

# LUMILOOP



User's Manual

———— LSPM 1.0 ————

**9 kHz - 6(12) GHz**

**Triple High-Speed Power Meter**

All trade names are the registered namework of their respective owners. Specifications are subject to change without notice.

© Copyright 2020 LUMILOOP® GmbH. All rights reserved. No part of this document may be copied without written permission from LUMILOOP GmbH.

## Contents

1	System Overview . . . . .	9
1.1	Data Acquisition and Processing . . . . .	10
1.2	Power Waveforms . . . . .	10
1.3	Sweep Analysis . . . . .	11
1.4	Statistical Analysis . . . . .	11
1.4.1	Continuous Statistics . . . . .	11
1.4.2	Triggered Statistics . . . . .	13
1.5	Stream Recording . . . . .	13
2	LSPM Hardware . . . . .	14
2.1	Principle of Operation . . . . .	14
2.1.1	Video Bandwidth Considerations . . . . .	15
2.1.2	Sampling Rate Reduction . . . . .	15
2.2	Components, Connectors and Indicators . . . . .	16
2.3	Systems with Multiple Power Meters and Field Probes . . . . .	18
2.4	Trigger Inputs and Outputs . . . . .	19
3	LSPM Software . . . . .	20
3.1	LSPM 1.0 TCP Server and GUI Installation . . . . .	20
3.2	USB Driver Installation . . . . .	21
3.2.1	Troubleshooting USB Driver Installation . . . . .	23
3.3	LabView Run Time Environment Installation . . . . .	23
4	Measuring Power . . . . .	24
4.1	Getting Ready to Measure . . . . .	24
4.1.1	Making Electrical Connections . . . . .	24
4.2	Power Meter Start-Up and Mode Selection . . . . .	24
4.2.1	Starting the LSPM 1.0 TCP Server . . . . .	24
4.2.2	Interacting with the LSPM 1.0 TCP Server . . . . .	26
4.2.3	General Notes on the LSPM 1.0 GUI . . . . .	26
4.2.4	Mode Selection Using the GUI . . . . .	27
4.2.5	Mode Selection Using SCPI Commands . . . . .	28
4.3	Continuous Power Measurements . . . . .	28
4.3.1	Continuous Measurements Using the GUI . . . . .	28
4.3.2	Continuous Measurements Using SCPI Commands . . . . .	30
4.4	Triggered Power Measurements . . . . .	30
4.4.1	Power Waveform Acquisition Using the GUI . . . . .	30
4.4.2	Power Waveform Acquisition Using SCPI Commands . . . . .	31
4.4.3	Pulse Measurements Using the GUI's Radar Tab . . . . .	33
4.4.4	Pulse Measurements Using SCPI Commands . . . . .	36
4.4.5	Sweep Measurements Using the GUI . . . . .	37
4.4.6	Sweep Measurements Using SCPI Commands . . . . .	39

4.5	Power Statistics . . . . .	40
4.5.1	Continuous Statistics using the GUI . . . . .	40
4.5.2	Continuous Statistics using SCPI Commands . . . . .	42
4.5.3	Triggered Statistics using the GUI . . . . .	43
4.5.4	Triggered Statistics using SCPI Commands . . . . .	43
4.6	Stream Recording . . . . .	43
4.6.1	Stream Recording Using the GUI . . . . .	44
4.6.2	Stream Recording Using SCPI Commands . . . . .	45
4.7	Saving Log Files using the GUI . . . . .	45
5	Third Party EMC Software . . . . .	46
5.1	EMC32 . . . . .	47
5.1.1	CW fields . . . . .	47
5.2	BAT-EMC . . . . .	50
5.2.1	CW Power Measurements . . . . .	51
5.2.2	Pulsed Power Measurements . . . . .	52
6	Virtual Power Meters . . . . .	53
6.1	Controlling Virtual Power Meters Using the GUI . . . . .	54
6.2	Controlling Virtual Power Meters Using SCPI Commands . . . . .	55
7	Power Meter Calibration . . . . .	55
7.1	In-house Calibration . . . . .	56
7.2	External Calibration . . . . .	56
7.3	Automated Calibration Data Import . . . . .	57
8	SCPI Communication Basics . . . . .	59
8.1	National Instruments VISA . . . . .	59
8.2	Raw TCP socket communication using PuTTY . . . . .	62
9	SCPI Command Reference . . . . .	62
9.1	Multi-Power Meter Behavior . . . . .	62
9.2	Generic Commands . . . . .	63
9.2.1	*CLS . . . . .	63
9.2.2	*ESE <ESR> . . . . .	63
9.2.3	*ESE? . . . . .	63
9.2.4	*ESR? . . . . .	63
9.2.5	*IDN? . . . . .	64
9.2.6	*OPC . . . . .	64
9.2.7	*OPC? . . . . .	64
9.2.8	*RST . . . . .	64
9.2.9	*SRE <int> . . . . .	64
9.2.10	*SRE? . . . . .	64
9.2.11	*STB? . . . . .	65
9.2.12	*TST? . . . . .	65
9.2.13	*WAI . . . . .	65

9.3	:SYSTem Commands	65
9.3.1	:SYSTem:WAIT <Sec>	65
9.3.2	:SYSTem:ERRor[:NEXT]?	66
9.3.3	:SYSTem:ERRor:COUNT?	66
9.3.4	:SYSTem:AUTOCONnect <State>	66
9.3.5	:SYSTem:AUTOCONnect?	66
9.3.6	:SYSTem:SERial <Value>	66
9.3.7	:SYSTem:SERial? [<MPMeter>]	67
9.3.8	:SYSTem:MAKer? [<MPMeter>]	67
9.3.9	:SYSTem:DEVice? [<MPMeter>]	67
9.3.10	:SYSTem:VERsion? [<MPMeter>]	67
9.3.11	:SYSTem:REVision? [<MPMeter>]	68
9.3.12	:SYSTem:FWUPdate?	68
9.3.13	:SYSTem:DEBUg <Value/Flag1[,Flag2]...>	68
9.3.14	:SYSTem:DEBUg?	69
9.3.15	:SYSTem:DFLags?	70
9.3.16	:SYSTem:MODE <Mode>[,<MPMeter>]	70
9.3.17	:SYSTem:MODE? [<MPMeter>]	70
9.3.18	:SYSTem:FREQUency <Frequency>[,<MPMeter>]	70
9.3.19	:SYSTem:FREQUency? [<MPMeter>]	71
9.3.20	:SYSTem:FREQUency:MINimum? [<MPMeter>]	71
9.3.21	:SYSTem:FREQUency:MAXimum? [<MPMeter>]	71
9.3.22	:SYSTem:SSKip? [<MPMeter>]	71
9.3.23	:SYSTem:SRAtE? [<MPMeter>]	72
9.3.24	:SYSTem:TImE?	72
9.4	:CALibration Commands	72
9.4.1	:CALibration:LOGging <Value>	72
9.4.2	:CALibration:LOGging?	72
9.4.3	:CALibration:EXTernal <Value>[,<MPMeter>]	73
9.4.4	:CALibration:EXTernal? [<MPMeter>]	73
9.4.5	:CALibration:CERTificate? [<MPMeter>]	73
9.4.6	:CALibration:TStamp? [<MPMeter>]	73
9.5	:MEASure Commands	74
9.5.1	:MEASure:TCold? [<MPMeter>]	74
9.5.2	:MEASure:VPeltier? [<MPMeter>]	74
9.5.3	:MEASure:IPeltier? [<MPMeter>]short-cut	74
9.5.4	:MEASure:P[1]/P2/P3/ALL? [<MPMeter>]	74
9.5.5	:MEASure:MIN P[1]/P2/P3/ALL? [<MPMeter>]	75
9.5.6	:MEASure:MAX P[1]/P2/P3/ALL? [<MPMeter>]	75
9.5.7	:MEASure:LPFrequency <Frequency>[,<MPMeter>]	75
9.5.8	:MEASure:LPFrequency? [<MPMeter>]	76
9.5.9	:MEASure:AUTOVBW <State>[,<MPMeter>]	76

9.5.10	:MEASure:AUTOVBW? [<MPMeter>] . . . . .	76
9.5.11	:MEASure:VBW <Frequency>[,<MPMeter>] . . . . .	77
9.5.12	:MEASure:VBW? [<MProbe>] . . . . .	77
9.5.13	:MEASure:RSsi:P[1]/P2/P3/ALL? [<MPMeter>] . . . . .	77
9.6	:TRIGger Commands . . . . .	78
9.6.1	:TRIGger:BEgin <Index>[,<MPMeter>] . . . . .	78
9.6.2	:TRIGger:BEgin? [<MPMeter>] . . . . .	78
9.6.3	:TRIGger:LEngh <Length>[,<MPMeter>] . . . . .	78
9.6.4	:TRIGger:LEngh? [<MPMeter>] . . . . .	78
9.6.5	:TRIGger:FLengh? [<MPMeter>] . . . . .	79
9.6.6	:TRIGger:POInt <Points>[,<MPMeter>] . . . . .	79
9.6.7	:TRIGger:POInt? [<MPMeter>] . . . . .	79
9.6.8	:TRIGger:PROgress? [<MPMeter>] . . . . .	80
9.6.9	:TRIGger:PTProgress? [<MPMeter>] . . . . .	80
9.6.10	:TRIGger:PTTimes? [<MPMeter>] . . . . .	80
9.6.11	:TRIGger:EVCNT? <Samples>[,<MPMeter>] . . . . .	81
9.6.12	:TRIGger:STATe? [<Timeout>,<MPMeter>] . . . . .	81
9.6.13	:TRIGger:ARM [<MPMeter>] . . . . .	81
9.6.14	:TRIGger:ARMed? [<Timeout>,<MPMeter>] . . . . .	82
9.6.15	:TRIGger:CLear [<MPMeter>] . . . . .	82
9.6.16	:TRIGger:FORce [<MPMeter>] . . . . .	82
9.6.17	:TRIGger:DONE? [<Timeout>,<MPMeter>] . . . . .	82
9.6.18	:TRIGger:COUnt? [<MPMeter>] . . . . .	83
9.6.19	:TRIGger:SOURce <Source>[,<MPMeter>] . . . . .	83
9.6.20	:TRIGger:SOURce? [<MPMeter>] . . . . .	83
9.6.21	:TRIGger:LEVel <Level>[,<MPMeter>] . . . . .	84
9.6.22	:TRIGger:LEVel? [<MPMeter>] . . . . .	84
9.6.23	:TRIGger:FALLing <0/1>[,<MPMeter>] . . . . .	84
9.6.24	:TRIGger:FALLing? [<MPMeter>] . . . . .	84
9.6.25	:TRIGger:OUTput <0/1>[,<MPMeter>] . . . . .	85
9.6.26	:TRIGger:OUTput? [<MPMeter>] . . . . .	85
9.6.27	:TRIGger:INVert <0/1>[,<MPMeter>] . . . . .	85
9.6.28	:TRIGger:INVert? [<MPMeter>] . . . . .	85
9.6.29	:TRIGger:SYNC <0/1>[,<MPMeter>] . . . . .	86
9.6.30	:TRIGger:SYNC? [<MPMeter>] . . . . .	86
9.6.31	:TRIGger:BPOUTput <0/1>[,<MPMeter>] . . . . .	86
9.6.32	:TRIGger:BPOUTput? [<MPMeter>] . . . . .	86
9.6.33	:TRIGger:BPINVert <0/1>[,<MPMeter>] . . . . .	86
9.6.34	:TRIGger:BPINVert? [<MPMeter>] . . . . .	87
9.6.35	:TRIGger:BPSYNC <0/1>[,<MPMeter>] . . . . .	87
9.6.36	:TRIGger:BPSYNC? [<MPMeter>] . . . . .	87

9.7	:TRIGger[:WAVEform] Commands	87
9.7.1	:TRIGger[:WAVEform][:Power]:P[1]/:P2/:P3? [<MPMeter>]	87
9.7.2	:TRIGger[:WAVEform][:Power]:ALL? [<MPMeter>]	87
9.7.3	:TRIGger[:WAVEform]:RSsi:P[1]/:P?:/P3? [<MPMeter>]	88
9.7.4	:TRIGger[:WAVEform][:Power]:BINary? [<MPMeter>]	88
9.7.5	:TRIGger[:WAVEform][:Power]:BINWait? [<Timeout>,<MPMeter>]	89
9.8	[:TRIGger]:RADar, Commands	89
9.8.1	[:TRIGger]:RADar:TRIM <State>[,<MProbe>]	89
9.8.2	[:TRIGger]:RADar:TRIM? [<MProbe>]	90
9.8.3	[:TRIGger]:RADar:MINTime <MinT>[<MPMeter>]	90
9.8.4	[:TRIGger]:RADar:MINTime? [<MPMeter>]	90
9.8.5	[:TRIGger]:RADar:MINSamples <MinS>[<MPMeter>]	90
9.8.6	[:TRIGger]:RADar:MINSamples? [<MPMeter>]	90
9.8.7	[:TRIGger]:RADar:THMethod <Method>[<MPMeter>]	91
9.8.8	[:TRIGger]:RADar:THMethod? [<MPMeter>]	91
9.8.9	[:TRIGger]:RADar:ATHold <Threshold>[<MPMeter>]	92
9.8.10	[:TRIGger]:RADar:ATHold? [<MPMeter>]	92
9.8.11	[:TRIGger]:RADar:RTHold <Threshold>[<MPMeter>]	92
9.8.12	[:TRIGger]:RADar:RTHold? [<MPMeter>]	92
9.8.13	[:TRIGger]:RADar:CLEARance <Clearance>[<MPMeter>]	93
9.8.14	[:TRIGger]:RADar:CLEARance? [<MPMeter>]	93
9.8.15	[:TRIGger]:RADar:THold:P[1]/P2/P3/ALL? [<MPMeter>]	93
9.8.16	[:TRIGger]:RADar:MAVG <Count>[<MPMeter>]	93
9.8.17	[:TRIGger]:RADar:MAVG? [<MPMeter>]	94
9.8.18	[:TRIGger]:RADar:P[1]/P2/P3? [<MPMeter>]	94
9.8.19	[:TRIGger]:RADar:APOWer:P[1]/:P2/:P3/:ALL? [<MPMeter>]	94
9.8.20	[:TRIGger]:RADar:MPOWer:P[1]/:P2/:P3/:ALL? [<MPMeter>]	94
9.8.21	[:TRIGger]:RADar:COUnt:P[1]/:P2/:P3/:ALL? [<MPMeter>]	95
9.8.22	[:TRIGger]:RADar:PULses:STArt:P[1]/:P2/:P3? [<MPMeter>]	95
9.8.23	[:TRIGger]:RADar:PULses:LENGth:P[1]/:P2/:P3? [<MPMeter>]	95
9.8.24	[:TRIGger]:RADar:PULses[:APOWer]:P[1]/:P2/:P3? [<MPMeter>]	96
9.8.25	[:TRIGger]:RADar:DUTY:P[1]/:P2/:P3/:ALL? [<MPMeter>]	96
9.8.26	[:TRIGger]:RADar:WPowEr:P[1]/:P2/:P3/:ALL? [<MPMeter>]	96
9.8.27	[:TRIGger]:RADar:BINary? <Wave>[,<MPMeter>]	96
9.9	[:TRIGger]:SWEEP, Commands	99
9.9.1	[:TRIGger]:SWEEP:TStep <TStep>[,<MPMeter>]	99
9.9.2	[:TRIGger]:SWEEP:TStep? [<MPMeter>]	100
9.9.3	[:TRIGger]:SWEEP:TCNT? [<MPMeter>]	100
9.9.4	[:TRIGger]:SWEEP:TBegin <TBegin>[,<MPMeter>]	100
9.9.5	[:TRIGger]:SWEEP:TBegin? [<MPMeter>]	101
9.9.6	[:TRIGger]:SWEEP:TEnd <TEnd>[,<MPMeter>]	101
9.9.7	[:TRIGger]:SWEEP:TEnd? [<MPMeter>]	101

9.9.8	[:TRIGger]:SWeep:MODe <Mode>[,<MPMeter>]	101
9.9.9	[:TRIGger]:SWeep:MODe? [<MPMeter>]	102
9.9.10	[:TRIGger]:SWeep:BEgin <Freq>[,<MPMeter>]	102
9.9.11	[:TRIGger]:SWeep:BEgin? [<MPMeter>]	102
9.9.12	[:TRIGger]:SWeep:COUnt <Count>[,<MPMeter>]	103
9.9.13	[:TRIGger]:SWeep:COUnt? [<MPMeter>]	103
9.9.14	[:TRIGger]:SWeep:STEP <Step>[,<MPMeter>]	103
9.9.15	[:TRIGger]:SWeep:STEP? [<MPMeter>]	104
9.9.16	[:TRIGger]:SWeep:ARBAdd <Freq>[,<MPMeter>]	104
9.9.17	[:TRIGger]:SWeep:ARBclear [<MPMeter>]	104
9.9.18	[:TRIGger]:SWeep:ARbitrary? [<MPMeter>]	104
9.9.19	[:TRIGger]:SWeep:LIST? [<MPMeter>]	105
9.9.20	[:TRIGger]:SWeep:IDX? [<MPMeter>]	105
9.9.21	[:TRIGger]:SWeep[:P[1]/:P2/:P3/[:ALL]ower]:P? [<MPMeter>]	105
9.9.22	[:TRIGger]:SWeep:RSsi:P[1]/:P2/:P3/[:ALL]? [<MPMeter>]	106
9.9.23	[:TRIGger]:SWeep:WPower:P[1]/:P2/:P3/[:ALL]? [<MPMeter>]	106
9.9.24	[:TRIGger]:SWeep:BINary?	107
9.10	:STATistics Commands	109
9.10.1	:STATistics:MAster <State>	109
9.10.2	:STATistics:MAster? [<MPMeter>]	109
9.10.3	:STATistics:ENable <State>[,<MPMeter>]	109
9.10.4	:STATistics:ENable? [<MPMeter>]	110
9.10.5	:STATistics:SNAPshot [<Triggered>][,<MPMeter>]	110
9.10.6	:STATistics:COUnt? [<MPMeter>]	110
9.10.7	:STATistics:RESolution <Resolution>[,<MPMeter>]	111
9.10.8	:STATistics:RESolution? [<MPMeter>]	111
9.10.9	:STATistics:HISTogram:SIze? [<Triggered>][,<MPMeter>]	111
9.10.10	:STATistics:HISTogram:OFFset? [<Triggered>][,<MPMeter>]	112
9.10.11	:STATistics:SAMples? [<Triggered>][,<MPMeter>]	112
9.10.12	:STATistics:MINimum:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]	112
9.10.13	:STATistics:MAXimum:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]	113
9.10.14	:STATistics:MEAN:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]	113
9.10.15	:STATistics:RMS:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]	113
9.10.16	:STATistics:SDEVIation:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]	113
9.10.17	:STATistics:Power? [<Triggered>][,<MPMeter>]	114
9.10.18	:STATistics:HISTogram:P[1]/:P2/:P3? [<Triggered>][,<MPMeter>]	114
9.10.19	:STATistics:P[1]/:P2/:P3DF:P? [<Triggered>][,<MPMeter>]	114
9.10.20	:STATistics:CDF:P[1]/:P2/:P3? [<Triggered>][,<MPMeter>]	115
9.10.21	:STATistics:CCDF:P[1]/:P2/:P3? [<Triggered>][,<MPMeter>]	115
9.10.22	:STATistics:BINary? [<Triggered>][,<MPMeter>]	115
9.11	:MProbe Commands	117
9.11.1	:MPMeter:SERial <MPMeter>,<SN1>[,<SN2>[,<SN3>[,<SN4>[,<SN5>[,<SN6>[,<SN7>[,<SN8>[,<SN9>[,<SN10>]]]]]]]]]	117



9.11.2	:MPMeter:SERial? <MPMeter>	118
9.12	:VIRTual Power Meter Commands	118
9.12.1	:VIRTual:SERial?	118
9.12.2	:VIRTual:CONnect [<SER>]	118
9.12.3	:VIRTual:DISConnect	119
9.12.4	:VIRTual:CW <RSSI1>,<RSSI2>,<RSSI3>	119
9.12.5	:VIRTual:CW?	119
9.12.6	:VIRTual:NOIse <NOISE1>,<NOISE2>,<NOISE3>	119
9.12.7	:VIRTual:NOIse?	119
9.12.8	:VIRTual:PULse [<RSSI1>],[<RSSI2>],[<RSSI3>],[<T>],[<Ton>]	120
9.12.9	:VIRTual:PULse?	120
9.12.10	:VIRTual:PLIST <P1_1>,<P2_1>,<P3_1>[,...,<P1_N>,<P2_N>,<P3_N>]	120
9.12.11	:VIRTual:PLIST?	121
9.12.12	:VIRTual:LIST <RSSI1_1>,<RSSI2_1>,<RSSI3_1>[,...,<RSSI2_N>,<RSSI3_N>]	121
9.12.13	:VIRTual:LIST?	121
9.12.14	:VIRTual:LCnt?	121
9.12.15	:VIRTual:LCLear	121
9.13	:STReam Recording Commands	121
9.13.1	:STReam:MAster <State>	121
9.13.2	:STReam:MAster? [<MPMeter>]	122
9.13.3	:STReam:LENgth <Length>[,<MPMeter>]	122
9.13.4	:STReam:LENgth? [<MPMeter>]	122
9.13.5	:STReam:ENable <State>[,<MPMeter>]	123
9.13.6	:STReam:ENable? [<MPMeter>]	123
9.13.7	:STReam:PROgress? [<MPMeter>]	123
9.13.8	:STReam:SKIp <SkipCnt>[,<MPMeter>]	123
9.13.9	:STReam:SKIp? [<MPMeter>]	124
9.13.10	:STReam:PREfix <String>[,<MPMeter>]	124
9.13.11	:STReam:PREfix? [<MPMeter>]	124
9.13.12	:STReam:SYNC <Sync>[,<MPMeter>]	124
9.13.13	:STReam:SYNC? [<MPMeter>]	125
10	File Formats	125
10.1	LSPM 1.0 GUI Log Files	125
10.1.1	Basic Data Logger	126
10.1.2	Power Scope Data Logger	127
10.1.3	Radar Data Logger	127
10.1.4	Sweep Data Logger	129
10.1.5	Statistics Data Logger	131
10.1.6	Stream Files in Binary Format	132
10.1.7	Stream Files in CSV Format	134
10.2	extCalLog TCP-Server Logger	135
10.3	Generic Calibration Result Files	136

---

10.4 Calibration Files . . . . .	137
10.4.1 In-House Linearity and Frequency Compensation Files . . . . .	137
10.4.2 External Power Calibration Files . . . . .	138
11 Specifications . . . . .	140
11.0.1 Typical Dynamic Range . . . . .	141
12 Warranty Conditions . . . . .	142
13 EC Declaration of Conformity . . . . .	143
14 Revision History . . . . .	145

# 1 System Overview

The Triple High-Speed Power Meter LSPM 1.0 enables continuous RF power measurements with high resolution, high speed and low noise. LSPM 1.0 is not limited to the acquisition of quasi-static power levels. With its sampling rate of 2 MSamples/s, LSPM 1.0 Power Meters are able to measure rapidly changing RF signals. The power meter's large dynamic range and fast pulse response make it ideally suited for the analysis of pulsed signals and sweep measurements.

Every LSPM 1.0 Power Meter comes with a complete set of in-house calibration data for high linearity and fine-grained frequency compensation. Temperature stability is guaranteed by actively controlling the power sensors' temperature. The high-accuracy in-house calibration data can be further enhanced by calibration at accredited laboratories.

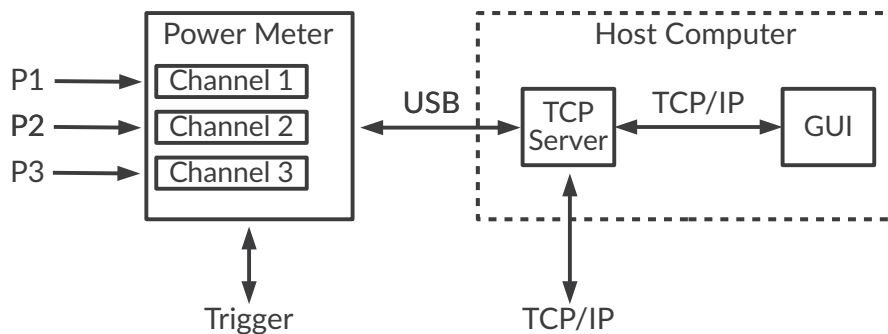


Figure 1: LSPM system block diagram

As shown in Figure 1, LSPM 1.0 features up to three channels which record power values using logarithmic power detectors and dedicated analog-to-digital converters. The digital sample values are transmitted to a host computer via USB. External trigger signals are handled by the LSPM's hardware. The LSPM 1.0 TCP Server handles USB communication, application of calibration data, data post-processing and communication with client programs via TCP/IP SCPI commands, e.g., the LSPM 1.0 Graphical User Interface (GUI).

Moreover, the TCP server receives SCPI commands via its standard input and replies via its standard output. This enables simple manual configuration and verification of the LSPM's operation by typing or pasting commands into a command window. This feature can also be used to connect the TCP server to third party software using the standard input and output of the TCP server.

Multiple LSPM 1.0 Power Meters may be connected to one TCP server instance to form a LSPM 1.0 Multi-Power Meter System consisting of synchronized power meters.

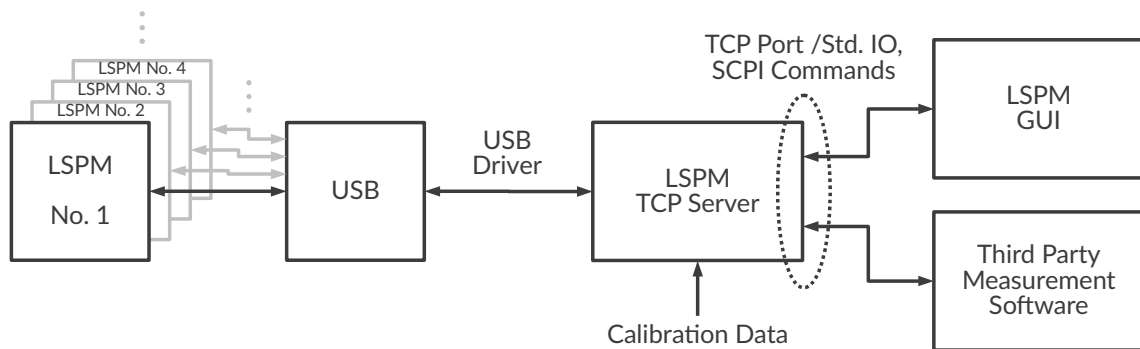


Figure 2: LSPM software block diagram

## 1.1 Data Acquisition and Processing

The software delivered with the LUMILOOP LSPM 1.0 Power Meter consists of:

- the LSPM 1.0 USB driver,
- the LSPM 1.0 TCP Server,
- the LSPM 1.0 Graphical User Interface (GUI),
- the CallImport tool,
- programming examples and
- third-party EMC software drivers.

As shown in Figure 2, one instance of the LSPM 1.0 TCP Server communicates with all LSPM 1.0 Power Meters connected to the host computer via USB 2.0. The TCP server configures the power meter, streams all power values and auxiliary data, applies calibration data to the received power values, handles trigger events, performs sensor data buffering and post-processing. The TCP server provides a comprehensive set of SCPI commands for simple and reliable text-based integration into test and measurement automation solutions, see Section 9. Concurrent access to all power meters is offered to up to 32 TCP clients.

All third party EMC software accesses the LSPM 1.0 TCP Server through SCPI commands, examples are given in Section 4.3.2, 4.4.2, etc. When operating multiple TCP client programs in parallel, the user is responsible for avoiding undesired interference. For example, one client must not change the measurement mode while another client relies on a different setting. Generally, concurrent access to the LSPM 1.0 TCP Server is discouraged except for debugging purposes.

The LSPM 1.0 GUI detailed in Section 4 is a graphical user interface for configuring and monitoring all device settings as well as measuring and logging data. As any client, the GUI connects to a local or remote TCP server.

## 1.2 Power Waveforms

In addition to the continuous measurement of RF power and output of the instantaneous power value with the optional application of a low-pass filter, the LSPM TCP-Server is able to record a

user specified length of power values starting from a specific point in time, i.e., power waveforms. Subsequent signal analysis, such as radar pulse detection, evaluation of frequency/power sweeps and statistical analysis, are based on previously recorded power waveforms.

