THORLABS

DC2200 V1.2
Write Your Own Application

- Drivers
- Samples
- Programmer's Reference

1 Write Your Own Application

In order to write your own application, you need a specific instrument driver and some tools for use in different programming environments. The driver and tools are included in the installer package that can be downloaded from the website.

In this section the location of drivers and files, required for programming in different environments, are given for installation under Windows 7 (32 and 64 bit)

Note

DC2200 software and drivers contains 32 bit and 64 bit applications.

In 32 bit systems, only the 32 bit components are installed to

```
C:\Program Files\...
```

In 64 bit systems the 64 bit components are being installed to

```
C:\Program Files\...
```

while 32 bit components can be found at

```
C:\Program Files (x86)\...
```

In the table below you will find a summary of what files you need for particular programming environments.

Programming environment	Necessary files
C, C++, CVI	*.fp (function panel file; CVI IDE only) *.h (header file) *.lib (static library) *.dll (dynamic linked library)
C#	.net wrapper dll
Visual Studio	*.h (header file) *.lib (static library) or .net wrapper dll
LabView	*.fp (function panel) and NI VISA instrument driver Beside that, LabVIEW driver vi's are provided with the *.llb container file

Note

All above environments require also the NI VISA instrument driver dll !

During NI-VISA Runtime installation, a system environment variable `VXIPNPPATH` for including files is created. It contains the information where the drivers are installed to, usually to `C:\Program Files\IVI Foundation\VISA\WinNT\`.

This is the reason, why after installation of a NI-VISA Runtime a system reboot is required: This environment variable is necessary for installation of the instrument driver software components.

In the next sections the location of above files is described in detail.

1.1 32 bit Systems

NI VISA instrument driver

```
C:\Program Files\IVI Foundation\VISA\WinNT\Bin\TLDC2200_32.dll
```

Online help for NI VISA instrument driver

```
C:\Program Files\IVI Foundation\VISA\WinNT\TLDC2200\Manual...
...\TLDC2200.html
```

NI LabVIEW driver

The LabVIEW Driver is a 32 bit driver and compatible with 32 bit NI LabVIEW versions 8.5 and higher only.

```
C:\Program Files\National Instruments\LabVIEW <version>\instr.lib...
...\TLDC2200\TLDC2200.llb
```

(LabVIEW container file with driver vi's. "LabVIEW <version>" stands for actual LabVIEW installation folder.)

Header file

```
C:\Program Files\IVI Foundation\VISA\WinNT\include\TLDC2200.h
C:\Program Files\IVI Foundation\VISA\WinNT\include...
...\TLDC2200_defines.h
```

Static library

```
C:\Program Files\IVI Foundation\VISA\WinNT\lib\msc\TLDC2200_32.lib
```

Function panel

```
C:\Program Files\IVI Foundation\VISA\WinNT\TLDC2200\TLDC2200.fp
```

.net wrapper dll

```
C:\Program Files\IVI Foundation\VISA\VisaCom...
...\Primary Interop Assemblies\Thorlabs.TLDC2200_32.Interop.dll
```

Example LabWindows CVI

```
C:\Program Files\IVI Foundation\VISA\WinNT\TLDC2200\Examples...
...\CSample\TLDC2200Sample.prj
```

Basic C Sample to get one measurement

Example Visual Studio .Net 2010

```
C:\Program Files\IVI Foundation\VISA\WinNT\TLDC2200\Examples...
...\DotNet\Thorlabs DC2200 DotNet Sample.csproj
```

C Sharp sample to get one measurement

1.2 64 bit Systems

Note

According to the VPP6 (Rev 6.1) Standard the installation of the 64 bit VXIpnp driver includes the WINNT, WIN64, GWINNT and GWIN64 frameworks. That means, that the 64 bit driver includes the 32 bit driver as well.

On a 64 bit operating system, 64 bit drivers and applications are installed to

"C:\Program Files"

while the 32 bit files - to

"C:\Program Files (x86)"

Below are listed both installation locations, so far applicable.

NI VISA instrument driver

```
C:\Program Files (x86)\IVI Foundation\VISA\WinNT\Bin\TLDC2200_32.dll
C:\Program Files\IVI Foundation\VISA\Win64\Bin\TLDC2200_64.dll
```

Online help for NI VISA instrument driver

```
C:\Program Files (x86)\IVI Foundation\VISA\WinNT\TLDC2200\Manual...
...\TLDC2200.html
```

NI LabVIEW Driver

The LabVIEW Driver supports both 32 bit and 64 bit NI LabVIEW 2009 and higher.

```
C:\Program Files\National Instruments\LabVIEW <version>\instr.lib...
...\TLDC2200\TLDC2200.llb
```

(LabVIEW container file with driver vi's. "LabVIEW <version>" stands for actual LabVIEW installation folder.)

