (specbos 1201)

Example of answer:

```
Tint:5.000
Tint:100.000
Tint:493.000
Tint:2710.000
```

4.5.3. Measurement and Output of Radiometric and Photometric Values

***MEASure:SPRADiance tint av format<CR>**

Spectral radiance, radiance, luminance, x, y, u', v' will be calculated in addition

Run spectral radiometric measurement.

Output in chosen wavelength range with a chosen step (set with *CONF:WRAN) in a chosen format.
Device must be calibrated, otherwise the command returns an error. The calibration file (and therefore the calibration mode) is chosen eigher automatically on the basis of Hall sensor signals or set manually (see *PARA:CALIBN).

All arguments are optional.

If format is omitted, its predefined value (see *PARA:FORM and *CONF:FORM) will be taken.
If tint or av are omitted, their previous values (see *CONF:TINT) will be taken.

Examples of answer:

```
format = 7 or 10
wbeg = 400, wstp = 1
Radiance mode
```

format = 4 or 9
wbeg = 400, wend = 450, wstp = 5
(that's why 11 values)
Irradiance mode

```
WL[nm] Sp Radiance[W/(sr*m^2*nm)]

400 4.46559534e-06
401 2.39596084e-05
402 8.00025646e-06
403 1.30781673e-05
404 1.25305132e-05
405 9.61789647e-06
406 1.41615910e-05...etc.
WL[nm] Sp Irradiance[W/(m^2*nm)]

1.10894134e-05
1.54641421e-05
2.42171282e-05
3.30743678e-05
6.60183432e-05
1.33644702e-04
2.90529802e-04
6.23540138e-04
1.24893908e-03
1.73620414e-03
1.63459103e-03
```

*MEASure:RADIOmetric tint av<CR>

Run radiometric measurement
Example of answer:

```
radiance[W/(sr*m^2)]:1.452e^2
```

*MEASure:PHOTOmetric tint av<CR>

Run photometric measurement.

For specbos 1201 wbeg and wend values set by *CONF:WRAN *must* be 380 and 780 nm correspondingly.

For specbos 1211 wbeg and wend values set by *CONF:WRAN *must* be within calibrated range (otherwise the command returns an error). Setting wbeg > 380 or wend < 780 is possible, but hardly makes sense. In this case the photometric value is calculated not from the entire spectrum, but from a part of it.
Example of answer:

```
luminance[cd/m^2]:2.417e^3
```

*PARAmeter:DISTAnce arg<CR>

Get/ set the measurement distance for luminous intensity measurement in mm.

arg: 1 ... 65535

Unsigned short; bytes 240–241 of the parameter block.
Example of answer for "*PARA:DISTA?<CR>":

```
Measurement distance:316
```

4.5.4. Measurement and Output of Colorimetric Values

*MEASure:CHROMXY tint av<CR>

Run xy measurement.
Example of answer:

```
x:0.4423
y:0.4067
```

*MEASure:CHROMUV tint av<CR>

Run u'v' measurement.
Example of answer:

```
u':0.2521
v':0.5229
```

*MEASure:DWLPE tint av<CR>

Run dominant wavelength and color purity measurement.
Example of answer:

```
dominant wavelength [nm]:583.0
colour purity:53.9
```

*MEASure:XYZ tint av<CR>

(Available from firmware version 1.53 (specbos 1211))

Run XYZ measurement
Example of answer:

```
X:104.5400
Y:108.7868
Z:79.8987
```

*MEASure:CCT tint av<CR>

Run Correlated Color Temperature (CCT) measurement
Example of answer:

```
color temperature[°K]:3007
```

*MEASure:CRI tint av<CR>

Run CRI measurement (according to CIE 13.3).
Example of answer:

```
color temperature[°K]:3016
color temperature of reference source:3016 °K
CRI _DC:3.109599e-04
CRI_Ra:99.2
CRI_R01:99.1
...
...
CRI_R14:98.3
```

*MEASure:EXTCRI tint av<CR>

Run CRI measurement, including R 15
Example of answer:

```
color temperature[°K]:3016
color temperature of reference source:3016 °K
CRI _DC:3.109599e-04
CRI_Ra:99.2
CRI_R01:99.1
...
...
CRI_R14:98.3
```

```
CRI_R15:99.5
colour purity:53.4
```

***MEASure:ALLVAlue tint av<CR>**

Run a measurement for radiometric-, photometric-, xy-, u'v'-values, dominant wavelength and color purity.
Example of answer:

```
radiance[W/(sr*m^2)]:1.452e^2
luminance[cd/m^2]:2.417e^3
x:0.4392
y:0.4053
u':0.2515
v':0.5222
```

4.6. Special Measurements

4.6.1. Synchronized Measurements of Modulated Sources

***PARAmeter:CYCTIM arg<CR>**

Get/ Set predefined cycle time in µs.

arg: 301 ... 65 535 µs (thus, the minimal synchronization frequency is 1000000 ÷ 65535 = 15.3 Hz and the maximal – 1000000 ÷ 301 = 3.3 kHz)

Default value: 1000
Unsigned short; bytes 234–235 of the parameter block.
Example of answer for "*PARA:CYCTIM ?<CR>":

```
Cycle time:3500
```

***PARAmeter:OPTICal arg<CR>**

Get/ Set availability of optical trigger (additional sensor in front plate of specbos 1211 to detect the repetition rate of pulsed sources). Is necessary for the commands *CONTR:CYCTIM , *CONF:CYCMOD and *CONF:CYCTIM.

arg:

0 – not available

1 – available

Default value:

1 for specbos 1211,

0 for other devices

One bit; bit #17 of a 32-bytes field

(see *PARA:CONFIG); bytes 144–147 of the parameter block.
Example of answer for "*PARA:OPTIC?<CR>":

```
Optical trigger:1
```

***CONTRol:CYCTIMe n t<CR>**

Get cycle time from optical trigger in ms with n averages and t timeout in ms.

n – number of modulation cycles to wait for
t – overtime in ms
It is recommended to use n = 200 and t = 5000 for display measurements.
Example of answer:

```
cyctim:10.23
```

***CONFigure:CYCMODe arg<CR>**

Get/ Set cycle mode.