Power waveforms are recorded upon receiving a trigger event. They can be queried until being replaced by a newly recorded power waveform. The waveform is recorded relative to a trigger event. Trigger events can be generated by software, by crossing a specified power threshold or by externally generated trigger signals. See Section 4.4 on page 30 for more details.

Note that power waveforms require memory in proportion to the recorded duration of time, i.e., approximately 12 MB per second. For a single power meter this limits the maximum waveform length to approximately 2.5 minutes on a 32 bit system. On 64 bit systems, the maximum waveform length is only limited by the amount of available memory.

### 1.3 Sweep Analysis

The LSPM 1.0 TCP Server is able to analyze frequency and power sweeps recorded in power waveforms. Information about the sweep's timing and frequency steps must be provided as sweep parameters. Each step of the sweep is evaluated individually, taking into account its frequency and settling times. For each part of the waveform that corresponds to a specific generator step the calibration data is newly computed and applied in agreement with the associated frequency for the respective generator step. The LSPM 1.0 TCP Server returns the averaged power value for each step. Typically, the signal generator and LSPM 1.0 Power Meter are synchronized using a hardware trigger line when acquiring power waveforms for sweep analysis for precise timing control. See Section 4.4.5 on page 37 for more details.

### 1.4 Statistical Analysis

The LSPM 1.0 TCP Server is able to perform a statistical evaluation of power data originating from one or more LSPM 1.0 Power Meters, thus reducing the programming effort, communication overhead, memory requirements and CPU load. Both scalar statistics values and histogram-like distributions are accessible through the LSPM 1.0 TCP Server. Figure 3 shows a simplified data flow diagram for both continuous and triggered statistics. See Section 4.5 on page 40 for more details.

#### 1.4.1 Continuous Statistics

Continuous statistics are based on three histograms generated from all incoming power values. There is one histogram for each power input. Histograms are created at a resolution of 0.005 dB. Data collection is enabled and disabled using dedicated signal lines described in Section 2.3. Continuous statistics consume a minimal amount of memory since timing information is discarded in the process of creating histogram data. Consequently, continuous statistics can be recorded for arbitrary durations of time. Scalar statistics values and histogram-like distributions use statistics

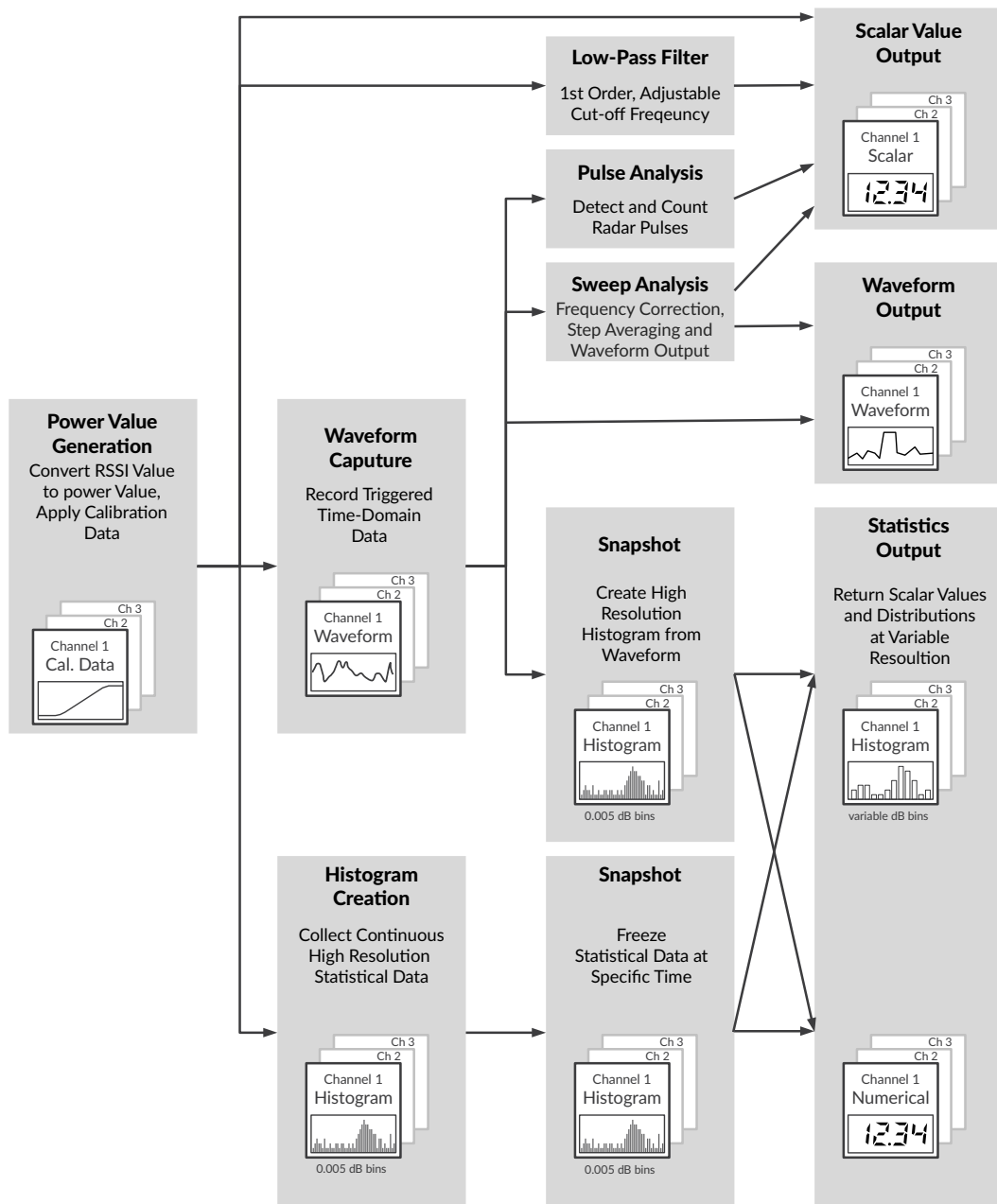


Figure 3: Data flow diagram of power value processing

snapshots, i.e., copies of the continuously updated histogram, created at specific times. Snapshot creation for multiple LSPM 1.0 Power Meters is synchronized using dedicated signal lines detailed in Section 2.3. Hardware-based snapshot creation ensures that statistics of multiple power meters will not be distorted by latencies introduced by buffering, the USB hardware or the operating system's data processing. Histogram-like distributions and associated power values can be queried at a lower level of detail than the 0.005 dB default of the snapshot histograms. See Section 4.5.2 and 4.5.4 for details.

### 1.4.2 Triggered Statistics

Triggered statistics create snapshot histograms from triggered waveforms. Consequently, a set of power waveforms must be recorded in full before statistics evaluation can take place. When compared to continuous statistics, triggered statistics have the advantage of preserving timing information in the form of the triggered waveforms. However, triggered statistics also have a number of disadvantages relative to continuous statistics:

- Triggered statistics require memory in proportion to the length of time to be evaluated and the number of power meters in a system, see Section 1.2 for more details.
- For the same duration of time creating a snapshot histogram from triggered waveforms is significantly slower than a continuous statistics snapshot. This is due to the fact that triggered statistics must evaluate all samples in the recorded waveforms while continuous statistics merely require a copy of the continuously updated histogram for every snapshot.
- Triggered statistics are only available for the recorded waveforms as a whole, continuous statistics may take statistics snapshots as data is being recorded.
- Triggered statistics may introduce significant delays in TCP server to client communication, especially when recording large waveforms.

It is therefore generally preferable to rely on continuous statistics when timing information is not essential.

## 1.5 Stream Recording

The LSPM 1.0 TCP Server is able to save a stream of continuously recorded power values directly to disk. This can be done for one or multiple LSPM 1.0 Power Meters. LSPM 1.0 trigger signals can be employed to synchronize multiple streams. In this case one LSProbe 1.2 Field Probe or LSPM 1.0 Power Meter acts as the stream master, sending one synchronization pulse for every 500  $\mu$ s. All other power meters act as stream slaves and use the master's synchronization pulses to ensure the sampling rate dictated by the master. See Figures 11, 12 and 13 for the required physical connections to the BNC connector and Ext1 RJ45 socket. See Section 4.6 on page 43 for more details.

## 2 LSPM Hardware

### 2.1 Principle of Operation

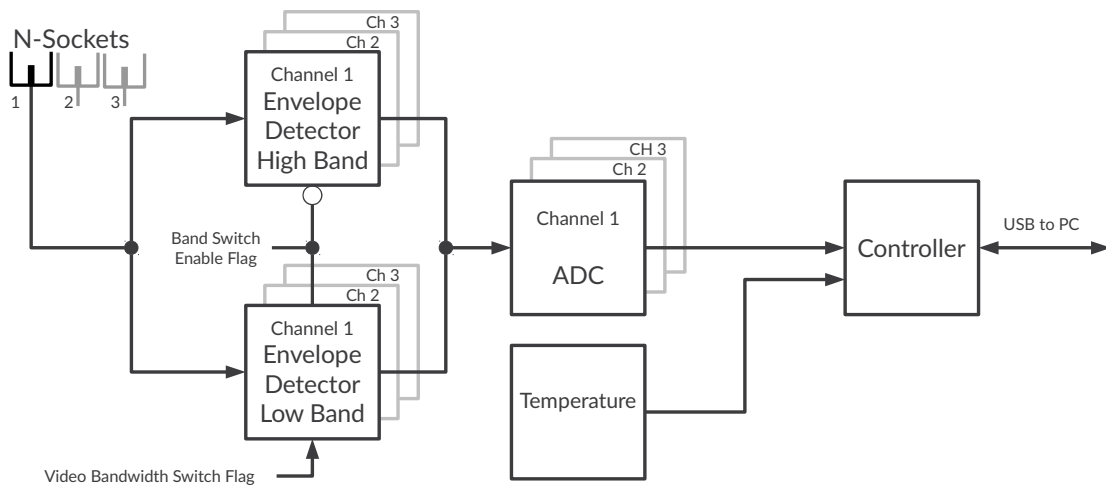


Figure 4: Power meter block diagram

Figure 4 shows the simplified block diagram of the power meter. Three 50 Ω N sockets feed the power signals to dedicated detectors. Each channel has got one dedicated low-band and one dedicated high-band logarithmic power detector, of which one is selectable at a time using a mode flag (see Table 1). The low-band stretches from 9 kHz to 400 MHz, the high-band from 30 MHz to 6(12<sup>2</sup>) GHz.

The video bandwidth of LSPM 1.0 Power Meters is defined as the -3 dB cut-off frequency of the first order low pass filter operating on the envelope of each logarithmic power detector. For accurate operation the detected frequency should be at least ten times larger than the video bandwidth. The low-band detector allows to choose between low and high video bandwidth, i.e., approximately 500 Hz and 1 MHz selectable using a mode flag (see Table 1). The high-band detector has a fixed video bandwidth of approximately 3 MHz.

For each axis a 14 bit analog-to-digital converter (ADC) is used to digitize the detected signal level.

The LSPM 1.0 Power Meter offers four operating modes. Each mode is characterized by a specific frequency range, video bandwidth, sampling rate and sample timing. An overview is given in in Table 1.

In mode 0 only the high band envelope detector is active. In mode 2 and 3 only the low band envelope detector is active. In mode 1 the appropriate detector is selected according to the set frequency. Mode 1 spans the power meter's entire frequency range but requires approximately 1 ms of additional settling time when crossing 30 MHz.

When using SCPI Commands or third party EMC software special care must be taken to ensure an appropriate mode setting and low-pass filtering for the measurement task at hand.

<sup>2</sup>Operation up to 12 GHz with reduced performance



Table 1: LSPM measurement modes overview

Mode	Minimum Frequency	Maximum Frequency	Video Bandwidth	Sampling Rate
0	30 MHz	$6(12^2)$ GHz	3 MHz	2 MS/s
1	9 kHz	29.9 MHz	500 Hz	2 MS/s
	30 MHz	$6(12^2)$ GHz	3 MHz	
2	9 kHz	400 MHz	1 MHz	2 MS/s
3	9 kHz	400 MHz	500 Hz	2 MS/s

### 2.1.1 Video Bandwidth Considerations

As highlighted in Figure 5, most applications are covered by operating modes 0, 1, and 3. For these modes the detectors' video bandwidths are sufficient. However, since mode 2 has a video bandwidth of 1 MHz and will only yield sufficiently ripple-free results for frequencies larger than 10 MHz, lower frequencies cannot be measured reliably without further measures.

For this reason, the LSPM 1.0 TCP Server supports software-based video bandwidth reduction, which will be applied automatically when needed. Figure 5 summarizes the supported operating modes. The excluded frequency range is 800 kHz to 10 MHz for mode 2.

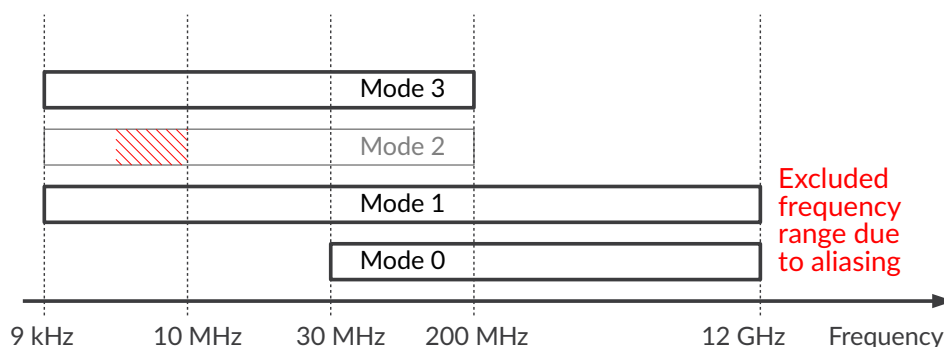


Figure 5: Operating modes versus frequency

### 2.1.2 Sampling Rate Reduction

In case the host computer is unable to sustain a continuous USB data rate of approximately 13 MB/s, which is required for operating the LSPM 1.0 Power Meter at its nominal sampling rate of 2 MSamples/s, the LSPM 1.0 TCP Server will automatically reduce the sampling rate, first to

1 MSamples/s, and if this data rate is still unsustainable, to 0.5 MSamples/s. The LSPM 1.0 TCP Server and the LSPM 1.0 GUI will display notification messages as shown in Figure 6.

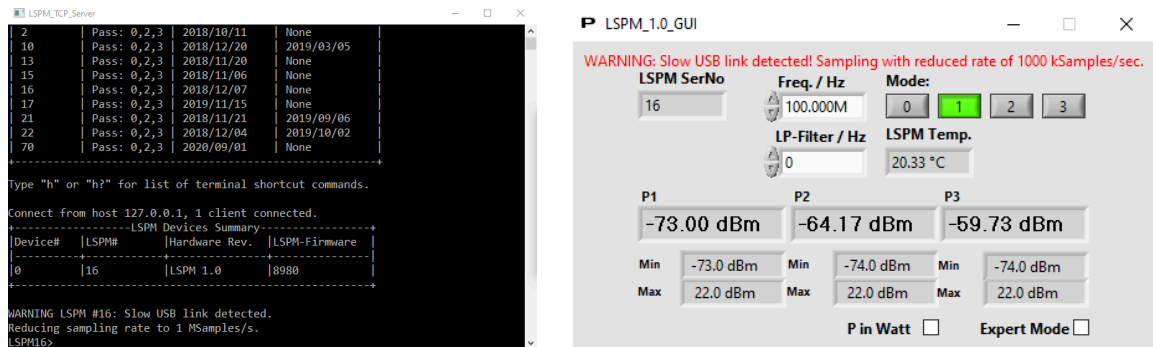


Figure 6: Reduced sampling rate operation of LSPM 1.0 TCP Server and GUI

Insufficient USB speeds can be caused by connecting too many high data rate USB devices to the same bus. Moving the LSPM 1.0 Power Meter to a different USB root hub or adding a USB root hub can help to fix this issue. Also, many USB extension products are known for being unable to operate at full USB 2.0 speeds for prolonged periods of time. At the time of writing the only USB 2.0 extension known to work is optoUSB2.0 manufactured by mk-messtechnik GmbH.

The current sampling rate can be queried using the SCPI commands »:SYSTEM:SSkip? [<MPMeter>]« and »:SYSTEM:SRate? [<MPMeter>]«.

## 2.2 Components, Connectors and Indicators

The LSPM 1.0 Power Meter's housing contains:

- up to three power detectors,
- a thermoelectric temperature controller for maintaining a constant power detector temperature,
- a BNC trigger input/output connector for synchronization and
- a USB 2.0 interface connecting to the host computer.

As shown in Figure 7, the main switch is located on the left side of the front panel. In "0" position, it disconnects the power meter's external 5 V supply. The right side of the front panel is occupied by the N sockets and the air outlet of the detector temperature controller, the latter must not be obstructed. Two labeled LED indicators display the LSPM's operating state as follows:



Figure 7: LSPM 1.0 Power Meter front panel

Power (green)

**Flashing**

Main switch is on, power meter is inactive.

**Continuously on**

USB connection to TCP server has been established.

**Continuously off**

Main switch is off, power supply is disconnected or power meter firmware is compromised.

Temp (red)

**Continuously off**

Temperature is being controlled within the power detectors' optimum operating temperature range.

**Continuously on**

Temperature of the power detectors is above its optimum operating range, detectors are being cooled.

**Flashing**

Temperature of the power detectors is below its optimum operating range, detectors are being heated.

The LSPM 1.0 Power Meter's back panel shown in Figure 8 contains the air inlet of the power detector temperature controller and must not be obstructed.

The following electrical connectors are located at the bottom edge of the back panel, left to right:

USB

USB B connector attaching the power meter to the host computer.

5 V 3 A

External DC power supply, barrel jack 2.1/5.5 mm.

Trigger

Trigger input/output BNC connector using 5 V CMOS logic levels.

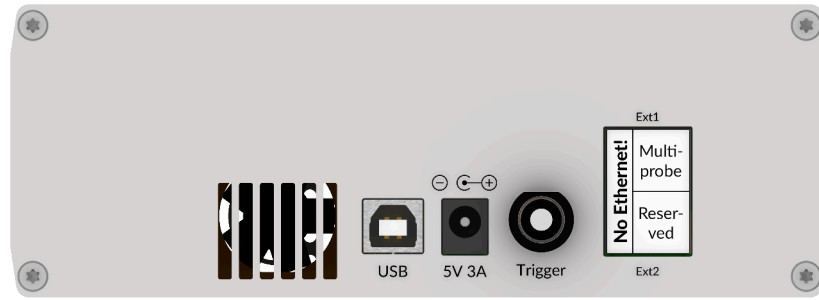


Figure 8: LSPM 1.0 Power Meter back panel

Ext 1

RJ45 extension connector for systems with multiple LSPM 1.0 Power Meters and LSProbe 1.2 Field Probes. **No Ethernet interface!**

Ext 2

Reserved for future use. **No Ethernet interface, do not connect!**

### 2.3 Systems with Multiple Power Meters and Field Probes

Continuous power statistics for systems with multiple LSPM 1.0 Power Meters and LSProbe 1.2 Field Probes require a hardware link via the “Ext 1” extension connector of every power meter. For systems with two power meters or a power meter and a compatible E-field probe a straight shielded RJ45 (EIA/TIA 568) patch cable is sufficient, see Figure 9. For larger Multiprobe systems an LSFrame Basic interconnect and power supply unit as shown in Figure 10 connects to every LSPM 1.0 Power Meter and LSProbe 1.2 Field Probe’s Computer Interface. Note that the barrel plug patch cables used for supplying the power meter and field probe computer interfaces via the LSFrame Basic interconnect and power supply unit are not shown.

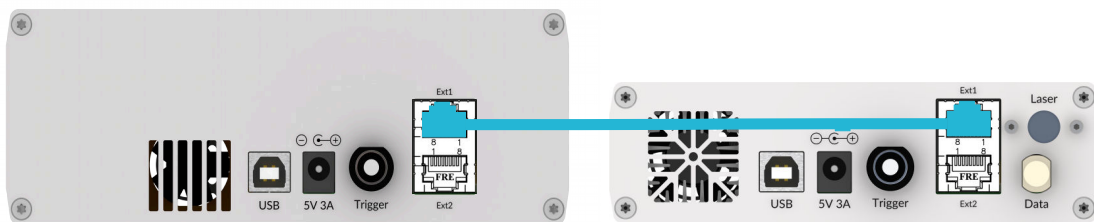


Figure 9: Connection of one LSPM 1.0 Power Meter and one LSProbe 1.2 Field Probe

In a system with multiple LSPM 1.0 Power Meters and LSProbe 1.2 Field Probes one LSPM 1.0 Power Meter or LSProbe 1.2 Field Probe Computer Interface is configured as the statistics master. The statistics master controls continuous statistics collection via dedicated enable and snapshot

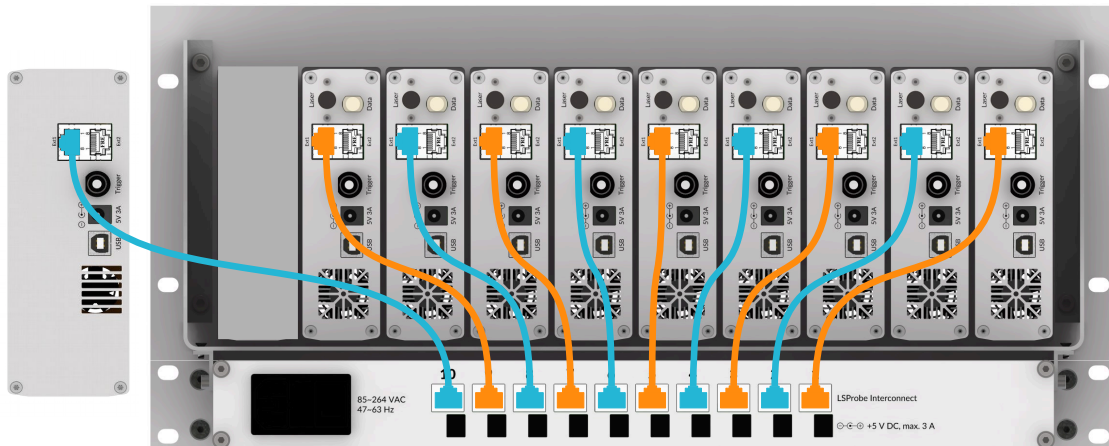


Figure 10: Connections for one LSPM 1.0 Power Meter and nine LSProbe 1.2 Field Probes

lines. The hardware link also carries lines indicating the master/slave status of each power meter and computer interface in a Multiprobe System. There are dedicated indicator LEDs on the front of the LSFrame Basic interconnect and power supply unit. The master is indicated by a flashing LED. Multiprobe slaves' LEDs are on continuously if the statistics collection is turned on. Continuously off LEDs indicate that statistics collection is off.

Upon enumeration the first devices are automatically set to master status. Therefore, the default LSPM 1.0 Power Meter statistics-master or the default LSProbe 1.2 Computer Interface statistics-master must be set to slave.

## 2.4 Trigger Inputs and Outputs

The LSPM 1.0 Power Meter features two independent trigger inputs and outputs. The BNC connector on the back of the power meter uses a single-ended 5 V CMOS logic trigger signal. Figures 11 and 12 show the basic point-to-point setup for using an external device as either trigger target or trigger source. The external device can be another power meter or an electrically compatible third-party device.

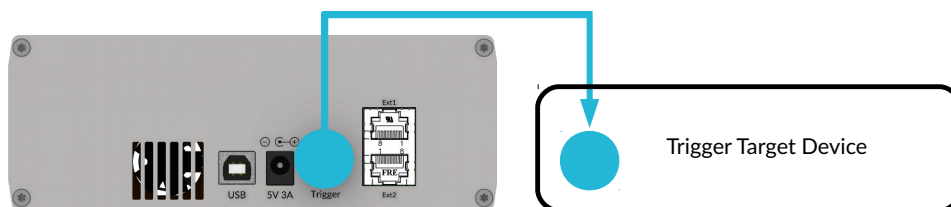


Figure 11: External trigger output using BNC connector

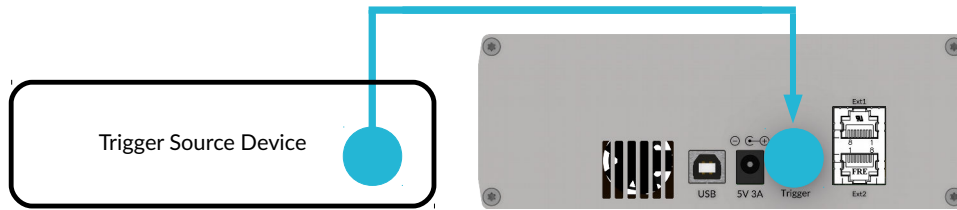


Figure 12: External trigger input using BNC connector

The Ext1 RJ45 socket on the back of the power meter uses a differential 3.3V CMOS logic trigger signal. This signal can be used to exchange trigger signals in a Multiprobe setup containing two or more LSPM 1.0 Power Meters or LSProbe 1.2 Computer Interfaces as shown in Figure 13. When a LSFrame Basic is connected all power meters and computer interfaces automatically share a pair of dedicated differential logic lines. The Ext1 RJ45 sockets are not recommended for use with third-party devices.

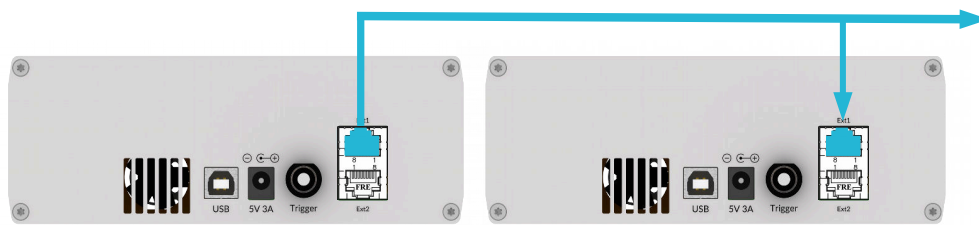


Figure 13: External trigger input and output using Ext1 RJ45 sockets

### 3 LSPM Software

#### 3.1 LSPM 1.0 TCP Server and GUI Installation

Software installation consists of the following steps:

1. Run the LSPM 1.0 installer. Follow the displayed installation instructions carefully.
2. Copy the supplied calibration data into the directory selected during TCP server and GUI installation, e.g., if the calibration data path is  
`C:\Program Files (x86)\LSPM_1.0\cal` and the serial number of the LSPM 1.0 Power Meter is 42, please copy the folder named `sn42` from the installation medium into the `cal` directory. Repeat the procedure for all LSPM 1.0 Power Meters connected to the host computer.

The installation path of the LSPM 1.0 software is stored in a system-wide environment variable named `LSPM_1.0_PATH` which is set during software installation. The installation path contains sub-directories named `bin`, `cal`, `doc` and `lib`. All executable files are stored in the `bin` sub-directory.

The `bin` sub-directory also contains the LSPM 1.0 configuration file named `LSPM_1.0.ini`. The configuration file must always be located in the same directory as the LSPM 1.0 TCP Server and GUI. `LSPM_1.0.ini` contains the following settings which are set during software installation and may be modified as required:

#### HOST

Defines the host name or IP address of the computer running the TCP server. This setting is used by the LSPM 1.0 GUI only, the default setting is `localhost`.

#### PORT

Defines the TCP port number of the LSPM 1.0 TCP Server. This setting is used by the LSPM 1.0 TCP Server and GUI, the default setting is `10,000`.

#### CAL\_PATH

Define the directory containing the calibration data folders and ZIP files for all LSPM 1.0 Power Meters. This setting is used by the LSPM 1.0 TCP Server only, it defaults to the path of the LSPM 1.0 TCP Server.

#### SAVE\_PATH

Defines the directory which is used to log files in. This setting is used by the LSPM 1.0 TCP Server and GUI, it defaults to the path of the LSPM 1.0 TCP Server and GUI.

#### UPDATE\_CHECK

Enable and disable checking for software updates on the LUMILOOP homepage when the LSPM 1.0 GUI is run. This setting is used by the LSPM 1.0 GUI only. If set to "1" update checking is enabled. If the set to "0" update checking is disabled, this is the default.