Header file

```
C:\Program Files (x86)\IVI Foundation\VISA\WinNT\include\TLDC2200.h
C:\Program Files (x86)\IVI Foundation\VISA\WinNT\include...
...\TLDC2200_defines.h
C:\Program Files\IVI Foundation\VISA\Win64\Include\TLDC2200.h
C:\Program Files\IVI Foundation\VISA\Win64\Include...
...\TLDC2200_defines.h
```

Static library

```
C:\Program Files (x86)\IVI Foundation\VISA\WinNT\lib\msc...
...\TLDC2200_32.lib
C:\Program Files\IVI Foundation\VISA\Win64\Lib_x64\msc...
...\TLDC2200_64.lib
```

Function panel

```
C:\Program Files (x86)\IVI Foundation\VISA\WinNT\TLDC2200...
...\TLDC2200.fp
```

.net wrapper dll

```
C:\Program Files (x86)\IVI Foundation\VISA\VisaCom...
...\Primary Interop Assemblies\Thorlabs.TLDC2200_32.Interop.dll
C:\Program Files\IVI Foundation\VISA\VisaCom64\Primary Interop
Assemblies\Thorlabs.TLDC2200_64.Interop.dll
```

Example LabWindows CVI

```
C:\Program Files (x86)\IVI Foundation\VISA\WinNT\TLDC2200...
...\Examples\CSample\TLDC2200Sample.prj
```

Basic C Sample to get one measurement

Example Visual Studio .Net 2010

```
C:\Program Files (x86)\IVI Foundation\VISA\WinNT\TLDC2200...
...\Example\DotNet\Thorlabs DC2200 DotNet Sample.csproj
```

C Sharp sample to get one measurement

1.3 Command Reference

1.3.1 IEEE488.2 Common Commands

Common commands are device commands that are common to all devices according to the IEEE488.2 standard. These commands are designed and defined by this standard. Most of the commands are described in detail in this section. The following common commands associated with the status structure are covered in the “Status Structure” section: *CLS, *ESE, *ESE?, *ESR?, *SRE, *SRE?, *STB?

Command summary

Mnemonic	Name	Description
***CLS**	Clear status	Clears all event registers and Error Queue
***ESE <NRf>**	Event enable command	Sets the Standard Event Enable Register
***ESE?**	Event enable query	Returns the Standard Event Enable Register
***ESR?**	Event status register query	Returns and clear the Standard Event Register
***IDN?**	Identification query	Returns the unit’s identification string
***OPC**	Operation complete command	Sets the Operation Complete bit in the Standard Event Register
***OPC?**	Operation complete query	Places a “1” into the output queue when all device operations have been completed
***RST**	Reset command	Returns the unit to the *RST default condition
***SRE <NRf>**	Service request enable command	Sets the Service Request Enable Register
***SRE?**	Service request enable query	Returns the Service Request Enable Register
***STB?**	Status byte query	Returns the Status Byte Register
***TST?**	Self-test query	Performs the unit’s self-test and returns the result.
***WAI**	Wait-to-continue command	Waits until all previous commands are executed

Command reference

1. *IDN? – identification query - read identification code

The identification code includes the manufacturer, model code, serial number, and firmware revision levels and is sent in the following format: `Thorlabs GmbH,MMM,SSS,X.X.X,` where

`MMM`	is the model code
`SSS`	is the serial number
`X.X.X`	is the instrument firmware revision level

Example: `THORLABS,DC2200,M00123456,1.0.1`

2. *OPC – operation complete - set OPC bit

3. *OPC? – operation complete query – places a “1” in output queue

When `*OPC` is sent, the OPC bit in the Standard Event Register will set after all pending command operations are complete. When `*OPC?` is sent, an ASCII “1” is placed in the Output Queue after all pending command operations are complete.

Typically, either one of these commands is sent after the `INITiate` command. The `INITiate` command is used to take the instrument out of idle in order to perform measurements. While operating within the trigger model layers, many sent commands will not execute. After all programmed operations are completed, the instrument returns to the idle state at which time all pending commands (including `*OPC` and/or `*OPC?`) are executed. After the last pending command is executed, the OPC bit and/or an ASCII “1” is placed in the Output Queue.

