

## PDMv5 Library

Generated by Doxygen 1.8.16



---

<b>1 File Index</b>	<b>1</b>
1.1 File List	1
<b>2 File Documentation</b>	<b>3</b>
2.1 pdmv5.h File Reference	3
2.1.1 Macro Definition Documentation	9
2.1.1.1 PDMv5_DLL	10
2.1.2 Function Documentation	10
2.1.2.1 ApplyRequest_PDMv5()	10
2.1.2.2 CloseCommunication_PDMv5()	10
2.1.2.3 GetLIVMeasures_PDMv5()	10
2.1.2.4 GetLIVParameters_PDMv5()	11
2.1.2.5 Measure1V3PositiveVoltage_PDMv5()	12
2.1.2.6 Measure2V5PositiveVoltage_PDMv5()	12
2.1.2.7 Measure3V3PositiveVoltage_PDMv5()	13
2.1.2.8 Measure5VNegativeVoltage_PDMv5()	13
2.1.2.9 Measure5VPositiveVoltage_PDMv5()	13
2.1.2.10 MeasureAlarms_PDMv5()	14
2.1.2.11 MeasureBFMCurrent_PDMv5()	15
2.1.2.12 MeasureBNCInterlockStatus_PDMv5()	15
2.1.2.13 MeasureCentralDriverEnableStatus_PDMv5()	16
2.1.2.14 MeasureCentralMMDEnableStatus_PDMv5()	16
2.1.2.15 MeasureComplianceVoltage_PDMv5()	16
2.1.2.16 MeasureCWCCurrentConsign_PDMv5()	17
2.1.2.17 MeasureDiodeAverageCurrent_PDMv5()	17
2.1.2.18 MeasureDiodeCWCCurrent_PDMv5()	18
2.1.2.19 MeasureDiodePulsedCurrent_PDMv5()	18
2.1.2.20 MeasureDiodeTemperature_PDMv5()	19
2.1.2.21 MeasureDiodeVoltage_PDMv5()	19
2.1.2.22 MeasureEXTInterlockStatus_PDMv5()	19
2.1.2.23 MeasureInputVoltage_PDMv5()	20
2.1.2.24 MeasureKeyStatus_PDMv5()	20
2.1.2.25 MeasureLIVStatus_PDMv5()	21
2.1.2.26 MeasureMOSTemperature_PDMv5()	21
2.1.2.27 MeasurePatchBatteryVoltage_PDMv5()	22
2.1.2.28 MeasurePatchPICOLASComplianceVoltage_PDMv5()	22
2.1.2.29 MeasurePatchPICOLASCurrent_PDMv5()	23
2.1.2.30 MeasurePatchTECCurrent_PDMv5()	23
2.1.2.31 MeasurePatchTECExternalTemperature_PDMv5()	23
2.1.2.32 MeasurePD_EXTCurrent_PDMv5()	24
2.1.2.33 MeasurePeakCurrentConsign_PDMv5()	24
2.1.2.34 MeasureTECCurrent_PDMv5()	25

2.1.2.35 MeasureTECExternalTemperature_PDMv5()	25
2.1.2.36 MeasureTECVoltage_PDMv5()	26
2.1.2.37 MeasureTECVoltageReference_PDMv5()	26
2.1.2.38 MeasureTemperatureConsign_PDMv5()	26
2.1.2.39 OpenCommunication_PDMv5()	27
2.1.2.40 ReadAddress_PDMv5()	27
2.1.2.41 ReadAPCHysteresisPercentage_PDMv5()	28
2.1.2.42 ReadAPCPhotodiode_PDMv5()	28
2.1.2.43 ReadAPCSamplingPeriod_PDMv5()	29
2.1.2.44 ReadBFMGain_PDMv5()	29
2.1.2.45 ReadComplianceVoltage_PDMv5()	29
2.1.2.46 ReadCurrent_PDMv5()	30
2.1.2.47 ReadCWCurrent_PDMv5()	30
2.1.2.48 ReadCWCurrentSource_PDMv5()	31
2.1.2.49 ReadCWLaserStatus_PDMv5()	31
2.1.2.50 ReadCWMMaximumCurrent_PDMv5()	32
2.1.2.51 ReadDelay_PDMv5()	32
2.1.2.52 ReadDelayLine_PDMv5()	33
2.1.2.53 ReadExternalMultiElementsBoardCommandVoltage_PDMv5()	33
2.1.2.54 ReadFrequency_PDMv5()	34
2.1.2.55 ReadHardwareType_PDMv5()	34
2.1.2.56 ReadHardwareVersion_PDMv5()	35
2.1.2.57 ReadLaserStatus_PDMv5()	35
2.1.2.58 ReadLIVCurrentStep_PDMv5()	36
2.1.2.59 ReadLIVMaximumCurrent_PDMv5()	36
2.1.2.60 ReadLIVMeasuresCount_PDMv5()	36
2.1.2.61 ReadLIVMinimumCurrent_PDMv5()	37
2.1.2.62 ReadLIVPauseInterval_PDMv5()	37
2.1.2.63 ReadLIVPhotodiode_PDMv5()	38
2.1.2.64 ReadLIVPulseWidth_PDMv5()	38
2.1.2.65 ReadMaximumAverageCurrent_PDMv5()	39
2.1.2.66 ReadMaximumCurrent_PDMv5()	39
2.1.2.67 ReadModulationCurrent_PDMv5()	39
2.1.2.68 ReadModulationExternalCurrentGain_PDMv5()	40
2.1.2.69 ReadModulationFrequency_PDMv5()	40
2.1.2.70 ReadModulationInternalType_PDMv5()	41
2.1.2.71 ReadModulationMaximumCurrent_PDMv5()	41
2.1.2.72 ReadModulationStatus_PDMv5()	42
2.1.2.73 ReadOperatingMode_PDMv5()	42
2.1.2.74 ReadPD_EXTGain_PDMv5()	43
2.1.2.75 ReadPULSE_INDetectionThreshold_PDMv5()	43
2.1.2.76 ReadPulseCurrentSource_PDMv5()	44

---

2.1.2.77 ReadPulseLaserStatusPDMv5()	44
2.1.2.78 ReadPulseWidth_PDMv5()	44
2.1.2.79 ReadSerialNumber_PDMv5()	45
2.1.2.80 ReadSoftwareType_PDMv5()	45
2.1.2.81 ReadSoftwareVersion_PDMv5()	46
2.1.2.82 ReadSynchro_PDMv5()	46
2.1.2.83 ReadTECMaximumCurrent_PDMv5()	47
2.1.2.84 ReadTECMaximumVoltage_PDMv5()	47
2.1.2.85 ReadTemperature_PDMv5()	48
2.1.2.86 Save_PDMv5()	48
2.1.2.87 SetAddressSpecific_PDMv5()	48
2.1.2.88 SetAPCHysteresisPercentage_PDMv5()	49
2.1.2.89 SetAPCPhotodiode_PDMv5()	49
2.1.2.90 SetAPCSamplingPeriod_PDMv5()	50
2.1.2.91 SetBFMGain_PDMv5()	50
2.1.2.92 SetComplianceVoltage_PDMv5()	51
2.1.2.93 SetCurrent_PDMv5()	51
2.1.2.94 SetCWCurrent_PDMv5()	51
2.1.2.95 SetCWCurrentSource_PDMv5()	52
2.1.2.96 SetCWLaserStatus_PDMv5()	52
2.1.2.97 SetCWMaximumCurrent_PDMv5()	53
2.1.2.98 SetDelay_PDMv5()	53
2.1.2.99 SetDelayLine_PDMv5()	54
2.1.2.100 SetExternalMultiElementsBoardCommandVoltage_PDMv5()	54
2.1.2.101 SetFrequency_PDMv5()	55
2.1.2.102 SetLaserStatus_PDMv5()	55
2.1.2.103 SetLIVCurrentStep_PDMv5()	56
2.1.2.104 SetLIVMaximumCurrent_PDMv5()	56
2.1.2.105 SetLIVMeasuresCount_PDMv5()	56
2.1.2.106 SetLIVMinimumCurrent_PDMv5()	57
2.1.2.107 SetLIVPauseInterval_PDMv5()	57
2.1.2.108 SetLIVPhotodiode_PDMv5()	58
2.1.2.109 SetLIVPulseWidth_PDMv5()	58
2.1.2.110 SetMaximumAverageCurrent_PDMv5()	59
2.1.2.111 SetMaximumCurrent_PDMv5()	59
2.1.2.112 SetModulationCurrent_PDMv5()	59
2.1.2.113 SetModulationExternalCurrentGain_PDMv5()	60
2.1.2.114 SetModulationFrequency_PDMv5()	60
2.1.2.115 SetModulationInternalType_PDMv5()	61
2.1.2.116 SetModulationMaximumCurrent_PDMv5()	61
2.1.2.117 SetModulationStatus_PDMv5()	62
2.1.2.118 SetOperatingMode_PDMv5()	62

2.1.2.119 SetPD_EXTGain_PDMv5()	63
2.1.2.120 SetPULSE_INDetectionThreshold_PDMv5()	63
2.1.2.121 SetPulseCurrentSource_PDMv5()	64
2.1.2.122 SetPulseLaserStatus_PDMv5()	64
2.1.2.123 SetPulseWidth_PDMv5()	64
2.1.2.124 SetTECMaximumCurrent_PDMv5()	65
2.1.2.125 SetTECMaximumVoltage_PDMv5()	65
2.1.2.126 SetTemperature_PDMv5()	66
2.1.2.127 StartCalibration_PDMv5()	66

<b>Index</b>	<b>69</b>
--------------	-----------

# Chapter 1

## File Index

### 1.1 File List

Here is a list of all files with brief descriptions:

<a href="#">pdmv5.h</a> . . . . .	3
-----------------------------------	---





## Chapter 2

# File Documentation

### 2.1 pdmv5.h File Reference

#### Macros

- `#define PDMv5_DLL`

#### Functions

- `PDMv5_DLL` enum `sp_return` `OpenCommunication_PDMv5` (char \*portName, struct sp\_port \*\*portPointer)  
*Open the communication with the device.*
- `PDMv5_DLL` void `CloseCommunication_PDMv5` (struct sp\_port \*port)  
*Close the communication with the device.*
- `PDMv5_DLL` int `ReadAddress_PDMv5` (struct sp\_port \*port, unsigned char \*address)  
*Read the adress of the device.*
- `PDMv5_DLL` int `SetAddressSpecific_PDMv5` (struct sp\_port \*port, unsigned char currentAddress, unsigned char newAddress)  
*Set the address of the device.*
- `PDMv5_DLL` int `ApplyRequest_PDMv5` (struct sp\_port \*port, unsigned char address)  
*Apply values and parameters (this function must be called in order to really change the state of the PDMv5 card, it can be called after each function or after a list of functions)*
- `PDMv5_DLL` int `Save_PDMv5` (struct sp\_port \*port, unsigned char address)  
*Save all values into the device memory.*
- `PDMv5_DLL` int `StartCalibration_PDMv5` (struct sp\_port \*port, unsigned char address, unsigned char calibrationID)  
*Start a calibration.*
- `PDMv5_DLL` int `GetLIVMeasures_PDMv5` (struct sp\_port \*port, unsigned char address, unsigned short measureStartIndex, unsigned short measureEndIndex, float \*voltages, float \*currents, float \*powers)  
*Get the LIV Measures.*
- `PDMv5_DLL` int `GetLIVParameters_PDMv5` (struct sp\_port \*port, unsigned char address, float \*currentMinimum, float \*currentMaximum, float \*currentStep, float \*pauseInterval, unsigned char \*count, unsigned char \*photodiode)  
*Get the parameters with which the LIV was done.*
- `PDMv5_DLL` int `ReadHardwareType_PDMv5` (struct sp\_port \*port, unsigned char address, unsigned int \*type)  
*Read the hardware type.*

- [PDMv5\\_DLL](#) int [ReadHardwareVersion\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*version)  
*Read the hardware version.*
- [PDMv5\\_DLL](#) int [ReadSerialNumber\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned int \*serial)  
*Read the serial number.*
- [PDMv5\\_DLL](#) int [ReadSoftwareType\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned int \*type)  
*Read the software type.*
- [PDMv5\\_DLL](#) int [ReadSoftwareVersion\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*version)  
*Read the software version.*
- [PDMv5\\_DLL](#) int [ReadSynchro\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*mode)  
*Read the synchronisation line which will trigger pulses.*
- [PDMv5\\_DLL](#) int [ReadDelayLine\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*mode)  
*Read the delay line.*
- [PDMv5\\_DLL](#) int [SetDelayLine\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char mode)  
*Set the delay line.*
- [PDMv5\\_DLL](#) int [ReadFrequency\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned int \*frequency)  
*Read the frequency (in Hz)*
- [PDMv5\\_DLL](#) int [SetFrequency\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned int frequency)  
*Set the frequency (in Hz)*
- [PDMv5\\_DLL](#) int [ReadPulseWidth\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned int \*width)  
*Read the pulse width (in ps)*
- [PDMv5\\_DLL](#) int [SetPulseWidth\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned int width)  
*Set the pulse width (in ps)*
- [PDMv5\\_DLL](#) int [ReadDelay\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned int \*delay)  
*Read the delay (in ps)*
- [PDMv5\\_DLL](#) int [SetDelay\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned int delay)  
*Set the delay (in ps)*
- [PDMv5\\_DLL](#) int [ReadCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)  
*Read the current (in % of maximum current)*
- [PDMv5\\_DLL](#) int [SetCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float current)  
*Set the current (in % of maximum current)*
- [PDMv5\\_DLL](#) int [ReadTemperature\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*temperature)  
*Read the temperature (in °C)*
- [PDMv5\\_DLL](#) int [SetTemperature\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float temperature)  
*Set the temperature (in °C)*
- [PDMv5\\_DLL](#) int [ReadMaximumAverageCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)  
*Read the mean current limit (in mA)*
- [PDMv5\\_DLL](#) int [SetMaximumAverageCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float current)  
*Set the mean current limit (in mA)*
- [PDMv5\\_DLL](#) int [ReadMaximumCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)  
*Read the maximum current limit (in mA)*
- [PDMv5\\_DLL](#) int [SetMaximumCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float current)  
*Set the maximum current limit (in mA)*
- [PDMv5\\_DLL](#) int [ReadLaserStatus\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*status)  
*Read the laser status.*

- [PDMv5\\_DLL](#) int [SetLaserStatus\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char status)  
*Set the laser status.*
- [PDMv5\\_DLL](#) int [ReadTECMaximumCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)  
*Read TEC maximum current (in mA)*
- [PDMv5\\_DLL](#) int [SetTECMaximumCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float current)  
*Set the TEC maximum current (in mA)*
- [PDMv5\\_DLL](#) int [ReadTECMaximumVoltage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*voltage)  
*Read the TEC maximum voltage (in mV)*
- [PDMv5\\_DLL](#) int [SetTECMaximumVoltage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float voltage)  
*Set the TEC maximum voltage (in mV)*
- [PDMv5\\_DLL](#) int [ReadComplianceVoltage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*voltage)  
*Read The Compliance Voltage.*
- [PDMv5\\_DLL](#) int [SetComplianceVoltage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float voltage)  
*Set the compliance voltage (in V)*
- [PDMv5\\_DLL](#) int [ReadOperatingMode\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*mode)  
*Read the operating mode.*
- [PDMv5\\_DLL](#) int [SetOperatingMode\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char mode)  
*Set the operating mode.*
- [PDMv5\\_DLL](#) int [ReadAPCPhotodiode\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*mode)  
*Read the APC photodiode.*
- [PDMv5\\_DLL](#) int [SetAPCPhotodiode\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char mode)  
*Set the APC photodiode.*
- [PDMv5\\_DLL](#) int [ReadAPCSamplingPeriod\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned short \*time)  
*Read the APC sampling period (in ms)*
- [PDMv5\\_DLL](#) int [SetAPCSamplingPeriod\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned short time)  
*Set the APC sampling period (in ms)*
- [PDMv5\\_DLL](#) int [ReadAPCHysteresisPercentage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*percentage)  
*Read the APC hysteresis percentage (in %)*
- [PDMv5\\_DLL](#) int [SetAPCHysteresisPercentage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float percentage)  
*Set the APC hysteresis percentage (in %)*
- [PDMv5\\_DLL](#) int [ReadPulseLaserStatusPDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*status)  
*Read the pulse laser status.*
- [PDMv5\\_DLL](#) int [SetPulseLaserStatus\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char status)  
*Set the pulse laser status.*
- [PDMv5\\_DLL](#) int [ReadPulseCurrentSource\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*source)  
*Read the pulse current source.*
- [PDMv5\\_DLL](#) int [SetPulseCurrentSource\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char source)

*Set the pulse current source.*

- [PDMv5\\_DLL](#) int [ReadPULSE\\_INDetectionThreshold\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*voltage)

*Read the PULSE\_IN detection threshold (in mV)*

- [PDMv5\\_DLL](#) int [SetPULSE\\_INDetectionThreshold\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float voltage)

*Set the PULSE\_IN detection threshold (in mV)*

- [PDMv5\\_DLL](#) int [ReadCWLaserStatus\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*status)

*Read the CW laser status.*

- [PDMv5\\_DLL](#) int [SetCWLaserStatus\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char status)

*Set the CW laser status.*

- [PDMv5\\_DLL](#) int [ReadCWCurrentSource\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*status)

*Read the CW current source.*

- [PDMv5\\_DLL](#) int [SetCWCurrentSource\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char status)

*Set the CW current source.*

- [PDMv5\\_DLL](#) int [ReadCWMMaximumCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)

*Read the CW maximum current (in mA)*

- [PDMv5\\_DLL](#) int [SetCWMMaximumCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float current)

*Set the CW maximum current (in mA)*

- [PDMv5\\_DLL](#) int [ReadCWCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)

*Read the CW current (in mA)*

- [PDMv5\\_DLL](#) int [SetCWCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float current)

*Set the CW current (in mA)*

- [PDMv5\\_DLL](#) int [ReadModulationStatus\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*status)

*Read the modulation status.*

- [PDMv5\\_DLL](#) int [SetModulationStatus\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char status)

*Set the modulation status.*

- [PDMv5\\_DLL](#) int [ReadModulationMaximumCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)

*Read the maximum modulation current (in mA)*

- [PDMv5\\_DLL](#) int [SetModulationMaximumCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float current)

*Set the maximum modulation current (in mA)*

- [PDMv5\\_DLL](#) int [ReadModulationCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)

*Read the modulation current (in mA)*

- [PDMv5\\_DLL](#) int [SetModulationCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float current)

*Set the modulation current (in mA)*

- [PDMv5\\_DLL](#) int [ReadModulationInternalType\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*type)

*Read the internal modulation type.*

- [PDMv5\\_DLL](#) int [SetModulationInternalType\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char type)

*Set the internal modulation type.*

- [PDMv5\\_DLL](#) int [ReadModulationFrequency\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*frequency)

- Read the modulation frequency.*
- [PDMv5\\_DLL](#) int [SetModulationFrequency\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float frequency)
- Set the modulation frequency (in Hz)*
- [PDMv5\\_DLL](#) int [ReadModulationExternalCurrentGain\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*gain)
- Read the external modulation current gain.*
- [PDMv5\\_DLL](#) int [SetModulationExternalCurrentGain\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float gain)
- Set the gain external modulation current.*
- [PDMv5\\_DLL](#) int [ReadLIVPauseInterval\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*time)
- Read the LIV pause interval.*
- [PDMv5\\_DLL](#) int [SetLIVPauseInterval\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float time)
- Set the LIV pause interval.*
- [PDMv5\\_DLL](#) int [ReadLIVMeasuresCount\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*value)
- Read the LIV measures count.*
- [PDMv5\\_DLL](#) int [SetLIVMeasuresCount\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char value)
- Set the LIV measures count.*
- [PDMv5\\_DLL](#) int [ReadLIVCurrentStep\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)
- Read the LIV current step (in mA)*
- [PDMv5\\_DLL](#) int [SetLIVCurrentStep\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float current)
- Set the LIV current step (in mA)*
- [PDMv5\\_DLL](#) int [ReadLIVMinimumCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)
- Read the LIV minimum current (in mA)*
- [PDMv5\\_DLL](#) int [SetLIVMinimumCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float current)
- Set the LIV minimum current (in mA)*
- [PDMv5\\_DLL](#) int [ReadLIVMaximumCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)
- Read the LIV maximum current (in mA)*
- [PDMv5\\_DLL](#) int [SetLIVMaximumCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float current)
- Set the LIV maximum current (in mA)*
- [PDMv5\\_DLL](#) int [ReadLIVPhotodiode\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*photodiode)
- Read the LIV photodiode choice.*
- [PDMv5\\_DLL](#) int [SetLIVPhotodiode\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char photodiode)
- Set the LIV photodiode choice.*
- [PDMv5\\_DLL](#) int [ReadLIVPulseWidth\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*time)
- Read the LIV pulse width (in ms)*
- [PDMv5\\_DLL](#) int [SetLIVPulseWidth\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float time)
- Set the LIV pulse width (in ms)*
- [PDMv5\\_DLL](#) int [ReadBFMGain\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*gain)
- Read the BFM gain.*
- [PDMv5\\_DLL](#) int [SetBFMGain\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float gain)
- Set the BFM gain.*
- [PDMv5\\_DLL](#) int [ReadPD\\_EXTGain\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*gain)
- Read the PD\_EXT photodiode gain.*
- [PDMv5\\_DLL](#) int [SetPD\\_EXTGain\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float gain)
- Set the PD\_EXT photodiode gain.*