Cycle mode is a special mode for the measurement of pulsed sources, the integration time is synchronized to the repetition rate of the source.
0 – tint as integration time in ms steps
1 – tint as cycle count with cycle time steps
Example of answer:

```
Cycle mode:0
```

***CONFigure:CYCTIMe arg<CR>**

Get/ Set cycle time in µs.

arg:301 ... 65 535 µs (thus, the minimal synchronization frequency is 1000000 ÷ 65535 = 15.3 Hz and the maximal – 1000000 ÷ 301 = 3.3 kHz)
Example of answer:

```
Cycle time:3500
```

***CONFigure:CYCDIV arg<CR>**

Get/ Set cycle divider (special solution to avoid overexposure, if the integration during one cycle ends with full scale; the cycle time will be divided into several synchronized scans until there is no overexposure any more)

Procedure is carried out automatically, only the result can be asked.

arg: 1 ... 20
Example of answer for "*CONF:CYCDIV ?<CR>":

```
CycDiv:3
```

4.6.2. Triggering of Measurements

Triggering of External Parts

*The following commands are used to set the status of the external trigger control (see Mini DIN connector pin 5 by *PARA:TRSL).*

***PARAmeter:TRIGger arg<CR>**

Get/ Set trigger mode

arg:
0 = disabled
1 = enabled: start of a configured measurement with hardware trigger (shortcut with switch or TTL signal), similar to the command *INITiate, last output:07 (measurement finished, data are ready)
2 = enquire: send enquire with hardware trigger

6 = complete a configured measurement with hardware trigger

Two bits; bits #4–5 of a 32-bytes-field (see *0:CONFIG); bytes 144–147 of the parameter block.
Example of answer for "*PARAmeter:TRIGger?<CR>":

```
Trigger on:1
```

***PARAmeter:TIMEOUT arg<CR>**

Set time-out for trigger mode 6 (time gap between trigger ready signal and received a hardware trigger in µs)

***CONFigure:EXPOsure arg<CR>**

Get/ Set handling of integration time

arg:
0 – uses previous tint (default value)
1 – always adaption of tint
2 – uses configured tint (see *CONF:TINT)
Example of answer:
```
Exposure:0
```

***CONFigure:ADAPtion arg<CR>**

Get/set adaption mode (is only valid for *CONF:EXPO0 or 2)

arg:
0 – no adaption if under or overexposition
1 – new adaption if overexposition
2 – new adaption if under- or overexposition
(the limits for over- and underexposure are set with *PARA:BORD)

Remark: If exposure is set to 1, the setting of adaption is without effect.

Example of answer:
```
Adaption:1
```

***PARAmeter:PREDARK arg<CR>**

Get/Set hold back time for triggered dark measurement

arg: 100 ... 65000 µs

Unsigned short; bytes 964–965 of the parameter block
Example of answer for *PARA:PREDARK ?<CR>:
```
PreDark:95000
```

***PARAmeter:PRELIGHT arg<CR>**

Get/Set hold back time for triggered light measurement

arg: 100 ... 65000 µs

Unsigned short; bytes 966–967 of the parameter block
Example of answer for *PARA:PRELIGHT ?<CR>:
```
PreLight:100
```

***MEASure:PRETRG<CR>**

Prepare fast triggered measurement in trigger mode 6

***PARAmeter:IGRTRIGger arg<CR> OBSOLETE since FW 2.3.4. Use *PARA:TRGCONF**

Get/ Set trigger source (three sensors = 0, igr = 1)
Example of answer:
```
IGR trigger (1-on, 0-off):   0
```

***PARAmeter:TRGPOSition pos func<CR>**

Get/ Set/ Delete igr trigger positions for Versa PS 2000

pos: 1 ... TotalEncoderIncrements-1 (set by *PARA:ENCTURN)
func:

D – dark

R – reference
T – transmission
C – delete all (pos must be = 0)
not specified – delete one position (existing position specified by pos)
Examples:

```
*PARA:TRGPOS 1000 R<CR> - set position #1000 as reference
*PARA:TRGPOS 1000<CR> - delete position #1000
*PARA:TRGPOS 0 C<CR> - delete all positions
```

19 positions are possible.

pos: array of 20 unsigned shorts; bytes 902–941 of the parameter block. Unused positions are sudden numbers.
func: array of 20 unsigned chars (values are 0 – unused position, 1 – reference, 2 – transmission, 3 – dark); bytes 942–961 of the parameter block.
Example of answer of "*PARA:TRGPOS?<CR>":

```
1       1088      D
2       1646      R
3       2359      T
```

*PARAmeter:ENCTURN arg<CR>

Get/ Set total encoder increments for Versa PS 2000

arg: 100 ... 10000

Unsigned short; bytes 962–963 of the parameter block
Example of answer for *PARA:ENCTURN?<CR>:

```
Encoder turn:3600
```

*PARAmeter:TRGCONF arg<CR>

Get/ Set trigger mode

arg:
0 – no trigger
1 – normal trigger; measure directly after a trigger signal comes.
2 – send enquire for a software after a trigger signal comes without measurement.
3 – trigger with 3 signals: for dark, reference and transmission; measure directly after a trigger signal comes.
4 – encoder; measure directly after a trigger signal comes.
5 – fast triggering; if trigger signal comes periodically, the cycle time is calculated and measurements are started with a hold back time set by *PARA:PREDARK and *PARA:PRELIGHT.

Note:	This parameter is a general setting of the used trigger type. To switch triggering on/off use *PARA:TRIG 0 (i.e. **not** *PARA:TRGCONF 0).

Unsigned short; bytes 80–81 of the parameter block
Example of answer for "*PARA:TRGCONF?<CR>":

```
TriggerConf:4
```

*PARAmeter:TRGDELay arg<CR>

Get/ Set igr trigger delay
Example of answer of "*PARA:TRGDEL?<CR>":

```
trigger delay:  5000
```

***PARAmeter:TRMODE arg<CR>**

Get/ Set trigger output with line interpolation (FOS) flag

arg: 0 – off; 1 – on

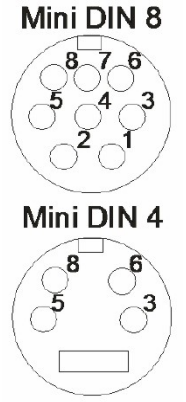
One bit; bit #23 of a 32-bytes-field (see *0:CONFIG); bytes 144–147 of the parameter block.

Example of answer for "*PARA:TRMODE?<CR>":

```
interpolation by trigger:1
```

***PARAmeter:TRSLope arg<CR>**

Pin out (view to female connector Mini DIN4 or Mini DIN8 on instrument):



Get/ Set the trigger slope

arg:
0 – triggering with switch opening/ rising TTL signal
1 – triggering with switch closing/ falling TTL signal
2 – TTL UART receive input (optional)
3 – Lamp out 5 V CMOS signal **(flash lamp trigger)**
5 – Meas. trigger input
6 – VCC +5 V
7 – TTL UART transmit output (optional)
8 – Ground

Default value: 0

One bit; bit #6 of a 32-bytes-field (see *0:CONFIG); bytes 144–147 of the parameter block.

Example of answer for "*PARA:TRSL?<CR>":

```
Trigger falling edge:0
```

4.6.3. Triggering with Encoder

***CONTRol:IGRVOLume?<CR>OBSOLETE see *PARA:ENCTURN**

Get increment volume of IGR

Example of answer:

```
IGR_Vol:1025
IGR_positions:5
```

***CONTRol:IGRPOSition?<CR>**

Get IGR position

Example of answer:

```
IGR_Position:560
```

4.6.4. Measurement of Flash Lamps

***PARAmeter:FLAShlight interval pulselength<CR>**

Get/ Set parameters for control of an external flash lamp

interval: 20 ... 65535 ms
pulselength: 1 ... 65535 µs
Number of flashes = tint/ flash interval

Default values: undefined
interval: unsigned short; bytes 244–245 of the parameter block;
pulse length: unsigned short; bytes 246–247 of the parameter block.