4. *RST – reset – return instrument to defaults

When the `*RST` command is sent, the instrument performs the following operations:

- Returns the instrument to default conditions
- Cancels all pending commands.
- Cancels response to any previously received `*OPC` and `*OPC?` commands.

5. *TST? – self-test query – run self test and read result

Use this query command to perform the instrument self-test routine. The command places the coded result in the Output Queue. A returned value of zero (0) indicates that the test passed, other values indicate that the test failed and an error code is placed into the error queue.

6. *WAI – wait-to-continue – wait until previous commands are completed

The *WAI command is a no operation command for the instrument and thus, does not need to be used. It is there for conformance to IEEE488.2.

1.3.2 SCPI Command Reference

SYSTem subsystem commands

Command	Description	SCPI
SYSTem	Path to SYSTem subsystem	Vol.2 §21
:BEEPer		Vol.2 §21.2
[:IMMediate]	Issues an audible signal	Vol.2 §21.2.2
:STATe {ON|1|OFF|0}	Activates/deactivates the beeper	Vol.2 §21.2.3
:STATe?	Returns the state of the beeper	Vol.2 §21.2.3
:VOLume <value>	Sets the beeper volume in a range from 0.0 to 1.0	Vol.2 §21.2.5
:VOLume?	Returns the beeper volume	Vol.2 §21.2.5
:ERRor		Vol.2 §21.8
[:NEXT]?	Returns the latest error code and message	Vol.2 §21.8.8
:VERSion?	Returns level of SCPI standard (1999.0)	Vol.2 §21.21
:TERMinal{[1]|2}	Information about the LED heads connected to the LED terminals LED1 or LED2	
[:HTYPe]?	LED head type identification query. The query response is in the format "<vendor_name_string>,<led_head_model_name_string>,<led_head_serial_no_string>,<fw_version_major_num>,<fw_version_minor_num>,<fw_version_subminor_num>". The firmware version indicates the head's onboard memory mapping version. A major version number -1 indicates a custom head without onboard head info memory. A major version number – 2 indicates that there is no head connected on the terminal Sample of a LED with onboard head info memory: Thorlabs,M365F1,M00123456,1.0.0 Sample of a LED without onboard head info memory: Thorlabs,custom,n/a,-1.0.0 Sample of no LED head connected: Thorlabs,no head,no head,-2.0.0	
:HEAD		
:TEMPerature		
[:COUNt]?	Get the number of temperature sensor available in the LED head currently connected to the specified terminal.	
:LABel? [{0|<sensidx>}]	Query LED head temperature sensor label. This is a query with parameter. The optional numeric parameter is the index of the LED head temperature sensor to query. Parameter value range is 0 <= sensidx < senscnt with default parameter value is 0.	
:VOLTage?	Query maximum LED forward voltage specified by the head's onboard info memory	
:CURRent?	Query maximum LED forward current specified by the head's onboard info memory	
:SPECtrum?	Query the spectrum information of the LED head. Positive values describe the nominal LED center wavelength in nm. Negative values describe the color temperature in K. The value 0.0 describes "information not available"	

DISPlay subsystem commands

Command	Description	SCPI
DISPlay	Path to DISPlay subsystem.	Vol.2 §8
:BRIGhtness <value>	Sets the display brightness in a range from 0.0 to 1.0	Vol.2 §8.2
:BRIGhtness?	Returns the display brightness value	Vol.2 §8.2
:CALibration[:TOUCh]	Initiates Touchscreen calibration	
[:INITiate]		
:FADeout		
[:STATe] {ON|1|OFF|0}	Activates/deactivates automatic dimming	
[:STATe]?	Returns the state of automatic dimming	