- [PDMv5\\_DLL](#) int [ReadExternalMultiElementsBoardCommandVoltage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned short \*voltage)  
*Read the external multielements board command voltage (in mV)*
- [PDMv5\\_DLL](#) int [SetExternalMultiElementsBoardCommandVoltage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned short voltage)  
*Set the external multielements board command voltage (in mV)*
- [PDMv5\\_DLL](#) int [MeasureKeyStatus\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*status)  
*Read the key status.*
- [PDMv5\\_DLL](#) int [MeasureBNCInterlockStatus\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*status)  
*Read the BNC interlock status.*
- [PDMv5\\_DLL](#) int [MeasureEXTInterlockStatus\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*status)  
*Read the external interlock status.*
- [PDMv5\\_DLL](#) int [MeasureLIVStatus\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned int \*percentage)  
*Read the LIV status (in %)*
- [PDMv5\\_DLL](#) int [MeasurePeakCurrentConsign\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)  
*Read the measured peak current consign (in mA)*
- [PDMv5\\_DLL](#) int [MeasureCWCurrentConsign\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)  
*Read the measured CW consign (in mA)*
- [PDMv5\\_DLL](#) int [MeasureTemperatureConsign\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*temperature)  
*Read the measured temperature consign (in °C)*
- [PDMv5\\_DLL](#) int [MeasureDiodeTemperature\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*temperature)  
*Read the measured diode temperature (in °C)*
- [PDMv5\\_DLL](#) int [MeasureTECVoltageReference\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*voltage)  
*Read the measured TEC Voltage reference (in mV)*
- [PDMv5\\_DLL](#) int [MeasureTECCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)  
*Read the measured TEC Current (in mA)*
- [PDMv5\\_DLL](#) int [MeasureTECExternalTemperature\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*temperature)  
*Read the measured TEC external temperature (in °C)*
- [PDMv5\\_DLL](#) int [MeasureMOSTemperature\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*temperature)  
*Read the measured MOS temperature (in °C)*
- [PDMv5\\_DLL](#) int [MeasureTECVoltage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*voltage)  
*Read the measured TEC Voltage (in mV)*
- [PDMv5\\_DLL](#) int [MeasureDiodeVoltage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*voltage)  
*Read the measured diode voltage (in V)*
- [PDMv5\\_DLL](#) int [MeasureDiodeCWCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)  
*Read the measured diode CW current (in mA)*
- [PDMv5\\_DLL](#) int [MeasureDiodeAverageCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)  
*Read the measured diode average current (in mA)*
- [PDMv5\\_DLL](#) int [MeasureDiodePulsedCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)

- Read the measured diode pulsed current (in mA)*
- [PDMv5\\_DLL](#) int [MeasureBFMCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)
- Read the measured BFM current (in  $\mu A$ )*
- [PDMv5\\_DLL](#) int [MeasurePD\\_EXTCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)
- Read the measured PD\_EXT current (in  $\mu A$ )*
- [PDMv5\\_DLL](#) int [MeasureCentralDriverEnableStatus\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*status)
- Read the Central Driver enable status.*
- [PDMv5\\_DLL](#) int [MeasureCentralMMDEnableStatus\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned char \*status)
- Read the Central MMD enable status.*
- [PDMv5\\_DLL](#) int [MeasurePatchTECExternalTemperature\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*temperature)
- Read the measured patch board TEC external temperature (in  $^{\circ}C$ )*
- [PDMv5\\_DLL](#) int [MeasurePatchPICOLASCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)
- Read the measured patch board PICOLAS current (in mA)*
- [PDMv5\\_DLL](#) int [MeasurePatchPICOLASComplianceVoltage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*voltage)
- Read the measured patch board PICOLAS compliance voltage (in V)*
- [PDMv5\\_DLL](#) int [MeasurePatchTECCurrent\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*current)
- Read the measured patch board TEC current (in mA)*
- [PDMv5\\_DLL](#) int [MeasurePatchBatteryVoltage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*voltage)
- Read the measured patch board ALIM battery voltage (in V)*
- [PDMv5\\_DLL](#) int [MeasureInputVoltage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*voltage)
- Read the measured input voltage (in V)*
- [PDMv5\\_DLL](#) int [MeasureComplianceVoltage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*voltage)
- Read the measured compliance voltage (in V)*
- [PDMv5\\_DLL](#) int [Measure5VPositiveVoltage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*voltage)
- Read the 5V alimentation voltage (in V)*
- [PDMv5\\_DLL](#) int [Measure3V3PositiveVoltage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*voltage)
- Read the 3.3V alimentation voltage (in V)*
- [PDMv5\\_DLL](#) int [Measure2V5PositiveVoltage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*voltage)
- Read the 2.5V alimentation voltage (in V)*
- [PDMv5\\_DLL](#) int [Measure1V3PositiveVoltage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*voltage)
- Read the 1.3V alimentation voltage (in V)*
- [PDMv5\\_DLL](#) int [Measure5VNegativeVoltage\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, float \*voltage)
- Read the -5V alimentation voltage (in V)*
- [PDMv5\\_DLL](#) int [MeasureAlarms\\_PDMv5](#) (struct sp\_port \*port, unsigned char address, unsigned int \*alarms)
- Read the alarms.*

### 2.1.1 Macro Definition Documentation

### 2.1.1.1 PDMv5\_DLL

```
#define PDMv5_DLL
```

## 2.1.2 Function Documentation

### 2.1.2.1 ApplyRequest\_PDMv5()

```
PDMv5_DLL int ApplyRequest_PDMv5 (  
    struct sp_port * port,  
    unsigned char address )
```

Apply values and parameters (this function must be called in order to really change the state of the PDMv5 card, it can be called after each function or after a list of functions)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device

#### Returns

An error code or 0

### 2.1.2.2 CloseCommunication\_PDMv5()

```
PDMv5_DLL void CloseCommunication_PDMv5 (  
    struct sp_port * port )
```

Close the communication with the device.

#### Parameters

in	<i>port</i>	The port to close
----	-------------	-------------------

### 2.1.2.3 GetLIVMeasures\_PDMv5()

```
PDMv5_DLL int GetLIVMeasures_PDMv5 (  
    struct sp_port * port,
```



```

    unsigned char address,
    unsigned short measureStartIndex,
    unsigned short measureEndIndex,
    float * voltages,
    float * currents,
    float * powers )

```

Get the LIV Measures.

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>measureStartIndex</i>	The first index of the measures that will be read
in	<i>measureEndIndex</i>	The last index of the measures that will be read
out	<i>voltages</i>	The voltages measured
out	<i>currents</i>	The currents measured
out	<i>powers</i>	The powers measured

#### Returns

An error code or 0

#### 2.1.2.4 GetLIVParameters\_PDMv5()

```

PDMv5_DLL int GetLIVParameters_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * currentMinimum,
    float * currentMaximum,
    float * currentStep,
    float * pauseInterval,
    unsigned char * count,
    unsigned char * photodiode )

```

Get the parameters with which the LIV was done.

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>currentMinimum</i>	The minimum current for the LIV (in mA)
out	<i>currentMaximum</i>	The maximum current for the LIV (in mA)
out	<i>currentStep</i>	The current step for the LIV (in mA)
out	<i>pauseInterval</i>	The pause interval for the LIV (in ms)
out	<i>count</i>	The number of the measures
out	<i>photodiode</i>	The photodiode used, 0 for BFM, 1 for PD_EXT

**Returns**

An error code or 0

**2.1.2.5 Measure1V3PositiveVoltage\_PDMv5()**

```
PDMv5_DLL int Measure1V3PositiveVoltage_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * voltage )
```

Read the 1.3V alimentation voltage (in V)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>voltage</i>	The 1.3V alimentation voltage (in V)

**Returns**

An error code or 0

**2.1.2.6 Measure2V5PositiveVoltage\_PDMv5()**

```
PDMv5_DLL int Measure2V5PositiveVoltage_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * voltage )
```

Read the 2.5V alimentation voltage (in V)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>voltage</i>	The 2.5V alimentation voltage (in V)

**Returns**

An error code or 0

### 2.1.2.7 Measure3V3PositiveVoltage\_PDMv5()

```
PDMv5_DLL int Measure3V3PositiveVoltage_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * voltage )
```

Read the 3.3V alimentation voltage (in V)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>voltage</i>	The 3.3V alimentation voltage (in V)

#### Returns

An error code or 0

### 2.1.2.8 Measure5VNegativeVoltage\_PDMv5()

```
PDMv5_DLL int Measure5VNegativeVoltage_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * voltage )
```

Read the -5V alimentation voltage (in V)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>voltage</i>	The -5V alimentation voltage (in V)

#### Returns

An error code or 0

### 2.1.2.9 Measure5VPositiveVoltage\_PDMv5()

```
PDMv5_DLL int Measure5VPositiveVoltage_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * voltage )
```

Read the 5V alimentation voltage (in V)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>voltage</i>	The 5V alimentation voltage (in V)

**Returns**

An error code or 0

**2.1.2.10 MeasureAlarms\_PDMv5()**

```
PDMv5_DLL int MeasureAlarms_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned int * alarms )
```

Read the alarms.