Example of answer for "*PARA:FLAS?<CR>":

```
FlashPara:20 20
```

***PARAmeter:FLMODe arg<CR>**

Get/ Set enable flash mode flag

arg:
0 – no flash
1 – scan will be executed with control an external flash lamp, the real integration time depends on flashligth parameters.

Default value: 0
One bit; bit #18 of a 32-bytes-field (see *0:CONFIG); bytes 144–147 of the parameter block.
Example of answer for "*PARA:FLMOD?<CR>":

```
Flash mode:0
```

4.6.5. Measurement of FFT

***PARAmeter:FFTPARAmeter freq ffttint count<CR>**

Get/Set parameters for fast measurement.

freq: measuring frequency ([Hz] = [measurements pro second])
ffttint: integration time in **micro**seconds
count: number of spectra to wait for; should be a power of 2
Example of answer:

```
Frequency:3000
FFT_Tint:10
FFT_Count:1024
(Frequency in Hz, FFT_Tint in µs)
```

***MEASure:FSTMEASure<CR>**

Run fast dark measurement with FFT-parameters.
Real frequency is depending on the setting frequency, the integration time and the pixel counts.
If FFT_Count = 0, a continuous measurement will be run.
The measurement will be stopped after all FFT-Counts or a "break command" (ESC) was given.
Output is binary data as 16-bit integers for example,
512 bytes = 256 pixels.

4.6.6. Measurement of Temperature

***MEASure:TEMPErature<CR>**

Measure PDA temperature, if a temperature sensor is present.
If it is not present, the output is a senseless (usually negative) number like -33.3.
Example of answer for "*meas:tempe<CR>":

```
Temperature:32.6
```

5. Fetch of Measured Data

A *FETCH command can only be used if the appropriate measurement was proceeded before.

***FETCH?<CR>**

Get a help list of the fetch commands

***FETCH format<CR>**

Output of previous measurement

***FETCH:LIGHT format<CR>**

Output of exposured spectrum values

***FETCH:DARK format<CR>**

Output of dark spectrum values

***FETCH:REFErence format<CR>**

Output of reference values

***FETCH:TRANSmission format<CR>**

Output of transmission values

***FETCH:SPRADiance format<CR>**

Output of spectral radiance values

***FETCH:RADIOmetric<CR>**

Output of radiometric value

***FETCH:PHOTOmetric<CR>**

Output of photometric value

***FETCH:CHROMXY<CR>**

Output of xy values

***FETCH:CHROMUV<CR>**

Output of u'v' values

***FETCH:DWLPE<CR>**

Output of dominant wavelength and color purity

***FETCH:XYZ<CR>**

(Available from firmware version 1.53 (specbos 1211))

Output XYZ values

***FETCH:ALLVAlue<CR>**

Output of radiometric-, photometric-, xy- and uv values, dominant wavelength and colour purity

***FETCH:CCT<CR>**

Output of CCT , can only be applied after *CALC:CCT or *CONF:FORM 13 and *INIT

Example of answer:

```
color temperature[°K]:3007
```

***FETCH:CRI<CR>**

Output of CRI values, can only be applied after *CALC:CRI or *CONF:FORM 14 and *INIT

Example of answer:

```
color temperature of reference source:
2968 K
CRI _DC:3.681963e-04
CRI_Ra:99.3
CRI_R01:99.2
CRI_R02:99.8
CRI_R03:99.1
...
```

***FETCH:EXTCRI<CR>**

Output of CRI values, can only be applied **after** *CALC:CRI **or** *CONF:FORM 14 and *INIT, includes R15

Example of answer:

```
color temperature of reference source:
2968 K
CRI _DC:3.681963e-04
CRI_Ra:99.3
CRI_R01:99.2
CRI_R02:99.8
CRI_R03:99.1
...
CRI_R15:98.7
```

***FETCH:LDCURRent<CR>**

Output last measured laser diode current

Example of answer:

```
29.5
```

***FETCH:CYCNUM<CR>**

Get time points, where the optical trigger has toggled by the last *CONTR :CYCTIM.

If the measurement was successful, it returns the number of triggering points (set by the **n** argument of *CONTR :CYCTIM) and the list of 200 time points in special processor units. Returns always 200 values, although only **n** of them have sense.

To calculate the difference between 2 time points in milliseconds, use: $\Delta t\ [ms] = (point_2 - point_1) \div 29481$.

Example of answer:

```
    50
4294967264
4294361234
4293790609
4293355451
4292923720
4251177817
4250618820
4250134726
4249623991
etc.
```

6. Calculations After Proceeding a Measurement

The calculation commands use the measuring spectral data obtained before.

***CALCulate:LINT:DARK wbeg wend wstp<CR>**

Linear interpolation of dark values.

wbeg and wend must be within the range set by *PARA:WAVLIMIT, and wend > wbeg

1 nm ≤ wstp ≤ 20 nm, integer; if not integer, will be rounded

All arguments are optional. If they are not specified, values set by *CONF:WRAN are used.

Example of answer:

```
400.0 4775.31
402.0 4764.11
404.0 4752.92
406.0 4786.45
408.0 4829.23...
```

***CALCulate:LINT:LIGHT wbeg wend wstp<CR>**

Linear interpolation of light values (see *0:LINT:DARK)

***CALCulate:LINT:REFER wbeg wend wstp<CR>**

Linear interpolation of reference values (see *CALC:LINT:DARK)

***CALCulate:LINT:TRANS wbeg wend wstp<CR>**

Linear interpolation of transmission (see *CALC:LINT:DARK)

***CALCulate:SPLIN:DARK wbeg wend splinestp<CR>**

Spline interpolation of dark values

0.1 nm ≤ splinestp ≤ 20 nm

The step count is limited by the available memory (set by pixel count of sensor). If exceeded, then error 230 ... 232.

Example of answer:

```
400.0 4.94943506e+03
402.0 4.78587012e+03
404.0 4.74870703e+03
406.0 4.78368457e+03
...
```

***CALCulate:SPLIN:LIGHT wbeg wend splinestp<CR>**

Spline interpolation of light values (see *CALC:SPLIN:DARK)

***CALCulate:SPLIN:REFER wbeg wend splinestp<CR>**

Spline interpolation of reference values (see *CALC:SPLIN:DARK).

***CALCulate:SPLIN:TRANS wbeg wend splinestp<CR>**

Spline interpolation of transmission values (see *CALC:SPLIN:DARK).

***CALCulate:RADIO metric wbeg wend<CR>**

Run radiometric calculation.

Example of answer:

```
radiance[W/(sr*m^2)]:1.452e^2
```

***CALCulate:PHOTO metric wbeg wend<CR>**

Run photometric calculation.
Example of answer:
```
luminance[cd/m^2]:2.417e^3
```

***CALCulate:CHROMXY wbeg wend<CR>**

Run xy calculation.