STATus subsystem commands

Command	Description	SCPI
STATus	Path to STATus subsystem.	Vol.2 §20
:MEASurement	Path to control measurement event registers	
[:EVENt]?	Returns the event register	
:CONDition?	Returns the condition register	
:PTRansition <value>	Sets the positive transition filter	
:PTRansition?	Returns the positive transition filter	
:NTRansition <value>	Sets the negative transition filter	
:NTRansition?	Returns the negative transition filter	
:ENABle <value>	Sets the enable register	
:ENABle?	Returns the enable register	
:OPERation	Path to control operation event registers	Vol.2 §20.1
[:EVENt]?	Returns the event register	Vol.2 §20.1.4
:CONDition?	Returns the condition register	Vol.2 §20.1.2
:PTRansition <value>	Sets the positive transition filter	Vol.2 §20.1.7
:PTRansition?	Returns the positive transition filter	Vol.2 §20.1.7
:NTRansition <value>	Sets the negative transition filter	Vol.2 §20.1.6
:NTRansition?	Returns the negative transition filter	Vol.2 §20.1.6
:ENABle <value>	Sets the enable register	Vol.2 §20.1.3
:ENABle?	Returns the enable register	Vol.2 §20.1.3
:QUEStionable	Path to control questionable event registers	Vol.2 §20.3
[:EVENt]?	Returns the event register	Vol.2 §20.3.4
:CONDition?	Returns the condition register	Vol.2 §20.3.2
:PTRansition <value>	Sets the positive transition filter	Vol.2 §20.3.7
:PTRansition?	Returns the positive transition filter	Vol.2 §20.3.7
:NTRansition <value>	Sets the negative transition filter	Vol.2 §20.3.6
:NTRansition?	Returns the negative transition filter	Vol.2 §20.3.6
:ENABle <value>	Sets the enable register	Vol.2 §20.3.3
:ENABle?	Returns the enable register	Vol.2 §20.3.3
:AUXiliary	Path to control auxiliary event registers	
[:EVENt]?	Returns the event register	
:CONDition?	Returns the condition register	
:PTRansition <value>	Sets the positive transition filter	
:PTRansition?	Returns the positive transition filter	
:NTRansition <value>	Sets the negative transition filter	
:NTRansition?	Returns the negative transition filter	
:ENABle <value>	Sets the enable register	
:ENABle?	Returns the enable register	
:PRESet	Return status registers to default states.	Vol.2 §20.2

OUTPut subsystem commands

Command	Description	SCPI
OUTPut[1]	Path to OUTPut subsystem	Vol.2 §15
[:STATe] {ON|1|OFF|0}	Enables (ON) or disables (OFF) LED output	Vol.2 §15.12
[:STATe]?	Returns output state	Vol.2 §15.12
:TERMinal {1|2}	Select output terminal: 1 = 10A/12Pin connector (LED1); 2 = 2A/4Pin connector (LED2);	
:TERMinal?	Returns selected output terminal	
:TERMinal:ABORt	Abort any running LED head presence test	
:TERMinal{[1]|2}		
:TEST	Testing presence and properties of LEDs	
[:INITiate]	Initiate presence test procedure on the specified terminal	
:STATus?	Query status of the presence test procedure as an <NF1> value: 0 = The presence test procedure is running 1 = The latest presence test detected no LED 2 = The latest presence test detected a custom LED without onboard head info memory 3 = The latest presence test detected a LED head with onboard head info memory	
:PROTection		
:INTLock[:TRIPped]?	Returns interlock circuit protection tripped (1) or untripped (0)	
:TEMPerature		
[:DRIVer][:TRIPped]?	Returns driver (console) over temperature protection tripped tripped (1) or untripped (0)	
:HEAD[:TRIPped]?	Returns head over temperature protection tripped tripped (1) or untripped (0)	

SENSe3 LED current sensing subsystem commands

Command	Description	SCPI
SENSe3	Path to LED current sensing	
[:CURRent][:DC]		
[:DATA]? [{MIN|MAX}]	Returns the measured LED current. Parameters MIN and MAX return the measurement value range possible in the currently active driver configuration (depends on the compliance voltage required by the LED head)	

SENSe4 LED voltage sensing subsystem commands

Command	Description	SCPI
SENSe4	Path to LED voltage sensing	
[:VOLTage][:DC]		
[:DATA]? [{MIN|MAX}]	Returns the measured LED voltage. Parameters MIN and MAX return the measurement value range possible.	

SENSe5 LED temperature sensing subsystem commands

Command	Description	SCPI
SENSe5	Path to LED temperature sensing	
[:TEMPerature]? [{MIN|MAX}]	Returns the measured LED temperature. Parameters MIN and MAX return the measurement value range possible.	