If the LSPM 1.0 TCP Server and LSPM 1.0 GUI are supposed to run on different host computers, the installer must be run on both systems. The setting `HOST` of the system running the LSPM 1.0 GUI must be set to point to the host computer running the LSPM TCP server. Moreover, the setting `PORT` must be set identically.

## 3.2 USB Driver Installation

If the host computer has access to the online Microsoft Windows Update, the drivers should be installed automatically when a Computer Interface is connected and powered-up for the first time. Note that for normal operation no Internet access is required.

After successful driver installation the Device Manager will list the Computer Interface as "USB Serial Converter A" and "USB Serial Converter B" as shown in Figure 14 (a) and (b). Note that the device naming is generic and references neither LUMILOOP nor LSPM. However, this does not affect the proper operation of the LSPM 1.0 Power Meter.

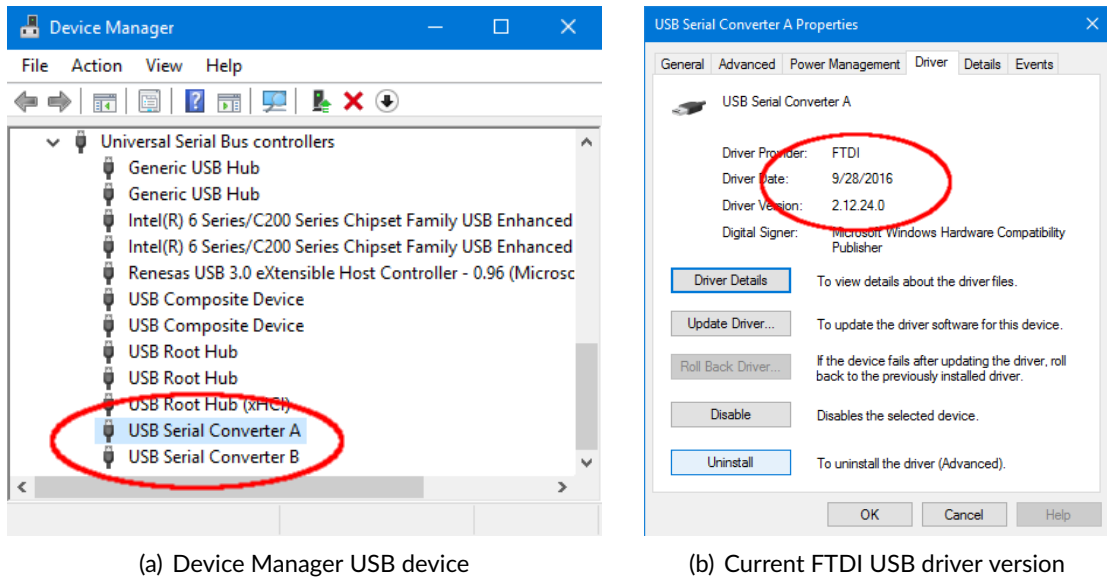


Figure 14: Correctly installed FTDI USB driver

If the automated install fails, or if the host computer has no Internet connection, execute the FTDI USB driver installer CDM v2.12.28 WHQL Certified.exe contained in the the LSPM installation path's lib directory as shown in Figure 15.

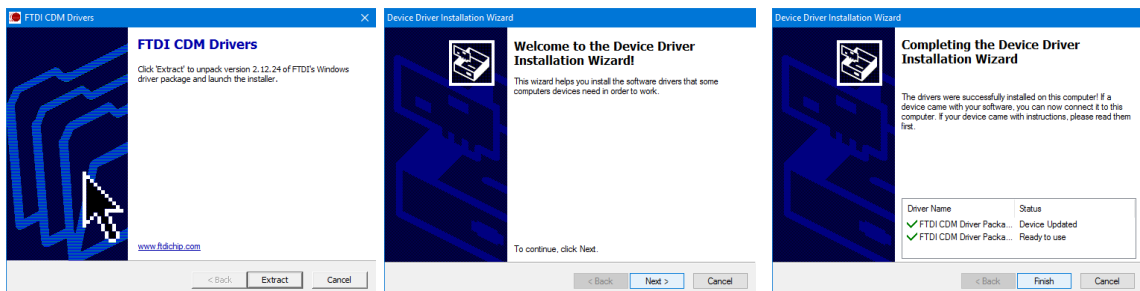


Figure 15: Manual FTDI USB device driver installation

It is strongly advised to observe the following recommendations:

- Plug the LSPM 1.0 Power Meter directly into the computer. Do not use a USB hub or docking station.
- Do not connect other high bandwidth USB devices to the same USB root hub. In rare cases this may reduce read performance significantly, resulting in unreliable operation and eventual loss of measurement data.
- Especially, do not operate the LSPM 1.0 Power Meter at a USB port where a USB graphics adapter is installed or was previously installed. The USB graphics driver may disturb communications even if the hardware is no longer attached.



### 3.2.1 Troubleshooting USB Driver Installation

If no LSPM 1.0 Power Meter and no other FTDI hardware have previously been connected to the computer and automatic Windows driver installation is deactivated or no Internet connection is available the error message shown in Figure 16 (a) will be displayed. In this case the Device Manager's "Other devices" section will give an output similar to Figure 16 (b), listing the LSPM 1.0 Power Meter's USB end points as unknown devices.

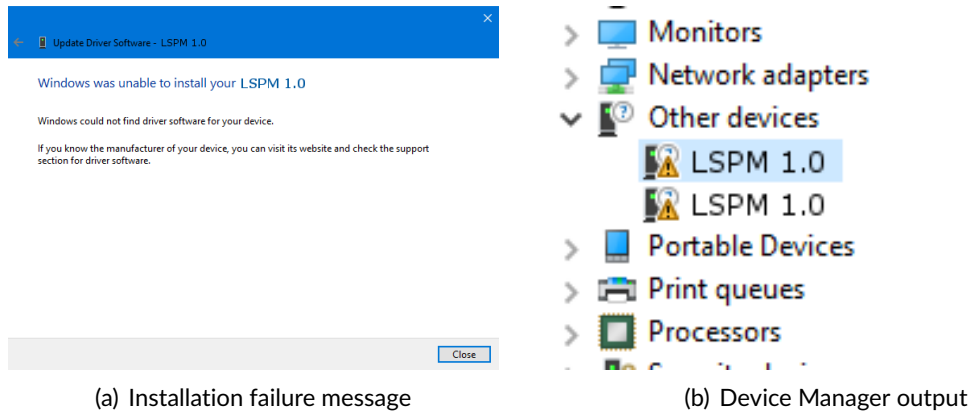


Figure 16: USB device driver failure messages

Make sure that you are using the most recent FTDI USB driver. The driver version at the time of writing is 2.12.24. Using out-of-date USB drivers may result in improper operation.

Check the driver version by opening the Device Manager and extending the "USB-Controller" category. Right-click "USB Serial Converter A" and select "Properties". Open the "Driver" tab to view the FTDI USB driver version as shown in Figure 14 (b). If an older version of the FTDI driver has been installed click on "Uninstall". In the following dialog, make sure to check "Uninstall the driver software for this device" as shown in Figure 17, failing to do so will prevent driver updates.

Make sure to repeat the uninstall process for all LSPM 1.0 devices using old FTDI USB driver versions. Note that the process must be executed for both "USB Serial Converter A" and "USB Serial Converter B" of every LSPM 1.0. Power-cycle each LSPM 1.0 and repeat the installation procedure as described above in Section 3.2.

### 3.3 LabView Run Time Environment Installation

The LSPM 1.0 GUI requires an installed 32 bit LabView 2012 Run-Time Engine or 32 bit LabView 2012 Development System. The LabView 2012 Run-Time Engine is included with the LSPM 1.0 installer and can be installed during LSPM 1.0 software installation. If either software package is already installed on the host computer, installation of the LabView 2012 Run-Time Engine is not required. The LabView 2012 Run-Time Engine may also be installed after LSPM software installation, its installer can be found in the `lib` sub-directory of the LSPM 1.0 installation path.

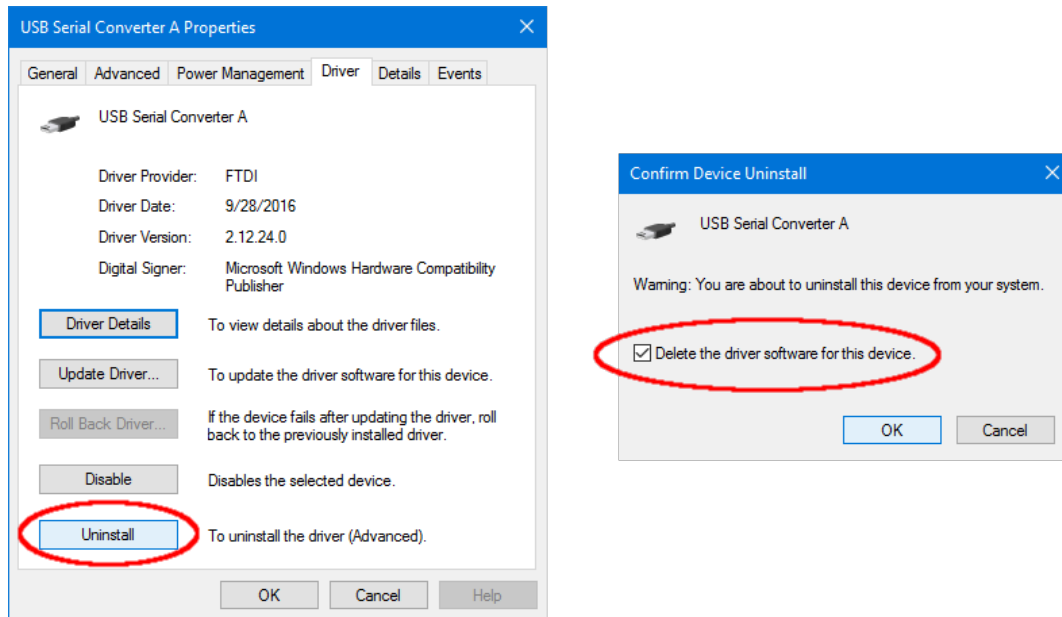


Figure 17: Uninstalling the FTDI USB driver

## 4 Measuring Power

### 4.1 Getting Ready to Measure

#### 4.1.1 Making Electrical Connections

When installing the LUMILOOP LSPM 1.0 Power Meter for the first time make the following electrical connections:

1. Connect the supplied mains adapter.
2. Connect the LSPM 1.0 Power Meter to the host computer using the supplied USB cable.
3. Optionally, connect the LSPM 1.0 Power Meter to any trigger sources or sinks via the BNC trigger connector.

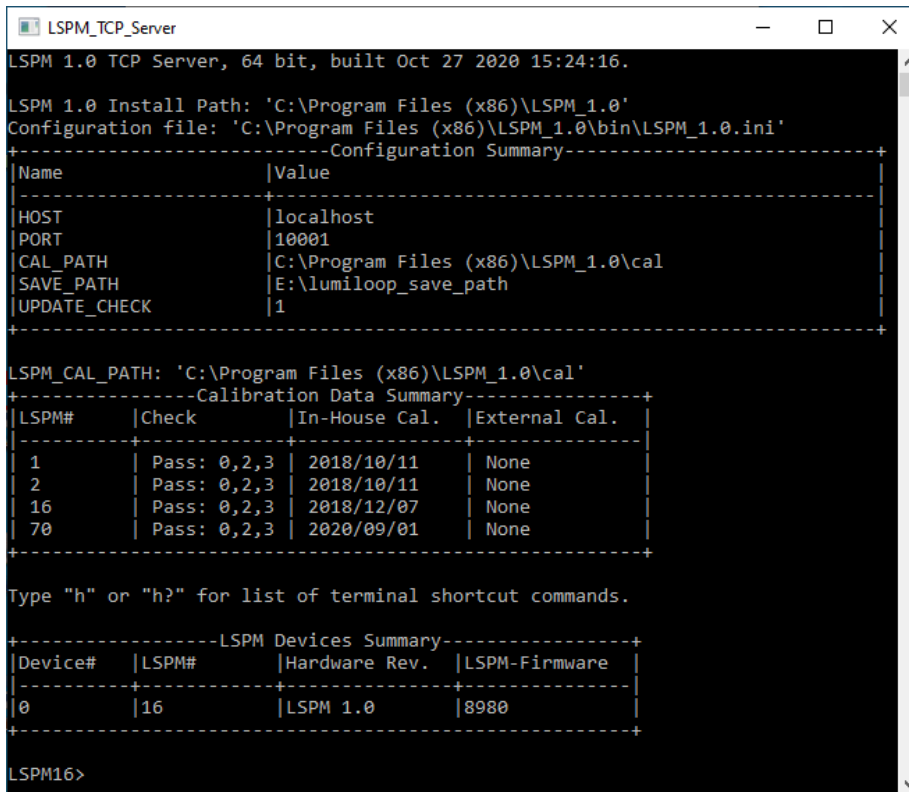
Turn on the LSPM 1.0 Power Meter by flipping the front panel switch to “1” and observe the green power LED starting to flash.

### 4.2 Power Meter Start-Up and Mode Selection

#### 4.2.1 Starting the LSPM 1.0 TCP Server

Carefully follow all instructions in the previous sections. Start the LSPM 1.0 TCP Server and ensure correct operation by verifying that the green power LED is constantly on and the LSPM 1.0

TCP Server has enumerated all connected power meters, listing their serial numbers resembling Figure 18. The LSPM 1.0 TCP Server will open inside a terminal window displaying status information, debugging output and error messages.



```

LSPM 1.0 TCP Server, 64 bit, built Oct 27 2020 15:24:16.

LSPM 1.0 Install Path: 'C:\Program Files (x86)\LSPM_1.0'
Configuration file: 'C:\Program Files (x86)\LSPM_1.0\bin\LSPM_1.0.ini'
-----Configuration Summary-----
+-----+
|Name      |Value      |
+-----+
|HOST      |localhost  |
|PORT      |10001      |
|CAL_PATH  |C:\Program |
|          |Files (x86)|
|          |\LSPM_1.0\|
|          |cal        |
|SAVE_PATH |E:\lumiloop|
|          |_save_path|
|UPDATE_CHECK|1          |
+-----+

LSPM_CAL_PATH: 'C:\Program Files (x86)\LSPM_1.0\cal'
-----Calibration Data Summary-----
+-----+
|LSPM#  |Check      |In-House Cal. |External Cal. |
+-----+
| 1      | Pass: 0,2,3 | 2018/10/11   | None         |
| 2      | Pass: 0,2,3 | 2018/10/11   | None         |
| 16     | Pass: 0,2,3 | 2018/12/07   | None         |
| 70     | Pass: 0,2,3 | 2020/09/01   | None         |
+-----+

Type "h" or "h?" for list of terminal shortcut commands.

-----LSPM Devices Summary-----
+-----+
|Device#  |LSPM#  |Hardware Rev. |LSPM-Firmware |
+-----+
| 0       | 16     | LSPM 1.0     | 8980          |
+-----+

LSPM16>

```

Figure 18: LSPM 1.0 TCP Server terminal window

Upon start-up, the LSPM 1.0 TCP Server will display the set environment variable `LSPM_1.0_PATH`, a tabular summary of the settings read from the configuration file and a tabular summary of all available calibration data.

During device enumeration, the firmware of each LSPM 1.0 Power Meter is loaded onto each device. After device enumeration the LSPM 1.0 TCP Server will list all detected power meters with their respective serial numbers and firmware revision numbers. The TCP server executable incorporates the firmware image required for proper operation. The version number can be found in the LSPM 1.0 GUI's "Connection" tab and the LSPM Devices Summary table of the LSPM 1.0 TCP Server.

When LSPM 1.0 Power Meters are added to or removed from the host computer after starting the LSPM 1.0 TCP Server, the server will detect these events, update the respective LSPM 1.0 Power Meter's firmware and display an updated LSPM 1.0 Devices Summary Table. Re-enumeration may also be forced by sending an »\*RST« SCPI command.

Since the LSPM 1.0 TCP Server needs to open a TCP port, the system's firewall may ask for permission for network access. Access must be granted to operate the LSPM 1.0 TCP Server (see Figure 19).



Figure 19: Microsoft Windows Firewall requesting TCP port access permissions

## 4.2.2 Interacting with the LSPM 1.0 TCP Server

The LSPM 1.0 TCP Server can be used to execute SCPI commands interactively. As shown in Figure 18, it will display a command prompt indicating the serial number of the power meter which SCPI commands are exchanged with. The prompt will change when a different power meter is selected.

Entering »h« will display a list of available shortcut commands. The prompt features a basic command history that can be accessed using the up and down cursor keys.

The command prompt can also be used for repeated execution of any SCPI command or shortcut command, e.g., polling queries. This can be achieved by entering »l«, i.e., lower case L, followed by an optional number of milliseconds, a space and a command. If the polling interval is omitted, a default value of 500 ms will be used. For example, the simple looped shortcut command »l p« can be used to query the power of all channels every 500 ms. The power of channel 1 can be polled every 100 ms by entering »l100 :meas:p?«. Note that shortcut commands, polling and command history are not supported for TCP client connections.

## 4.2.3 General Notes on the LSPM 1.0 GUI

The LSPM 1.0 GUI has two display modes: Basic Mode and Expert Mode. It will always start in Basic Mode as shown in Figure 20. Expert Mode, as shown in Figure 22, can be toggled using the check box "Expert Mode" in the lower right corner of the main window.

The LSPM 1.0 GUI is intended as an easy to use demonstration software for all LSPM capabilities. The GUI is designed in such a way that it will not issue any configuration commands to the TCP server unless the user changes a setting using one of the controls. This feature allows for running the LSPM 1.0 TCP Server in parallel with any third-party EMC software and observing all power meter settings and measurement results. This feature is especially useful during third party EMC software integration and function testing.

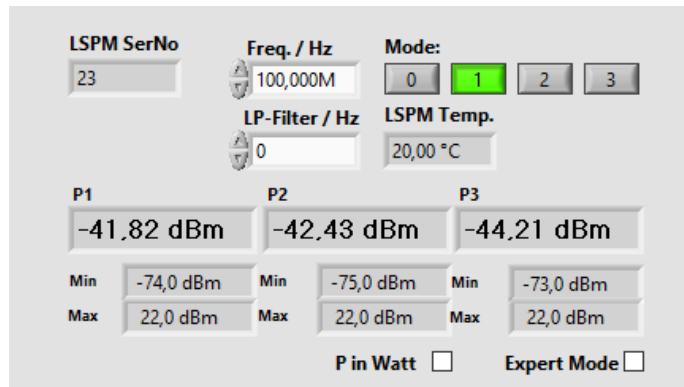


Figure 20: LSPM Basic GUI



Figure 21: Frequency control for matching (a) and mismatching (b) TCP server and GUI settings

When there is a mismatch between a setting of the LSPM 1.0 TCP Server and the expected setting of the GUI, the font color of the control element will turn red. A black font color indicates that the GUI's settings are identical with the TCP server. Figure 21 shows the behavior of the GUI for matching and mismatching frequency and mode settings.

If the computer running the LSPM 1.0 TCP Server is unable to process incoming values quickly enough, the data rate will be reduced automatically, first to 1 MSamples/s or 0.5 MSamples/s thereafter. Both the LSPM 1.0 TCP Server and the LSPM 1.0 GUI will display warning messages.

#### 4.2.4 Mode Selection Using the GUI

For accurate power measurements, the signal's frequency must be specified using the "Freq./Hz" entry field. Values are entered in hertz, SI unit prefixes may be used, e.g., "1.8G" for 1.8 GHz. The actual decimal separator is determined by the host computer's system language. When selecting a mode via the buttons "0" through "3", the user must ensure that "Frequency/Hz" is appropriate for the selection. Frequency values outside a mode's supported frequency range will not be accepted. See Table 1 on page 15 for a list of supported modes, their frequency ranges and sampling rates. Hovering the mouse pointer over any of the mode selection buttons of the GUI will display a tool-tip containing a brief description of the mode.

### 4.2.5 Mode Selection Using SCPI Commands

After establishing the TCP/IP connection, the SCPI commands »:SYSTem:SERial? [<MPMeter>]« and »:SYSTem:SERial <Value>« can be used to query all enumerated LSPM 1.0 Power Meters and set the serial number of the power meter to be accessed. If only a single LSPM 1.0 Power Meter is attached to the host computer it will be selected automatically and its serial number can be queried using »:SYSTem:SERial? [<MPMeter>]«. »:SYSTem:SERial? [<MPMeter>]« can be used with the MPMeter parameter set to zero to list all enumerated power meters.

The desired operating mode of the selected power meter is set using »:SYSTem:MODE <Mode>[,<MPMeter>]«. Refer to Table 1 on page 15 for a list of valid modes. Set the operating frequency in hertz using »:SYSTem:FREQUency <Frequency>[,<MPMeter>]«. Frequencies outside a mode's supported frequency range will be diverted to the nearest supported frequency. »:SYSTem:FREQUency? [<MPMeter>]« can be used to verify the frequency setting.

## 4.3 Continuous Power Measurements

While the LSPM 1.0 Power Meter is capable of exceptionally high speed measurements it is also able to perform high precision measurements of quasi-static electric fields. For continuous power measurements the TCP server receives all field strength values, applies calibration data and performs low-pass filtering if configured accordingly.

### 4.3.1 Continuous Measurements Using the GUI

As shown in Figure 22, continuous measurements are configured through the "Log" tab. Power values are displayed both textually and graphically. Values are polled continuously, resulting in an update rate in the order of 100 samples per second. The actual polling frequency depends on the speed of the host computer and the speed of the network connection, in case of remote operation.

The calibrated range of power values, indicating the smallest and largest power value that can be measured for a channel at a given frequency is displayed below each channel's power value. For single and dual channel versions of the LSPM 1.0 Power Meter "NaN" will be displayed for the unpopulated channels.

In the tab area at the upper right of the window "Graph Length" determines the maximum number of samples that will be displayed in the plot pane at the bottom of the window. To display the elapsed time instead of sample indexes for graphs' x-axis and graph length, select the "Display x-axis as time" check box. The plot can be paused using "Pause Graph" and cleared using the "Clear" button. Through the "Log" tab the logging of continuously polled values and of the "Trigger", "Radar", "Sweep" and "Statistics" data can be turned on and off individually. If enabled, newly arriving data of the specific subsystem gets immediately written to a log file. The button "Quick Save" offers a shortcut for the logging of the currently viewed data without going through the "Log" tab. The data gets logged to a file whose prefix is determined by the "File Prefix" text entry field. Log files

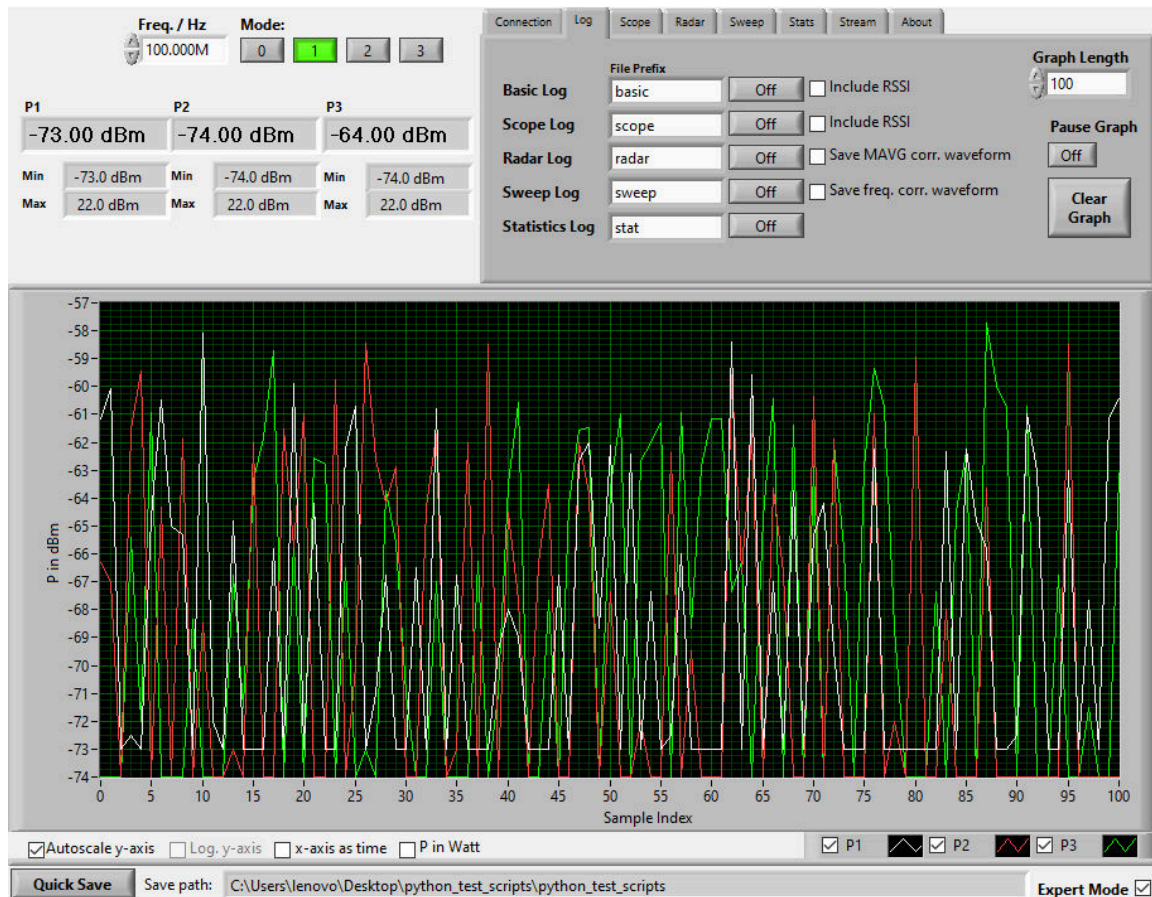


Figure 22: LSPM 1.0 GUI data logger view

are saved according to the setting `SAVE_PATH` defined in the `LSPM_1.0.ini` configuration file. The power meter's serial number, a date and time string and a CSV file suffix will be appended to every newly created log file. See Section 10.1.1 on page 126 for detailed information on all file format details.

Low-pass filter settings are made through the "LP-Filter" numeric input field. A value of 0 disables low-pass filtering. A non-zero value sets the  $-3$  dB cut-off frequency for the first order low-pass filter used for power values, for example 10 Hz. When changing the frequency, the low-pass filtered power value will be updated directly, i.e., low-pass filtered values will see a step response instead of a slewing of values.

The controls below the graph offer shortcuts for enabling and disabling automatic y-axis scaling, logarithmic and linear display of the y-axis, choosing between sample index and time as the unit for the graphs' x-scale and choosing between displaying power in dBm or Watts. The plot pane also supports adjusting the x- and y-axis by clicking the first or last of the axis labels and setting its value. Individual graphs can be disabled by un-checking the graph labels at the bottom right of the graph. Right-clicking the graph exposes a number of generic functions including data export to the host computer's clipboard.

### 4.3.2 Continuous Measurements Using SCPI Commands

After setting the operating mode and frequency as described in Section 4.2.5, the low-pass filter frequency is configured through »:MEASure:LPFrequency <Frequency>[,<MPMeter>]«. A synchronized set of power values can be queried through the ALL variant of the »:MEASure:P[1]/P2/P3/ALL? [<MPMeter>]« command. This is the recommended method ensuring that values have been acquired at the same time. Power values can be queried individually through »:MEASure:P[1]/P2/P3/ALL? [<MPMeter>]«.

## 4.4 Triggered Power Measurements

Triggered power measurements allow the user to take full advantage of the LSPM's exceptionally high-speed measurements. Waveform acquisition can be triggered by software, by edge-sensitive hardware trigger signals originating from the power meter's BNC/RJ45 connectors or a channel's power level with a set threshold for power level triggering. The trigger system has a built-in dead-time of 100  $\mu$ s, i.e., it can process up to 10,000 events per second.

### 4.4.1 Power Waveform Acquisition Using the GUI

Figure 23 shows the LSPM 1.0 GUI in scope mode which is entered by selecting the "Scope" tab. When the scope tab is selected, the textual display shows the averaged power values for the displayed waveforms. The calibrated range of power values, indicating the smallest and largest power value that can be measured for a channel at a given frequency is displayed below each channel's power value. For single and dual channel versions of the power meter "NaN" will be displayed for the unpopulated channels.