***CALCulate:CHROMUV wbeg wend<CR>**

Run u'v' calculation.

***CALCulate:DWLPE wbeg wend<CR>**

Run dominant wavelength and color purity calculation.

***CALCulate:XYZ wbeg wend<CR>**[2]

Run XYZ calculation.
Example of answer:
```
X:2.3666
Y:2.3702
Z:1.1399
```

***CALCulate:ALLVAlue wbeg wend<CR>**

Run calculation of radiometric-, photometric-, xy-, uv-values as well as of dominant wavelength and color purity.

***CALCulate:CCT wbeg wend<CR>**[3]

Run Correlated Color Temperature (CCT) calculation.

***CALCulate:CRI temp<CR>**

Run CRI measurement (the argument temp is used for the reference source calculation, without argument the measured CCT will be used), calculation according to CIE 13.3 publication.
Example of answer:
```
color temperature of reference source:4000 °K
CRI _DC:2.984050e-02
CRI_Ra:79.0
CRI_R01:84.9
CRI_R02:95.4
...
...
CRI_R14:82.3
```

2(Available from firmware version 1.53 (specbos 1211))
3New calculation of Spectral Radiance, Radiance, Luminance, x, y, u', v' only if the arguments are different from the previous measurement

*CALCulate:EXTCRI temp<CR>

Run CRI measurement (the argument temp is used for the reference source calculation, without argument the measured CCT will be used), including R15

Example of answer:

```
color temperature of reference source:4000 °K
CRI _DC:2.984050e-02
CRI_Ra:79.0
CRI_R01:84.9
CRI_R02:95.4
...
...
CRI_R15:79.8
```

7. Help-, Status- and Error-Messages

7.1. Help commands

The help commands give all information to the SCPI-commands.

***HELP:PARA<CR>**

Get a summary of all parameter commands
Example of answer:

```
*RST <CR>:softwarereset
*IDN ?:get device ID
*vers?:get firmware version
*PARAmeter:CHANnel?:get channel count
*PARAmeter:PIXel:get/set pixel count
*PARAmeter:SENSor:get/set sensor type
*PARAmeter:SDELay:get/set scan delay
```

***HELP:PARA:<cmd2><CR>**

Get the description of the selected parameter command (cmd2)
Example of answer for "HELP:PARA:tint<CR>":

```
predefined integration time/ms,
valid values 1 ... 60 000
```

***HELP:CONF<CR>**

Get a summary of all configuration commands
Example of answer:

```
*CONFigure:TINT ?<CR>:get last integration time
*CONFigure:TINT arg<CR>:set preset integration time
*CONFigure:AVERage?<CR>:get last average counts
...
```

***HELP:CONF:<cmd2><CR>**

Get the description of the selected configuration command (cmd2)
Example of answer for "HELP:CONF:expos<CR>":

```
exposition mode
0-exposition with last integrationtime
1-exposition with new adaption integration time
2-exposition with configured integration time
```

***HELP:CONTR<CR>**

Get a summary of all control commands
Example of answer:

```
*CONTRol:LAMP ?<CR>:get lamp/shutter state
*CONTRol:LAMP arg<CR>:set lamp/shutter on/off
...
```

***HELP:READ<CR>**

Get a description of the read commands
Example of answer:

```
*READ format<CR>:initiate and output a pre configured
measurement
...
```

***HELP:FETCH<CR>**

Get a summary of all fetch commands

Example of answer:

```
*FETCH   format<CR>:output last measurement
*FETCH:LIGHT  format<CR>:output light values
*FETCH:DARK  format<CR>:output dark values
*FETCH:REFErence format<CR>:output reference values
...
```

*HELP:MEAS<CR>

Get a summary of all measurement commands

Example of answer:

```
*MEASure tint av format<CR>:run measurement with parameters
*MEASure:DARKspectra tint av format<CR>:run dark measurement
*MEASure:LIGHTspectra tint av format<CR>:run light
measurement
*MEASure:REFERence tint avformat<CR>:run reference
measurement
*MEASure:SPRADIance tint av format<CR>:run spectro
radadiometric measurement
...
```

*HELP:CALC<CR>

Get a summary of all calculate commands

Example of answer:

```
*CALCulate:SPRADIance wbeg wend<CR>:run spectral radiance
calculation
*CALCulate:RADIOmetric wbeg wend<CR>:run radiometric
calculation
*CALCulate:PHOTOmetric wbeg wend<CR>:run photometric
calculation
...
```

*HELP:STAT<CR>

Get a summary of all status commands (only for specbos 1211)

Example of answer:

```
*STATus:ERRor?<CR>:get last error code
*STATus:TXTERror?<CR>:get last error text
*STATus:ENQUire?<CR>:get enquire state
*STATus:CON4?<CR>:get status flags
*STATus:EXPO ?<CR>:get exposition state (1-under 2-
overexposure)
...
```

*HELP:STAT:ERR<CR>

Get a summary of error codes

Example of answer:

```
Error codes:
 0:no error
 4:command error
 7:error password
 8:digit error
10:error argument 1
11:error argument 2
...
For the full list of error messages – see meaning of error
codes.
```

***HELP:STAT:ENQU<CR>**

Get a summary of enquire codes (see *STAT:ENQU)
Example of answer:

```
EnqCode:
0:EnqCode enquire none
1:EnqCode trigger event
2:EnqCode switch power off
...
```

The following commands display a table with all available commands of the category:

***PARAmeter?<CR>**

Get a help list of the parameter commands
Example of answer:

```
*RST <CR>:softwarereset
*IDN ?:get device ID
*VERS ?:get firmware version
*PARAmeter:CHANnel?:get channel count
*PARAmeter:PIXel:get/set pixelcount
*PARAmeter:SENSor:get/set sensor type
*PARAmeter:SDELay:get/set scan delay
*PARAmeter:ADCType:get/set adc parameters
etc.
```

***CONFigure?<CR>**

Get a help list of the configuration commands

***CALCulate?<CR>**

Get a help list of the calculation commands

7.2. Status Commands

The status commands are used to get the information of the error and configuration conditions.

***STATus:ERRor?<CR>**

Get the error code (see '7' table_errorcodes)
Example of answer:

```
Error Code:0
```

***STATus:TXTERRor?<CR>**

Get error code and description of the error
Example of answer:

```
0:error none
```

***STATus:EXPOsure?<CR>**

Get error code and description of the error

0 – correct exposure

1 – under exposure

2 – over exposure

3 – not used

4 – overexposure in cycle mode (still overexposure with *CONF:CYCDIV 20)

5 – combination of 4 and 1

6 – combination of 4 and 2

Example of answer:

```
Exposition state:0
```

***STATus:ENQUire?<CR>**

Get enquire state

An enquire code (0x05) will be send, if any event occurs. With this command the reason of the enquire event can be detected.