SOURce1 LED source subsystem commands

Command	Description	SCPI
SOURce[1]		Vol.2 §19
:MODe {CC*|1|CB|2|PWM|3|PULS|4| IMOD|5|EMOD|6|TTL|7}	Set the LED operating mode: 1 = CC = Constant Current 2 = CB = Constant Brightness 3 = PWM = Pulse Width Modulation 4 = PULS = Pulse Modulation 5 = IMOD = Internal Modulation 6 = EMOD = External Modulation 7 = TTL = TTL Input Controlled	
:MODe?	Query the LED operation mode	
[:CURRent]		Vol.2 §19.5
:LIMit		Vol.2 §19.5.5
[:AMPLitude] {MIN|MAX|<amps>}	Sets the limit current value	Vol.2 §19.5.5.1
[:AMPLitude]? [{MIN|MAX}]	Query the limit current value	Vol.2 §19.5.5.1
:TRIPped?	Returns limit detection tripped (1) or untripped (0)	
[:AMPLitude] {MIN|MAX|<amps>}	Set the constant current mode forward current value	
[:AMPLitude]? [{MIN|MAX}]	Query the constant current mode forward current value	
:CCURent	LED operating mode “Constant Current” specific settings. These settings only have an effect while in “Constant Current” mode	
[:CURRent][:LEVel]		
[:AMPLitude] {MIN|MAX|<amps>}	Set the constant current mode forward current value	
[:AMPLitude]? [{MIN|MAX}]	Query the constant current mode forward current value	
:CBRightness	LED operating mode “Constant Brightness” specific settings. These settings only have an effect while in “Constant Brightness” mode	
[:BRIGhtness][:LEVel]		
[:AMPLitude] {MIN|MAX|<perc>}	Sets LED brightness set value in percent of currently set limit current	
[:AMPLitude]? [{MIN|MAX}]	Query LED brightness set value in percent of currently set limit current	
:PWM	LED operating mode “Pulse Width Modulation” specific settings. These settings only have an effect while in PWM mode	
[:CURRent][:LEVel]		
[:AMPLitude] {MIN|MAX|<amps>}	Set the PWM mode forward current value	
[:AMPLitude]? [{MIN|MAX}]	Query the PWM mode forward current value	
:FREQency		
[:CW|:FIXed] {MIN|MAX|<hertz>}	Set PWM mode modulation frequency	
[:CW|:FIXed]? [{MIN|MAX}]	Query PWM mode modulation frequency	
:DCYCle {MIN|MAX|<perc>}	Set the duty cycle in percent	
:DCYCle? [{MIN|MAX}]	Query the duty cycle in percent	
:COUNt {MIN|MAX|<numeric_value>}	Set the number of pulses. 0 = infinite pulses	
:COUNt? [{MIN|MAX}]	Query the number of pulses	
:PULSe	LED operating mode “Pulse Modulation” specific settings. These settings only have an effect while in Pulse mode	
[:BRIGhtness][:LEVel]		
[:AMPLitude] {MIN|MAX|<perc>}	Sets LED brightness set value in percent of currently set limit current	
[:AMPLitude]? [{MIN|MAX}]	Query LED brightness set value in percent of currently set limit current	

Command	Description	SCPI
:ONTime {MIN|MAX|<seconds>}	Set pulse on time in seconds	
:ONTime? [{MIN|MAX}]	Query pulse on time in seconds	
:OFFTime {MIN|MAX|<seconds>}	Set pulse off time in seconds	
:OFFTime? [{MIN|MAX}]	Query pulse off time in seconds	
:COUNt {MIN|MAX|<numeric_value>}	Set the number of pulses. 0 = infinite pulses	
:COUNt? [{MIN|MAX}]	Query the number of pulses	
:IMODulation	LED operating mode “Internal Modulation” specific settings. These settings only have an effect while in internal modulation mode	
[:BRIGhtness]		
:HIGH {MIN|MAX|<num>}	Set the maximum brightness in % of limit current	
:HIGH? [{MIN|MAX}]	Query the maximum brightness in % of limit current	
:LOW {MIN|MAX|<num>}	Set the minimum brightness in % of limit current	
:LOW? [{MIN|MAX}]	Query the minimum brightness in % of limit current	
:FREQency		
[:CW|:FIXed] {MIN|MAX|<hertz>}	Set internal modulation mode modulation frequency	
[:CW|:FIXed]? [{MIN|MAX}]	Query internal modulation mode modulation frequency	
:FUNCtion		
[:SHAPe] {SINusoid*|1|SQUare|2|TRIangle|3}	Set modulation shape function	
[:SHAPe]?	Query modulation shape function 1 = SIN = sinusoid 2 = SQU = square 3 = TRI = triangular	
:TTL	LED operating mode “TTL” specific settings. These settings only have an effect while in TTL mode	
[:CURRent][:LEVel]		
[:AMPLitude] {MIN|MAX|<amps>}	Set the TTL mode forward current value	
[:AMPLitude]? [{MIN|MAX}]	Query the TTL mode forward current value	

UNIT subsystem commands

Command	Description	SCPI
UNIT		Vol.2 §25
:TEMPerature {C|CEL|CELSius|F|FAR|FAHReinheit|K|KELVin}	Sets the temperature unit	Vol.2 §25.3
:TEMPerature?	Returns the temperature unit	Vol.2 §25.3