The "Trigger Source" drop-down box is used for selecting rising or falling edge external BNC, external RJ45 and channel 1/2/3 power value triggering. For the latter, the threshold value in dBm or Watts can be set via "P1/P2/P3 Threshold" By changing "Mode" automatic free-running triggering, normal event-based triggering and one-shot single triggering can be selected. "Trigger State" displays the condition of the trigger system. The length of the acquired waveform is determined by the "Trigger Length" numeric input field, setting the number of samples for each triggered waveform. "Trigger Begin" sets the start of the saved waveform relative to the position of the trigger event. "Trigger Begin", "Trigger Length" and the graph's x-axis can be displayed/entered as time values by selecting the "Display x-axis as time" check box. Time values are displayed according to the sampling rates in Table 1 on page 15. The "Arm" and "Force" buttons serve to prepare the trigger system and to force triggering regardless of actual trigger events. The "BNC Trigger Output" and "RJ45 Trigger Output" drop-down menus are used to enable trigger signal output via the power meter's BNC connector and "Ext1" RJ45 socket, and set their respective signal polarities. Triggers can be output either when encountering a trigger event, including forced triggering, or for synchronization triggering as described in Section 9.6.29.



Point triggering enables recording of waveforms consisting of multiple sub-waveforms of equal size, based on multiple trigger events. Point triggering is enabled by selecting more than the default single trigger point in the “Trigger Points” input field. The number of recorded trigger points is displayed by “Point Progress”. After processing the set number of trigger points, the trigger state will reach DONE and the waveform will be displayed.

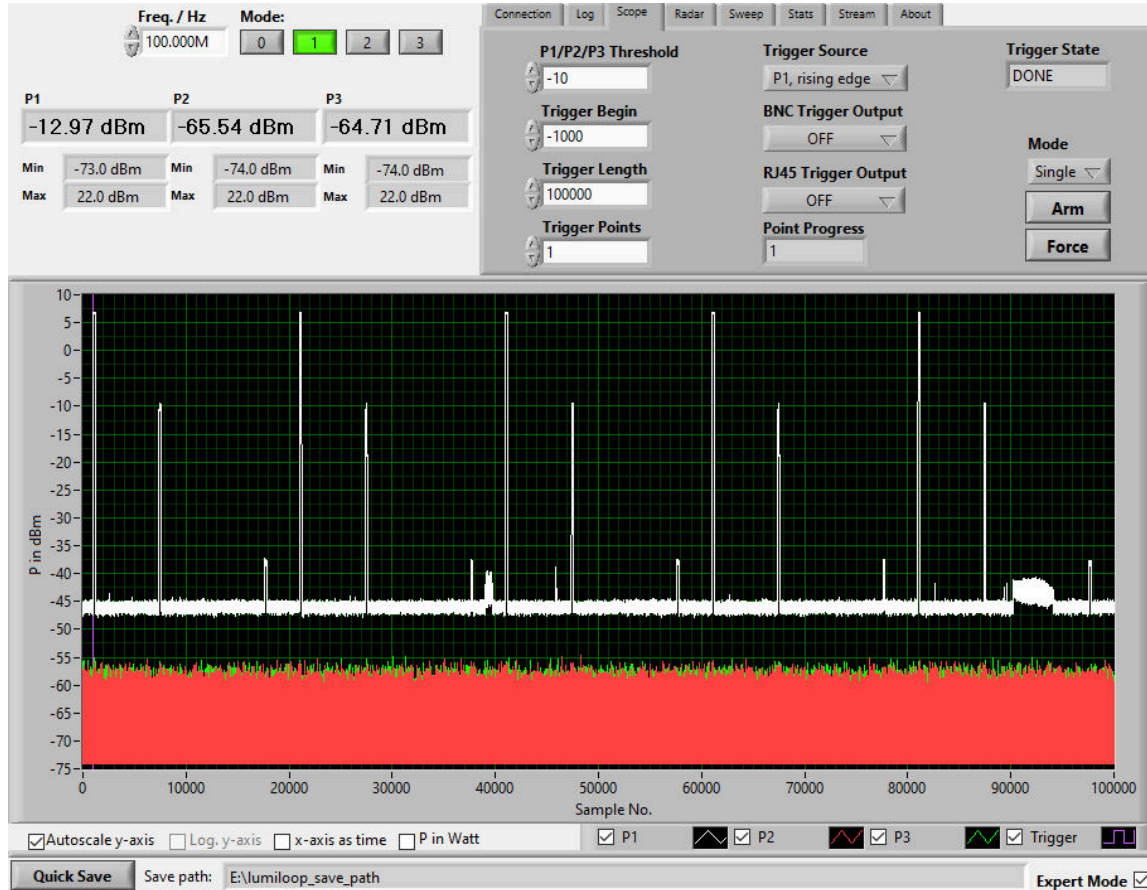


Figure 23: LSPM 1.0 GUI, Scope tab

#### 4.4.2 Power Waveform Acquisition Using SCPI Commands

The state of the trigger system is queried using `»:TRIGger:STATE? [<Timeout>,<MPMeter>]«`. The configuration of triggered measurements must take place in IDLE state. Waveform query must take place when the trigger system is in DONE state. The SCPI commands `»:TRIGger:Clear [<MPMeter>]«`, `»:TRIGger:ARM [<MPMeter>]«` and `»:TRIGger:FORce [<MPMeter>]«` are used for directly manipulating the state of the trigger system. Figure 24 shows all valid trigger states and state transitions.

After receiving `»:TRIGger:ARM [<MPMeter>]«`, the trigger subsystem first transitions to the ARM state before entering the state ARMED, readying the power meter for trigger event processing. Since trigger events will only be processed in state ARMED, the queries `»:TRIGger:STATE? [<Timeout>,<MPMeter>]«` or `»:TRIGger:ARMed? [<Timeout>,<MPMeter>]«` must be used to verify the

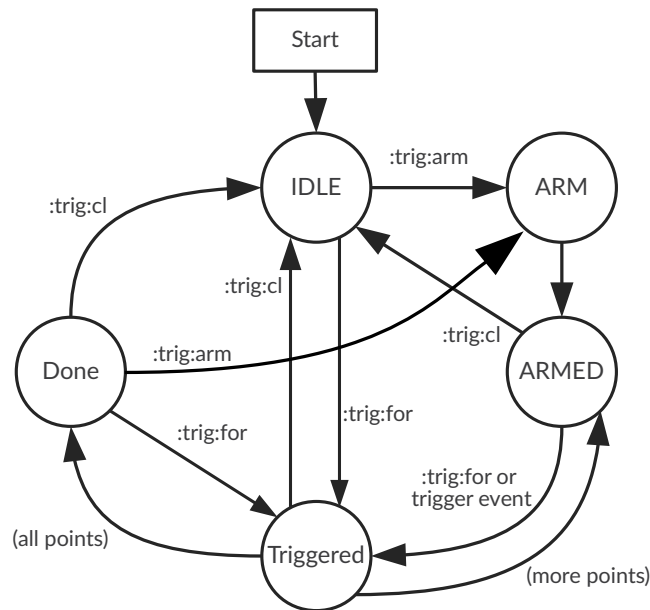


Figure 24: Trigger system states and state transitions

state ARMED before generating trigger events. Similarly, the command »:TRIGger:DONE? [<Timeout>,<MPMeter>]« can be used to wait for the trigger subsystem to reach the DONE state. The command »:TRIGger:PROgress? [<MPMeter>]« can be used to query the number of recorded samples.

The trigger source and polarity are set using »:TRIGger:SOURce <Source>[,<MPMeter>]« and »:TRIGger:FALLing <0/1>[,<MPMeter>]«. If power value triggering is employed, the trigger level is set via »:TRIGger:LEVel <Level>[,<MPMeter>]«. The trigger length is set using »:TRIGger:LENgth <Length>[,<MPMeter>]« and »:TRIGger:BEgin <Index>[,<MPMeter>]«. The corresponding query commands are »:TRIGger:LENgth? [<MPMeter>]« and »:TRIGger:BEgin? [<MPMeter>]«. The progress of waveform acquisition can be checked using »:TRIGger:PROgress? [<MPMeter>]«.

Trigger output is configured using :TRIGger:OUTput <0/1>[,<MPMeter>], :TRIGger:INVert <0/1>[,<MPMeter>], :TRIGger:SYnc <0/1>[,<MPMeter>] for the BNC connector and :TRIGger:BPouTput <0/1>[,<MPMeter>], :TRIGger:BPINVert <0/1>[,<MPMeter>] and :TRIGger:BPsync <0/1>[,<MPMeter>] for the RJ45 connector.

The number of trigger points is set via »:TRIGger:POINt <Points>[,<MPMeter>]«, the number of recorded points is queried using »:TRIGger:PTProgress? [<MPMeter>]«. The full length of the waveform for all trigger points can be queried using »:TRIGger:FLENgth? [<MPMeter>]«. The command »:TRIGger:PTTimes? [<MPMeter>]« can be used to retrieve the relative timing of trigger events.

In DONE state the waveform values can be queried using the commands »:TRIGger[:WAVEform][:Power]:P[1]/:P2/:P3? [<MPMeter>]«.

Average waveform values can be queried using »:TRIGger[:WAVEform][:Power]:ALL? [<MPMeter>]«. The »:TRIGger[:WAVEform][:Power]:BINary? [<MPMeter>]« command is available for fast and com-

putationally efficient waveform readout.

#### 4.4.3 Pulse Measurements Using the GUI's Radar Tab

The LSPM 1.0 Power Meter is able to scan a previously recorded power waveform for pulses to find their positions, lengths, averaged power values and maximum power values. See Section 4.4.1 for details regarding triggered value acquisition. Pulses can be detected if they are at least 0.5  $\mu$ s long and 0.5  $\mu$ s apart. Pulse measurements in mode 1 below 30 MHz and in mode 3 are not recommended due to the small video bandwidth of the low-band detector.

In order to enable pulse detection and evaluation within the LSPM 1.0 GUI select the “Radar” tab and activate “Enable Radar” as shown in Figure 25.

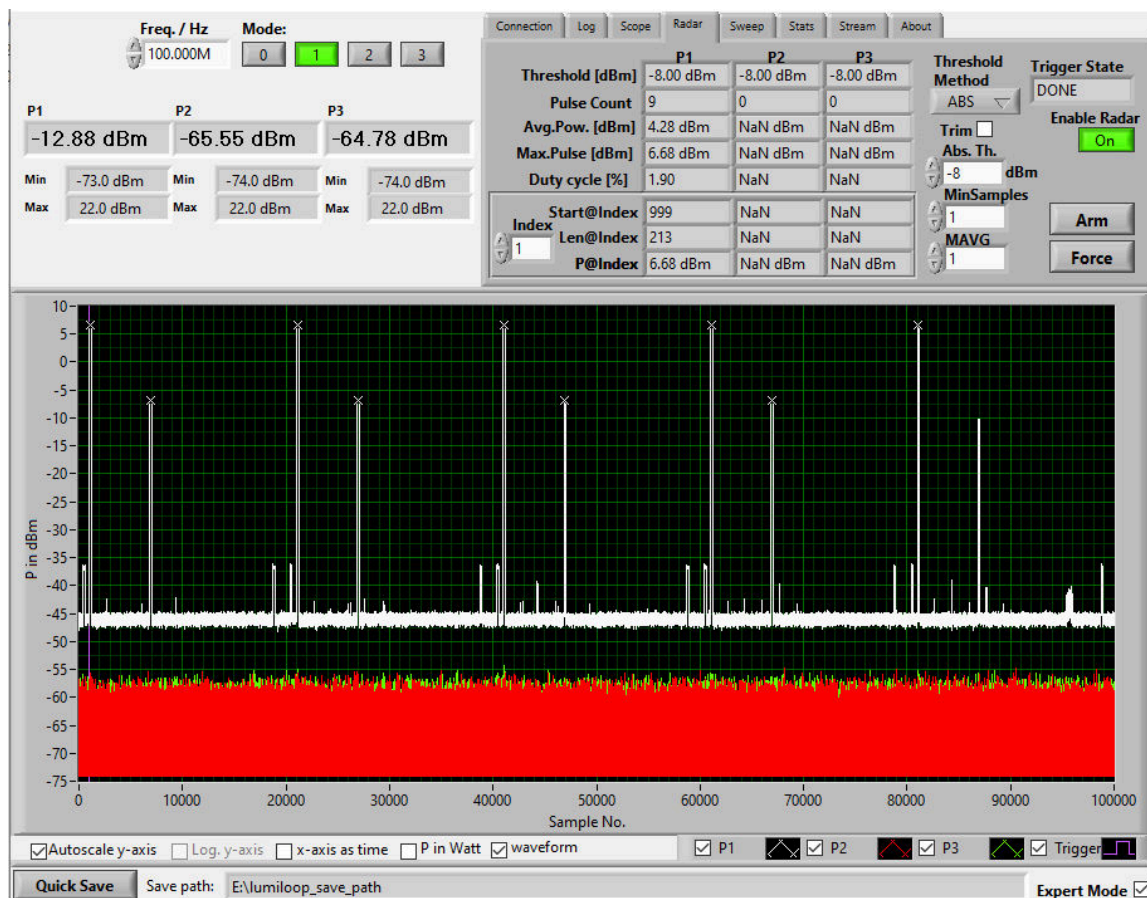


Figure 25: LSPM 1.0 GUI, Radar tab

Pulses can be evaluated with or without pulse trimming, use the “Pulse Trimming” checkbox to enable pulse trimming. If pulse trimming is disabled, all samples above the threshold will be treated as a part of the pulse, as shown in Figure 26. If pulse trimming is enabled, samples at the start and end of the pulse will be treated distinctly, this is useful for constant peak power evaluation of short pulses with short rise and fall times. In this case, for pulses containing one or two samples only the larger sample

value will be used. For pulses containing at least three samples the pulse's power value is defined as the arithmetic mean of all but the first and last sample value of the pulse, Figure 27 depicts this behavior and denotes these samples using squares. Moreover, pulses starting before the beginning of a waveform will only be detected if they contain at least two samples. The last sample of the pulse is always discarded and the pulse's power in this special case is defined as the arithmetic mean of all preceding power values. Pulses ending after the end of a waveform are handled in the same way, discarding the pulse's first sample value instead of the pulse's last sample value. The "MinSamples" input field sets the minimum number of samples per pulse of the transmitter. All pulses that are shorter will be discarded, as shown in Figure 28.

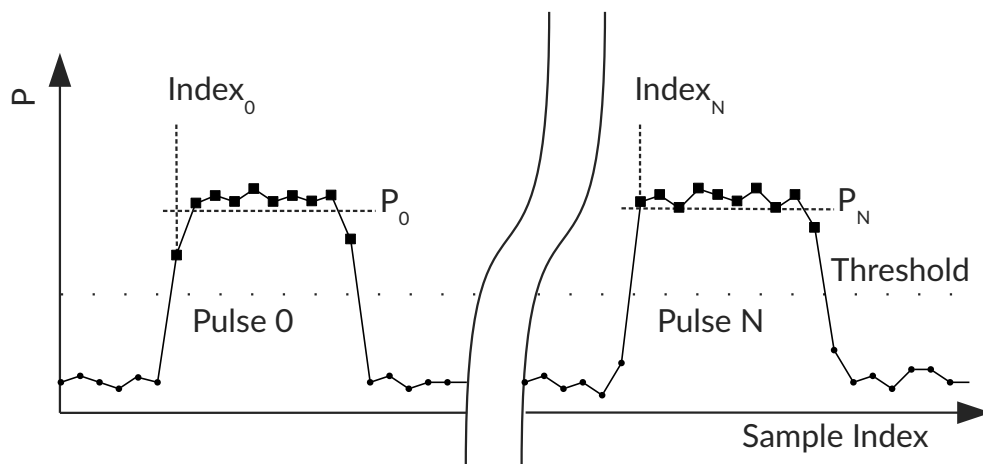


Figure 26: Radar pulse detection when pulse trimming is disabled. All samples above the threshold are used for power calculation, leading to a lower average pulse power.

The "Threshold Method" entry field is used to set one of four threshold methods:

**AVG**

The pulse threshold is automatically set to the arithmetic mean of the maximum power value and minimum power value in the waveform.

**ABS**

Pulse detection will use the fixed power level set via the "Abs. Th." entry field as the threshold.

**REL**

Pulse detection will use a power level relative to the maximum value found in the waveform as the threshold. The threshold level in dB relative to the maximum is set via the "Rel. Th." entry field. Figure 28 left demonstrates the principle – pulse 3 will be ignored because its peak power is too low.

**HIST**

Pulse detection will use a histogram-based threshold, based on the minimum of the power

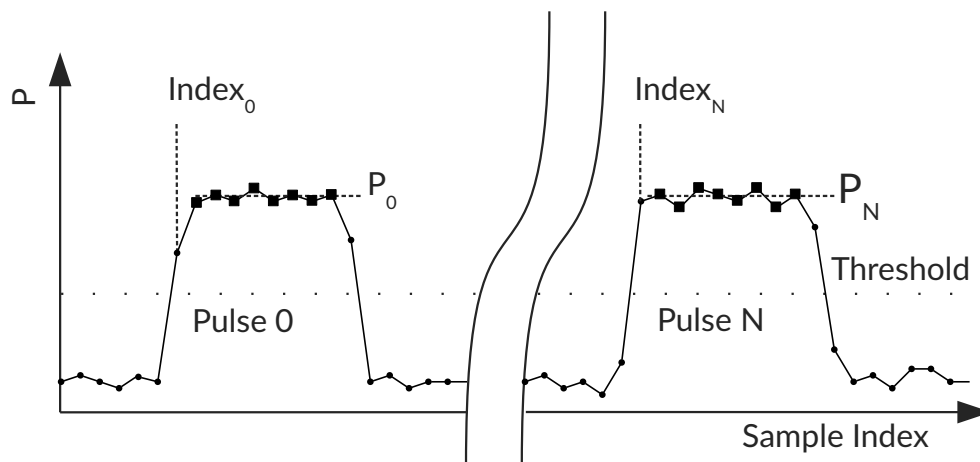


Figure 27: Principle of radar pulse detection when pulse trimming is enabled

value probability distribution which is located between the probabilities of the noise floor and the active transmitter's power level, as shown in Figure 28. The histogram which is created in the background has a resolution of 1 dB.

The "Clearance" entry field can be used to ensure that the threshold is located an integer-valued multiple of 1 dB away from the probability peaks of the probability distribution. This way a minimum signal-to-noise ratio can be guaranteed.

The "MAVG" input field sets the number of samples used for a moving average filter that is applied to the waveform values. A value of "1" will disable the moving average filter, a larger value will calculate the moving average of the set number of samples, thus reducing both video bandwidth and noise. A value of two is most common for standards-compliant measurements which require at least 1 MSamples/s.

Results are presented independently for all channels of the power meter. Averaged power values for all axes are displayed for channel 1, 2 and 3. Averaged values over all pulses are shown on top, the start index of individual pulses and their averaged power value are displayed below. If no pulses are detected, the averaged power values will display "NaN". The sample index denotes the position of the pulse within the waveform, it represents the first sample exceeding the given pulse threshold. The numeric entry field "Index" can be used to step through individual pulses, the first value has an index of one. If an invalid index is chosen, "NaN" will be displayed for sample index and power value.

The number of detected pulses can differ for each channel. Manually set or automatically calculated threshold values are shown in the row labeled "Threshold [dBm]". The row labeled "Pulse Count" gives the number of detected pulses. The arithmetic means of all pulses' power values is given in the row "Avg.Pow. [dBm]". The power of the pulse with the largest averaged power can be found in the row labeled "Max.Pul. [dBm]". The duty cycle, i.e., the ratio of the number of samples in the

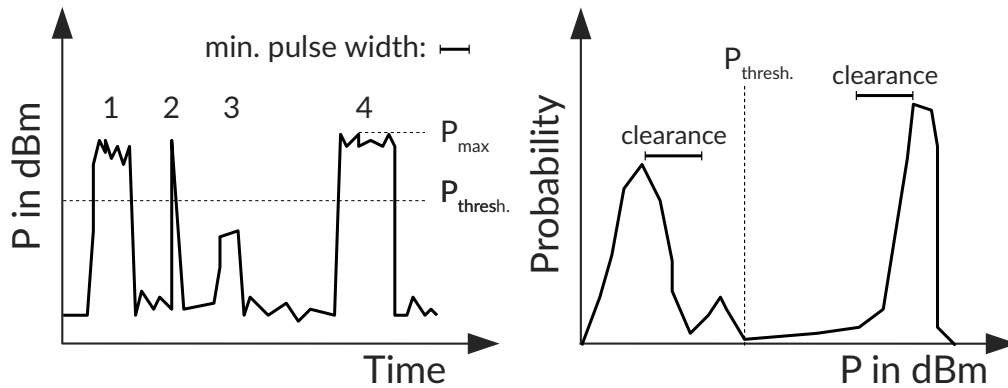


Figure 28: Principle of pulsed power measurements' threshold calculation. Pulse 2 is omitted since it is too short. Pulse 3 is omitted since it is too low. Diagram on the right is the corresponding histogram.

waveform belonging to pulses and the total number of samples in the waveform is displayed in the row labeled "Duty Cycle [%]". If no valid data is available, indicator fields will display "NaN".

By selecting the "Display x-axis as time" check box, the "Index" values of pulses and x-axis labels of the waveform graph will be displayed in seconds. If the radar subsystem is turned off, the "Pulse Count" fields will be set to zero and all other fields be cleared.

The "Arm" and "Force" buttons are duplicated from the "Scope" tab to enable waveform acquisition without switching to the "Scope" tab. All trigger settings are performed via the "Scope" tab. Pulse evaluation is repeated for every update of the power waveform and with every change in the threshold value.

#### 4.4.4 Pulse Measurements Using SCPI Commands

See Section 4.4.2 for the details of triggered measurements. Pulse detection requires the trigger system to be in DONE state.

Pulse trimming is enabled and disabled using »[:TRIGger]:RADar:TRIM <State>[,<MProbe>]«. The pulse detection method is set using »[:TRIGger]:RADar:THMethod <Method>[<MPMeter>]«, allowing for minimum/maximum-based threshold calculation, absolute threshold setting, relative threshold setting and histogram-based threshold calculation. The latter are configured using the commands »[:TRIGger]:RADar:ATHold <Threshold>[<MPMeter>]«, »[:TRIGger]:RADar:RTHold <Threshold>[<MPMeter>]« and »[:TRIGger]:RADar:CLEARance <Clearance>[<MPMeter>]«. The threshold values can be queried using »[:TRIGger]:RADar:THold:P[1]/P2/P3/ALL? [<MPMeter>]«. Radar pulse evaluation can be performed multiple times using different threshold values.

The moving average filter is configured using the »[:TRIGger]:RADar:MAVG <Count>[<MPMeter>]« command. The minimum pulse length is set by »[:TRIGger]:RADar:MINSamples <MinS>[<MPMeter>]«.

Results for the entire waveform can be queried using »[:TRIGger]:RADar[:APOWer]:P[1]/:P2/:P3/:ALL? [<MPMeter>]« for averaged pulse power values, »[:TRIGger]:RADar:COUnt:P[1]/:P2/:P3/:ALL? [<MPMeter>]« for the number of pulses, »[:TRIGger]:RADar:DUTY:P[1]/:P2/:P3/:ALL? [<MPMeter>]« for the duty cycle and »[:TRIGger]:RADar:MPOWer:P[1]/:P2/:P3/:ALL? [<MPMeter>]« for the power of the largest pulse.

Measurement results for individual pulses can be queried using »[:TRIGger]:RADar:PULses[:APOWer]:P[1]/:P2/:P3? [<MPMeter>]« for the combined start times, pulse lengths and pulse power values, times returned as seconds. Alternatively, »[:TRIGger]:RADar:PULses[:APOWer]:P[1]/:P2/:P3? [<MPMeter>]«, »[:TRIGger]:RADar:PULses:STArT:P[1]/:P2/:P3? [<MPMeter>]« and »[:TRIGger]:RADar:PULses:LENgth:P[1]/:P2/:P3? [<MPMeter>]« can be used to retrieve these values individually, with times expressed as samples.

The »[:TRIGger]:RADar:BINary? <Wave>[,<MPMeter>]« command is available for fast and computationally efficient pulse measurement readout in binary format.

#### 4.4.5 Sweep Measurements Using the GUI

The LSPM 1.0 Power Meter supports the evaluation a power waveform containing a level or frequency sweep, thus enabling sped-up measurements. For sweep measurements the acquired waveform typically contains multiple sections of equal size, where each section is characterized by constant measurement conditions. Most commonly, the frequency setting of a setup is altered in a step-wise manner in order to measure the frequency-dependent power of a setup under test. In this case, each section of the power waveform will have a different, constant frequency associated with it. Sweep measurements return the averaged power value for each section of the waveform. Dead-times at the beginning and end of each section may be applied to account for the setting characteristics of the setup under test. Due to the high sampling rate of the LSPM 1.0 Power Meter, waveform sections may be shorter than a millisecond, enabling the evaluation of more than a thousand distinct frequencies per second. This method is much faster than setting the frequencies individually and measuring power values via multiple discrete queries. See Section 4.4.1 for details regarding triggered value acquisition.

The sweep subsystem should be used in power meter operating modes 0, 2 and 3 only. One or more sections of the waveform, which align with the sweep's steps, are analyzed independently, yielding a set of averaged power values for each section. Waveform sections have a uniform length and spacing relative to each other. They are spaced in such a way to guarantee stable conditions for each section and must take into account the switching and settling characteristics of the setup under test.

In order to enable sweep evaluation within the LSPM 1.0 GUI select the "Sweep" tab and activate "Enable Sweep", as shown in Figure 29. Sweep evaluation requires the trigger system to be in DONE

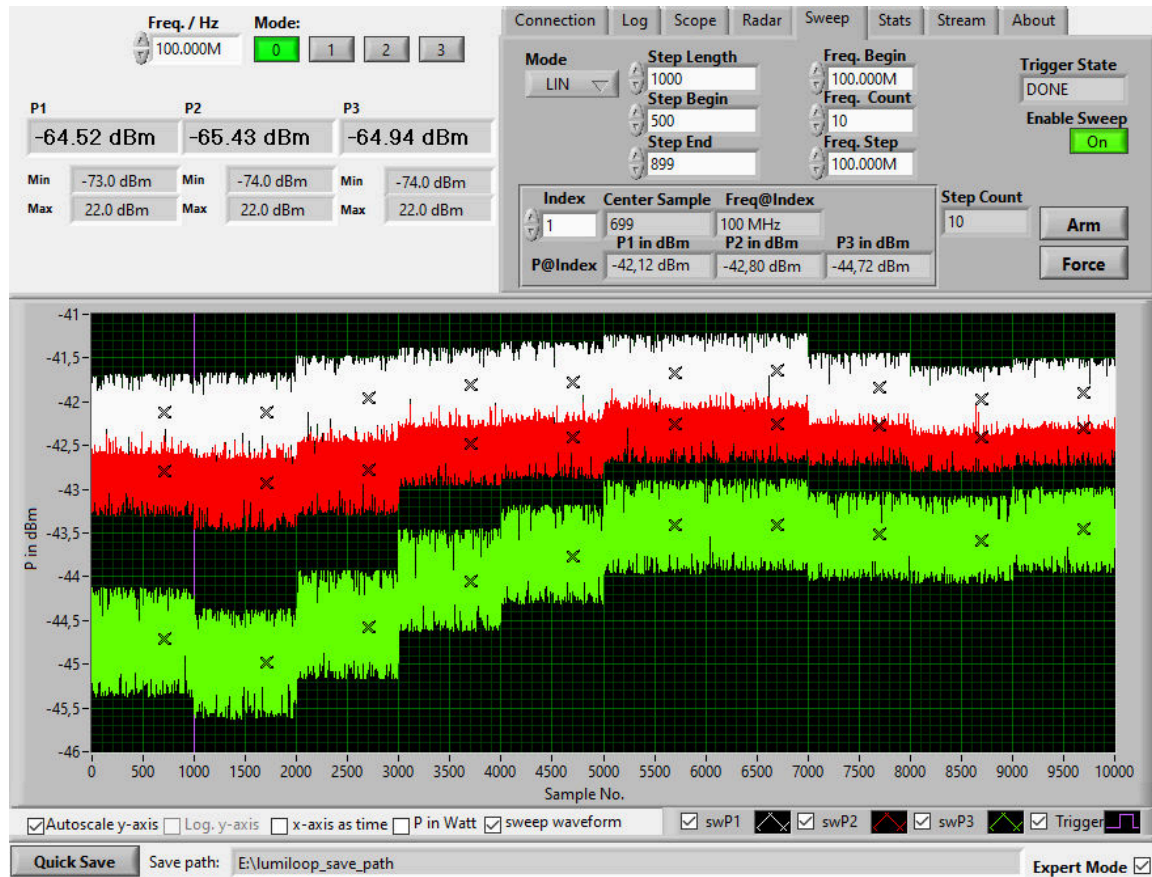


Figure 29: LSPM 1.0 GUI, Sweep tab



state. The “Trigger State” indicator field, the “Force” button and the “Arm” button are identical with the “Scope” tab. Frequency sweep re-evaluation is performed upon every change of an evaluation parameter change and when the power waveform is updated.