0 – none

1 – trigger event with *PARA:TRIG 2

2 – device has been switched off

3 – switched off device due to low battery

4 – measurement head has been changed

5 – laser state has been changed

6 – not used

7 – on time has been expired; is sent shortly before the device switches off (see *PARA:PONTIM)

8 – warning low battery

9 – break scan due to overexposure

10 – break scan due to instability; is sent if using split time, if there is overexposure in one of the split parts

11 – time out during cycle time measurement

12 – measuring error during cycle time measurement

Example of answer:

```
EnqCode:1
```

***STATus:BATTery?<CR>**

Get the information, whether a battery is available (0 – no battery, 1 – battery available)

Example of answer:

```
Battery available:1
```

7.3. Structure of Data Stream

All data transmitted via UART (RS232 , no parity, 1 stop-bit, no Handshake; USB virtual COM port; BT)

Format = 1

(L/H binary output without length and checksum)

All data transmitted as 16 bit word, low-byte first (Little Endian)

Byte	Value	Definition
0	*xx yy*	*first Pixel*
2	*xx yy*	*second Pixel*
		:
		:
*n*2*	*xx yy*	*last Pixel*

n:number of pixel per line

Format = 2

(ASCII output, separated by space)

938 930 919 967 944 911 941 971 933 941 959 941 903 947 952
951 916 945 947 951 957 941 954 961 947 951 959 962 977 939
960 981 977 966 949 962 951 988 1012 978 1022 1072 1072 1091
1107 1144 1130 1138 1209 1180 1244 1219 1264 1287 1281 1336 1312
1407 1406 1383 1451 1440 1451 1450 1472 1451 1497 1494 1511 1547
1508 1531 1546 1534 1554 1546 1559 1523 1513 1561 1566 1552 ....

Format = 3

(L/H binary output with length and checksum)

All data transmitted as 16 bit word, low-byte first (Little Endian)

Byte	Value	Definition
0	xx yy	length
2	xx yy	first pixel
4	xx yy	second pixel
		:
		:
2+ n*2	xx yy	last pixel
4+n*2	xx yy	checksum

n: number of pixel per line

Format = 4

(ASCII output, separated by <CR>)

819
858
807
841
793
.....

Format = 5

(H/L binary output without length and checksum)

All data transmitted as 16 bit word, high-byte first (Big Endian)

Byte	Value	Definition
0	xx yy	first pixel
2	xx yy	second pixel
		:
		:
n*2	xx yy	last pixel

n: number of pixel per line

Format = 6

(H/L binary output with length and checksum)

All data transmitted as 16 bit word, high-byte first (Big Endian)

Byte	Value	Definition
0	xx yy	length
2	xx yy	first Pixel

4	xx yy	second pixel
		:
		:
2 + n*2	xx yy	last pixel
4 + n*2	xx yy	checksum

n: number of pixel per line

Format = 7

(ASCII output with wavelength, separated by <CR>)

250.1	4153
250.5	4118
250.9	4126
251.4	3690
251.8	2425
252.2	1765
252.6	1452
253.0	1317
253.4	1277
253.8	1345
254.2	1525
254.6	1680
255.0	1728
.....	.....

Format = 9

(Line interpolated ASCII output without wavelength, separated by <CR>)

2316.75
2405.70
2505.48
2592.03
2614.35
2630.52
2584.60
2737.96
2814.95
....

Format = 10

or by command “*CALC:LINT:dark”
(Line interpolated ASCII output with wavelength, separated by <CR>)

400.0	2316.75
405.0	2405.70
410.0	2505.48
415.0	2592.03
420.0	2614.35
425.0	2630.52
430.0	2584.60
435.0	2737.96
440.0	2814.95
...	....

Format = 11

(H/L binary integer interpolated output with length and checksum)

All data transmitted as 16 bit word, high-byte first

Byte	Value	Definition
0	xx yy	length
2	xx yy	first wavelength value
4	xx yy	second wavelength value
		:
2+n*2	xx yy	last wavelength value
4+n*2	xx yy	checksum

n: 1+(wavmax-wavmin)/wavstep

Format = 12

(H/L binary float interpolated output with length and checksum)

All data transmitted as 32 bit word, high-byte first

Byte	Value	Definition
0	xx yy	length
2	uu vv xx yy	first wavelength value
6	uu vv xx yy	second wavelength value
		:
2+n*4	uu vv xx yy	last wavelength value
6+n*4	xx yy	checksum

n: 1+(wavmax-wavmin)/wavstep

Line interpolated output by dual trigger (binary output)

Byte	Value	Definition
1	<ACK>	command ok
2	<BEL>	measurement complete
3	cc	trigger type 0-dark, 1-reference, 2-transmission
5-6	xx yy	wavbegin
7-8	xx yy	wavend
9	ss	wavstep
10-11	xx yy	value by wavbegin
		:
10+n*2	xx yy	value by wavend
12+n*2	xx yy	checksum

n = 1 + (wavend – wavbegin) / (wavstep)

Format = 13

(Line interpolated ASCII output without wavelength, separated by <CR>)

789
804
799
794
806
781
….

7.4. Calibration Commands

*** CALIBrate?**

Get a help list of the calibration commands

***CALIBrate:GET filenr<CR>**

Read selected (filenr) calibration file

Example of answer:

```
Radiance
Remark
380
780
1
538
3.637039e+03
3.690917e+03
3.758527e+03
…..
```

***CALIBrate:SET filenr<CR>**

Write selected (filenr) calibration file

***CALIBrate:CLEAr filenr<CR>**

Clear selected (filenr) calibration file

***CALIBrate:EXECute lampnr<CR>**

Run calibration with selected lamp file to head or manually selected calibration file (with integration time adaption)

***CALIBrate:STATus?<CR>**

Get calibration status (hex number of bit code), file n = 2^(n-1)
Example:

```
23:file 1, file 2 and file 6 are valid
```

*** CALIBrate:LAMP lamp number<CR>**

Output of stored data of lamp files
(number 1..3, 0 – user lamp), wavelength
step 1 nm
Example of answer:

```
Radiance
Remark
380
780
1
1.52205391e+01
1.78924637e+01
2.09196072e+01
2.41626911e+01
….
```

*** CALIBrate:LAMP:SET<CR>**

Write lamp file (user lamp, 0); format – see above

8. Default Settings for JETI Instruments

(settings which should not be modified by the user are marked in grey)

Parameter	Meaning	Default settings		Unit	Command	Remark
		scb1201	scb1211			
tint	Integration time (exposure time of detector)	100	100	ms	*PARA: TINT, *CONF: TINT	minimum: *CONF: MINTINT? (depending from detector) maximum: 60 000 ms
split time	splitting time, due to high dark signal at long integration times	5000	1000	ms	*PARA: SPLIT	used for ELIS and S9840 detectors, split of integration into several parts, allows to measure with long tint
av	Number of scans for averaging	1	1		*CONF: AVER	up to 10000 average scans
format	data output format	7	7		*PARA: FORM, *CONF: FORM	
function	kind of measurement	12	12		*PARA: FUNC, *CONF: FUNC	
exposition	defines, which integration time will be used	0	0		*PARA: EXPO, *CONF: EXPO	use previous integration time
sens	type of sensor	28	34		*PARA: SENS	[4]
ADCR	resolution of ADC	14	15	bit	*PARA: ADCR	Can be set to 14, 15 or 16 bit (16 bit is not allowed for radiometric and transmission/ reflexion calculations)
ADCV	Input voltage of ADC	2	4	V	*PARA: ADCV	2 or 4 V
ADPW	Power down modus of ADC	1	0		*PARA: ADPW	on
Baud	interface Baud rate	921600	921600[5]	bd	*PARA: BAUD	