Each alarm is coded by a bit

- b0 : Interlock
- b1 : Average Current
- b2 : Diode Temperature
- b3 : ICHG Temperature (CW mode) or MOS Temperature (Pulsed mode)
- b4 : TEC Ext
- b5 : BNC Interlock
- b6 : Ext Interlock
- b7 : Key Interlock
- b8 : MOS CW
- b9 : Reserved
- b10 : Open Circuit

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>alarms</i>	The alarms

**Returns**

An error code or 0

### 2.1.2.11 MeasureBFMCurrent\_PDMv5()

```
PDMv5_DLL int MeasureBFMCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * current )
```

Read the measured BFM current (in  $\mu\text{A}$ )

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The measured BFM current (in $\mu\text{A}$ )

#### Returns

An error code or 0

### 2.1.2.12 MeasureBNCInterlockStatus\_PDMv5()

```
PDMv5_DLL int MeasureBNCInterlockStatus_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * status )
```

Read the BNC interlock status.

0 : OFF, 1 : ON

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>status</i>	The BNC interlock status

#### Returns

An error code or 0

### 2.1.2.13 MeasureCentralDriverEnableStatus\_PDMv5()

```
PDMv5_DLL int MeasureCentralDriverEnableStatus_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * status )
```

Read the Central Driver enable status.

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>status</i>	The Central Driver enable status

#### Returns

An error code or 0

### 2.1.2.14 MeasureCentralMMDEnableStatus\_PDMv5()

```
PDMv5_DLL int MeasureCentralMMDEnableStatus_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * status )
```

Read the Central MMD enable status.

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>status</i>	The Central MMD enable status

#### Returns

An error code or 0

### 2.1.2.15 MeasureComplianceVoltage\_PDMv5()

```
PDMv5_DLL int MeasureComplianceVoltage_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * voltage )
```

Read the measured compliance voltage (in V)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>voltage</i>	The measured compliance voltage (in V)

## Returns

An error code or 0

**2.1.2.16 MeasureCWCurrentConsign\_PDMv5()**

```
PDMv5_DLL int MeasureCWCurrentConsign_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * current )
```

Read the measured CW consign (in mA)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The measured CW consign (in mA)

## Returns

An error code or 0

**2.1.2.17 MeasureDiodeAverageCurrent\_PDMv5()**

```
PDMv5_DLL int MeasureDiodeAverageCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * current )
```

Read the measured diode average current (in mA)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The measured diode average current (in mA)

**Returns**

An error code or 0

**2.1.2.18 MeasureDiodeCWCurrent\_PDMv5()**

```
PDMv5_DLL int MeasureDiodeCWCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * current )
```

Read the measured diode CW current (in mA)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The measured diode CW current (in mA)

**Returns**

An error code or 0

**2.1.2.19 MeasureDiodePulsedCurrent\_PDMv5()**

```
PDMv5_DLL int MeasureDiodePulsedCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * current )
```

Read the measured diode pulsed current (in mA)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The measured diode pulsed current (in mA)

**Returns**

An error code or 0



### 2.1.2.20 MeasureDiodeTemperature\_PDMv5()

```
PDMv5_DLL int MeasureDiodeTemperature_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * temperature )
```

Read the measured diode temperature (in °C)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>temperature</i>	The measured diode temperature (in °C)

#### Returns

An error code or 0

### 2.1.2.21 MeasureDiodeVoltage\_PDMv5()

```
PDMv5_DLL int MeasureDiodeVoltage_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * voltage )
```

Read the measured diode voltage (in V)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>voltage</i>	The measured diode voltage (in V)

#### Returns

An error code or 0

### 2.1.2.22 MeasureEXTInterlockStatus\_PDMv5()

```
PDMv5_DLL int MeasureEXTInterlockStatus_PDMv5 (
    struct sp_port * port,
```

```

    unsigned char address,
    unsigned char * status )

```

Read the external interlock status.

0 : OFF, 1 : ON

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>status</i>	The external interlock status

#### Returns

An error code or 0

### 2.1.2.23 MeasureInputVoltage\_PDMv5()

```

PDMv5_DLL int MeasureInputVoltage_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * voltage )

```

Read the measured input voltage (in V)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>voltage</i>	The measured input voltage (in V)

#### Returns

An error code or 0

### 2.1.2.24 MeasureKeyStatus\_PDMv5()

```

PDMv5_DLL int MeasureKeyStatus_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * status )

```

Read the key status.

0 : UNTURNED, 1 : TURNED

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>status</i>	The key status

## Returns

An error code or 0

**2.1.2.25 MeasureLIVStatus\_PDMv5()**

```
PDMv5_DLL int MeasureLIVStatus_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    unsigned int * percentage )
```

Read the LIV status (in %)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>percentage</i>	The LIV status (in %)

## Returns

An error code or 0

**2.1.2.26 MeasureMOSTemperature\_PDMv5()**

```
PDMv5_DLL int MeasureMOSTemperature_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    float * temperature )
```

Read the measured MOS temperature (in °C)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>temperature</i>	The measured MOS temperature (in °C)

**Returns**

An error code or 0

**2.1.2.27 MeasurePatchBatteryVoltage\_PDMv5()**

```
PDMv5_DLL int MeasurePatchBatteryVoltage_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * voltage )
```

Read the measured patch board ALIM battery voltage (in V)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>voltage</i>	The measured patch board ALIM battery voltage (in V)

**Returns**

An error code or 0

**2.1.2.28 MeasurePatchPICOLASComplianceVoltage\_PDMv5()**

```
PDMv5_DLL int MeasurePatchPICOLASComplianceVoltage_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * voltage )
```

Read the measured patch board PICOLAS compliance voltage (in V)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>voltage</i>	The measured patch board PICOLAS compliance voltage (in V)

**Returns**

An error code or 0

### 2.1.2.29 MeasurePatchPICOLASCurrent\_PDMv5()

```
PDMv5_DLL int MeasurePatchPICOLASCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * current )
```

Read the measured patch board PICOLAS current (in mA)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The measured patch board PICOLAS current (in mA)

#### Returns

An error code or 0

### 2.1.2.30 MeasurePatchTECCurrent\_PDMv5()

```
PDMv5_DLL int MeasurePatchTECCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * current )
```

Read the measured patch board TEC current (in mA)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The measured patch board TEC current (in mA)

#### Returns

An error code or 0

### 2.1.2.31 MeasurePatchTECExternalTemperature\_PDMv5()

```
PDMv5_DLL int MeasurePatchTECExternalTemperature_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * temperature )
```

Read the measured patch board TEC external temperature (in °C)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>temperature</i>	The measured patch board TEC external temperature (in °C)

## Returns

An error code or 0

**2.1.2.32 MeasurePD\_EXTCurrent\_PDMv5()**

```
PDMv5_DLL int MeasurePD_EXTCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * current )
```

Read the measured PD\_EXT current (in µA)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The measured PD_EXT current (in µA)

## Returns

An error code or 0

**2.1.2.33 MeasurePeakCurrentConsign\_PDMv5()**

```
PDMv5_DLL int MeasurePeakCurrentConsign_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * current )
```

Read the measured peak current consign (in mA)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The measured peak current consign (in mA)

**Returns**

An error code or 0

**2.1.2.34 MeasureTECCurrent\_PDMv5()**

```
PDMv5_DLL int MeasureTECCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * current )
```

Read the measured TEC Current (in mA)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The measured TEC Current (in mA)

**Returns**

An error code or 0

**2.1.2.35 MeasureTECExternalTemperature\_PDMv5()**

```
PDMv5_DLL int MeasureTECExternalTemperature_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * temperature )
```

Read the measured TEC external temperature (in °C)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>temperature</i>	The measured TEC external temperature (in °C)

**Returns**

An error code or 0

### 2.1.2.36 MeasureTECVoltage\_PDMv5()

```
PDMv5_DLL int MeasureTECVoltage_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * voltage )
```

Read the measured TEC Voltage (in mV)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>voltage</i>	The measured TEC Voltage (in mV)

#### Returns

An error code or 0

### 2.1.2.37 MeasureTECVoltageReference\_PDMv5()

```
PDMv5_DLL int MeasureTECVoltageReference_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * voltage )
```

Read the measured TEC Voltage reference (in mV)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>voltage</i>	The measured TEC Voltage reference (in mV)

#### Returns

An error code or 0

### 2.1.2.38 MeasureTemperatureConsign\_PDMv5()

```
PDMv5_DLL int MeasureTemperatureConsign_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * temperature )
```

Read the measured temperature consign (in °C)



## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>temperature</i>	The measured temperature consign (in °C)

## Returns

An error code or 0

**2.1.2.39 OpenCommunication\_PDMv5()**

```
PDMv5_DLL enum sp_return OpenCommunication_PDMv5 (
    char * portName,
    struct sp_port ** portPointer )
```

Open the communication with the device.

## Parameters

in	<i>portName</i>	The name of the COM port
out	<i>portPointer</i>	A pointer to a port that will be initialized correctly

## Returns

A value indicating the status error (RS232 error code) = 0 if no error

**2.1.2.40 ReadAddress\_PDMv5()**

```
PDMv5_DLL int ReadAddress_PDMv5 (
    struct sp_port * port,
    unsigned char * address )
```

Read the adress of the device.

## Parameters

in	<i>port</i>	The port used
out	<i>address</i>	The address of the device

## Returns

An error code or 0

#### 2.1.2.41 ReadAPCHysteresisPercentage\_PDMv5()

```
PDMv5_DLL int ReadAPCHysteresisPercentage_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * percentage )
```

Read the APC hysteresis percentage (in %)

##### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>percentage</i>	The APC hysteresis percentage (in %)

##### Returns

An error code or 0

#### 2.1.2.42 ReadAPCPhotodiode\_PDMv5()

```
PDMv5_DLL int ReadAPCPhotodiode_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * mode )
```

Read the APC photodiode.

0 : BFM, 1 : PD\_EXT

##### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>mode</i>	The APC photodiode

##### Returns

An error code or 0

### 2.1.2.43 ReadAPCSamplingPeriod\_PDMv5()

```
PDMv5_DLL int ReadAPCSamplingPeriod_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned short * time )
```

Read the APC sampling period (in ms)

0 : BFM, 1 : PD\_EXT

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>time</i>	The APC sampling period (in ms)

#### Returns

An error code or 0

### 2.1.2.44 ReadBFMGain\_PDMv5()

```
PDMv5_DLL int ReadBFMGain_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * gain )
```

Read the BFM gain.

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>gain</i>	The BFM gain

#### Returns

An error code or 0

### 2.1.2.45 ReadComplianceVoltage\_PDMv5()

```
PDMv5_DLL int ReadComplianceVoltage_PDMv5 (
    struct sp_port * port,
```

```
    unsigned char address,  
    float * voltage )
```

Read The Compliance Voltage.