The timing of the sweep is configured using the fields “Step”, “Begin” and “End”. “Step” specifies the length of each sweep step. “Begin” and “End” specify the first and the last sample of each sweep step to be averaged during sweep evaluation. Samples before “Begin” and samples after “End” will be discarded. The power waveform is divided into as many sweep steps as will fit into the power waveform starting at its beginning. Note that the entire waveform can be shifted relative to the trigger signal by using of a different value for “Trigger Begin” in the “Scope” tab.

There are four frequency sweep modes selectable under “Mode”:

#### FIXED

uses the frequency set under “Freq./Hz”.

#### LIN

selects a linear frequency sweep, defined by its start frequency “Freq. Begin” in hertz, the number of frequency steps “Freq. Count” and the linear frequency increment “Freq. Step” in hertz.

#### LOG

selects a logarithmic frequency sweep, defined by its start frequency “Freq. Begin” in hertz, the number of frequency steps “Freq. Count” and the incremental factor between two steps “Freq. Step”, e.g., 1.1 for increasing the by 10% from one step to the next.

#### LIST

allows adding arbitrary frequency using the “Add” button to a list of frequency values. The list can be reset using the “Clear” button.

If there are more sweep steps than frequency steps, the frequency list will be evaluated from the beginning after reaching its last value.

The total number of sweep steps is displayed in the field “Step Count”. The selector field “Step” is used to select individual sweep steps. The first step has the index one. The center of each step relative to the trigger event is displayed in the field “Center Sample”. The corresponding frequency is shown under “Freq@Step”. Averaged channel 1, 2 and 3 power values for the selected sweep step are displayed below. Selecting an invalid sweep step will display “NaN” for all corresponding values.

By selecting the “x-axis as time” check box the waveform’s x-axis as well as the fields “Step Length”, “Step Begin”, “Step End” and “Center Sample” are expressed in seconds instead of samples.

### 4.4.6 Sweep Measurements Using SCPI Commands

See Section 4.4.2 for a detailed description of triggered measurements. To set the sweep step length and start/end index of averaging within each step the SCPI commands »[:TRIGger]:SWeep:TStep <TStep>[,<MPMeter>]« and »[:TRIGger]:SWeep:TBegin <TBegin>[,<MPMeter>]« and

»[:TRIGger]:SWEEP:TEnd <TEnd>[,<MPMeter>]« are used. The sweep mode is set via »[:TRIGger]:SWEEP:MODE <Mode>[,<MPMeter>]«, choosing »FIXED«, »LIN«, »LOG« or »LIST«. For linear and logarithmic sweeps, the start frequency, number of frequency steps and frequency increment is set via »[:TRIGger]:SWEEP:BEGIN <Freq>[,<MPMeter>]«, »[:TRIGger]:SWEEP:COUNT <Count>[,<MPMeter>]« and »[:TRIGger]:SWEEP:STEP <Step>[,<MPMeter>]«. A list of arbitrary frequencies can be created incrementally via the »[:TRIGger]:SWEEP:ARBADD <Freq>[,<MPMeter>]« command. »[:TRIGger]:SWEEP:ARBCLEAR [<MPMeter>]« is used to clear the arbitrary frequency list. In any mode, the command »[:TRIGger]:SWEEP:LIST? [<MPMeter>]« returns the list of frequencies in accordance with the selected sweep mode. The command »[:TRIGger]:SWEEP:IDX? [<MPMeter>]« returns the center sample indices of all sweep steps in the waveform. The averaged power and RSSI values for each step are queried using the commands »[:TRIGger]:SWEEP:P[1]/:P2/:P3/[:ALL]POWER:P? [<MPMeter>]« and »[:TRIGger]:SWEEP:RSSI:P[1]/:P2/:P3/[:ALL]? [<MPMeter>]«.

Frequency-corrected power waveforms can be obtained using »[:TRIGger]:SWEEP:WPOWER:P[1]/:P2/:P3/[:ALL]? [<MPMeter>]«. The command »[:TRIGger]:SWEEP:BINARY? is available for efficient data readout in binary format.

## 4.5 Power Statistics

Two types of power statistics are available for LUMILOOP LSPM 1.0 Power Meters, continuous statistics as described in Section 4.5.1 and triggered statistics based on acquired waveforms as described in Section 4.5.3. Continuous statistics evaluate all measured power values from the time that statistics collection is enabled to the time that a statistics snapshot is created. Continuous statistics are collected in the background and can be performed over arbitrary periods of time. Triggered statistics evaluate only power values of waveforms in memory. All SCPI commands of the statistics subsystem are available for both continuous and triggered collected and triggered power values.

### 4.5.1 Continuous Statistics using the GUI

All statistics functions are controlled via the “Stats” tab of the LSPM 1.0 GUI, see Figure 30. See Section 4.5 for an explanation of the operating principle.

As explained in Section 2.3, continuous statistics use a physical connection for enabling statistics collection and snapshot creation. Consequently, one master LSPM 1.0 Power Meter or LSProbe 1.2 Field Probe Computer Interface must be set up for continuous power statistics. This also applies to single probe systems where the power meter must always be configured to be the statistics master. The first enumerated power meter will be configured automatically as the statistics master, all others will be configured as statistics slaves. A different power meter may be configured as the statistics master via the “Stats Master” drop-down list.

In order to enable continuous statistics in the LSPM 1.0 GUI, select “Continuous” from the “Mode” drop-down list, this will enable additional controls for continuous statistics.

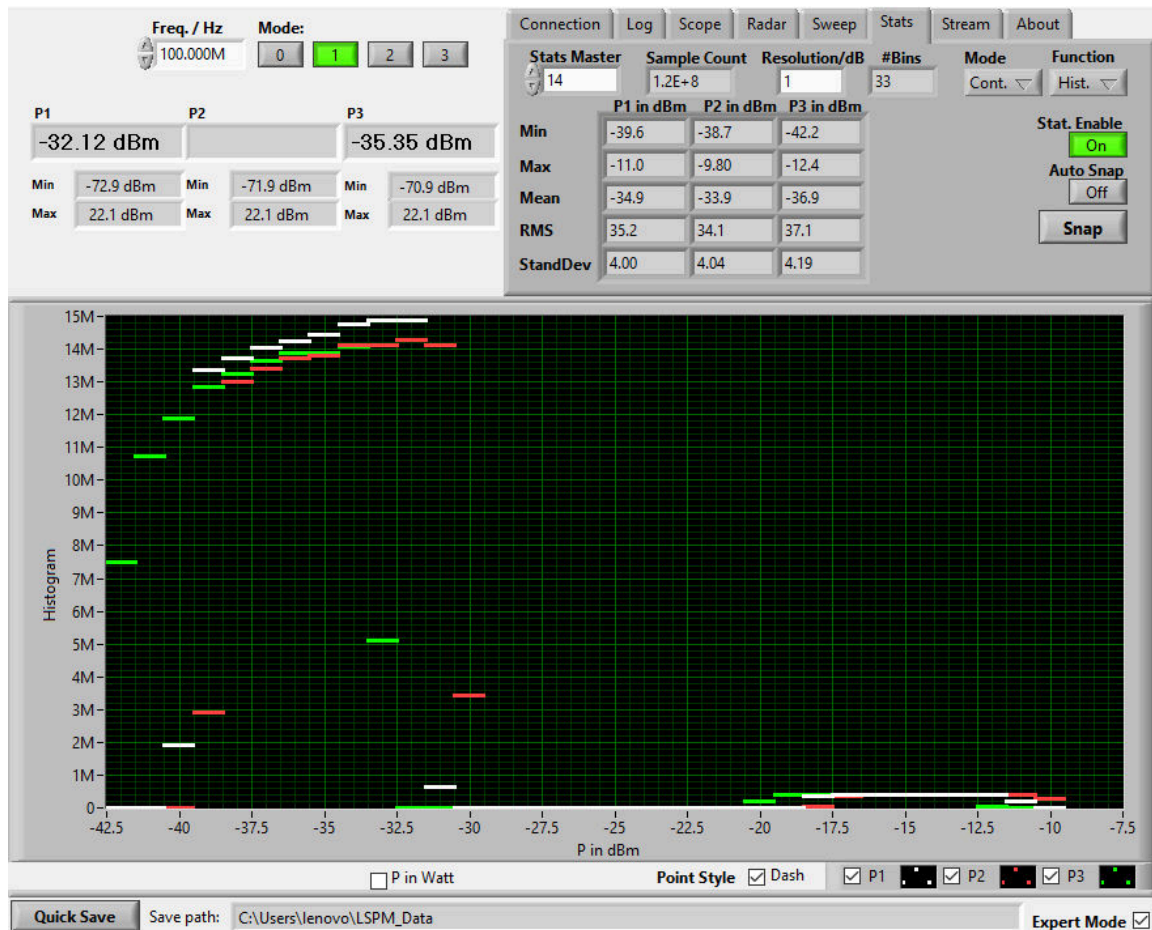


Figure 30: LSPM 1.0 GUI, Statistics tab

The “Stat. Enable” button is used to enable and disable continuous power statistics collection via the statistics master power meter. The results of continuous statistics measurements can be viewed in the form of statistics snapshots as described in Section 4.5. A statistics snapshot can be created manually by clicking on the “Snap” button. A statistics snapshot will also be created when statistics collection is disabled via the “Stat. Enable” button. Automatic statistics snapshot creation is enabled via the “Auto Snap” button, instructing the GUI to trigger a new statistics snapshot after each update of the statistics display. The total number of samples used for the most recent statistics snapshot is displayed in the “Sample Count” display field. For setups employing multiple LSPM 1.0 Power Meters and Multiprobe systems this number may vary slightly due to minimally different local clock frequencies of each power detector and/or field strength detector.

All output is based on the most recent statistics snapshot. See the SCPI command reference in Section 9.10 for a detailed description of scalar and histogram-like statistics values.

The table which is displayed in the “Stats” tab, lists all scalar statistics values for the currently selected power meter. Maximum, minimum, arithmetic mean, root mean square and standard deviation are displayed for channels 1, 2 and 3.

Histogram-like statistics are selected via the “Function” drop-down list. Available choices are histogram, discrete relative probability distribution, discrete cumulative probability distribution and discrete complementary cumulative probability distribution.

Histogram-like statistics are displayed in the main plot area. The style of the displayed graphs is adjustable via the “Point Style” checkbox below the plot area. If selected, dashes stretching each power value bin will be displayed, otherwise one point will be plotted for every power value bin.

The power value resolution in dB can be set via the “Resolution” input field, determining the size of the power value bins. Its smallest permissible value, yielding the maximum power value resolution, is 0.005 dB. The number of bins resulting from the set power value resolution and power value distribution is shown in the “#Bins” display field.

## 4.5.2 Continuous Statistics using SCPI Commands

Continuous statistics and triggered statistics are accessible using a common set of SCPI commands. An SCPI command's function is determined by the parameter “Triggered”. If set to “0” an SCPI command applies to continuous statistics, if set to “1” an SCPI command applies to triggered statistics. For most commands, the parameter “Triggered” is optional, making the SCPI command default to continuous statistics.

The master power meter for continuous statistics is set using `»:SYSTem:SERial <Value>«` followed by setting its master status to one using `»:STATistics:MAster <State>«`. The master/slave status of any LSPM 1.0 Power Meter may be queried using `»:STATistics:MAster? [<MPMeter>]«`.

Continuous power statistics collection is started by issuing `»:STATistics:ENable <State>[,<MPMeter>]«` with the parameter State set to “1” for the statistics master power meter. Statistics snapshots will be generated on receiving either `»:STATistics:SNAPshot [<Triggered>][,<MPMeter>]«` or `»:STATistics:ENable <State>[,<MPMeter>]«` with the parameter State set to “0”. The snapshot counter will be incremented by one for every new snapshot, the counter(s) can be queried using `»:STATistics:COUnt? [<MPMeter>]«`. Using this query enables snapshot synchronization since snapshot query and execution are inherently asynchronous for continuous E-field statistics. Enabling continuous statistics will reset the snapshot counter to zero. `»:STATistics:SAMPles? [<Triggered>][,<MPMeter>]«` returns the number of samples used for the most recent statistics snapshot.

Scalar statistics values can be read using the commands described in Section 9.10.12 onward. Histogram-like statistics values are returned by the commands described in Section 9.10.18 onward.

The resolution for histogram-like values is set using `»:STATistics:RESolution <Resolution>[,<MPMeter>]«`. The resulting number of bins, the offset of the bin with the smallest power value and the center power value of each bin can be queried via `»:STATistics:HISTogram:SIZE? [<Triggered>][,<MPMeter>]«`, `»:STATistics:HISTogram:OFFset? [<Triggered>][,<MPMeter>]«` and `»:STATistics:Power? [<Triggered>][,<MPMeter>]«` respectively. All statistics values are also available in binary format, see `»:STATistics:BINary? [<Triggered>][,<MPMeter>]«` for details.

### 4.5.3 Triggered Statistics using the GUI

Triggered statistics use waveform data for building the scalar and histogram-like values discussed in the previous section. See Section 4.4.1 for a description of waveform acquisition. Triggered power statistics do not rely on the physical connections required for continuous power statistics. For triggered power statistics there is no statistics master, no statistics enable function and no hardware-based snapshot feature.

In order to access triggered power statistics in the LSPM 1.0 GUI select “Triggered” from the “Mode” drop-down list, this will enable additional controls for triggered power statistics.

A statistics snapshot based on the most recently acquired waveforms can be created manually by clicking on the “Snap” button. Automatic snapshot creation is enabled via the “Auto Snap” button, making the GUI take a new triggered snapshot upon receiving a new waveform. The “Arm” and “Force” buttons are provided for ease of use and are identical in function to the buttons described in Section 4.4.

Triggered statistics data is viewed in the form of statistics snapshots as described in Section 4.5. All scalar and histogram-like values are controlled and displayed as described in the previous sections. When displaying triggered statistics data, the “Function” drop-down list may additionally be set to “Scope” for more convenient viewing of scalar statistics values and triggered waveforms at the same time.

### 4.5.4 Triggered Statistics using SCPI Commands

Triggered statistics SCPI commands require the parameter Triggered to be set to “1” for all related SCPI commands. Statistical evaluation requires a valid set of triggered waveforms, see Section 4.4.2 for details about waveform acquisition.

Snapshot histograms are generated using »:STATistics:SNAPshot [<Triggered>][,<MPMeter>]«. There is no trigger snapshot counter. Instead, triggered snapshot generation is performed synchronously making statistics values available immediately after issuing the SCPI snapshot command.

Scalar and histogram-like statistics values can be obtained using the same SCPI commands as described in Section 4.5.4 with the parameter Triggered set to “1” at all times.

## 4.6 Stream Recording

The LSPM 1.0 Power Meter supports recording data at the power meter’s full sampling rate for virtually unlimited durations of time. Recording time is only limited by the disk space available.

Stream files are stored in the path specified by the LSPM\_1.0.ini configuration file setting SAVE\_PATH and adhere to the naming convention in Section 10.1.6. File names start with an arbitrary prefix string followed by the power meter’s serial number and a unique time stamp. A new stream file or set of stream files, containing an updated time stamp, will be created every 1,000,000,000 samples.

The stream recording feature stores power data in a binary format in order to reduce both disk space and CPU load. The binary file format is detailed in Section 10.1.6. Binary stream files can be converted into CSV files using the “Bin2csv” conversion tool described in Section 10.1.7.

### 4.6.1 Stream Recording Using the GUI

To configure and perform stream recordings select the “Stream” tab of the LSPM 1.0 GUI, as depicted in Figure 31.

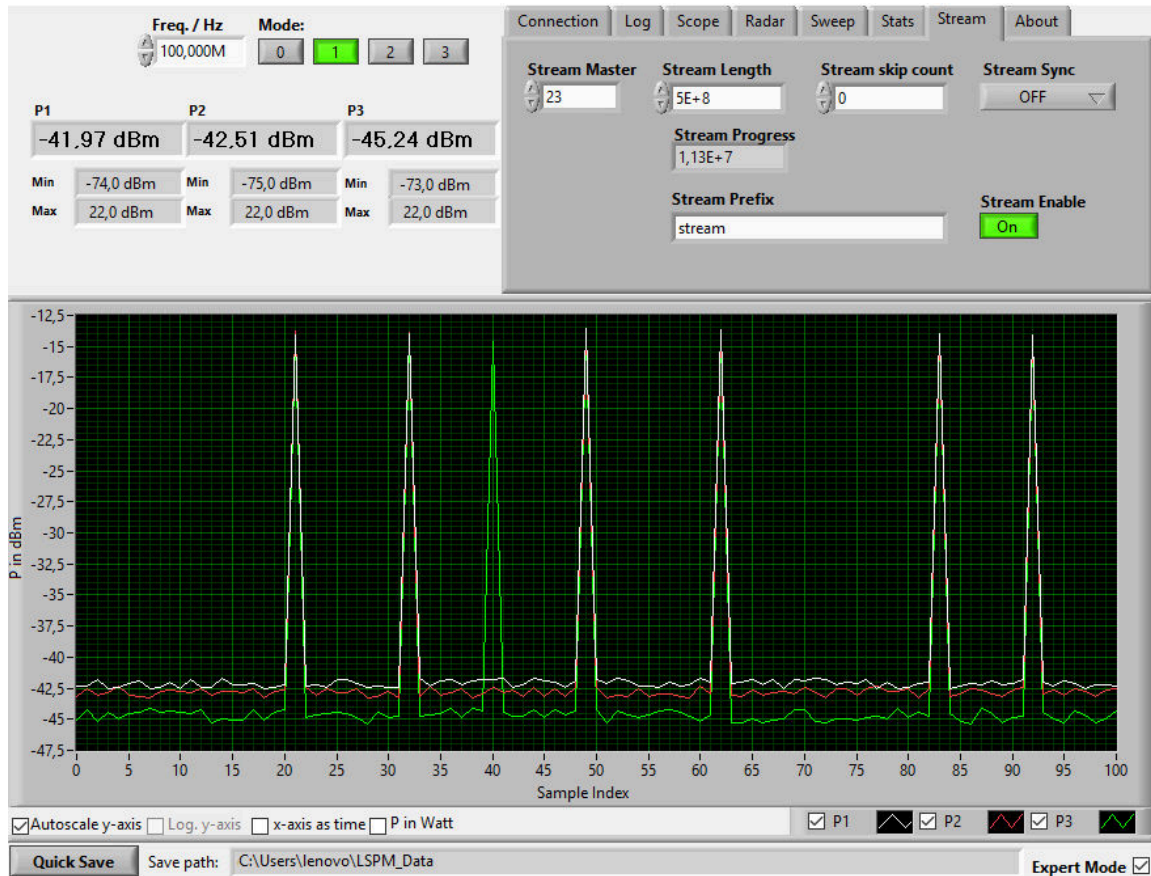


Figure 31: LSPM 1.0 GUI, Stream tab for stream recording

If synchronization between multiple LSPM 1.0 Power Meters in combination with LSProbe 1.2 Computer Interfaces is required, make sure to connect the appropriate signal lines. Set the synchronization power meter using the “Stream Master” drop-down list, the setting defaults to the first enumerated LSPM 1.0 Power Meter.

The number of samples to be recorded and the number of power values to be skipped after every recorded sample are set via the “StreamLength” and “Stream Skip Count” numeric input fields. Make sure to enable the “ALL” button when using multiple LSPM 1.0 Power Meters.

The synchronization source can be selected via the “Stream Sync” drop-down list. The stream file

prefix is set via the “Stream Prefix” input field. The number of samples recorded during a stream recording session is displayed in the “StreamProgress” field. If the “Stream Length” input field is set to a non-zero value, pressing the “Stream Enable” button will initiate the recording of the set number of samples. A stream recording may be terminated via the “Stream Enable” button before the set number of samples has been reached. If the “Stream Length” input field is set to “0”, stream recording must be terminated manually via the “Stream Enable” button. Manual termination of a stream recording can lead to stream files containing a disparate number of samples.

#### 4.6.2 Stream Recording Using SCPI Commands

Stream synchronization can be configured using the `:STReam:SYNC <Sync>[,<MPMeter>]` SCPI command. When synchronization is disabled, the stream master power meter can be changed by setting the former master power meter to slave, selecting the new power meter and setting it to stream master using the `:STReam:MAster <State>` and `:SYSTem:SERial <Value>` commands.

To set the maximum number of samples to be recorded, use the `:STReam:LENgth <Length>[,<MPMeter>]` command. If set to a non-zero number, stream recording will terminate automatically after reaching the maximum number of samples.

The stream data rate can be reduced using the `:STReam:SKIp? [<MPMeter>]` command, specifying the number of skipped samples following each stored sample. The `:STReam:PREfix <String>[,<MPMeter>]` command can be used to change the stream files' prefix string.

Stream recording is initiated and terminated using the `:STReam:ENable <State>[,<MPMeter>]`. The progress of stream recording can be monitored using the `:STReam:PROgress? [<MPMeter>]` command.

### 4.7 Saving Log Files using the GUI

The “Log” tab contains controls for the file name prefix of LSPM 1.0 GUI log files. The prefix applies to both single-shot and continuous logging, see Figure 32. Separate log files are created for the “Log”, “Scope”, “Radar”, “Sweep” and “Statistics” tabs. Single-shot log files are created using the “Quick Save” button in the lower left corner of the GUI window. The “Quick Save” button will create a new log file for the currently active tab and power meter of the LSPM 1.0 GUI. Continuous logging is enabled in the “Log” tab using the buttons next to the file name prefixes. It saves the respective data for all enumerated power meters to disk. For scope and sweep logging every new set of data will be written to a separate file. For all other log data continuous logging will append new sets of data to the respective log file as long as continuous logging is enabled.

Log files are stored in the path specified by the configuration file setting `SAVE_PATH`. The path is displayed in the lower right corner of the GUI window. Log file names consist of the file name prefix extended by the power meter's serial number and a timestamp as detailed in Section 10.

Logging of RSSI values, i.e., raw ADC sample values, for all three channels can be enabled by selecting the “include RSSI” check box. The feature is available for “Basic Log”, “Scope Log” and “Sweep Log” files.

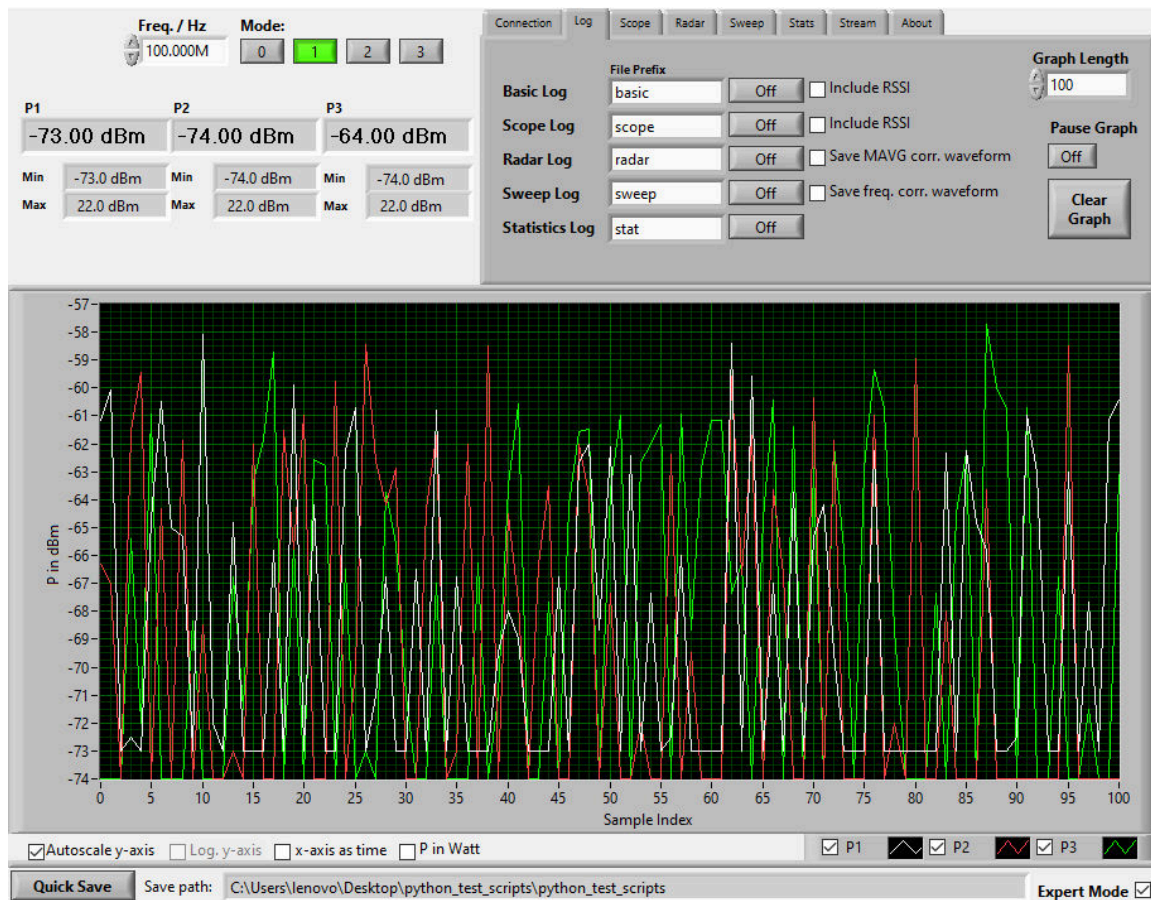


Figure 32: LSPM 1.0 GUI, Data Logger tab

## 5 Third Party EMC Software

This section describes the setup of the LSPM 1.0 Power Meter in third party EMC test automation software. Third party support-files are installed in separate directories of the lib sub-directory of the LSPM 1.0 installation path.

Before running any third party EMC software follow the hardware setup instructions detailed in Section 4.1 on page 24 and start the TCP server as described in Section 4.2.1 on page 24.

Starting the TCP server, setting the operating mode and starting third party software can be automated by the “LSPM-EMC-Starscript.pl” Perl script in the bin sub-directory of the LSPM 1.0 installation path. “Strawberry Perl” has to be installed in order to use the start script. In Line 28 and 29 the paths for the LSPM TCP-Server and the third party software must be set appropriately. The



measurement mode is defined in line 30, with a default value of “1”. After closing the third party software, the TCP-Server will be terminated as well.

The LSPM 1.0 GUI is not required when using third party EMC software. A notable exception is setting the operating mode. However, the LSPM 1.0 GUI may be run in parallel as long as it is used solely for monitoring and no settings are changed using the GUI, i.e., the user must not modify any LSPM 1.0 GUI controls, e.g., mode, frequency, low-pass filter frequency, etc. The LSPM 1.0 GUI is designed in such a way that it will not apply any settings on its own.

## 5.1 EMC32

LUMILOOP recommends using the most recent version of the R&S EMC32 measurement software since EMC32 integration is always tested against the most recent EMC32 release. The LSPM 1.0 Power Meter is supported by EMC32 version 10.3 and later.

Copy all device configuration files ending in `DeviceConfiguration` from the `EMC32-10.3` sub-directory of the LSPM installation path's `lib` directory to the EMC32 program data path's `Configuration\Power Meters` sub-directoy, a typical location is `C:\ProgramData\EMC32\Configuration\Power Meters`.

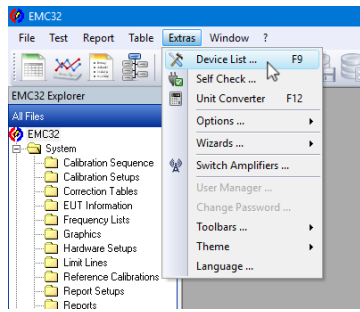
### 5.1.1 CW fields

To set up the LSPM 1.0 Power Meter in EMC32 run the TCP server, then start EMC32.