4 ELIS for specbos1201 and 9840 for specbos1211, SDC_BB
5 115200 for specbos1211-RS and specbos1211-BT

Parameter	Meaning	Default settings		Unit	Command	Remark
		scb 1201	scb1211			
Flash length	Flashlight length	20	20	ms	*PARA: FLASH	
Flash interval	Flashlight interval	20	20	ms	*PARA: FLASH	
Flash mode	Flash mode	0	0		*PARA. FLMOD	
Lamp polarity	Polarity of lamp/ shutter control	1	1		*PARA. LAMPE	Lamp/ shutter low active
Lamp enable	Set lamp/ shutter control on/ off	1	1		*PARA. LAMPP	Lamp/ shutter signal enabled
Fast scan	Continuous scan without data output	50	50	ms	*PARA. FAST	
offset	ADC offset	-	-		*PARA. OFFS	Individually adjusted
pix	number of pixel	512	2048		*PARA: PIX	will be changed, if pixel binning is used, wavelength fit has to be changed too
wavstep	Predefined wavelength step	5	5	nm	*PARA: WAVSTEP	valid after program start
wavend	predefined end of wavelength range	780	1000	nm	*PARA: WAVEND	valid after program start
wavbeg	predefined begin of wavelength range	380	350[6]	nm	*PARA: WAVBEG	valid after program start
Fit n	Wavelength fit coefficients	-	-	-	*PARA: FITN	individually calibrated for each unit (has to be changed if pixel binning is changed, see sens)
adaption	defines the behaviour in case of over or under exposure	1	1		*PARA: ADAPT, *CONF: ADAPT	new adaption of integration time only in case of over exposure

6 specbos1211UV: 250nm

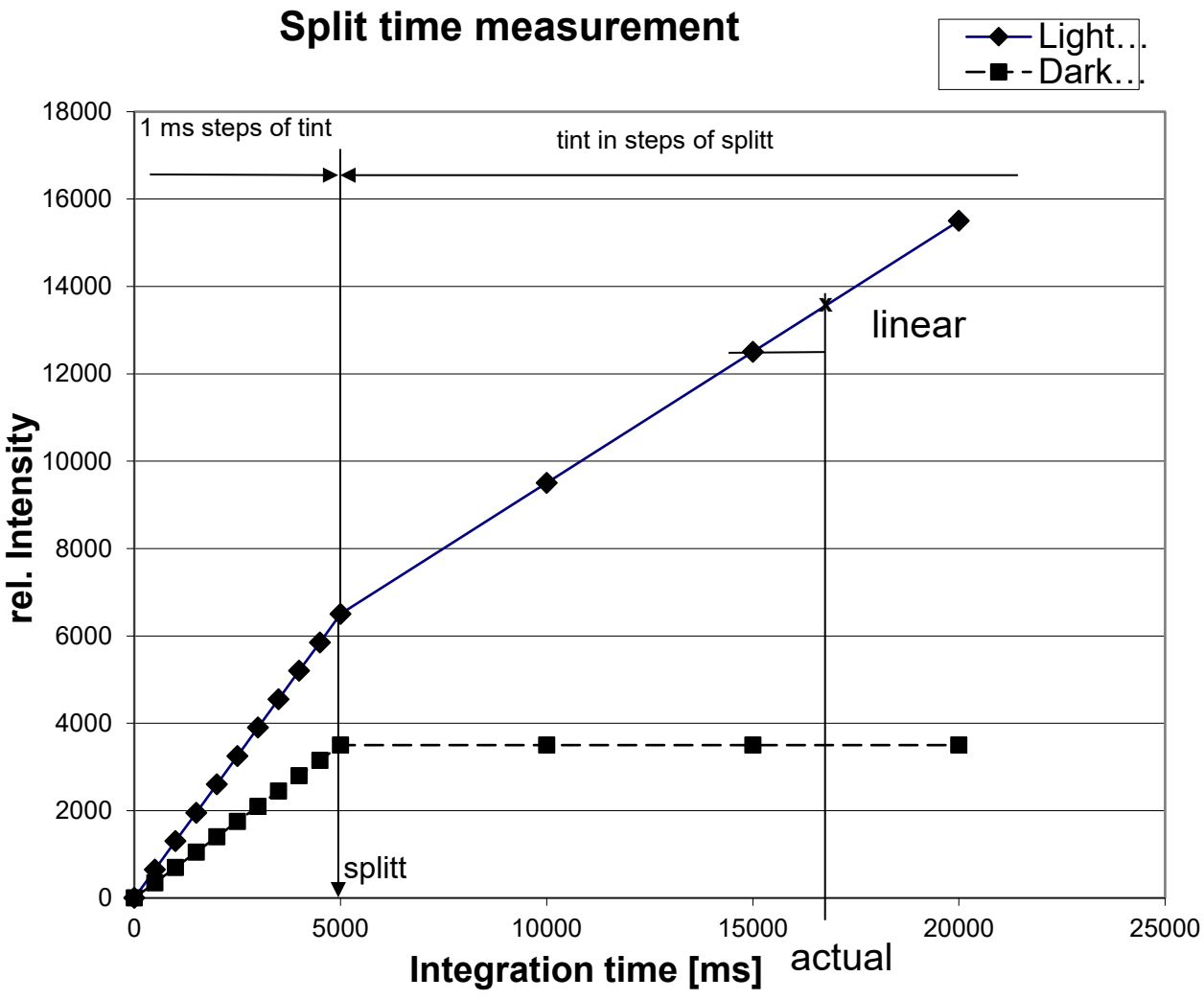
9. Explanation of Split Time Concept

Some low light applications demand a long an integration time of the detector. But detector arrays as S9840 (specbos 1211) and ELIS (specbos 1201) already saturate by the dark signal of such long integration times. Therefore, the split of the integration time was introduced. Above the preset split time (splitt) the measurement scan is divided into integer parts of the split time. E.g. 17 s will be split into three measurements of 5 s in case of splitt = 5000 ms (see diagram below).

The dark signal is calculated by averaging the individual split scans. Therefore, the dark signal does not rise any more if *tint* > *splitt*!

The light signal is calculated by a summation of the different scans. Furthermore, there is done a linear interpolation to get the data for integration times between multiples of the split time.

The following diagram shows the scheme of the split time measurement.



10. Examples of Measurement Procedures using Firmware Commands

The following code sequences demonstrate the usage of the commands for five selected applications. Two examples use raw data operations (non-calibrated remission and transmission measurements) and three examples are radiometric measurements (continuous source and pulsed source measurements).