CALibration subsystem commands

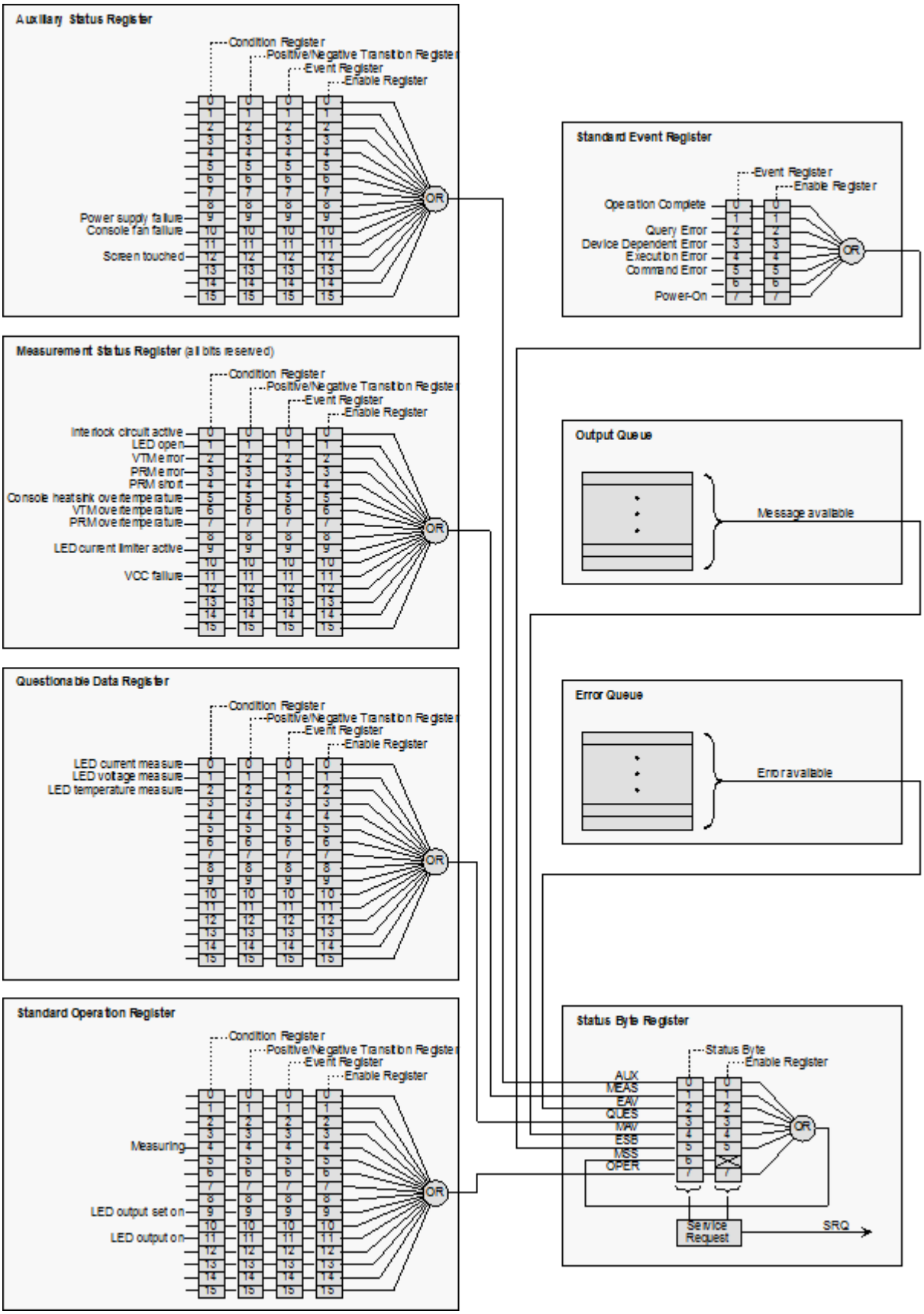
Command	Description	SCPI
CALibration		
:STRing?	Returns the console’s calibration string that was written to the console at the latest calibration procedure (usually the calibration date). Example: “15-Jul-2015”	