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>voltage</i>	The Compliance Voltage

#### Returns

An error code or 0

#### 2.1.2.46 ReadCurrent\_PDMv5()

```
PDMv5_DLL int ReadCurrent_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    float * current )
```

Read the current (in % of maximum current)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The current (in % of maximum current)

#### Returns

An error code or 0

#### 2.1.2.47 ReadCWCurrent\_PDMv5()

```
PDMv5_DLL int ReadCWCurrent_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    float * current )
```

Read the CW current (in mA)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The CW current (in mA)

## Returns

An error code or 0

**2.1.2.48 ReadCWCurrentSource\_PDMv5()**

```
PDMv5_DLL int ReadCWCurrentSource_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * status )
```

Read the CW current source.

0 : Internal DAC; 1 : Potentiometer POT; 2 : External (SMA or connector)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>status</i>	The CW current source

## Returns

An error code or 0

**2.1.2.49 ReadCWLaserStatus\_PDMv5()**

```
PDMv5_DLL int ReadCWLaserStatus_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * status )
```

Read the CW laser status.

0 : OFF, 1 : ON

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>status</i>	The CW laser status

**Returns**

An error code or 0

**2.1.2.50 ReadCWMaximumCurrent\_PDMv5()**

```
PDMv5_DLL int ReadCWMaximumCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * current )
```

Read the CW maximum current (in mA)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The CW maximum current (in mA)

**Returns**

An error code or 0

**2.1.2.51 ReadDelay\_PDMv5()**

```
PDMv5_DLL int ReadDelay_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned int * delay )
```

Read the delay (in ps)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>delay</i>	The delay (in ps)

**Returns**

An error code or 0

**2.1.2.52 ReadDelayLine\_PDMv5()**

```
PDMv5_DLL int ReadDelayLine_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * mode )
```

Read the delay line.

0 : NONE (SMA TTL/LVTTL input, duration of pulses will be the same as the applied signal), 1: Internal

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>mode</i>	The delay line

**Returns**

An error code or 0

**2.1.2.53 ReadExternalMultiElementsBoardCommandVoltage\_PDMv5()**

```
PDMv5_DLL int ReadExternalMultiElementsBoardCommandVoltage_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned short * voltage )
```

Read the external multielements board command voltage (in mV)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>voltage</i>	The external multielements board command voltage (in mV)

**Returns**

An error code or 0

### 2.1.2.54 ReadFrequency\_PDMv5()

```
PDMv5_DLL int ReadFrequency_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned int * frequency )
```

Read the frequency (in Hz)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>frequency</i>	The frequency (in Hz)

#### Returns

An error code or 0

### 2.1.2.55 ReadHardwareType\_PDMv5()

```
PDMv5_DLL int ReadHardwareType_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned int * type )
```

Read the hardware type.

- 'A' is for PDM
- 'B' is for MMD
- 'C' is for Cristal
- 'D' is for Central
- 'E' is for Pulsepicker
- 'F' is for Shaper
- 'G' is for PDMv5

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>type</i>	An integer corresponding to the hardware type



**Returns**

An error code or 0

**2.1.2.56 ReadHardwareVersion\_PDMv5()**

```
PDMv5_DLL int ReadHardwareVersion_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * version )
```

Read the hardware version.

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>version</i>	Three integers corresponding to the hardware version

**Returns**

An error code or 0

**2.1.2.57 ReadLaserStatus\_PDMv5()**

```
PDMv5_DLL int ReadLaserStatus_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * status )
```

Read the laser status.

0 for OFF, 1 for ON

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>status</i>	The laser status

**Returns**

An error code or 0

### 2.1.2.58 ReadLIVCurrentStep\_PDMv5()

```
PDMv5_DLL int ReadLIVCurrentStep_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * current )
```

Read the LIV current step (in mA)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The LIV current step (in mA)

#### Returns

An error code or 0

### 2.1.2.59 ReadLIVMaximumCurrent\_PDMv5()

```
PDMv5_DLL int ReadLIVMaximumCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * current )
```

Read the LIV maximum current (in mA)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The LIV maximum current (in mA)

#### Returns

An error code or 0

### 2.1.2.60 ReadLIVMeasuresCount\_PDMv5()

```
PDMv5_DLL int ReadLIVMeasuresCount_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * value )
```

Read the LIV measures count.

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>value</i>	The LIV measures count

## Returns

An error code or 0

**2.1.2.61 ReadLIVMinimumCurrent\_PDMv5()**

```
PDMv5_DLL int ReadLIVMinimumCurrent_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    float * current )
```

Read the LIV minimum current (in mA)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The LIV minimum current (in mA)

## Returns

An error code or 0

**2.1.2.62 ReadLIVPauseInterval\_PDMv5()**

```
PDMv5_DLL int ReadLIVPauseInterval_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    float * time )
```

Read the LIV pause interval.

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>time</i>	the LIV pause interval (in ms)

**Returns**

An error code or 0

**2.1.2.63 ReadLIVPhotodiode\_PDMv5()**

```
PDMv5_DLL int ReadLIVPhotodiode_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * photodiode )
```

Read the LIV photodiode choice.

0 for BFM, 1 for PD\_EXT

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>photodiode</i>	The LIV photodiode choice

**Returns**

An error code or 0

**2.1.2.64 ReadLIVPulseWidth\_PDMv5()**

```
PDMv5_DLL int ReadLIVPulseWidth_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * time )
```

Read the LIV pulse width (in ms)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>time</i>	The LIV pulse width (in ms)

**Returns**

An error code or 0

### 2.1.2.65 ReadMaximumAverageCurrent\_PDMv5()

```
PDMv5_DLL int ReadMaximumAverageCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * current )
```

Read the mean current limit (in mA)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The mean current limit (in mA)

#### Returns

An error code or 0

### 2.1.2.66 ReadMaximumCurrent\_PDMv5()

```
PDMv5_DLL int ReadMaximumCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * current )
```

Read the maximum current limit (in mA)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The maximum current limit (in mA)

#### Returns

An error code or 0

### 2.1.2.67 ReadModulationCurrent\_PDMv5()

```
PDMv5_DLL int ReadModulationCurrent_PDMv5 (
    struct sp_port * port,
```

```

    unsigned char address,
    float * current )

```

Read the modulation current (in mA)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The modulation current (in mA)

#### Returns

An error code or 0

### 2.1.2.68 ReadModulationExternalCurrentGain\_PDMv5()

```

PDMv5_DLL int ReadModulationExternalCurrentGain_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * gain )

```

Read the external modulation current gain.

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>gain</i>	The external modulation current gain (in mA/V)

#### Returns

An error code or 0

### 2.1.2.69 ReadModulationFrequency\_PDMv5()

```

PDMv5_DLL int ReadModulationFrequency_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * frequency )

```

Read the modulation frequency.

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>frequency</i>	The modulation frequency (in Hz)

## Returns

An error code or 0

**2.1.2.70 ReadModulationInternalType\_PDMv5()**

```
PDMv5_DLL int ReadModulationInternalType_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * type )
```

Read the internal modulation type.

0 : Sinusoidal, 1 : Triangular, 2 : Square

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>type</i>	The internal modulation type

## Returns

An error code or 0

**2.1.2.71 ReadModulationMaximumCurrent\_PDMv5()**

```
PDMv5_DLL int ReadModulationMaximumCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * current )
```

Read the maximum modulation current (in mA)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The maximum modulation current (in mA)

**Returns**

An error code or 0

**2.1.2.72 ReadModulationStatus\_PDMv5()**

```
PDMv5_DLL int ReadModulationStatus_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * status )
```

Read the modulation status.

0 : Off; 1 : On

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>status</i>	modulation status

**Returns**

An error code or 0

**2.1.2.73 ReadOperatingMode\_PDMv5()**

```
PDMv5_DLL int ReadOperatingMode_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * mode )
```

Read the operating mode.

0 : ACC, 1 : APC

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>mode</i>	The operating mode



**Returns**

An error code or 0

**2.1.2.74 ReadPD\_EXTGain\_PDMv5()**

```
PDMv5_DLL int ReadPD_EXTGain_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    float * gain )
```

Read the PD\_EXT photodiode gain.

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>gain</i>	The PD_EXT photodiode gain

**Returns**

An error code or 0

**2.1.2.75 ReadPULSE\_INDetectionThreshold\_PDMv5()**

```
PDMv5_DLL int ReadPULSE_INDetectionThreshold_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    float * voltage )
```

Read the PULSE\_IN detection threshold (in mV)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>voltage</i>	The PULSE_IN detection threshold (in mV)

**Returns**

An error code or 0

### 2.1.2.76 ReadPulseCurrentSource\_PDMv5()

```
PDMv5_DLL int ReadPulseCurrentSource_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * source )
```

Read the pulse current source.

0 : Internal DAC; 1 : Potentiometer POT; 2 : External (SMA or connector)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>source</i>	The pulse current source

#### Returns

An error code or 0

### 2.1.2.77 ReadPulseLaserStatusPDMv5()

```
PDMv5_DLL int ReadPulseLaserStatusPDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * status )
```

Read the pulse laser status.

0 : OFF, 1 : ON

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>status</i>	The pulsed mode laser status

#### Returns

An error code or 0

### 2.1.2.78 ReadPulseWidth\_PDMv5()

```
PDMv5_DLL int ReadPulseWidth_PDMv5 (
    struct sp_port * port,
```

```
unsigned char address,  
unsigned int * width )
```

Read the pulse width (in ps)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>width</i>	The pulse width (in ps)

#### Returns

An error code or 0

#### 2.1.2.79 ReadSerialNumber\_PDMv5()

```
PDMv5_DLL int ReadSerialNumber_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    unsigned int * serial )
```

Read the serial number.

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>serial</i>	An integer corresponding to the serial number

#### Returns

An error code or 0

#### 2.1.2.80 ReadSoftwareType\_PDMv5()

```
PDMv5_DLL int ReadSoftwareType_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    unsigned int * type )
```

Read the software type.

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>type</i>	An integer corresponding to the software type

**Returns**

An error code or 0

**2.1.2.81 ReadSoftwareVersion\_PDMv5()**

```
PDMv5_DLL int ReadSoftwareVersion_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * version )
```

Read the software version.

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>version</i>	3 integers corresponding to the software version

**Returns**

An error code or 0

**2.1.2.82 ReadSynchro\_PDMv5()**

```
PDMv5_DLL int ReadSynchro_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char * mode )
```

Read the synchronisation line which will trigger pulses.