All examples require a shutter for control of the light source for dark measurements.

10.1. Raw Data Operations

10.1.1. Raw Data Emission Measurement

Task:

- Measurement of raw data of a light source
- Application of default values except wavelength range, step width and format
- The appropriate integration time *tint* has to be determined before
- The necessary dark measurement will be included automatically. That's not the case if a dark scan with the same *tint* was already proceeded
- Output of interpolated wavelengths and counts in the wavelength range of 400 to 600 nm with a step width of 5 nm

Program sequence:

```
*CONF:WRANGE 400 600 1<CR>   / setting of wavelength range and step
*CONF:FORM  10<CR>           / output format: interpolated ASCII values with
wavelengths
*MEAS :REFER  tint<CR>       / start of measurement and data output
```

Output:

```
400   877/ first wavelength and value
401   921
402   929
403   959
404   1022
405   1074
…
597   1183
598   1068
599    959
600    806/ last wavelength and value
```

Variations:

- Wavelength output can be suppressed by changing *CONF:FORMASCII format can be changed into binary short (format 11) or binary float (format 12) – faster data transfer
- The integration time tint can be determined automatically using the value 0 (*MEAS:REFER 0<CR>).

10.1.2. Transmission Measurement (Pre-configured)

Task:

- Reference measurement (100 % line) and sample measurement with a fixed integration time tint
- Dark measurement is obligatory
- All measurements with 3 average scans
- Transmission data output from 400 to 700 nm in steps of 5 nm in promille in binary short format

Program sequence:

```
*CONF:FORM11<CR>                 / data output in binary short format
*CONF:FUNCT 3<CR>                / setting: reference spectrum
*CONF:WRANGE 400 700 5<CR>       / setting of wavelength range and step
*CONF:TINTtint<CR>               / setting: tint
*CONF:AVER  3<CR>                / setting of average scan number
*INIT<CR>/ start of measurement (adaption of tint and final dark scan)
Insertion of measuring object
*MEAS :TRANS <CR>                / measurement and output of data
```

Output:

Line interpolated (binary output)

```
0x00    / length high byte
0x7A    / length low byte
0x03    / value 400nm high byte (810)
0x2A    / value 400nm low byte
  …
0x02    / value 700nm high byte (763)
0xFB    / value 700nm low byte
0x05    / checksum high byte
0x43    / checksum low byte
```

10.2. Radiometric Measurements

10.2.1. Simple xyY Measurement

Task:

- Measure the xyY values of a light source

Program sequence:

```
*CONF:EXPO 1<CR> / setting: adaption of integration time
*CONF:WRAN 380 780 1<CR> / setting: wavelength range and step width
*MEAS :PHOTO <CR> / tint adaption, light and dark measurement and output of
photometric value
*FETCH :CHROMXY <CR> / output of x y
```

Output:

```
luminance [cd/m2]: 9.001e^2
x: 0.3127
y: 0.3290
```

10.2.2. Continuous Source Measurement

Task:

- Measure the xyY values and the radiometric spectrum of a light source in Irradiance mode (with attached cosine diffusor)

- Delay of 1 s between initialization and start of measurement (e.g. heating up time of measuring object)
- Determination of tint in reduced limits (between 40 and 50 % of saturation level – to speed up the measurement)
- Adaption of tint in case of over and under exposure
- Average over 3 scans
- Additional output of Color Rendering Index for with a reference source of Illuminant A, exposure level and serial number of instrument

Program sequence:

*PARA:CALIBN ?<CR> / request of calibration file number
*PARA:SDEL 1000<CR> / permanent setting of scan delay
*PARA:BORD 40 50<CR> / permanent setting of min and max exposure limits for adaption of tint
*CONF:EXPO1<CR> / setting: adaption of tint
*CONF:AVER 3<CR> / setting: average 3 scans
*MEAS :PHOTO <CR> / measurement and output phototmetric value Y (Illuminance)
*FETCH :CHROMXY <CR / output of chromaticity coordinates x and y
*FETCH :SPRAD <CR> / output of radiometric spectrum
*CALC:CRI 2856<CR> / calculation and output of CRI for a reference of illuminant A
*CONF:LEVEL <CR> / output of exposure level
*PARA:SPNUM <CR> / output of serial number

Output:

Calib number: 0 / output of calibration file number (0: select file automatically according to Hall sensor signal)
Illuminance[lx]:1.528e+03
Chrom_x:0.3534
Chrom_y:0.3739
WL[nm] Sp Irradiance[W/(m^2*nm)]
380 4.87725949e-03 / first wavelength and value
385 4.49027959e-03
390 5.99744590e-03
…
775 2.98083741e-02
780 2.95560490e-02 / last wavelength and value
Color temperature of reference source: 2856 K
CRI _DC: 2.2e-04
CRI _Ra: 57.49
CRI _R01: 55.0
CRI _R02: 56.7
…
CRI _R11: 47.2
CRI _R12: 53.6
CRI _R13: 44.8
CRI _R14: 72.4
Level/ cnt: 26377 / maximum counts in raw spectrum
Level/ %: 0088 / saturation level of measurement in %
Spectrometer number: 2012666

Variation:

- Output of radiometric spectrum in steps of 1 nm – insertion of command *CONF:WSTP 1 before the FETCHcommand

- The *CALC:CRI command can be used without argument to calculate the reference spectrum with the measured CCT .

10.2.3. Pulsed Source Measurement (e.g. Monitor)

Task:

- Adjustment of measuring spot on the monitor
- u´v´and Y (Lv) measurement of a monitor
- Measurement synchronized to the monitor repetition rate (measurement of the cycle time with 200 averages and a timeout time of 1 s)

Program sequence:

*CONTR :LASER 1<CR> / switch on the target laser
adjustment of the measuring area on the monitor
*CONTR :LASER 0<CR> / switch off the target laser
*CONF:EXPO 1<CR> / setting: adaption of tint
*CONF:CYCMOD 1<CR> / switching to synchronized measuring mode
*CONTR :CYCTIM 200 4000<CR> / measurement of cycle time
*CONF:CYCTIM …<CR> / setting: cycle time to measured value (in µs)
*MEAS :CHROMUV <CR> / measurement and output of u´and v´
*FETCH :PHOTO <CR> / output of photometric value

Output:

cyctim: 19.99 / measuring result for cycle time in ms
u´: 0.2521
v´: 0.5229
luminance [cd/m2]: 9.241e^2

11. License Agreement

License Agreement for JETI Technische Instrumente GmbH Software Products
IMPORTANT -- READ CAREFULLY BEFORE USING THE JETI SOFTWARE
JETI hereby grants to you a non-exclusive license to use one copy of the delivered JETI software program (the "software") on a single computer according to the following items:
The software is in "use" on the computer when it is loaded into temporary memory (i.e. RAM) or installed into permanent memory (e.g. hard disk, CD-ROM, or other storage device) of the computer. However, installation on a network server for the sole purpose of distribution to one or more other computer(s) shall not constitute "use" for which a separate license is required, provided you have separate license for each computer to which the software is distributed.
It is only allowed to use the software in connection with the JETI product (instrument, spectrometer or OEM electronics), which was delivered with this software. For additional JETI products you need an extra software license.
The software is owned by JETI or its suppliers and is protected by German copyright laws and international treaty provisions and all other applicable national laws. Therefore, you must treat the software like any other copyrighted material (e.g. a book or musical recording) except that if the software is not copy protected you may either

(a) make one copy of the software solely for backup or archival purposes, or

(b) transfer the software to a single hard disk provided you keep the original solely for backup or archival purposes. You may not copy the user
documentation provided with the software and the connected JETI product.