Measurement commands

Command	Description	SCPI
ABORt	Aborts current measurement	Vol.2 §24.5
CONFigure[:SCALar]		Vol.2 §3.1
:CURRent[1][:DC]	Configures instrument LED current measurement	
:VOLTage[1][:DC]	Configures instrument LED voltage measurement	
:TEMPerature[1]	Configures instrument for LED head temperature measurement	
CONFigure?	Query configuration	Vol.2 §3.1
INITiate[:IMMediate]	Starts measurement	Vol.2 §24.7.2
FETCh?	Returns last measurement data	Vol.2 §3.2
FETCh		Vol.2 §3.2
:CURRent[1][:DC]?	Get last LED current measurement	
:VOLTage[1][:DC]?	Get last LED voltage measurement	
:TEMPerature[1]?	Get last LED head temperature measurement	
READ?	Starts new measurement (as configured) and read data	Vol.2 §3.3
MEASure[:SCALar]		Vol.2 §3.4
:CURRent[1][:DC]?	Perform LED current measurement	
:VOLTage[1][:DC]?	Perform LED voltage measurement	
:TEMPerature[1]?	Perform LED head temperature measurement	

1.3.3 Status Reporting

The figure below gives an overview of the device's status reporting structure. See also section STATus subsystem commands for a detailed description of the related commands and their syntax.



Status Byte Register

The Status Byte Register gives a summary of all underlying status structures. See also IEEE488.2-1992-§11.2.

Bit #	Mnemonic	Description
7	**OPER**	Standard Operation Status Structure Summary Bit
6	**RQS/MSS**	Request Service / Master Summary Status
5	**ESB**	Standard Event Status Bit
4	**MAV**	Message Available. There is response data available for readout
3	**QUES**	Questionable Status Structure Summary Bit
2	**EAV**	Error Available. There is at least one error in the error queue.
1	**MEAS**	Measurement Status Structure Summary Bit
0	**AUX**	Auxiliary Status Structure Summary Bit

Standard Event Status Structure

The Standard Event Status Structure is described in IEEE488.2-1992-§11.5.

Standard Operation Register

The Standard Operation Status Structure is described in SCPI1999.0-Vol1-§9.3. In addition bit 8..12 are used as output state/on indicators.

Bit #	Mnemonic	Description
15..13		See SCPI1999.0-Vol1-§9.3
12		reserved, read as 0
11	**LEDON**	LED output is currently ON
10		reserved, read as 0
9	**LEDST**	LED output state is ON
8		reserved, read as 0
7..0		See SCPI1999.0-Vol1-§9.3

Questionable Data Register

The Questionable Data Status Structure is described in SCPI1999.0 Vol1 §9.4.

Bit #	Mnemonic	Description
15..13		See SCPI1999.0-Vol1-§9.4 – flags are not implemented
12..3		reserved, read as 0
2	**LEDT**	LED temperature measure
1	**LEDV**	LED voltage measure
0	**LEDC**	LED current measure

Measurement Status Register

The Measurement Status Register Status Byte Register reports device operation and measurement states.

Bit #	Description
15	reserved, read as 0
14	Over temperature (Instrument is too hot)
13	Socket connection failure (Socket missing or unknown socket detected)
12	reserved, read as 0
11	TEC output compliance voltage reached
10	Temperature sensor failure.
9	Temperature window failure.
8	Temperature protection is active.
7	TEC current limit reached
6	reserved, read as 0
5	reserved, read as 0
4	reserved, read as 0
3	LD current limit reached
2	LD interlock is active
1	LD output compliance voltage reached
0	Keylock protection is active

Auxiliary Status Register

The Auxiliary Status Register Status Byte Register reports auxiliary device states.

Bit #	Description
15..13	reserved, read as 0
12	Screen touched
11	reserved, read as 0
10	Console fan failure
9	Power supply failure
8..0	reserved, read as 0

1.3.4 Error Reporting

The device stores errors in a queue containing up to 10 entries. The error queue may be read out by the `'SYSTem:ERRor[:NEXT]?'` command. The following table lists all error numbers and the according descriptive messages. Note: negative numbers are defined by SCPI while positive error numbers are device dependent.

Error	Description
0	No error
1	The error couldn't be specified more precisely
2	Floating point domain error
3	Device temperature too high
4	General GUI error
5	Authentication required for operation
6	Authentication process failed
7	Operation is not allowed in service mode
8	Operation is allowed in service mode only
9	A measurement is currently in process
14	LED head is missing or it is of unknown type
15	Power supply error
20	Operation not allowed while LED output is on
21	Wrong operating mode for this operation
22	INTERLOCK circuit is open
23	LED is overheated
24	Operation not allowed because of a 'OPEN CIRCUIT' condition
26	VTM module error
27	PRM module error
28	PRM module short circuit detected
29	VTM module overheated
30	PRM module overheated
31	Output current limit reached
32	Sensor failed
33	Supply 3.3V digital failed
34	Supply 1.2V digital failed
35	Supply 12V analog failed
36	Supply -12V analog failed
37	Supply 5V analog failed
38	Supply 12V VTM module and internal fan failed