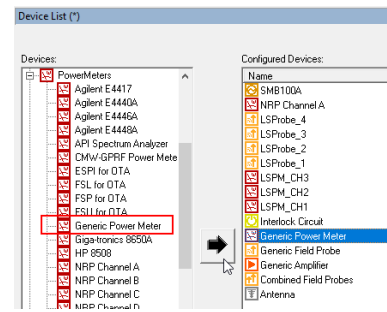
Open the EMC32 Device List via “Extras→Device List” in the menu bar as shown in Figure 33 (a). In the “Device List” window select “Generic Power Meter” from the “Devices:” list's “Power Meters” category and create a new “Configured Device” by clicking on the right-pointing arrow in the center as shown in Figure 33 (b). This will create a new entry named “Generic Power Meter”. Optionally add a second and a third “Generic Power Meter” entry by clicking the right-pointing arrow an additionally one or two times. The first power meter will be used for channel 1 power value queries, the second for channel 2 and the third for channel 3. Rename the first “Generic Power Meter” entry to “LSPM\_Ch1” and the optional second and third entries to either “LSPM\_Ch2” or “LSPM\_Ch3” via “right-click→Rename” as shown in Figure 33(c).

Use “right-click→Edit” to open the Generic Power Meter's settings. In the “General” tab shown in Figure 34 (a) edit “VISA Device Identifier” to configure the IP address and TCP port of the LSPM 1.0 TCP Server. The identifier string has the generalized format “TCPIP0::<IP address>::<TCP Port>::SOCKET”. Usually, the LSPM 1.0 TCP Server is run on the same computer and listening to the default TCP port 10,001. Consequently, the default identifier string is “TCPIP0::127.0.0.1::10001::SOCKET”. All other settings in the “General” tab are optional and may be left unchanged.

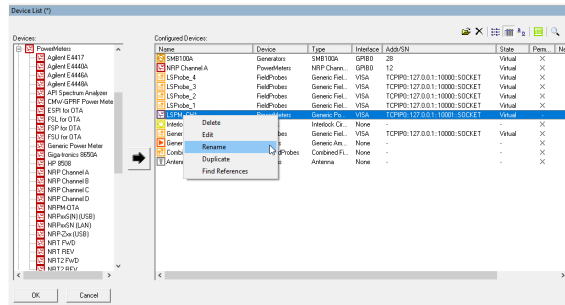
Select the “Properties” tab shown in Figure 34(b) through (d), edit all parameters as detailed in Table 2. Select the appropriate file for “Configuration File” located in `...\ProgramData\EMC32\Configuration\Power Meters`. The low pass filter value in the



(a) Opening the Device List



(b) Adding Generic Power Meter(s)



(c) Renaming Power Meters

Figure 33: Adding the LSPM 1.0 Power Meter in EMC32

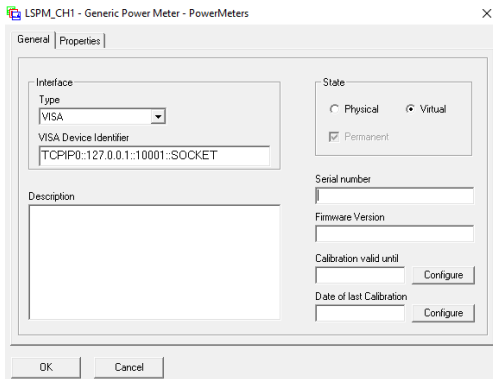
.DeviceConfiguration files can be adjusted by the user to accommodate longer or shorter settling times.

Table 2: EMC32 Property tab values for Channel 1 Power Meter

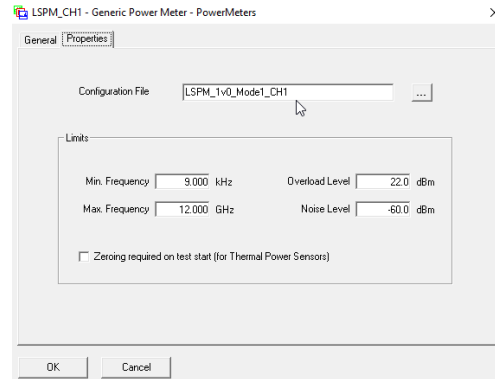
Setting	Channel 1
Min. Frequency	9.000 kHz
Max. Frequency	12.000 GHz
Overload Level	22 dBm
Noise Level	-60 dBm
Zeroing required on test start	unchecked
Configuration File	LSPM_1v0_Mode1_CH1.Device-Configuration

To finish the setup restart EMC32 and open the EMC32 Device List via “Extras→Device List”. Use “right-click→Edit” for the “LSPM 1.0” entry appropriate for the measurement task and change “State” from “Virtual” to “Physical”. This will prompt EMC32 to connect to the LSPM 1.0 TCP Server. After establishing the correct mode the “Serial number” value turns blue.

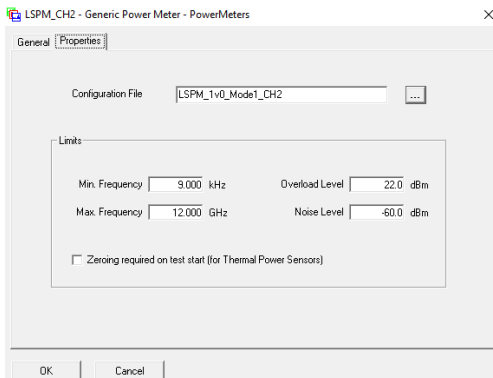
For multiple LSPM 1.0 Power Meters, the configuration files have to be adapted so that a specific LSPM gets selected by a EMC32 client connection for communication. The first “[Initialize]” block encompassing line 49 to 53 has to be commented. The second “[Initialize]” block encompassing line



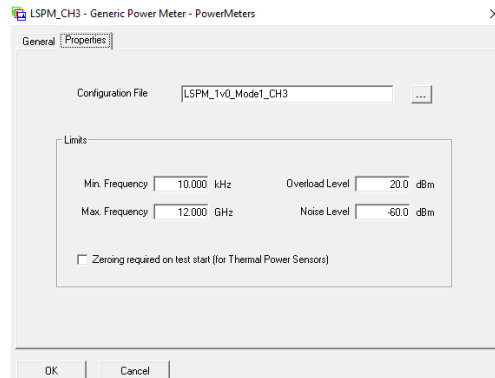
(a) General Settings



(b) Channel 1 Power Meter Properties



(c) Channel 2 Power Meter Properties



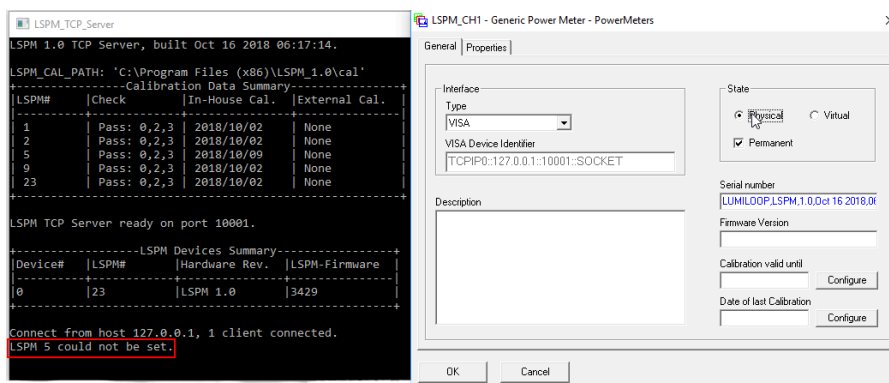
(d) Channel 3 Power Meter Properties

Figure 34: Configuring LSPM 1.0 Power Meters in EMC32

55 to 61 must not be commented out. An additional SCPI command is sent to the LSPM 1.0 TCP Server setting the active power meter. The specific serial number is set in line 59. After changing “State” from “Virtual” to “Physical” check the TCP-Server window. If the setting of the active LSPM failed, an error message will be printed to the TCP server window, as depicted in Figure 35 (b). If

```
LSPM_1v0_Mode1...ceConfiguration
33 ;Parit 0=None, 1=Odd, 2=Even
34 Baud=9600
35 Data=8
36 Stop=1
37 Parity=0
38
39 ;Visa strings can have leading characters:
40 ;---@n wait n milliseconds after this command
41
42 [Identify]
43 ;Identification Query1
44 Count = 1
45 GpibLine1=*IDN?
46 GpibResponse1=LUMILOOP
47
48 ;COMMENT IF SECOND INITIALIZE IS TO BE UTILIZED FOR SETTING SPECIFIC LSPM SERIAL NUMBER
49 ;[Initialize]
50 ;Reset on system start, count may be > 1
51 ;Count=2
52 ;GpibLine1=@50@SYST:MOD 1
53 ;GpibLine2=MEAS:P:LPF 150
54
55 ;UNCOMMENT IF SPECIFIC LSPM IS TO BE SET, ADJUST SERIAL NUMBER
56 ;[Initialize]
57 ;Reset on system start, count may be > 1
58 ;Count=3
59 GpibLine1=SYST:SER 5
60 GpibLine2=SYST:MOD 1
61 GpibLine3=MEAS:P:LPF 150
62
63 [channel]
64 ;Command for setting the measurement channel (only for multi-channel devices)
65 ;count may be > 1
66 Count=0
```

(a) Modification of LSPM\_1v0\_Mode1\_CH1.Device-Configuration file



(b) Check LSPM 1.0 TCP Server for error messages

Figure 35: Adapting LSPM device configuration files for selecting specific LSPM

## 5.2 BAT-EMC

The BAT-EMC software supports CW and pulsed power measurements. BAT-EMC requires a DLL file, make sure that “FieldP\_Lumliloop\_LS12.dll” is present in the BAT-EMC directory “...\BAT-EMC\BAT-EMS”.

Please import the provided "One-input Power Meter" equipment models listed in Table 3 by right-clicking on “One-input Power Meter” inside the “Equipment” tree structure as shown in Figure 36.

If the IP address and/or the port number of the LSPM connection differ from the default values of

Table 3: BAT-EMC equipment model files for CW and pulsed powers

CW	Pulsed
LSPM_1.0_CW1.xml	LSPM_1.0_Pulse1.xml
LSPM_1.0_CW2.xml	LSPM_1.0_Pulse2.xml
LSPM_1.0_CW3.xml	LSPM_1.0_Pulse3.xml

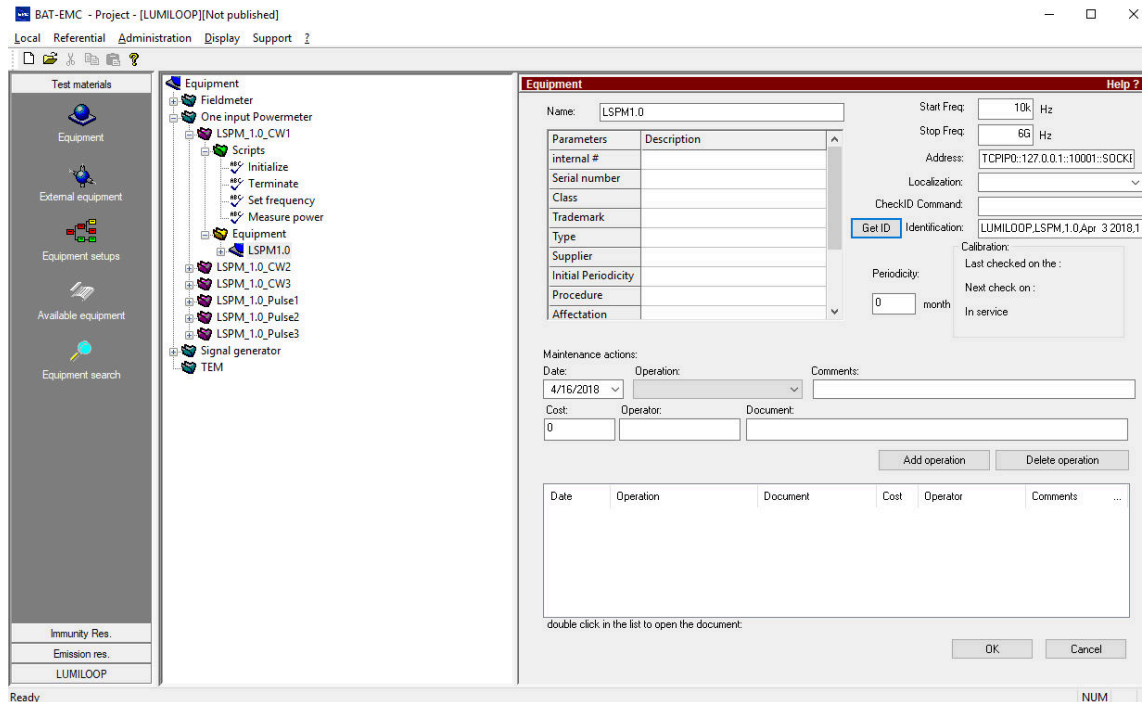


Figure 36: BAT-EMC Equipment editor, network configuration

localhost and port 10,001 go to the “Equipment” subsection and change the “Address” input field appropriately, see also Figure 36.

BAT-EMC requires the LSPM 1.0 Power Meter to be configured before performing measurements. Before starting BAT-EMC, start the LSPM TCP server and optionally LSPM 1.0 GUI to set a mode other than 1 as described in Sections 4.2.1 through 4.2.5.

## 5.2.1 CW Power Measurements

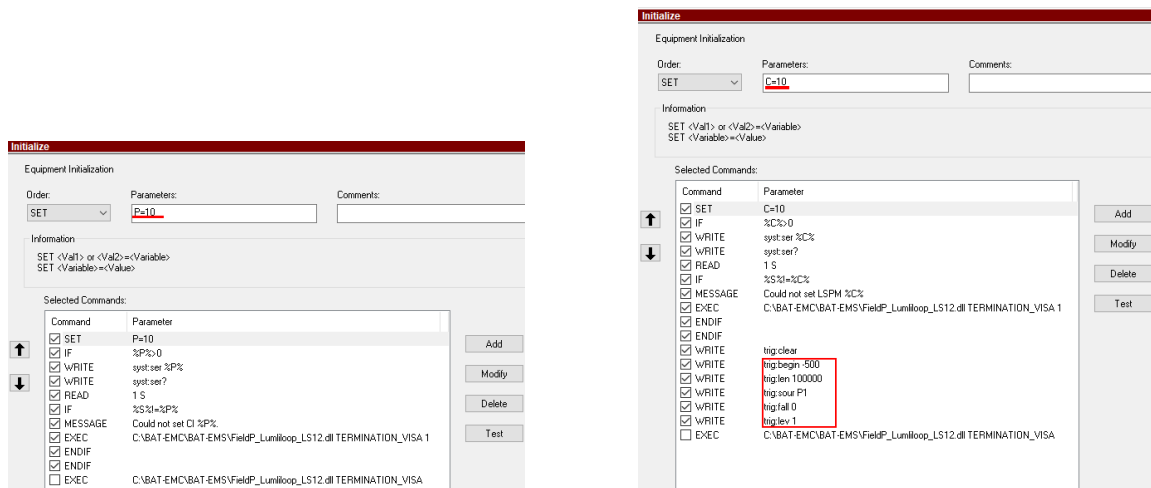
The “One-input Power Meter” model “LSPM\_1.0\_CW1” handles all communication including setting/checking the operating frequency as well as retrieving channel 1, channel 2 and channel 3 power values. The power meter returns the channel 1 power value and stores the power values of channel 2 and channel 3 in global variables named “AB” and “AC”. The two “One-input Power Meter” models “LSPM\_1.0\_CW[2/3]” retrieve the global variables and return the respective channels’ power values.

When measuring channel 1, channel 2 and channel 3 power values, the “LSPM\_1.0\_CW1” Power Meter model must be called first.

### 5.2.2 Pulsed Power Measurements

The “One-input Power Meter” model “LSPM\_1.0\_Pulse1” handles all communication including setting/checking the operating frequency as well as retrieving channel 1, channel 2 and channel 3 power values. Additionally, the “One-input Power Meter” model includes commands for trigger subsystem configuration, trigger detection and pulse property retrieval. The “One-input Power Meter” returns the channel 1 power value and stores the power values of channel 2 and 3 using global variables named “AB” and “AC”. The two “One-input Power Meter” models “LSPM\_1.0\_Pulse[2/3]” read these global variables and return the respective channels’ power values. When measuring channel 1, channel 2 and channel 3 power values, the “LSPM\_1.0\_Pulse1” “One-input Power Meter” model must be called first.

The trigger subsystem’s configuration can be modified via the “One-input Power Meter” model’s “Initialize” script, the relevant commands are shown inside the red frame in Figure 37 (b). See Section 4.4.2 for more information about the trigger subsystem’s SCPI commands.



(a) LSPM\_1.0\_CW1 Initialize script

(b) LSPM\_1.0\_Pulse1 Initialize script

Figure 37: Configuring the Initialize scripts for LSPM 1.0 CW and pulse measurements in BAT-EMC

Figure 38 shows the “Measure level” script of the “LSPM\_1.0\_Pulse” “One-input Power Meter” model. Every call to the “Measure level” script arms the trigger subsystem, waits for the trigger subsystem to acquire a waveform, checks the number of pulses and queries the averaged pulsed power values.

The default setting of the “LSPM\_1.0\_Pulse” “One-input Power Meter” model is suitable for the GMW-3097 standard. It will record 400,000 samples (see the trigger length setting in Figure 37 (b)), verify that there are a total of 50 pulses (see the topmost red frame in Figure 38) and retrieve the

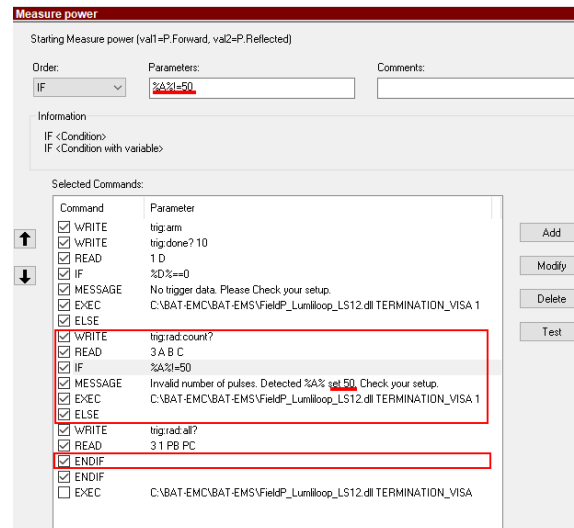


Figure 38: Addressing a specific power meter using BAT-EMC

averaged pulse power values for channel 1, 2 and 3. The trigger source, trigger length and expected pulse count can be modified by editing the respective scripts. Pulse count checking can be disabled by unchecking all boxes in the two lower red frames in Figure 38.

If multiple LSPM 1.0 Power Meters are attached to the host computer, set the variable P in Figure 37 to a value other than zero. Doing so will enable the setting and verification of the power meter's serial number. The Figure 37 demonstrates setting the LSPM 1.0 Power Meter's serial number to 10, the serial number must be changed to match the desired power meter's serial number. The variable is available for both CW and pulsed "One-input Power Meter" models.

For monitoring multiple power meters in parallel, create a copy of all "One-input Power Meter" models by right-clicking on the "One-input Power Meter" models and choosing "duplicate". Adjust all model names and power meter serial number variable settings appropriately.

## 6 Virtual Power Meters

The LSPM 1.0 TCP Server is capable of instantiating virtual power meters including the simulation of arbitrary power patterns.

Virtual power meters can replace physical meters and signal generators during measurement setup, feature demonstration, third party EMC software development and off-line signal analysis, including setups using multiple power meters.

The following virtual power meter properties can be can be configured:

- virtual power meter serial number,
- virtual power meter channel 1, 2 and 3 power value, see pattern description below.

Virtual power meter serial numbers must be unique, i.e., must not duplicate the serial numbers of any physical or virtual devices.

Power value patterns for channel 1, 2 and 3 are simulated by summing up the RSSI patterns of the following types:

CW (continuous wave)

Generates constant RSSI values.

Noise

Generates random RSSI values with a configurable maximum amplitude whose time-average is zero.

Pulse

Generates periodically pulsed RSSI values whose OFF-value is zero and whose ON-value is configurable for each channel. The pattern's ON-time and period are configurable and apply to all channels.

List

Generates a sequence of arbitrary RSSI values, optionally calculated from a list of power values, using the present mode and operating frequency. The sequence is repeated indefinitely.

Virtual power meters support all operating modes. Triggering is supported with the exception of external trigger input and output. Virtual power meters support the collection of continuous statistics. By default virtual power meters are configured as statistics slaves. Statistics snapshots are recorded when the statistics master power meters triggers a snapshot, the master power meter may be physical or virtual. Note that a virtual power meter, that is configured as a statistics master, cannot control physical power meters as statistics slaves.

## 6.1 Controlling Virtual Power Meters Using the GUI

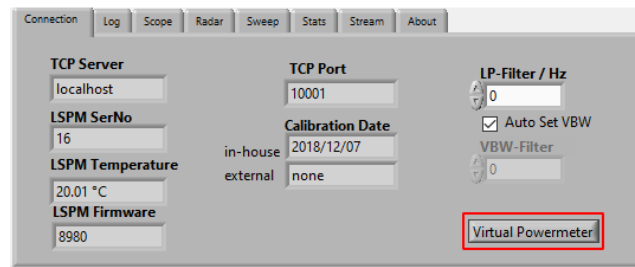
To open the "Virtual Power Meter Control Panel" press the "Virtual Power Meter" button in the "Connection" tab as shown in Figure 39.

Click the "Add" button to add a new virtual power meter with the specified serial number. The "Virtual PM#" ring lists all connected virtual power meters by their serial numbers. Click the "Remove" button to disconnect the currently selected power meter.

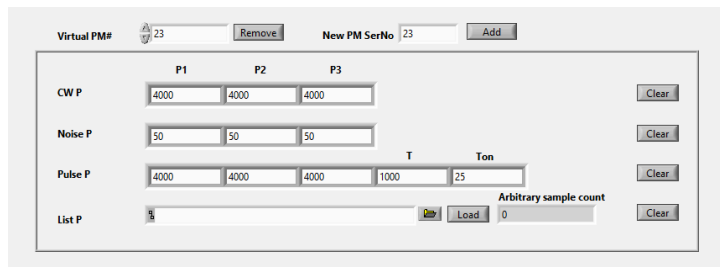
The selected virtual power meter can be configured using the controls in the frame below. The power pattern of channel 1, 2 and 3 are set using the controls inside the frame below. Settings take effect as soon as they are entered. The "Clear" buttons can be used to reset the corresponding settings to their default values.

Power value lists can be loaded from Scope-Log CSV files whose format is described in Section 10.1.2 on page 127. The file name is selected via the "List P" control. Clicking "Load" will append all power values to the virtual power meter's list buffer. The list can be emptied using the corresponding "Clear" button.





(a) LSPM 1.0 GUI



(b) Virtual Power Meters Control Panel

Figure 39: Opening the Virtual Power Meters Control Panel of the LSPM 1.0 GUI

## 6.2 Controlling Virtual Power Meters Using SCPI Commands

Virtual power meters are added using »:VIRTual:CONnect [<SER>]«. »:VIRTual:SERial?« lists the serial numbers of all virtual power meters. The currently selected virtual power meter can be removed using »:VIRTual:DISConnect«. The virtual power meter's serial number and parametric power patterns are set/queried using the following SCPI commands:

- »:VIRTual:CW <RSSI1>,<RSSI2>,<RSSI3>«/»:VIRTual:CW?«,
- »:VIRTual:NOIse <NOISE1>,<NOISE2>,<NOISE3>«/»:VIRTual:NOIse?« and
- »:VIRTual:PULse [<RSSI1>],[<RSSI2>],[<RSSI3>],[<T>],[<Ton>]«/»:VIRTual:PULse?«.

Arbitrary power values are appended to the virtual power meter's list using »:VIRTual:LIST <RSSI1\_1>,<RSSI2\_1>,<RSSI3\_1>[,...,<RSSI2\_N>,<RSSI3\_N>]« for arbitrary RSSI, »:VIRTual:PLIST <P1\_1>,<P2\_1>,<P3\_1>[,...,<P1\_N>,<P2\_N>,<P3\_N>]« for arbitrary power values. The complete list of power values is queried using »:VIRTual:LIST?«. »:VIRTual:LCnt?« returns the number of samples in the list. »:VIRTual:LCLear« clears the list of values.

## 7 Power Meter Calibration

The LSPM 1.0 Power Meter uses in-house and external calibration files for calculating accurate power values based on the ADC values generated by each of the power meter's channels, see Figure 4 on page 14 for a principle block diagram of the power meter.

## 7.1 In-house Calibration

In-house linearity and frequency compensation files, whose format is detailed in Section 10.4.1, contain the relationship between detector input power and returned ADC value. They cover all channels, modes and frequencies. The LSPM 1.0 TCP Server interpolates between these recorded data points to obtain a highly linear measurement results for each channel.

Table 4 lists the default in-house calibration frequencies. The input power is adjusted to cover the input level of the detector between the its noise floor and level of saturation in 1 dB steps.

Table 4: LSPM 1.0 default in-house calibration frequencies

Frequency range	Calibration Frequencies	Modes
10 kHz to 20 MHz	10 kHz, 15 kHz, 22 kHz, 33 kHz, 47 kHz, 68 kHz, 100 kHz, 150 kHz, 220 kHz, 330 kHz, 470 kHz, 680 kHz, 1 MHz, 1.5 MHz, 2.2 MHz, 3.3 MHz, 4.7 MHz, 6.8 MHz, 10 MHz, 20 MHz	3
30 MHz to 400 MHz	30 MHz to 100 MHz in 10 MHz steps, 100 MHz to 400 MHz in 20 MHz steps	0, 3
420 MHz to 8.2 GHz	420 MHz to 1000 MHz in 20 MHz steps, 1000 MHz to 8200 MHz in 50 MHz steps	0
8.25 GHz to 12 GHz	8.25 GHz to 12 GHz in 50 MHz steps	0

In-house calibration records the digitized output voltage of both logarithmic RF detector circuits covering all channels modes, frequencies and signal levels. As detailed in Table 4 calibration is performed in modes 0 and 3 only.

The reference power value for the in-house power value calibration is established using a transfer standard, i.e., a reference power meter. This represents no accredited procedure. Therefore, LSPM supports the inclusion of externally generated, accredited calibration data.

## 7.2 External Calibration

External power calibration files, whose format is detailed in Section 10.4.2, contain the power value correction factors in Decibels for a constant calibration power and desired number of frequencies. Typically, calibration is performed at a constant calibration power of 0 dBm. If required, external calibration can be performed at multiple constant calibration power values.

External calibration of LSPM 1.0 Power Meters must be performed in mode 0 and mode 3 to cover the entire frequency range, other modes are not permissible during calibration. Ahead of external calibration, two SCPI commands must be sent to the LSPM 1.0 TCP Server, requiring the September 2020 release or later:

```
:CALibration:EXternal 0
```

disables the application of external calibration data. Thus ensuring that (re-)calibration is performed relative to the power meter's in-house calibration data.

`:CALibration:LOGging 1`

enables automatic logging of power values, operating modes, etc. by the TCP server. For each measurement that is initiated by the lab's calibration software, one line will be added to the log file. Log files are stored in the directory defined by the `LSPM_1.0.ini` configuration file setting `CAL_PATH`. By default, log files follow the `extCalLog_X_YYYYMMDD_hhddss.csv` naming scheme, where `X` denotes the serial number, and `YYYYMMDD_hhddss` denotes the creation date and time of the log file.

External power calibration file names take either the format `snX.csv` or `snX_S.csv`, where `S` denotes an arbitrary string, for distinguishing different reference power values and power-meter modes. `CallImport` will generate files adhering to the format `snX_P_mM.csv`, where `P` denotes the nominal calibration power and `M` denotes the power-meter mode. For example, `sn1_0_m0.csv` and `sn1_0_m3.csv` are the calibration files for power meter serial number 1, at 0 dBm in mode 0 and mode 3. The LSPM 1.0 TCP Server will parse all external calibration files according to their contents, ignoring the individual file names.