You may not:

i. permit other individuals to use the software except under the terms listed above;
ii. modify, translate, reverse engineer, decompile, disassemble (except to the extent that this restriction is expressly prohibited by law) or create derivative works based upon the software or documentation;
iii. copy the software or documentation (except for back-up or archival purposes);
iv. rent, lease, transfer, or otherwise transfer rights to the software or documentation;
v. remove any proprietary notices or labels on the software or documentation.

Any such forbidden use shall immediately terminate your license to the Software.

You agree that you shall only use the software and documentation in a manner that complies with all applicable laws in the jurisdictions in which you use the software and documentation, including, but not limited to, applicable restrictions concerning copyright and other intellectual property rights.
You may not use the software in an attempt to, or in conjunction with, any device, program or service designed to circumvent technological measures employed to control access to, or the rights in, a content file or other work protected by the copyright laws of any jurisdiction.
If you receive the first copy of the software electronically and a second copy on physical media (e.g., CD, diskette, etc.), the second copy may be used for archival purposes only and may not be transferred to or used by any other person. This license does not grant you any right to any enhancement or update.

The software may include certain external software components ("Add-Ons"), including the Lab Windows user interface and drivers. You may only call to or otherwise use such Add-Ons through the use of the JETI applications. Any direct use of Add-Ons through a non-JETI proprietary application, including a custom or user-written application is prohibited by this Agreement.
If you have purchased JETI software in connection with a JETI product, you have the right of purchasing a software update, if it is available, for a priority price.
Title, ownership, rights, and intellectual property rights in and to the software and documentation shall remain in JETI and/or its suppliers. Title, ownership rights and intellectual property rights in and to the content accessed through the software including the content contained in the software media demonstration files shall be retained by the applicable content owner and may be protected by applicable copyright or other law. This license gives you no rights to such content.
For the JETI operating environment, the following terms apply:
JETI warrants that for a period of ninety (90) days from the date of acquisition the software, if operated as directed, will substantially achieve the functionality described in the documentation. JETI does not warrant however that your use of the software will be uninterrupted or that the operation of the software will be error-free or secure. JETI also warrants that the media containing the software, if provided by JETI, is free from defects in material and workmanship and will so remain for ninety (90) days from the date you acquire the software.
To the maximum extent permitted by applicable law JETI and its suppliers disclaim all other warranties either express or implied including but not limited to implied warranties of merchantability and fitness for a particular purpose with regard to the software, the accompanying written materials and any accompanying hardware.
If the media is subjected to accident abuse or improper use during the warranty period; or if you violate the terms of this License Agreement, this warranty shall immediately terminate. This warranty shall not apply if the software is used on or in conjunction with hardware or software other than the unmodified version of hardware and software with which the software was designed to be used as described in the documentation.
This limited warranty gives you specific legal rights.
Your remedies:
JETI's sole liability for a breach of this warranty shall be in JETI's sole discretion:

to replace your defective media; or
to advise you how to achieve substantially the same functionality with the software as described in the documentation through a procedure different from that set forth in the documentation; or
if the above remedies are impracticable, to refund the license fee, if any, you paid for the software. Repaired, corrected or replaced software and documentation shall be covered by this limited warranty for the period remaining under the warranty that covered the original software or if longer for thirty (30) days after the date JETI either shipped to you the repaired or replaced software or advised you as to how to operate the software so as to achieve the functionality described in the documentation, whichever is applicable. Only if you inform JETI of the problem with the software during the applicable warranty period and provide evidence of the date you acquired the software will JETI be obligated to honor this warranty.

Under no circumstances and under no legal theory whether in tort contract or otherwise shall JETI or its suppliers or resellers be liable to you or any other person for any indirect, special, incidental or consequential damages of any character including without limitation damages for loss of goodwill, work stoppage, computer failure or malfunction or any and all other commercial damages or losses even if JETI shall have been informed of the possibility of such damages or for any claim by any other party. Further, in no event shall JETI´s liability under any provision of this agreement exceed the license fee paid to JETI for the software and documentation.
This software is intended for use only with properly licensed media, content, and content creation tools. It is your responsibility to ascertain whether any copyright, patent or other licenses are necessary and to obtain any such licenses to serve and/or create, compress or download such media and content. You agree to record, play back and download only those materials for which you have the necessary patent, copyright and other permissions, licenses, and/or clearances. You agree to hold harmless, indemnify and defend

JETI, its officers, directors and employees, from and against any losses, damages, fines and expenses (including attorneys' fees and costs) arising out of or relating to any claims that you have

- viewed, downloaded, encoded, compressed, copied or transmitted any materials (other than materials provided by JETI) in connection with the software in violation of another party's rights or in violation of any law, or
- violated any terms of this License Agreement.

This License Agreement will automatically terminate if you fail to comply with any term hereof. No notice shall be required from JETI to affect such termination.
Technical support for the software, as made available by JETI, is given directly from the JETI office in Jena, Germany.
This License Agreement shall constitute the complete and exclusive agreement between JETI, it’s resellers and the end user, notwithstanding any variance with any purchase order or other written instrument submitted by you, whether formally rejected by JETI or not. The acceptance of any purchase order you place is expressly made conditional on your consent to the terms set forth herein. The terms and conditions contained in this License Agreement may not be modified except in a writing duly signed by you and an authorized representative of JETI. If any provision of this License Agreement is held to be unenforceable for any reason, such provision shall be reformed only to the extent necessary to make it enforceable, and such decision shall not affect the enforceability of such provision under other circumstances, or of the remaining provisions hereof under all circumstances. This License Agreement shall be governed by the laws of Germany, without regard to conflicts of law provisions, and you hereby consent to the exclusive jurisdiction of the provincial and federal courts sitting in Germany. Any and all unresolved disputes relating in any way to, or arising out of, the software, your use of the software or this License Agreement shall be submitted to arbitration in Germany; except that, to the extent that you have breached or have indicated your intention to breach this License Agreement in any manner which violates or may violate JETI's intellectual property rights, or may cause continuing or irreparable harm to JETI (including, but not limited to, any breach that may impact JETI's intellectual property rights, or a breach by reverse engineering), JETI may seek injunctive relief, or any other appropriate relief, in any court of competent jurisdiction.

12. Index Commands