0 : External TTL/LVTTL (SMA input), 1: External LVDS (optional SMA inputs), 2 : Internal clock

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>mode</i>	The synchronisation line

**Returns**

An error code or 0

**2.1.2.83 ReadTECMaximumCurrent\_PDMv5()**

```
PDMv5_DLL int ReadTECMaximumCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * current )
```

Read TEC maximum current (in mA)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>current</i>	The TEC maximum current (in mA)

**Returns**

An error code or 0

**2.1.2.84 ReadTECMaximumVoltage\_PDMv5()**

```
PDMv5_DLL int ReadTECMaximumVoltage_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * voltage )
```

Read the TEC maximum voltage (in mV)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>voltage</i>	The TEC maximum voltage (in mV)

**Returns**

An error code or 0

### 2.1.2.85 ReadTemperature\_PDMv5()

```
PDMv5_DLL int ReadTemperature_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float * temperature )
```

Read the temperature (in °C)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
out	<i>temperature</i>	The temperature (in °C)

#### Returns

An error code or 0

### 2.1.2.86 Save\_PDMv5()

```
PDMv5_DLL int Save_PDMv5 (
    struct sp_port * port,
    unsigned char address )
```

Save all values into the device memory.

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device

#### Returns

An error code or 0

### 2.1.2.87 SetAddressSpecific\_PDMv5()

```
PDMv5_DLL int SetAddressSpecific_PDMv5 (
    struct sp_port * port,
    unsigned char currentAddress,
    unsigned char newAddress )
```

Set the address of the device.

## Parameters

in	<i>port</i>	The port used
in	<i>currentAddress</i>	The current address of the device
in	<i>newAddress</i>	The new address of the device

## Returns

An error code or 0

**2.1.2.88 SetAPCHysteresisPercentage\_PDMv5()**

```
PDMv5_DLL int SetAPCHysteresisPercentage_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    float percentage )
```

Set the APC hysteresis percentage (in %)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>percentage</i>	The APC hysteresis percentage (in %)

## Returns

An error code or 0

**2.1.2.89 SetAPCPhotodiode\_PDMv5()**

```
PDMv5_DLL int SetAPCPhotodiode_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    unsigned char mode )
```

Set the APC photodiode.

0 : BFM, 1 : PD\_EXT

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>mode</i>	The APC photodiode

**Returns**

An error code or 0

**2.1.2.90 SetAPCSamplingPeriod\_PDMv5()**

```
PDMv5_DLL int SetAPCSamplingPeriod_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned short time )
```

Set the APC sampling period (in ms)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>time</i>	The APC sampling period (in ms)

**Returns**

An error code or 0

**2.1.2.91 SetBFMGain\_PDMv5()**

```
PDMv5_DLL int SetBFMGain_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float gain )
```

Set the BFM gain.

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>gain</i>	The BFM gain

**Returns**

An error code or 0



### 2.1.2.92 SetComplianceVoltage\_PDMv5()

```
PDMv5_DLL int SetComplianceVoltage_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float voltage )
```

Set the compliance voltage (in V)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>voltage</i>	The compliance voltage (in V)

#### Returns

An error code or 0

### 2.1.2.93 SetCurrent\_PDMv5()

```
PDMv5_DLL int SetCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float current )
```

Set the current (in % of maximum current)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>current</i>	The current (in % of maximum current)

#### Returns

An error code or 0

### 2.1.2.94 SetCWCurrent\_PDMv5()

```
PDMv5_DLL int SetCWCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float current )
```

Set the CW current (in mA)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>current</i>	The CW current (in mA)

**Returns**

An error code or 0

**2.1.2.95 SetCWCurrentSource\_PDMv5()**

```
PDMv5_DLL int SetCWCurrentSource_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char status )
```

Set the CW current source.

0 : Internal DAC; 1 : Potentiometer POT; 2 : External (SMA or connector)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>status</i>	The CW current source

**Returns**

An error code or 0

**2.1.2.96 SetCWLaserStatus\_PDMv5()**

```
PDMv5_DLL int SetCWLaserStatus_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char status )
```

Set the CW laser status.

0 : OFF, 1 : ON

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>status</i>	The CW laser status

## Returns

An error code or 0

**2.1.2.97 SetCWMaximumCurrent\_PDMv5()**

```
PDMv5_DLL int SetCWMaximumCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float current )
```

Set the CW maximum current (in mA)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>current</i>	The CW maximum current (in mA)

## Returns

An error code or 0

**2.1.2.98 SetDelay\_PDMv5()**

```
PDMv5_DLL int SetDelay_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned int delay )
```

Set the delay (in ps)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>delay</i>	The delay (in ps)

**Returns**

An error code or 0

**2.1.2.99 SetDelayLine\_PDMv5()**

```
PDMv5_DLL int SetDelayLine_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char mode )
```

Set the delay line.

0 : NONE (SMA TTL/LVTTL input, duration of pulses will be the same as the applied signal), 1: Internal

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>mode</i>	The delay line

**Returns**

An error code or 0

**2.1.2.100 SetExternalMultiElementsBoardCommandVoltage\_PDMv5()**

```
PDMv5_DLL int SetExternalMultiElementsBoardCommandVoltage_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned short voltage )
```

Set the external multielements board command voltage (in mV)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>voltage</i>	The external multielements board command voltage (in mV)

**Returns**

An error code or 0

### 2.1.2.101 SetFrequency\_PDMv5()

```
PDMv5_DLL int SetFrequency_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    unsigned int frequency )
```

Set the frequency (in Hz)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>frequency</i>	The frequency (in Hz)

#### Returns

An error code or 0

### 2.1.2.102 SetLaserStatus\_PDMv5()

```
PDMv5_DLL int SetLaserStatus_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    unsigned char status )
```

Set the laser status.

0 for OFF, 1 for ON

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>status</i>	The laser status

#### Returns

An error code or 0

### 2.1.2.103 SetLIVCurrentStep\_PDMv5()

```
PDMv5_DLL int SetLIVCurrentStep_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float current )
```

Set the LIV current step (in mA)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>current</i>	The LIV current step (in mA)

#### Returns

An error code or 0

### 2.1.2.104 SetLIVMaximumCurrent\_PDMv5()

```
PDMv5_DLL int SetLIVMaximumCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float current )
```

Set the LIV maximum current (in mA)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>current</i>	The LIV maximum current (in mA)

#### Returns

An error code or 0

### 2.1.2.105 SetLIVMeasuresCount\_PDMv5()

```
PDMv5_DLL int SetLIVMeasuresCount_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char value )
```

Set the LIV measures count.

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>value</i>	The LIV measures count

## Returns

An error code or 0

**2.1.2.106 SetLIVMinimumCurrent\_PDMv5()**

```
PDMv5_DLL int SetLIVMinimumCurrent_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    float current )
```

Set the LIV minimum current (in mA)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>current</i>	The LIV minimum current (in mA)

## Returns

An error code or 0

**2.1.2.107 SetLIVPauseInterval\_PDMv5()**

```
PDMv5_DLL int SetLIVPauseInterval_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    float time )
```

Set the LIV pause interval.

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>time</i>	The LIV pause interval (in ms)

**Returns**

An error code or 0

**2.1.2.108 SetLIVPhotodiode\_PDMv5()**

```
PDMv5_DLL int SetLIVPhotodiode_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char photodiode )
```

Set the LIV photodiode choice.

0 for BFM, 1 for PD\_EXT

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>photodiode</i>	The LIV photodiode choice

**Returns**

An error code or 0

**2.1.2.109 SetLIVPulseWidth\_PDMv5()**

```
PDMv5_DLL int SetLIVPulseWidth_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float time )
```

Set the LIV pulse width (in ms)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>time</i>	The LIV pulse width (in ms)

**Returns**

An error code or 0



### 2.1.2.110 SetMaximumAverageCurrent\_PDMv5()

```
PDMv5_DLL int SetMaximumAverageCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float current )
```

Set the mean current limit (in mA)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>current</i>	The mean current limit (in mA)

#### Returns

An error code or 0

### 2.1.2.111 SetMaximumCurrent\_PDMv5()

```
PDMv5_DLL int SetMaximumCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float current )
```

Set the maximum current limit (in mA)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>current</i>	The maximum current limit (in mA)

#### Returns

An error code or 0

### 2.1.2.112 SetModulationCurrent\_PDMv5()

```
PDMv5_DLL int SetModulationCurrent_PDMv5 (
    struct sp_port * port,
```

```
    unsigned char address,
    float current )
```

Set the modulation current (in mA)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>current</i>	The modulation current (in mA)

#### Returns

An error code or 0

### 2.1.2.113 SetModulationExternalCurrentGain\_PDMv5()

```
PDMv5_DLL int SetModulationExternalCurrentGain_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float gain )
```

Set the gain external modulation current.

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>gain</i>	The gain external modulation current (in mA/V)

#### Returns

An error code or 0

### 2.1.2.114 SetModulationFrequency\_PDMv5()

```
PDMv5_DLL int SetModulationFrequency_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float frequency )
```

Set the modulation frequency (in Hz)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>frequency</i>	The modulation frequency (in Hz)

## Returns

An error code or 0

**2.1.2.115 SetModulationInternalType\_PDMv5()**

```
PDMv5_DLL int SetModulationInternalType_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char type )
```

Set the internal modulation type.

0 : Sinusoidal, 1 : Triangular, 2 : Square

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>type</i>	The internal modulation type

## Returns

An error code or 0

**2.1.2.116 SetModulationMaximumCurrent\_PDMv5()**

```
PDMv5_DLL int SetModulationMaximumCurrent_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float current )
```

Set the maximum modulation current (in mA)

## Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>current</i>	The maximum modulation current (in mA)

**Returns**

An error code or 0

**2.1.2.117 SetModulationStatus\_PDMv5()**

```
PDMv5_DLL int SetModulationStatus_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char status )
```

Set the modulation status.

0 : Off; 1 : On

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>status</i>	The modulation status

**Returns**

An error code or 0

**2.1.2.118 SetOperatingMode\_PDMv5()**

```
PDMv5_DLL int SetOperatingMode_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char mode )
```

Set the operating mode.

0 : ACC, 1 : APC

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>mode</i>	The operating mode

**Returns**

An error code or 0

**2.1.2.119 SetPD\_EXTGain\_PDMv5()**

```
PDMv5_DLL int SetPD_EXTGain_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    float gain )
```

Set the PD\_EXT photodiode gain.