Though it is recommended to use the calibration frequencies detailed in Table 4 for external power calibration, it is permissible to use a different set of frequencies. The LSPM 1.0 TCP Server will interpolate the calibration factors as needed.

### 7.3 Automated Calibration Data Import

Figure 40 shows the flow of data during the calibration process. Traditionally, the calibration laboratory generates calibration result files in a format of their choosing, containing at least the calibration power and the indicated power. Calibration certificates are provided to customers based on this data. Customers are required to incorporate the appropriate correction factors into their setups on their own – this step is known to be both cumbersome and especially prone to human error.

As shown in Figure 40, LUMILOOP has enhanced the calibration data flow by supplying the calibration data import tool `CallImport` as part of the LSPM 1.0 software. `CallImport` automatically generates checksum-protected CSV files for the LSPM 1.0 TCP Server. These external calibration files contain the correction factors for each frequency expressed in decibel. Multiple external calibration files can be generated to accommodate different power meter modes and nominal calibration power values.

`CallImport` supports the automatic generation of checksum-protected external calibration files for the laboratories and file formats listed in Table 5. A generic CSV file format can be used as a fallback, see Section 10.3 of the LSPM 1.0 Power Meter User's Manual for a description of the file format.

Table 5: *File formats supported by CallImport*

Laboratory	Format	Cal. Log File Support
Generic	CSV	yes
AMETEK CTS Europe	PDF	yes

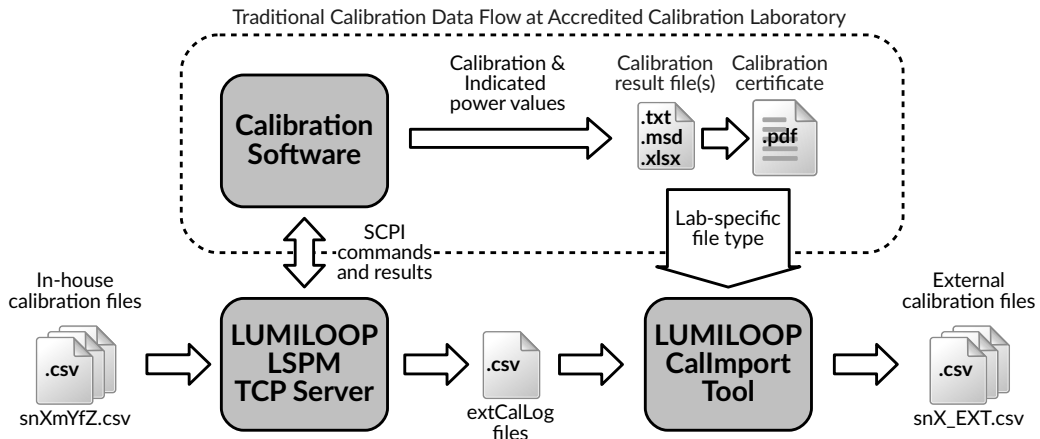


Figure 40: Enhanced calibration data flow

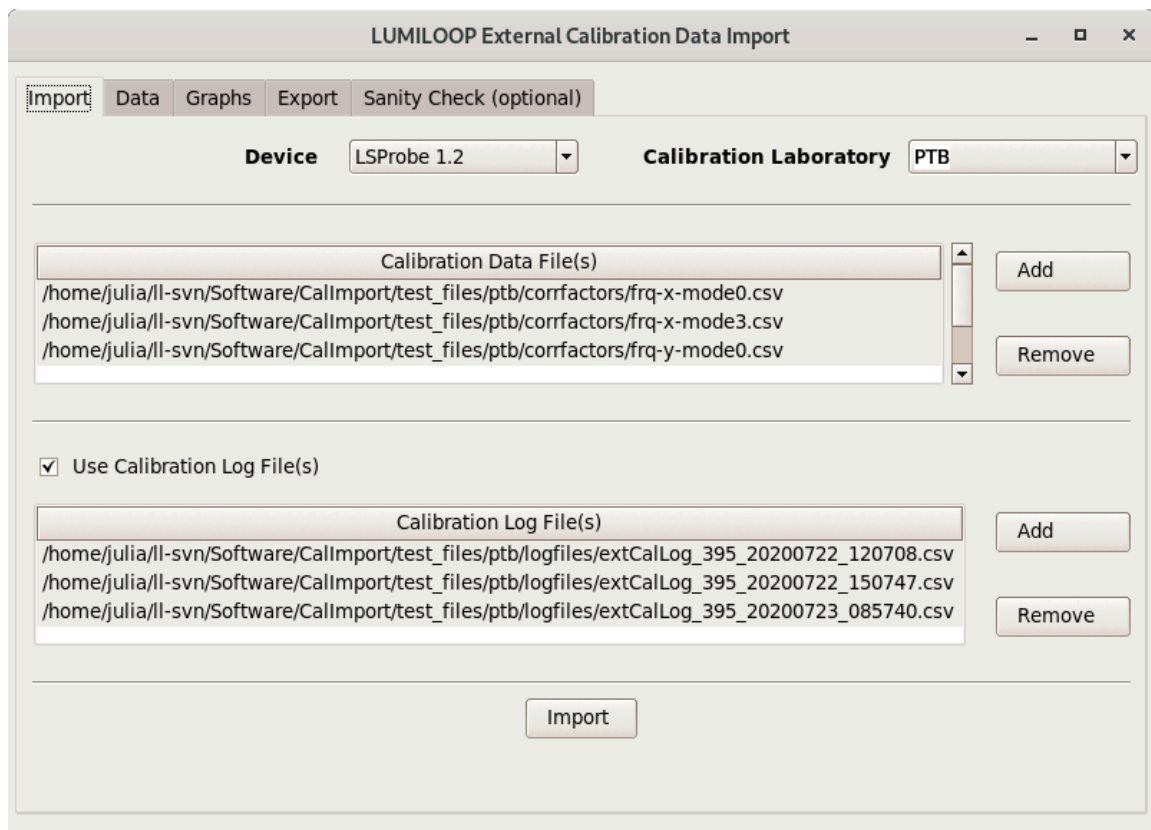


Figure 41: Using the calibration data import tool CallImport

Using the LSPM 1.0 TCP Server's calibration log files with CallImport is strongly recommended. Doing so will enable automatic serial number and time stamp detection. CallImport will also perform data sanity checks that avoid a number of common data handling errors. As shown in Figure 41, using

CallImport is straightforward:

1. Run CallImport
2. Select the appropriate calibration laboratory
3. Choose calibration data files and calibration log files
4. Import all calibration data
5. Optionally, review the correction factors and graphs
6. Export the external calibration file(s) to the appropriate `snX` folder in the `CAL_PATH` directory specified in the configuration file `LSPM_1.0.ini`

External calibration files are stored in the same folder as in-house calibration files. The LSPM 1.0 TCP Server will read and verify all available calibration data upon start-up. Therefore, after adding or modifying calibration data, (re-)start the LSPM 1.0 TCP Server and check the calibration data summary table for errors.

## 8 SCPI Communication Basics

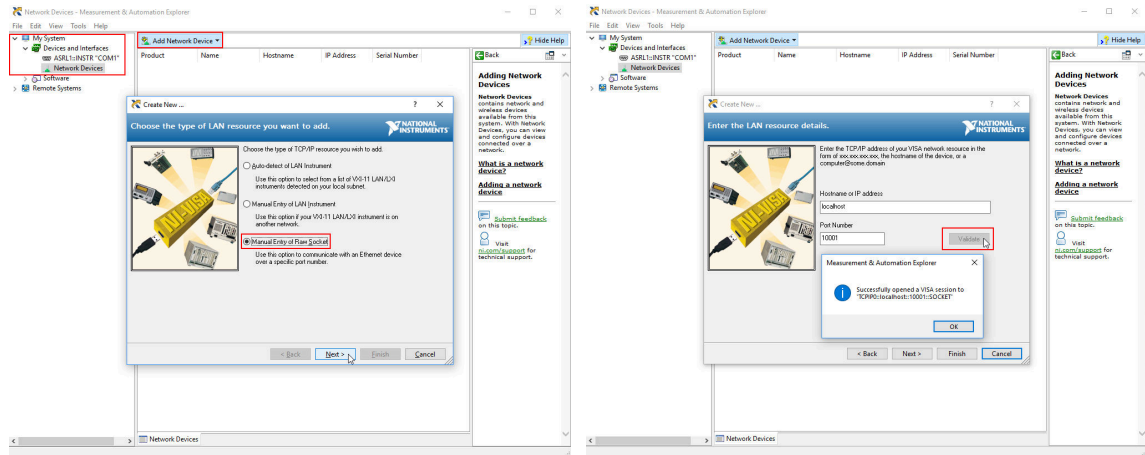
The LSPM 1.0 TCP Server provides a convenient text command-based interface to power meter measurement data, it supports up to 32 concurrent TCP/IP client connections. All commands sent to the TCP server are ASCII text commands which terminated by a newline (`\n`), carriage return (`\r`) or semicolon (`;`) character or any combination thereof. Replies sent by the TCP server in reply to queries are single lines of text terminated by a carriage return character followed by a newline character (`\r\n`). Binary replies deviate from this convention, see the individual commands' descriptions in Section 9 for further details.

This section gives examples of communication with the TCP server using standard libraries and utilities.

### 8.1 National Instruments VISA

NI VISA is a cross-platform library for unified communication with measurements connected via GPIB, serial port, Network socket, etc. NI VISA handles all low-level configuration and provides buffered bidirectional I/O streams. This sections explains how to configure a socket connection to the LSPM 1.0 TCP Server using the debug tool provided with the NI VISA library and how to test it. NI VISA needs be downloaded from the "National Instruments" homepage and installed first.

Open the NI VISA Measurement and Automation Explorer (NI MAX). Add a new network device by selecting the subsection "Network devices" of "Devices and Interfaces" next click on "Add Network Device". As shown in Figure 42(a), select "Manual Entry of Raw Socket" and click "Next". As shown in Figure 42(b) enter the correct "Hostname or IP" the TCP "Port Number", click "Validate" to connect to the LSPM 1.0 TCP Server. Both NI MAX and the TCP server's output will indicate a successful connection. Click "OK" and "Finish" to return to the NI MAX main window.



(a)

(b)

Figure 42: Connection to LSPM 1.0 TCP Server through NI MAX

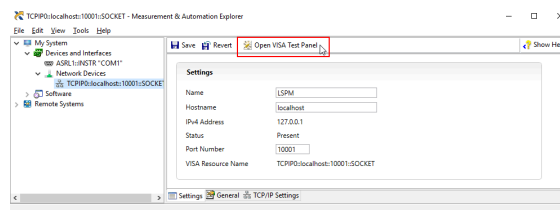


Figure 43: Starting NI VISA Test Panel

Right-click on the newly created network device and select “OPEN VISA Test Panel” as shown in Figure 43. No changes are required in the “TCP/IP Settings” tab. Set “Enable Termination Character” in the “I/O Settings” tab, click “Apply Changes” and observe the return data output as shown in Figure 44(a). This step needs to be repeated for every NI VISA Input/Output debug session. The “View Attributes” tab in Figure 44(b) shows shows the VISA parameters “VI\_ATTR\_TERMCHAR\_EN” set to “VI\_TRUE” and the “VI\_ATTR\_TERMCHAR” attribute set to “0xA”. When using the NI VISA library for connecting to the LSPM 1.0 TCP Server make sure to set all VISA parameters identically.

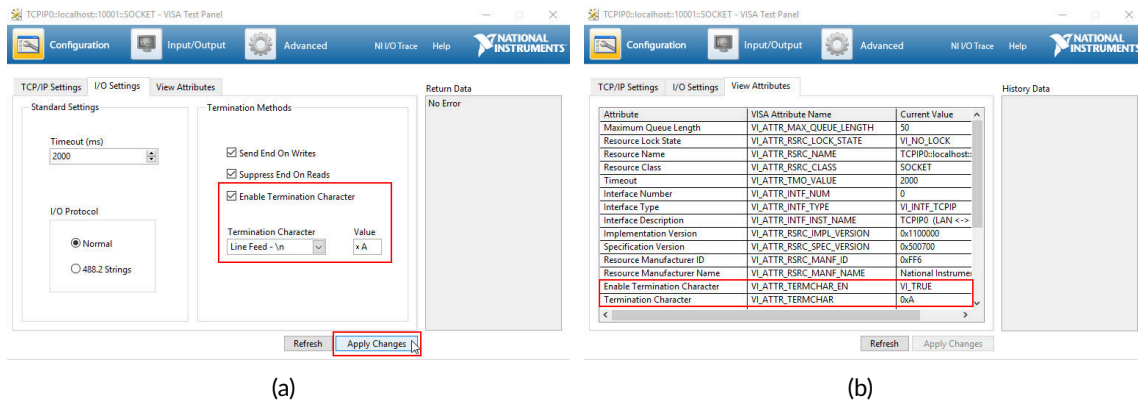
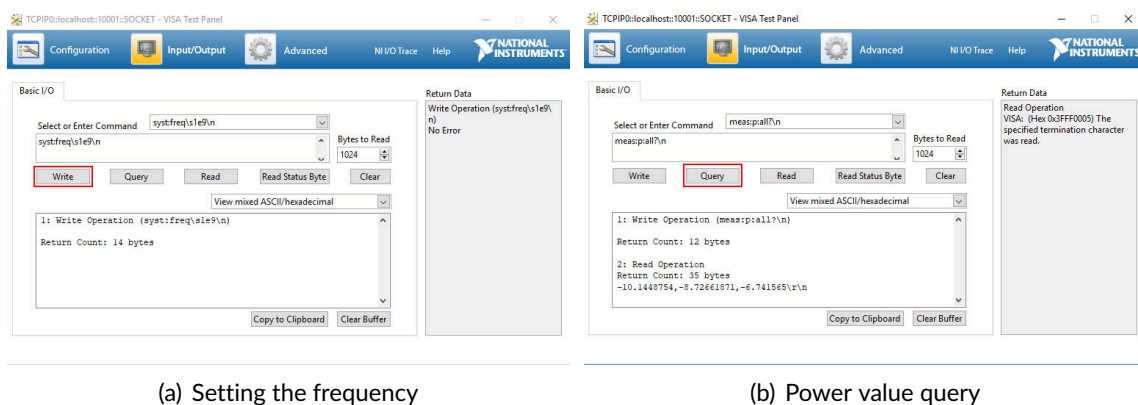


Figure 44: Configuring VISA TCP/IP socket parameters

Click on “Input/Output” to start testing NI VISA communication. Clicking “Query” will retrieve the identification string using the “\*idn?\n” command. As shown in Figure 45 the frequency is set to a specific value by entering and writing the “syst:freq 1e9\n” command. Set the command to “meas:p:all?\n” and click “Query” to obtain three power values. Make sure no errors are produced at any time.



(a) Setting the frequency

(b) Power value query

Figure 45: Testing NI VISA LSPM 1.0 TCP Server writes and queries

## 8.2 Raw TCP socket communication using PuTTY

Run PuTTY and enter the host name or IP address and the TCP port number. Set “Connection type” to “Raw” as shown in Figure 46(a). Optionally, save the session configuration for later use. Click “Open” to start the terminal session. Figure 46(b) shows the terminal window. Enter commands and press Return when done. Query commands will generate one reply line each. Multiple commands may be sent in rapid succession by separating them by semicolons.

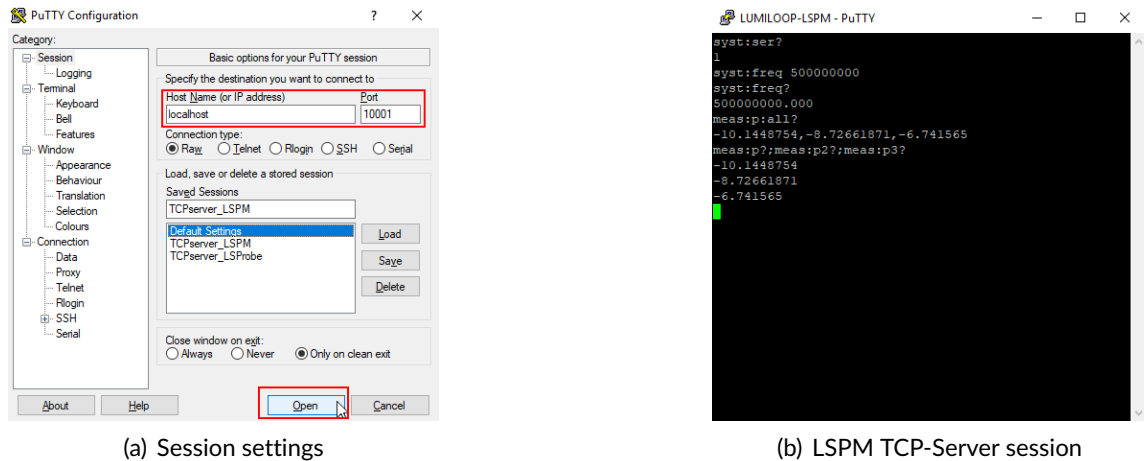


Figure 46: Using PuTTY

## 9 SCPI Command Reference

This section contains a list of all SCPI commands supported by the LSPM 1.0 TCP Server, grouped by sub-system. Each command is given with its mandatory, short-form syntax in upper-case letters. The long-form syntax is given by a combination of upper and lower-case letters. Optional parts of the command are enclosed in square brackets, i.e., []. Parameters are enclosed in angular brackets, i.e., <>.

### 9.1 Multi-Power Meter Behavior

Most SCPI commands support the optional MPMeter parameter determining the Multi-Power Meter behavior of the respective command. The following MPMeter parameters are available:

empty

If the MPMeter parameter is omitted the command will only be executed for the presently active power meter.

0

If the MPMeter parameter is set to zero the command is executed for all enumerated power



meters. Power meters will be enumerated and displayed sorted by their serial numbers starting with the smallest value.

## N

If the MPMeter parameter is set to a number N that is greater than zero the command will be executed for all power meters of the specified LSPM 1.0 Multi-Power Meter setup. LSPM 1.0 Multi-Power Meter Systems are defined using the :MPMeter:SERial <MPMeter>,<SN1>[,<SN2>,...,<SNN>] SCPI command, the identifier N can be any positive, non-zero integer value. Use :MPMeter:SERial? <MPMeter> to query the LSPM 1.0 Multi-Power Meter System.

When any SCPI command is executed for more than one power meter, i.e., the MPMeter parameter refers to multiple power meters, the command behaves as if executed for each power meter of the specified LSPM 1.0 Multi-Power Meter System successively. The order of execution is the same as the serial numbers returned by »:MPMeter:SERial? <MPMeter>«. All output will be joined on a single line by replacing line breaks between the output of different power meters with commas. For the sake of brevity the return value descriptions in the following sections do not explicitly state the return values for Multiprobe calls if they conform to the format explained above.

## 9.2 Generic Commands

### 9.2.1 \*CLS

Clear all status registers and structures, e.g., error queue.

### 9.2.2 \*ESE <ESR>

Set event status enable register. This feature is currently not implemented.

Parameters:

Integer value for event status register.

### 9.2.3 \*ESE?

Query event status enable register. This feature is currently not implemented.

Return value:

Returns the integer value of the event status register.

### 9.2.4 \*ESR?

Query the most recent error status register value. The error will be removed from error queue.

Return value:

Value of most recent errors in error queue.

### 9.2.5 \*IDN?

Query TCP server identification string.

Return value:

Comma-separated string, consisting of maker, product name, product version, TCP server build date and TCP server build time, e.g., »LUMILOOP,LSPM,1.0,Jun 2 2018,08:07:06«.

### 9.2.6 \*OPC

Set operation complete flag after the completion of the previously sent command. This feature is currently not implemented.

### 9.2.7 \*OPC?

Query operation complete flag. This feature is currently not implemented.

Return value:

Always 1.

### 9.2.8 \*RST

Reset TCP server. This will close all previously opened power meters, rescan the USB hardware and open all detected power meters. This will perform a power-on reset of all power meters.

The TCP server will print enumeration status information to its standard error output.

### 9.2.9 \*SRE <int>

Set service request enable register. This feature is currently not implemented.

Parameter:

Integer value of service request enable register.

### 9.2.10 \*SRE?

Query service request enable register. This feature is currently not implemented.

Return value:

Always 0.

### 9.2.11 \*STB?

Query status byte. Note that only bit 2 is currently implemented.

Return value:

The returned integer value contains the following status flags:

Bit 0

Unused bit.

Bit 1

Protection event flag, currently not implemented.

Bit 2

Error/Event queue message available.

Bit 3

Questionable status, currently not implemented.

Bit 4

Message available, currently not implemented.

Bit 5

Standard event status register, currently not implemented.

Bit 6

Service request, currently not implemented.

Bit 7

Operation status flag, currently not implemented.

### 9.2.12 \*TST?

Initiate self test and return test result. This feature is currently not implemented.

Return value:

0 on failing and 1 on passing the self test.

### 9.2.13 \*WAI

Wait for the completion of the previously issued command. This feature is currently not implemented.

## 9.3 :SYSTEM Commands

### 9.3.1 :SYSTEM:WAIT <Sec>

Pause processing of SCPI commands for Sec seconds.

### 9.3.2 :SYSTem:ERRor[:NEXT]?

Query most recent entry in system error queue and remove entry from error queue.

Return value:

Returns comma-separated error and error message string enclosed in quotes, e.g., "0,"No error".

### 9.3.3 :SYSTem:ERRor:COUNT?

Query number of entries in system error queue.

Return value:

Number of values in error queue.

### 9.3.4 :SYSTem:AUTOCONnect <State>

Enable/disable polling for new LSPM 1.0 devices.

Parameter:

Setting State to 1 activates polling for new LSPM devices, setting State to 0 disables polling.

By default, device polling is enabled. Disabling device polling can be useful if it is suspected or found to interfere with other USB devices.

### 9.3.5 :SYSTem:AUTOCONnect?

Query status of new device polling.

Return value:

The command returns an unsigned integer value containing the status of polling for new LSPM devices. If disabled, the command returns 0, if enabled 1.

### 9.3.6 :SYSTem:SERial <Value>

Select active power meter by serial number.

Parameter:

One serial number out of the list returned by :SYSTem:SERial? [<MPMeter>] with the MPMeter parameter set to 0.

### 9.3.7 :SYSTem:SERial? [<MPMeter>]

Query serial number of one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

Unsigned integer-valued serial number of selected power meter or comma-separated list of power meter serial numbers for LSPM 1.0 Multi-Power Meter Systems. If no power meters have been enumerated the command will return NAN.

### 9.3.8 :SYSTem:MAKer? [<MPMeter>]

Query maker identification string of one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

Name string of the device maker, e.g., »LUMILOOP«.

### 9.3.9 :SYSTem:DEVIce? [<MPMeter>]

Query device identification string of one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

Name string of the device, e.g., "LSPM".

### 9.3.10 :SYSTem:VERSion? [<MPMeter>]

Query device version string of one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

Version string of the device, e.g., »1.0«.

### 9.3.11 :SYSTem:REVision? [<MPMeter>]

Query firmware revision of one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

An unsigned integer value indicating the firmware's revision number is returned.

### 9.3.12 :SYSTem:FWUPdate?

Query firmware revision provided by TCP server.

Return value:

Unsigned integer value specifying the firmware revision number provided by the TCP server.

### 9.3.13 :SYSTem:DEBUG <Value/Flag1[,Flag2]...>

Set value of debug flags in debug register. Doing so makes the LSPM 1.0 TCP Server output debug information to standard error.

Parameter:

Either integer value whose individual bits are used to enable/disable the output of debugging information. Setting a flag to 1 enables debug output, setting a bit to 0 disables debug output. The bit positions of the debug flags are defined below.

Alternatively, a comma-separated list of strings can be supplied, as defined below.

Bit 0, Value 1, String "MEM")

Information about memory usage

Bit 1, Value 2, String "TIM"

Timing information

Bit 2, Value 4, String "IN"

Echo of all incoming TCP server messages

Bit 3, Value 8, String "OUT"

Echo all outgoing TCP server messages

Bit 4, Value 16, String "FIFO"

Information about USB burst and FIFO function calls

Bit 5, Value 32, String "USB"

Information about timing and data throughput of USB communications

Bit 6, Value 64, String "CAL"

Information about read-in and interpolation of calibration data

Bit 7, Value 128, String "CORR"

Information on the timing of correction data processing

Bit 8, Value 256, String "TRIG"

Information about trigger events

Bit 9, Value 512, String "STR"

Information about data streaming

Bit 10, Value 1024, String "STAT"

Information about statistics collection and processing

Bit 11, Value 2048, String "LOG"

Information on the logging of calibration data

Bit 12, Value 4096, String "POL"

Information on Computer Interface polling

Bit 13, Value 8192, String "FW"

Information on firmware programming

Bit 14, Value 16384, String "TRLUT"

Information on look-up-tables for triggered operation

Bit 15, Value 32768, String "LUT"

Information on processing of E-field look-up-tables

Bit 16, Value 65536, String "STLUT"

Information on processing of look-up-tables for stream saving

Bit 17, Value 131072, String "VBW"

Information on software-based RSSI value filtering to reduce video bandwidth

Bit 18, Value 262144, String "RADAR"

Information on radar computation

Bit 19, Value 524288, String "INIFILE"

Information on ini file path

E.g., to trace ingoing and outgoing SCPI commands issue ":syst:debug 12".

### 9.3.14 :SYSTEM:DEBUG?

Query value of debug register, containing all debug flags.

Return value:

Unsigned integer value containing all debug flags. See »:SYSTEM:DEBUG <Value/Flag1[,Flag2]...>« for the description of the individual debug flags and their values.

### 9.3.15 :SYSTem:DFLags?

Query mnemonic strings of all set debug flags in debug register

Return values:

List of string values for all set debug flags in the debug register. See »:SYSTem:DEBUG <Value/Flag1[,Flag2]...>« for the description of individual debug flags, their strings and their numeric values.

### 9.3.16 :SYSTem:MODe <Mode>[,<MPMeter>]

Set power meter operating mode of one or multiple power meters.

Parameters:

The unsigned integer parameter Mode specifies the power meter operating mode as described in Table 1 on page 15. Valid values are 0, 1, 2 and 3.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.3.17 :SYSTem:MODe? [<MPMeter>]

Get power meter operating mode of one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

An unsigned integer mode value as described in Table 1 on page 15 will be returned.

### 9.3.18 :SYSTem:FREQuency <Frequency>[,<MPMeter>]

Set frequency for frequency-compensated operation of one or multiple power meters.

Parameters:

The first double-precision, floating-point valued parameter sets the desired frequency. If the frequency exceeds the calibrated frequency range for the mode set via »:SYSTem:MODe <Mode>[,<MPMeter>]« the frequency will be forced to the nearest calibrated frequency.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1. E.g., «:syst:freq 1e9,0» will set the compensation frequency to 1 GHz for all enumerated power meters, »:syst:freq 2e9« will set the compensation frequency to 2 GHz for the currently selected power meter.



### 9.3.19 :SYSTem:FREQUency? [<MPMeter>]

Query frequency for frequency-compensated operation of one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

A floating-point value rounded to three decimal places indicating the set compensation frequency will be returned.

### 9.3.20 :SYSTem:FREQUency:MINimum? [<MPMeter>]

Query minimum calibrated frequency for frequency-compensated operation of one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

A floating-point value indicating the minimum calibrated compensation frequency of the current mode will be returned.

### 9.3.21 :SYSTem:FREQUency:MAXimum? [<MPMeter>]

Query maximum calibrated frequency for frequency-compensated operation of one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

A floating-point value indicating the maximum calibrated compensation frequency of the current mode will be returned.

### 9.3.22 :SYSTem:SSKip? [<MPMeter>]

Query sample skip count of one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

An unsigned integer value indicating the currently active sample skip rate of the USB communication.

### 9.3.23 :SYSTem:SRate? [<MPMeter>]

Query currently effective sampling rate of one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

An unsigned integer value indicating the current sampling rate of the power meter resulting from the skip counter value that can be queried via »:SYSTem:SSkip? [<MPMeter>]«.

### 9.3.24 :SYSTem:TIME?

Query current time stamp

Return value:

Float value specifying the number of seconds since 1904-01-01 00:00:00 UTC.

## 9.4 :CALibration Commands

### 9.4.1 :CALibration:LOGging <Value>

Enable or disable logging of status information when receiving measurement commands from current TCP session.

Parameter:

Enable logging of LSPM information for the current TCP session by setting Value to 1, disable logging by setting Value to 0.