Error	Description
39	Supply 15V external fan failed
40	Supply 5V digital failed
41	Supply 5V reference failed
42	Voltage supply failed
43	Supply fan failed
44	Supply touch screen failed
45	Power supply failed
51	User Current limitation by Max. Current limitation
52	User Current limitation by Power limitation
60	LED forward voltage measure procedure failed
91	Erroneous connection to LED driver A/D converter
100	I²C#0 wires stuck
101	I²C#0 bus error
102	I²C#0 slave address not acknowledged
103	I²C#0 incomplete write operation
104	I²C#0 bus arbitration lost
110	I²C#1 wires stuck
111	I²C#1 bus error
112	I²C#1 slave address not acknowledged
113	I²C#1 incomplete write operation
114	I²C#1 bus arbitration lost
120	I²C#2 wires stuck
121	I²C#2 bus error
122	I²C#2 slave address not acknowledged
123	I²C#2 incomplete write operation
124	I²C#2 bus arbitration lost
130	Nonvolatile memory timed out
131	Nonvolatile memory checksum error
132	Nonvolatile memory address overflow
133	Nonvolatile memory not supported
134	Nonvolatile memory missing

Error	Description
135	Async transfer is already running
140	FPGA configuration error
150	Fan controller not responding
151	Fan failure
170	RAM device failure
171	RAM address bus failure
172	RAM data bus failure
180	Touch controller INT signal failure
181	Touch controller INT signal timeout
182	Touch controller command error
183	Touch controller unrecognized command
184	Touch controller unrecognized header
185	Touch controller command timeout
186	Touch panel is not calibrated
187	Touch controller calibration canceled
188	Touch calibration already running
189	Touch calibration is not running
190	Touch calibration point is out of bounds
200	Value is not editable
201	Operation is not applicable
210	Numeric value error
211	Value minimum reached
212	Value maximum reached
213	Step size lower limit reached
214	Step size upper limit reached
220	Selection limit reached
230	Value is out of range
231	Not editable while output is ON
250	Unable to switch operating mode while LED output is on

Error	Description
251	A stored value is out of bounds and has been coerced
252	Empty storage - push and hold button to store setpoints
253	Setpoints stored
260	Unable to leave panel while LED output is on (safety mode)
261	Unable to switch LED output on in menu panel (safety mode)
270	No LED connected
271	Unknown LED head type
272	LED head memory data invalid
273	LED head memory version not supported
274	A mandatory LED head feature is not supported by the device
275	The forward voltage required by the LED head is not supported by the device. Occurrence of LED OPEN is highly possible
276	The maximum forward current allowed by the LED head can't be reached by the device
277	Can not assign LED head memory to terminals
278	The head data does not fit into the EEPROM
301	1-Wire line is shorted
302	No 1-Wire device found
304	No 1-Wire device on net
305	1-Wire bridge reset bit RST is set
307	1-wire file system: can't find path
308	1-wire file system: can't open file
309	1-wire file system: can't read file
310	1-wire file system: can't close file
312	Invalid 1-wire device handle
313	Invalid 1-wire device address
320	Invalid 1-Wire bridge channel
321	Invalid 1-Wire bridge index
-100	General command error
-101	Invalid character
-102	Syntax error

Error	Description
-103	Invalid separator
-104	Data type error
-105	GET not allowed
-108	Parameter not allowed
-109	Missing parameter
-110	Command header error
-111	Header separator error
-112	Program mnemonic too long
-113	Undefined header (Unknown command)
-114	Header suffix out of range
-115	Unexpected number of parameters
-120	Numeric data error
-130	Suffix error
-131	Invalid suffix
-150	String data error
-151	Invalid string data
-200	General execution error
-210	General trigger error
-211	Trigger ignored
-212	ARM ignored
-213	Init ignored
-220	Parameter error
-221	Settings conflict
-222	Data out of range
-223	Too much data
-224	Illegal parameter value
-230	Data corrupt or stale
-240	Hardware error
-310	System error
-311	Memory error
-313	Calibration memory lost
-314	Save/recall memory lost
-315	Configuration memory lost

Error	Description
-321	Out of memory
-330	Self-test failed
-340	Calibration failed
-350	Queue overflow
-363	Input buffer overrun
-365	Time out error
-410	Query INTERRUPTED