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>gain</i>	The PD_EXT photodiode gain

**Returns**

An error code or 0

**2.1.2.120 SetPULSE\_INDetectionThreshold\_PDMv5()**

```
PDMv5_DLL int SetPULSE_INDetectionThreshold_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    float voltage )
```

Set the PULSE\_IN detection threshold (in mV)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>voltage</i>	The PULSE_IN detection threshold (in mV)

**Returns**

An error code or 0

### 2.1.2.121 SetPulseCurrentSource\_PDMv5()

```
PDMv5_DLL int SetPulseCurrentSource_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char source )
```

Set the pulse current source.

0 : Internal DAC; 1 : Potentiometer POT; 2 : External (SMA or connector)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>source</i>	The pulse current source

#### Returns

An error code or 0

### 2.1.2.122 SetPulseLaserStatus\_PDMv5()

```
PDMv5_DLL int SetPulseLaserStatus_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char status )
```

Set the pulse laser status.

0 : OFF, 1 : ON

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>status</i>	The pulse laser status

#### Returns

An error code or 0

### 2.1.2.123 SetPulseWidth\_PDMv5()

```
PDMv5_DLL int SetPulseWidth_PDMv5 (
    struct sp_port * port,
```

```
unsigned char address,  
unsigned int width )
```

Set the pulse width (in ps)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>width</i>	The pulse width (in ps)

#### Returns

An error code or 0

#### 2.1.2.124 SetTECMaximumCurrent\_PDMv5()

```
PDMv5_DLL int SetTECMaximumCurrent_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    float current )
```

Set the TEC maximum current (in mA)

#### Parameters

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>current</i>	The TEC maximum current (in mA)

#### Returns

An error code or 0

#### 2.1.2.125 SetTECMaximumVoltage\_PDMv5()

```
PDMv5_DLL int SetTECMaximumVoltage_PDMv5 (  
    struct sp_port * port,  
    unsigned char address,  
    float voltage )
```

Set the TEC maximum voltage (in mV)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>voltage</i>	The TEC maximum voltage (in mV)

**Returns**

An error code or 0

**2.1.2.126 SetTemperature\_PDMv5()**

```
PDMv5_DLL int SetTemperature_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    float temperature )
```

Set the temperature (in °C)

**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>temperature</i>	The temperature (in °C)

**Returns**

An error code or 0

**2.1.2.127 StartCalibration\_PDMv5()**

```
PDMv5_DLL int StartCalibration_PDMv5 (
    struct sp_port * port,
    unsigned char address,
    unsigned char calibrationID )
```

Start a calibration.

Once a calibration is started, all commands will return an error code -30032 until the calibration is finished Calibration IDs are :

- 0x00 : RESERVED
- 0x01 : Measure LIV (Light, Intensity, Voltage)
- 0x02 : Measure IP (Intensity, Power)
- 0x03 : Measure IV (Intensity, Voltage)



**Parameters**

in	<i>port</i>	The port used
in	<i>address</i>	The address of the device
in	<i>calibrationID</i>	The identifier of the calibration

**Returns**

An error code or 0



# Index

ApplyRequest\_PDMv5  
pdmv5.h, [10](#)

CloseCommunication\_PDMv5  
pdmv5.h, [10](#)

GetLIVMeasures\_PDMv5  
pdmv5.h, [10](#)

GetLIVParameters\_PDMv5  
pdmv5.h, [11](#)

Measure1V3PositiveVoltage\_PDMv5  
pdmv5.h, [12](#)

Measure2V5PositiveVoltage\_PDMv5  
pdmv5.h, [12](#)

Measure3V3PositiveVoltage\_PDMv5  
pdmv5.h, [12](#)

Measure5VNegativeVoltage\_PDMv5  
pdmv5.h, [13](#)

Measure5VPositiveVoltage\_PDMv5  
pdmv5.h, [13](#)

MeasureAlarms\_PDMv5  
pdmv5.h, [14](#)

MeasureBFMCurrent\_PDMv5  
pdmv5.h, [15](#)

MeasureBNCInterlockStatus\_PDMv5  
pdmv5.h, [15](#)

MeasureCentralDriverEnableStatus\_PDMv5  
pdmv5.h, [15](#)

MeasureCentralMMDEnableStatus\_PDMv5  
pdmv5.h, [16](#)

MeasureComplianceVoltage\_PDMv5  
pdmv5.h, [16](#)

MeasureCWCCurrentConsign\_PDMv5  
pdmv5.h, [17](#)

MeasureDiodeAverageCurrent\_PDMv5  
pdmv5.h, [17](#)

MeasureDiodeCWCurrent\_PDMv5  
pdmv5.h, [18](#)

MeasureDiodePulsedCurrent\_PDMv5  
pdmv5.h, [18](#)

MeasureDiodeTemperature\_PDMv5  
pdmv5.h, [18](#)

MeasureDiodeVoltage\_PDMv5  
pdmv5.h, [19](#)

MeasureEXTInterlockStatus\_PDMv5  
pdmv5.h, [19](#)

MeasureInputVoltage\_PDMv5  
pdmv5.h, [20](#)

MeasureKeyStatus\_PDMv5  
pdmv5.h, [20](#)

MeasureLIVStatus\_PDMv5  
pdmv5.h, [21](#)

MeasureMOSTemperature\_PDMv5  
pdmv5.h, [21](#)

MeasurePatchBatteryVoltage\_PDMv5  
pdmv5.h, [22](#)

MeasurePatchPICOLASComplianceVoltage\_PDMv5  
pdmv5.h, [22](#)

MeasurePatchPICOLASCurrent\_PDMv5  
pdmv5.h, [22](#)

MeasurePatchTECCurrent\_PDMv5  
pdmv5.h, [23](#)

MeasurePatchTECExternalTemperature\_PDMv5  
pdmv5.h, [23](#)

MeasurePD\_EXTCurrent\_PDMv5  
pdmv5.h, [24](#)

MeasurePeakCurrentConsign\_PDMv5  
pdmv5.h, [24](#)

MeasureTECCurrent\_PDMv5  
pdmv5.h, [25](#)

MeasureTECExternalTemperature\_PDMv5  
pdmv5.h, [25](#)

MeasureTECVoltage\_PDMv5  
pdmv5.h, [25](#)

MeasureTECVoltageReference\_PDMv5  
pdmv5.h, [26](#)

MeasureTemperatureConsign\_PDMv5  
pdmv5.h, [26](#)

OpenCommunication\_PDMv5  
pdmv5.h, [27](#)

pdmv5.h, [3](#)

ApplyRequest\_PDMv5, [10](#)

CloseCommunication\_PDMv5, [10](#)

GetLIVMeasures\_PDMv5, [10](#)

GetLIVParameters\_PDMv5, [11](#)

Measure1V3PositiveVoltage\_PDMv5, [12](#)

Measure2V5PositiveVoltage\_PDMv5, [12](#)

Measure3V3PositiveVoltage\_PDMv5, [12](#)

Measure5VNegativeVoltage\_PDMv5, [13](#)

Measure5VPositiveVoltage\_PDMv5, [13](#)

MeasureAlarms\_PDMv5, [14](#)

MeasureBFMCurrent\_PDMv5, [15](#)

MeasureBNCInterlockStatus\_PDMv5, [15](#)

MeasureCentralDriverEnableStatus\_PDMv5, [15](#)

MeasureCentralMMDEnableStatus\_PDMv5, [16](#)

MeasureComplianceVoltage\_PDMv5, [16](#)

MeasureCWCCurrentConsign\_PDMv5, [17](#)

MeasureDiodeAverageCurrent\_PDMv5, 17  
 MeasureDiodeCWCCurrent\_PDMv5, 18  
 MeasureDiodePulsedCurrent\_PDMv5, 18  
 MeasureDiodeTemperature\_PDMv5, 18  
 MeasureDiodeVoltage\_PDMv5, 19  
 MeasureEXTInterlockStatus\_PDMv5, 19  
 MeasureInputVoltage\_PDMv5, 20  
 MeasureKeyStatus\_PDMv5, 20  
 MeasureLIVStatus\_PDMv5, 21  
 MeasureMOSTemperature\_PDMv5, 21  
 MeasurePatchBatteryVoltage\_PDMv5, 22  
 MeasurePatchPICOLASComplianceVoltage\_PDMv5, 22  
 MeasurePatchPICOLASCurrent\_PDMv5, 22  
 MeasurePatchTECCurrent\_PDMv5, 23  
 MeasurePatchTECEXternalTemperature\_PDMv5, 23  
 MeasurePD\_EXTCurrent\_PDMv5, 24  
 MeasurePeakCurrentConsign\_PDMv5, 24  
 MeasureTECCurrent\_PDMv5, 25  
 MeasureTECEXternalTemperature\_PDMv5, 25  
 MeasureTECVoltage\_PDMv5, 25  
 MeasureTECVoltageReference\_PDMv5, 26  
 MeasureTemperatureConsign\_PDMv5, 26  
 OpenCommunication\_PDMv5, 27  
 PDMv5\_DLL, 9  
 ReadAddress\_PDMv5, 27  
 ReadAPCHysteresisPercentage\_PDMv5, 28  
 ReadAPCPhotodiode\_PDMv5, 28  
 ReadAPCSamplingPeriod\_PDMv5, 28  
 ReadBFMGain\_PDMv5, 29  
 ReadComplianceVoltage\_PDMv5, 29  
 ReadCurrent\_PDMv5, 30  
 ReadCWCCurrent\_PDMv5, 30  
 ReadCWCCurrentSource\_PDMv5, 31  
 ReadCWLaserStatus\_PDMv5, 31  
 ReadCWMMaximumCurrent\_PDMv5, 32  
 ReadDelay\_PDMv5, 32  
 ReadDelayLine\_PDMv5, 33  
 ReadExternalMultiElementsBoardCommandVoltage\_PDMv5, 33  
 ReadFrequency\_PDMv5, 34  
 ReadHardwareType\_PDMv5, 34  
 ReadHardwareVersion\_PDMv5, 35  
 ReadLaserStatus\_PDMv5, 35  
 ReadLIVCurrentStep\_PDMv5, 35  
 ReadLIVMaximumCurrent\_PDMv5, 36  
 ReadLIVMeasuresCount\_PDMv5, 36  
 ReadLIVMinimumCurrent\_PDMv5, 37  
 ReadLIVPauseInterval\_PDMv5, 37  
 ReadLIVPhotodiode\_PDMv5, 38  
 ReadLIVPulseWidth\_PDMv5, 38  
 ReadMaximumAverageCurrent\_PDMv5, 39  
 ReadMaximumCurrent\_PDMv5, 39  
 ReadModulationCurrent\_PDMv5, 39  
 ReadModulationExternalCurrentGain\_PDMv5, 40  
 ReadModulationFrequency\_PDMv5, 40  
 ReadModulationInternalType\_PDMv5, 41  
 ReadModulationMaximumCurrent\_PDMv5, 41  
 ReadModulationStatus\_PDMv5, 42  
 ReadOperatingMode\_PDMv5, 42  
 ReadPD\_EXTGain\_PDMv5, 43  
 ReadPULSE\_INDetectionThreshold\_PDMv5, 43  
 ReadPulseCurrentSource\_PDMv5, 43  
 ReadPulseLaserStatus\_PDMv5, 44  
 ReadPulseWidth\_PDMv5, 44  
 ReadSerialNumber\_PDMv5, 45  
 ReadSoftwareType\_PDMv5, 45  
 ReadSoftwareVersion\_PDMv5, 46  
 ReadSynchro\_PDMv5, 46  
 ReadTECMaximumCurrent\_PDMv5, 47  
 ReadTECMaximumVoltage\_PDMv5, 47  
 ReadTemperature\_PDMv5, 47  
 Save\_PDMv5, 48  
 SetAddressSpecific\_PDMv5, 48  
 SetAPCHysteresisPercentage\_PDMv5, 49  
 SetAPCPhotodiode\_PDMv5, 49  
 SetAPCSamplingPeriod\_PDMv5, 50  
 SetBFMGain\_PDMv5, 50  
 SetComplianceVoltage\_PDMv5, 50  
 SetCurrent\_PDMv5, 51  
 SetCWCCurrent\_PDMv5, 51  
 SetCWCCurrentSource\_PDMv5, 52  
 SetCWLaserStatus\_PDMv5, 52  
 SetCWMMaximumCurrent\_PDMv5, 53  
 SetDelay\_PDMv5, 53  
 SetDelayLine\_PDMv5, 54  
 SetExternalMultiElementsBoardCommandVoltage\_PDMv5, 54  
 SetFrequency\_PDMv5, 55  
 SetLaserStatus\_PDMv5, 55  
 SetLIVCurrentStep\_PDMv5, 55  
 SetLIVMaximumCurrent\_PDMv5, 56  
 SetLIVMeasuresCount\_PDMv5, 56  
 SetLIVMinimumCurrent\_PDMv5, 57  
 SetLIVPauseInterval\_PDMv5, 57  
 SetLIVPhotodiode\_PDMv5, 58  
 SetLIVPulseWidth\_PDMv5, 58  
 SetMaximumAverageCurrent\_PDMv5, 59  
 SetMaximumCurrent\_PDMv5, 59  
 SetModulationCurrent\_PDMv5, 59  
 SetModulationExternalCurrentGain\_PDMv5, 60  
 SetModulationFrequency\_PDMv5, 60  
 SetModulationInternalType\_PDMv5, 61  
 SetModulationMaximumCurrent\_PDMv5, 61  
 SetModulationStatus\_PDMv5, 62  
 SetOperatingMode\_PDMv5, 62  
 SetPD\_EXTGain\_PDMv5, 63  
 SetPULSE\_INDetectionThreshold\_PDMv5, 63  
 SetPulseCurrentSource\_PDMv5, 63  
 SetPulseLaserStatus\_PDMv5, 64  
 SetPulseWidth\_PDMv5, 64  
 SetTECMaximumCurrent\_PDMv5, 65  
 SetTECMaximumVoltage\_PDMv5, 65  
 SetTemperature\_PDMv5, 66  
 StartCalibration\_PDMv5, 66

PDMv5\_DLL  
pdmv5.h, 9

ReadAddress\_PDMv5  
pdmv5.h, 27

ReadAPCHysteresisPercentage\_PDMv5  
pdmv5.h, 28

ReadAPCPhotodiode\_PDMv5  
pdmv5.h, 28

ReadAPCSamplingPeriod\_PDMv5  
pdmv5.h, 28

ReadBFMGain\_PDMv5  
pdmv5.h, 29

ReadComplianceVoltage\_PDMv5  
pdmv5.h, 29

ReadCurrent\_PDMv5  
pdmv5.h, 30

ReadCWCurrent\_PDMv5  
pdmv5.h, 30

ReadCWCurrentSource\_PDMv5  
pdmv5.h, 31

ReadCWLaserStatus\_PDMv5  
pdmv5.h, 31

ReadCWMMaximumCurrent\_PDMv5  
pdmv5.h, 32

ReadDelay\_PDMv5  
pdmv5.h, 32

ReadDelayLine\_PDMv5  
pdmv5.h, 33

ReadExternalMultiElementsBoardCommandVoltage\_PDMv5  
pdmv5.h, 33

ReadFrequency\_PDMv5  
pdmv5.h, 34

ReadHardwareType\_PDMv5  
pdmv5.h, 34

ReadHardwareVersion\_PDMv5  
pdmv5.h, 35

ReadLaserStatus\_PDMv5  
pdmv5.h, 35

ReadLIVCurrentStep\_PDMv5  
pdmv5.h, 35

ReadLIVMaximumCurrent\_PDMv5  
pdmv5.h, 36

ReadLIVMeasuresCount\_PDMv5  
pdmv5.h, 36

ReadLIVMinimumCurrent\_PDMv5  
pdmv5.h, 37

ReadLIVPauseInterval\_PDMv5  
pdmv5.h, 37

ReadLIVPhotodiode\_PDMv5  
pdmv5.h, 38

ReadLIVPulseWidth\_PDMv5  
pdmv5.h, 38

ReadMaximumAverageCurrent\_PDMv5  
pdmv5.h, 39

ReadMaximumCurrent\_PDMv5  
pdmv5.h, 39

ReadModulationCurrent\_PDMv5  
pdmv5.h, 39

ReadModulationExternalCurrentGain\_PDMv5  
pdmv5.h, 40

ReadModulationFrequency\_PDMv5  
pdmv5.h, 40

ReadModulationInternalType\_PDMv5  
pdmv5.h, 41

ReadModulationMaximumCurrent\_PDMv5  
pdmv5.h, 41

ReadModulationStatus\_PDMv5  
pdmv5.h, 42

ReadOperatingMode\_PDMv5  
pdmv5.h, 42

ReadPD\_EXTGain\_PDMv5  
pdmv5.h, 43

ReadPULSE\_INDetectionThreshold\_PDMv5  
pdmv5.h, 43

ReadPulseCurrentSource\_PDMv5  
pdmv5.h, 43

ReadPulseLaserStatusPDMv5  
pdmv5.h, 44

ReadPulseWidth\_PDMv5  
pdmv5.h, 44

ReadSerialNumber\_PDMv5  
pdmv5.h, 45

ReadSoftwareType\_PDMv5  
pdmv5.h, 45

ReadSoftwareVersion\_PDMv5  
pdmv5.h, 46

ReadSynchro\_PDMv5  
pdmv5.h, 46

ReadTECMaximumCurrent\_PDMv5  
pdmv5.h, 47

ReadTECMaximumVoltage\_PDMv5  
pdmv5.h, 47

ReadTemperature\_PDMv5  
pdmv5.h, 47

Save\_PDMv5  
pdmv5.h, 48

SetAddressSpecific\_PDMv5  
pdmv5.h, 48

SetAPCHysteresisPercentage\_PDMv5  
pdmv5.h, 49

SetAPCPhotodiode\_PDMv5  
pdmv5.h, 49

SetAPCSamplingPeriod\_PDMv5  
pdmv5.h, 50

SetBFMGain\_PDMv5  
pdmv5.h, 50

SetComplianceVoltage\_PDMv5  
pdmv5.h, 50

SetCurrent\_PDMv5  
pdmv5.h, 51

SetCWCurrent\_PDMv5  
pdmv5.h, 51

SetCWCurrentSource\_PDMv5  
pdmv5.h, 52

SetCWLaserStatus\_PDMv5  
pdmv5.h, 52

SetCWMaximumCurrent\_PDMv5  
pdmv5.h, [53](#)

SetDelay\_PDMv5  
pdmv5.h, [53](#)

SetDelayLine\_PDMv5  
pdmv5.h, [54](#)

SetExternalMultiElementsBoardCommandVoltage\_PDMv5  
pdmv5.h, [54](#)

SetFrequency\_PDMv5  
pdmv5.h, [55](#)

SetLaserStatus\_PDMv5  
pdmv5.h, [55](#)

SetLIVCurrentStep\_PDMv5  
pdmv5.h, [55](#)

SetLIVMaximumCurrent\_PDMv5  
pdmv5.h, [56](#)

SetLIVMeasuresCount\_PDMv5  
pdmv5.h, [56](#)

SetLIVMinimumCurrent\_PDMv5  
pdmv5.h, [57](#)

SetLIVPauseInterval\_PDMv5  
pdmv5.h, [57](#)

SetLIVPhotodiode\_PDMv5  
pdmv5.h, [58](#)

SetLIVPulseWidth\_PDMv5  
pdmv5.h, [58](#)

SetMaximumAverageCurrent\_PDMv5  
pdmv5.h, [59](#)

SetMaximumCurrent\_PDMv5  
pdmv5.h, [59](#)

SetModulationCurrent\_PDMv5  
pdmv5.h, [59](#)

SetModulationExternalCurrentGain\_PDMv5  
pdmv5.h, [60](#)

SetModulationFrequency\_PDMv5  
pdmv5.h, [60](#)

SetModulationInternalType\_PDMv5  
pdmv5.h, [61](#)

SetModulationMaximumCurrent\_PDMv5  
pdmv5.h, [61](#)

SetModulationStatus\_PDMv5  
pdmv5.h, [62](#)

SetOperatingMode\_PDMv5  
pdmv5.h, [62](#)

SetPD\_EXTGain\_PDMv5  
pdmv5.h, [63](#)

SetPULSE\_INDetectionThreshold\_PDMv5  
pdmv5.h, [63](#)

SetPulseCurrentSource\_PDMv5  
pdmv5.h, [63](#)

SetPulseLaserStatus\_PDMv5  
pdmv5.h, [64](#)

SetPulseWidth\_PDMv5  
pdmv5.h, [64](#)

SetTECMaximumCurrent\_PDMv5  
pdmv5.h, [65](#)

SetTECMaximumVoltage\_PDMv5  
pdmv5.h, [65](#)

SetTemperature\_PDMv5  
pdmv5.h, [66](#)

StartCalibration\_PDMv5  
pdmv5.h, [66](#)