When status logging is enabled, »:MEASure:P[1]/P2/P3/ALL? [<MPMeter>]« SCPI queries will log one line to the external calibration log.

See Section 10.2 for details about the log file format. Log files are located in the directory specified by the SAVE\_PATH setting in the LSPM\_1.2.ini configuration file. There is one log file for every active power meter.

### 9.4.2 :CALibration:LOGging?

Query state of logging of status information when receiving measurement commands from current TCP session.

Return value:

An unsigned integer value indicating if logging is enabled. The command returns 1 if logging is enabled and 0 otherwise.

### 9.4.3 :CALibration:EXternal <Value>[,<MPMeter>]

Enable or disable application of external calibration data of one or multiple power meters. After start-up the application of external calibration data is enabled.

Parameters:

Enable application of external calibration data by setting Value to 1, disable application by setting Value to 0.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.4.4 :CALibration:EXternal? [<MPMeter>]

Query application of external calibration data of one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

An unsigned integer value indicating if external calibration data is being applied or not will be returned. If external calibration data is being applied the command returns 1, otherwise zero will be returned. NAN will be returned if no power meter is connected.

### 9.4.5 :CALibration:CERTificate? [<MPMeter>]

Query external calibration certificate string of one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

An external calibration certificate string as given in the external calibration data CSV file will be returned. If the external calibration data CSV file does not contain a certificate string, no power meter is connected or there is no valid external calibration data "undefined" will be returned.

### 9.4.6 :CALibration:TStamp? [<MPMeter>]

Query in-house and external calibration time stamps of one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

A pair of unsigned integer values indicating the time stamp of in-house calibration and external calibration will be returned. Time stamps are expressed as the number of seconds since

Jan 1 1904 00:00:00. NAN will be returned if no power meter is connected or there is no valid in-house or external calibration data. NAN will also be returned for the external calibration time stamp if external calibration data has been disabled using »:CALibration:EXTernal <Value>[,<MPMeter>]«.

## 9.5 :MEASure Commands

### 9.5.1 :MEASure:TCold? [<MPMeter>]

Query the common cold plate temperature of the power sensors for one or multiple power meters. Temperature is controlled at 20 °C by a Peltier cooler for temperature-independent sensor operation.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns the float-valued cold plate temperature in °C.

### 9.5.2 :MEASure:VPeltier? [<MPMeter>]

Query Peltier cooler voltage for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns the float-valued Peltier cooler voltage in volt.

### 9.5.3 :MEASure:IPeltier? [<MPMeter>]short-cut

Query Peltier cooler current for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

The command returns the float-valued Peltier cooler current in ampere.

### 9.5.4 :MEASure:P[1]/P2/P3/ALL? [<MPMeter>]

Query power value for one or multiple power meters.

Return result for one of:

:P[1]?/:P2?/:P3?

Channel 1, 2 or 3 power value

:ALL?

all three results above as a list.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

The command returns a float-valued power value in dBm. Values are low-pass filtered as described in »:MEASure:LPFrequency <Frequency>[,<MPMeter>]«. -100 will be returned if the power sensor of the respective channel is not present. If ALL values are queried, a list of three values is returned. NAN will be returned if there is no valid calibration data.

### 9.5.5 :MEASure:MIN P[1]/P2/P3/ALL? [<MPMeter>]

Query minimum value power range for one or multiple power meters.

Return result for one of:

:P[1]?/:P2?/:P3?

Channel 1, 2 or 3 power value

:ALL?

all three results above as a list.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns a float-valued minimum calibrated power value in dBm. The value is determined using the calibration data for a given frequency and mode. -100 will be returned if the power sensor of the respective channel is not present. If ALL values are queried, a list of three values is returned. NAN will be returned if there is no valid calibration data.

### 9.5.6 :MEASure:MAX P[1]/P2/P3/ALL? [<MPMeter>]

Query maximum value power range for one or multiple power meters.

See »:MEASure:MIN P[1]/P2/P3/ALL? [<MPMeter>]« for a description of parameter and return values.

### 9.5.7 :MEASure:LPFrequency <Frequency>[,<MPMeter>]

Set power low-pass filter -3 dB cut-off frequency for one or multiple power meters.

**Parameter:**

Float value specifying the -3 dB cut-off frequency for the first order power low-pass filter in hertz. The filter is applied to calibrated and uncalibrated power values. Setting the value to 0 Hz disables low-pass filtering.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

**9.5.8 :MEASure:LPFrequency? [<MPMeter>]**

Query power low-pass filter -3 dB cut-off frequency for one or multiple power meters.

**Parameter:**

The optional unsigned integer parameter MPMeter is described in Section 9.1.

**Return value:**

The command returns a float value specifying the -3 dB cut-off frequency for the first order power low-pass filter in hertz. A value to 0 indicates that low-pass filtering is disabled.

**9.5.9 :MEASure:AUTOVBW <State>[,<MPMeter>]**

Enable or disable automatic, frequency-dependent setting of the software-defined video bandwidth

**Parameters:**

If the unsigned integer parameter State is set to 1 the LSPM 1.0 TCP Server will apply a low-pass filter with a frequency response suitable for the set operating frequency to avoid aliasing when the video bandwidth of the low-band detector is too large. If the parameter State is set to 0 the bandwidth set by the SCPI command »:MEASure:VBW <Frequency>[,<MPMeter>]« will be used.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

**9.5.10 :MEASure:AUTOVBW? [<MPMeter>]**

Query state automatic, frequency-dependent setting of the software-defined video bandwidth

**Parameter:**

The optional unsigned integer parameter MPMeter is described in Section 9.1.

**Return value:**

If automatic software-defined video bandwidth setting is enabled, the command returns 1, otherwise the command returns 0.

### 9.5.11 :MEASure:VBW <Frequency>[,<MPMeter>]

Set software-defined video bandwidth

Parameters:

The float-valued parameter Frequency sets 3 dB cut-off frequency in hertz for the software-based first-order RSSI filter. If set to zero, software-based video bandwidth filtering is disabled.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.5.12 :MEASure:VBW? [<MProbe>]

Query software-defined video bandwidth

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns a float value specifying the -3 dB cut-off frequency for the software-based first-order RSSI filter in hertz. A value to 0 indicates that low-pass filtering is disabled.

### 9.5.13 :MEASure:RSsi:P[1]/P2/P3/ALL? [<MPMeter>]

Query RSSI value for one or multiple power meters.

Return result for one of:

:P[1]?/:P2?/:P3?

Channel 1, 2 or 3 power value

:ALL?

all three results above as a list.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns an unsigned integer value representing the uncalibrated 14 bit ADC value the RSSI chip, i.e., received signal strength indicator, used to detect the power. The value is low-pass filtered using the low-pass filter as described in :MEASure:LPFrequency <Frequency>[,<MPMeter>]. 0 will be returned if the power sensor of the respective channel is not present. If ALL values are queried, a list of three values is returned.

## 9.6 :TRIGger Commands

### 9.6.1 :TRIGger:BEgin <Index>[,<MPMeter>]

Set index of first sample of power waveform for one or multiple power meters.

Parameters:

The first parameter is the integer-valued position of the beginning of the power waveform relative to the position of the trigger. E.g., »:trig:beg 0« will record samples starting at the trigger position, »:trig:beg -100« will record samples starting 100 samples before the trigger position, »:trig:beg 100« will record samples starting 100 samples after the trigger position.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.6.2 :TRIGger:BEgin? [<MPMeter>]

Query index of first sample of power waveform for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns an integer-valued position of the beginning of the power waveform relative to the position of the trigger, corresponds to the first parameter of »:TRIGger:BEgin <Index>[,<MPMeter>]«. If executed for multiple power meters the command returns a list of integer-valued positions of the beginning of the power waveform relative to the position of the trigger for each power meter of the respective list.

### 9.6.3 :TRIGger:LEnGth <Length>[,<MPMeter>]

Set length of the power waveform for one or multiple power meters.

Parameters:

The unsigned integer-valued parameter of the command specifies the length of a power waveform. E.g., »:trig:len 100« will record 100 samples starting at the sample index specified by :TRIGger:BEgin <Index>[,<MPMeter>].

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.6.4 :TRIGger:LEnGth? [<MPMeter>]

Query number of samples in power waveform for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.



Return value:

The command returns an unsigned integer-valued length of the power waveform, corresponding to the length set by »:TRIGger:LENgth <Length>[,<MPMeter>]«. If executed for multiple power meters the command returns a list of unsigned integer-valued lengths of the power waveforms for each power meter of the respective list.

### 9.6.5 :TRIGger:FLENgth? [<MPMeter>]

Query full number of samples in complete power waveform for all trigger points, for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns the unsigned integer-valued full length of the power waveform, for all trigger points, expressed as a number of samples.

to the length set by :TRIGger:LENgth <Length>[,<MPMeter>] multiplied by the number of set trigger points set by :TRIGger:POINt <Points>[,<MPMeter>]. The return value equals the number of samples per trigger point, returned by »:TRIGger:LENgth? [<MPMeter>]« multiplied by the number of trigger points returned by »:TRIGger:POINt? [<MPMeter>]«. If executed for multiple power meters the command returns a list of unsigned integer-valued full lengths of the power waveforms for each power meter of the respective list.

### 9.6.6 :TRIGger:POINt <Points>[,<MPMeter>]

Set number of the trigger points for one or multiple power meters.

Parameters:

The unsigned integer-valued parameter Points specifies the number of trigger points, i.e., the number of trigger events required for a full waveform. Consequently, reaching the trigger state DONE requires Points valid trigger events. The number of samples per sub-waveforms is specified using »:TRIGger:LENgth <Length>[,<MPMeter>]«.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.6.7 :TRIGger:POINt? [<MPMeter>]

Query number of power waveforms for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns the unsigned integer-valued number of trigger points, corresponding to the value set by »:TRIGger:POINT <Points>[,<MPMeter>]«. If executed for multiple LSPM 1.0 Power Meters the command returns a list of unsigned integer-valued point numbers of the power waveform lengths for each power meter of the respective list.

### 9.6.8 :TRIGger:PROgress? [<MPMeter>]

Query progress of waveform acquisition for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns the unsigned integer-valued number of samples that have been recorded to the power waveform buffer. Upon reaching the trigger state DONE the return value equals the number returned by »:TRIGger:FLENght? [<MPMeter>]«. If executed for multiple power meters the command returns a list of lengths for each power meter of the respective list.

### 9.6.9 :TRIGger:PTProgress? [<MPMeter>]

Query progress of point trigger acquisition for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns the unsigned integer-valued number of point trigger events that have been processed by the trigger sub-system. Upon reaching the trigger state DONE the return value equals the number returned by »:TRIGger:POINT? [<MPMeter>]«.

If executed for multiple power meters the command returns a list of lengths for each power meter of the respective list.

### 9.6.10 :TRIGger:PTTimes? [<MPMeter>]

Query point trigger sample offsets for a full waveform for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns a list of unsigned 64 bit integer-valued offset values, expressed as the number of samples relative to the first trigger point. The list has one element less than the there

are trigger points. E.g., operating in mode 0, for three trigger points whose trigger events are 1 ms apart, the command will return a list consisting of two values, 2000 and 4000, indicating that the second trigger event occurred 1 ms after the first trigger event and that the third trigger event occurred 2 ms after the first trigger event. NAN will be returned if the trigger system is not in DONE state or if there is only one trigger point.

#### 9.6.11 :TRIGger:EVCNT? <Samples>[,<MPMeter>]

Query the number of hardware-detected trigger events for one or multiple power meters for given number of samples.

Parameter:

The first mandatory unsigned integer parameter Samples defines the number of samples for which the trigger events are counted. The detection of trigger events is independent of the trigger state. Every trigger event increments the counter by one.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

An integer-valued number of hardware-detected trigger events is returned.

#### 9.6.12 :TRIGger:STATe? [<Timeout>,<MPMeter>]

Query the state of the trigger system for one or multiple power meters.

Parameter:

The first optional float-valued parameter Timeout sets the maximum number of seconds to wait for the trigger state DONE. The command will return immediately if the trigger state is DONE or IDLE, if Timeout is set to zero, or if Timeout is omitted.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1. The second parameter always requires the first parameter to be set.

Return value:

The command returns a string value giving the state of the trigger system, valid return values are IDLE, ARM, ARMED, TRIGGERED, DONE. See Figure 24 on page 32 for reference.

#### 9.6.13 :TRIGger:ARM [<MPMeter>]

Arm trigger, the trigger system will change state from any other trigger state to ARMED for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

#### 9.6.14 :TRIGger:ARMed? [<Timeout>,<MPMeter>]

Query if the trigger system is in state ARMED for one or multiple power meters.

Parameter:

The first optional float-valued parameter Timeout sets the maximum number of seconds to wait for the state ARMED. The command will return immediately if the trigger state is ARMED, if Timeout is set to zero, or if Timeout is omitted.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1. The second parameter always requires the first parameter to be set.

Return value:

The command returns an unsigned integer value indicating if the trigger state is ARMED. If the trigger state is ARMED, the return value is 1, otherwise it is 0.

#### 9.6.15 :TRIGger:CLear [<MPMeter>]

Clear trigger, the trigger system will change state from any other trigger state to IDLE for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

#### 9.6.16 :TRIGger:FORce [<MPMeter>]

Force trigger, the trigger system will change state to TRIGGERED independent of the trigger source set by »:TRIGger:SOURce <Source>[,<MPMeter>]« for one or multiple power meters. This command is used for software triggering.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

#### 9.6.17 :TRIGger:DONE? [<Timeout>,<MPMeter>]

Query if the trigger system is in state DONE for one or multiple power meters.

Parameter:

The first optional float-valued parameter Timeout sets the maximum number of seconds to wait for the state DONE. The command will return immediately if the trigger state is DONE, if Timeout is set to zero, or if Timeout is omitted.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1. The second parameter always requires the first parameter to be set.

Return value:

The command returns an unsigned integer value indicating if the trigger state is DONE. If the trigger state is DONE, the return value is 1, otherwise it is 0.

#### 9.6.18 :TRIGger:COUnT? [<MPMeter>]

Query number of detected trigger events for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

An integer-valued number of trigger events recorded is returned.

#### 9.6.19 :TRIGger:SOURce <Source>[,<MPMeter>]

Set trigger source for triggered operation for one or multiple power meters.

Parameter:

String parameter without quotes specifying the trigger source. Valid values are SOFT, EXT, EXT2, P1, P2 and P3. When set to SOFT triggering must occur by means of the »:TRIGger:FORce [<MPMeter>]« command. When set to EXT triggering uses the external trigger input configured by »:TRIGger:INVert <0/1>[,<MPMeter>]«, »:TRIGger:SYNC <0/1>[,<MPMeter>]« and »:TRIGger:OUTput <0/1>[,<MPMeter>]«. EXT refers to the trigger signal of the BNC connector. EXT2 refers to the trigger signal of the "Ext1" RJ45 socket of the respective power meter and is configured by »:TRIGger:BPINVert <0/1>[,<MPMeter>]«, »:TRIGger:BPSYNC <0/1>[,<MPMeter>]« and »:TRIGger:BPOUTput <0/1>[,<MPMeter>]«.

When set to either P1, P2 or P3 the field strength value of the selected channel is used for triggering, »:TRIGger:LEVel <Level>[,<MPMeter>]« and »:TRIGger:FALLing <0/1>[,<MPMeter>]« are used for configuration.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

#### 9.6.20 :TRIGger:SOURce? [<MPMeter>]

Query trigger source for triggered operation for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns a string value without quotes specifying the trigger source, see »:TRIGger:SOURce <Source>[,<MPMeter>]« for more details.

### 9.6.21 :TRIGger:LEVel <Level>[,<MPMeter>]

Set the trigger power level for P1, P2 and P3 triggering for one or multiple power meters.

Parameter:

Float parameter specifying the power level in dBm. Triggering occurs when the power value crosses the set power level.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.6.22 :TRIGger:LEVel? [<MPMeter>]

Query the trigger power level for P1, P2 and P3 triggering for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

The command returns the trigger level as a float-valued power value in dBm. NAN will be returned if the power meter is off or in start-up.

### 9.6.23 :TRIGger:FALLing <0/1>[,<MPMeter>]

Set the direction for external, P1, P2 and P3 triggering for one or multiple power meters.

Parameters:

Boolean value of either 0 or 1. If set to 0 the the rising edge of the external trigger signal or passing the threshold value in rising direction will bring the trigger system from the state ARMED to TRIGGERED. If set to 1 the falling edge will be used for triggering.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.6.24 :TRIGger:FALLing? [<MPMeter>]

Query the direction for external, P1, P2 and P3 triggering for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns the boolean value giving the trigger edge direction, see »:TRIGger:FALLing <0/1>[,<MPMeter>]« for details.

### 9.6.25 :TRIGger:OUTput <0/1>[,<MPMeter>]

Enable or disable the output of a trigger signal via the power meter's BNC connector for one or multiple power meters.

Parameters:

Boolean value of either 0 or 1. If set to 0 trigger output is disabled and the power meter's BNC connector can be used for trigger input. If set to 1 trigger output is enabled and the power meter's BNC connector cannot be used for trigger input.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.6.26 :TRIGger:OUTput? [<MPMeter>]

Query status of trigger output for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns a boolean value giving the state of the power meters BNC trigger connector, see »:TRIGger:OUTput <0/1>[,<MPMeter>]« for details.

### 9.6.27 :TRIGger:INVert <0/1>[,<MPMeter>]

Set the polarity for trigger output via the power meter's BNC trigger connector for one or multiple power meters.

Parameters:

Boolean value of either 0 or 1. If set to 0 trigger output uses a rising edge logic signal. If set to 1 a falling edge logic signal will be generated.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.6.28 :TRIGger:INVert? [<MPMeter>]

Query the polarity for trigger output via the power meter's BNC trigger connector for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns a boolean value of either 0 or 1. See »:TRIGger:INVert <0/1>[,<MPMeter>]« for details.

### 9.6.29 :TRIGger:SYNC <0/1>[,<MPMeter>]

Enable or disable synchronization trigger output using the power meter's BNC trigger connector for one or multiple power meters. This function is useful for synchronizing signal generators or transmitters with the power meter.

Parameters:

Boolean value of either 0 or 1. If set to 0 and external trigger output is enabled, the output trigger signal described for »:TRIGger:OUTput <0/1>[,<MPMeter>]« and »:TRIGger:INVert <0/1>[,<MPMeter>]«. If set to 1 and trigger output is enabled, a logic pulse will be generated synchronously with power value acquisition. A 250 ns long pulse will be generated once every 500 ns.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.6.30 :TRIGger:SYNC? [<MPMeter>]

Query synchronization trigger output for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns a boolean value of either 0 or 1. See »:TRIGger:SYNC <0/1>[,<MPMeter>]« for details.

### 9.6.31 :TRIGger:BPOUTput <0/1>[,<MPMeter>]

Enable or disable the output of a trigger signal via the power meter's upper RJ45 connector for one or multiple power meters. See »:TRIGger:OUTput <0/1>[,<MPMeter>]« for parameters.

### 9.6.32 :TRIGger:BPOUTput? [<MPMeter>]

Query status of trigger output via the power meter's upper RJ45 connector for one or multiple power meters. See »:TRIGger:OUTput? [<MPMeter>]« for parameter and return value.

### 9.6.33 :TRIGger:BPINVert <0/1>[,<MPMeter>]

Set the polarity for trigger output via the power meter's upper RJ45 connector for one or multiple power meters. See »:TRIGger:INVert <0/1>[,<MPMeter>]« for parameters.



#### 9.6.34 :TRIGger:BPINVert? [<MPMeter>]

Query the polarity for trigger output via the power meter's upper RJ45 connector for one or multiple power meters. See »:TRIGger:INVert? [<MPMeter>]« for parameter and return value.

#### 9.6.35 :TRIGger:BPSYNC <0/1>[,<MPMeter>]

Enable or disable synchronization trigger output via the power meter's upper RJ45 connector for one or multiple power meters. See »:TRIGger:SYNC <0/1>[,<MPMeter>]« for parameters.

#### 9.6.36 :TRIGger:BPSYNC? [<MPMeter>]

Query the configuration of the synchronization trigger output via the power meter's upper RJ45 connector for one or multiple power meters. See »:TRIGger:SYNC? [<MPMeter>]« for parameter and return value.

### 9.7 :TRIGger[:WAVEform] Commands

#### 9.7.1 :TRIGger[:WAVEform][:Power]:P[1]/:P2/:P3? [<MPMeter>]

Query power values of power waveform channel 1, 2 or 3 for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

The command returns a comma-separated list of float-valued channel 1, 2 or 3 power values of the power waveform in dBm. -100 will be returned if the power sensor of the respective channel is not present. NAN will be returned if there is no valid calibration data or if the trigger system state is not equal to DONE.

#### 9.7.2 :TRIGger[:WAVEform][:Power]:ALL? [<MPMeter>]

Query channel 1, 2 and 3 power values averaged over the present power waveform for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

The command returns a comma-separated list of three float values giving the arithmetic mean of channel 1, 2 and 3 power, in this order. -100 will be returned if the power sensor of the respective channel is not present. NAN will be returned if there is no valid calibration data or if the trigger system state is not equal to DONE.

### 9.7.3 :TRIGger[:WAVEform]:RSSI:P[1]/:P?:/P3? [<MPMeter>]

Query channel RSSI values of a waveform of channel 1, 2 or 3 for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

The command returns a list of unsigned integer values representing the uncalibrated 14 bit ADC value acquired by the channel 1, 2 or 3 RSSI chip, i.e., received signal strength indicator, used to detect the power level. 0 will be returned if the power sensor of the respective channel is not present. NAN will be returned if the trigger system state is not equal to DONE.

### 9.7.4 :TRIGger[:WAVEform][:Power]:BINary? [<MPMeter>]

Query channel 1, channel 2 and channel 3 power values of power waveform in binary format for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

Binary data block followed by a carriage return, newline sequence. The first four bytes specify the number of bytes of the binary data block which will be returned following the first four bytes.

For each power meter P a chunk of binary data will be sent. All values are encoded in little endian format. Data is ordered as follows:

power meter number

32 bit unsigned integer value giving the serial number of the corresponding power meter. If the power meter P is not defined the power meter serial number and sample count are set to zero and the binary data block ends.

sample count

32 bit unsigned integer value giving the number of samples S in the waveform of the corresponding power meter. The following binary data will contain S values for each of the following values. If there is no valid calibration data or if the trigger system state is not equal to DONE the number of samples is set to zero and the binary data block ends.

channel 1 power

S 32 bit single-precision, floating-point values giving a list of channel 1 power values in dBm of the power waveform, see »:TRIGger[:WAVEform][:Power]:P[1]/:P2/:P3? [<MPMeter>]«.

channel 2 power

S 32 bit single-precision, floating-point values giving a list of channel 2 power values in

dBm of the power waveform, see »:TRIGger[:WAVEform][:Power]:P[1]/:P2/:P3? [<MP-Meter>]«.

channel 3 power

S 32 bit single-precision, floating-point values giving a list of channel 3 power values in dBm of the power waveform, see »:TRIGger[:WAVEform][:Power]:P[1]/:P2/:P3? [<MP-Meter>]«.

channel 1 RSSI

S 16 bit unsigned integer values, encoded as 32 bit single-precision, floating-point values, giving a list of channel 1 RSSI values in LSB of the waveform, see »:TRIGger[:WAVEform]:RSsi:P[1]/:P?:/P3? [<MPMeter>]«.

channel 2 RSSI

S 16 bit unsigned integer values, encoded as 32 bit single-precision, floating-point values, giving a list of channel 2 RSSI values in LSB of the waveform, see »:TRIGger[:WAVEform]:RSsi:P[1]/:P?:/P3? [<MPMeter>]«.

channel 3 RSSI

S 16 bit unsigned integer values, encoded as 32 bit single-precision, floating-point values, giving a list of channel 3 RSSI values in LSB of the waveform, see »:TRIGger[:WAVEform]:RSsi:P[1]/:P?:/P3? [<MPMeter>]«.

### 9.7.5 :TRIGger[:WAVEform][:Power]:BINWait? [<Timeout>,<MPMeter>]

Convenience command combining »:TRIGger:DONE? [<Timeout>,<MPMeter>]« and »:TRIGger[:WAVEform][:Power]:BINary? [<MPMeter>]«. The command will wait for the trigger system to reach the DONE state for at most Timeout seconds. When the state DONE is reached, Timeout is set to 0, or when the Timeout is reached »:TRIGger[:WAVEform][:Power]:BINary? [<MPMeter>]« is executed. If the trigger system is in the DONE state the trigger system will be cleared and armed automatically.

This command is most useful for rapid and efficient waveform value polling.

## 9.8 [:TRIGger]:RADar, Commands

### 9.8.1 [:TRIGger]:RADar:TRIM <State>[,<MProbe>]

Enable/disable pulse trimming

Parameter:

The mandatory first unsigned integer parameter State controls the trimming of pulse edges. If set to 0 all samples exceeding the pulse threshold will be treated as belonging to a pulse. If set to 1, the first and last sample of the pulse, which exceed the threshold will be trimmed, i.e., removed from the pulse.

The optional second unsigned integer parameter MPMeter is described in Section 9.1.

### 9.8.2 `[:TRIGger]:RADar:TRIM? [<MProbe>]`

Query state of pulse trimming

Parameter:

The optional unsigned integer parameter `MPMeter` is described in Section 9.1.

Return value:

An unsigned integer value indicating state of pulse trimming, see `»[:TRIGger]:RADar:TRIM <State>[,<MProbe>]«`.

### 9.8.3 `[:TRIGger]:RADar:MINTime <MinT>[<MPMeter>]`

Set minimum required pulse duration in seconds.

Parameter:

The mandatory floating-point value `MinT` sets the minimum duration of a pulse in seconds. Shorter pulses will be discarded.

The optional unsigned integer parameter `MPMeter` is described in Section 9.1.

### 9.8.4 `[:TRIGger]:RADar:MINTime? [<MPMeter>]`

Query minimum required pulse duration in seconds.

Parameter:

The optional unsigned integer parameter `MPMeter` is described in Section 9.1.

Return value:

A floating-point value indicating minimum pulse duration in seconds.

### 9.8.5 `[:TRIGger]:RADar:MINSamples <MinS>[<MPMeter>]`

Set minimum required pulse duration in samples.

Parameter:

The mandatory integer value `MinS` sets the minimum required duration of a pulse in samples. Shorter pulses will be discarded. The optional unsigned integer parameter `MPMeter` is described in Section 9.1.

### 9.8.6 `[:TRIGger]:RADar:MINSamples? [<MPMeter>]`

Query minimum required pulse duration in samples.

**Parameter:**

The optional unsigned integer parameter MPMeter is described in Section 9.1.

**Return values:**

An unsigned integer value indicating minimum required pulse duration expressed in samples.

### 9.8.7 [:TRIGger]:RADar:THMethod <Method>[<MPMeter>]

Set the method for setting the pulse threshold

**Parameter:**

The mandatory string value Method sets the method of threshold detection. Parameters are given without quotation marks, valid methods are:

**AVG, max/min-based threshold**

When set to AVG, the pulse threshold is set to the arithmetic mean of the maximum power value and minimum power value in the waveform. This is the default method.

**ABS, absolute threshold**

When set to ABS, the threshold set by the SCPI command »[:TRIGger]:RADar:ATHold <Threshold>[<MPMeter>]« is used.

**REL, relative threshold**

When set to REL, the threshold set by the SCPI command »[:TRIGger]:RADar:RTHold <Threshold>[<MPMeter>]« is used.

**HIST, histogram-based threshold**

When set to HIST, the pulse threshold is determined based on the distribution of power values in the waveform. Typical probability distributions of power values have one peak at the power level of the inactive transmitter, i.e., noise level, and another peak at the power level of the active transmitter.

The threshold will be placed between these peaks, at the power level in the probability distribution that has the smallest probability according to the waveform's histogram at a resolution of 1 dB. The threshold must have a clearance greater than the value set via »[:TRIGger]:RADar:CLEARance <Clearance>[<MPMeter>]« to either probability peak.

The optional second unsigned integer parameter MPMeter is described in Section 9.1.

### 9.8.8 [:TRIGger]:RADar:THMethod? [<MPMeter>]

Query the method for setting the pulse threshold

**Parameter:**

The optional unsigned integer parameter MPMeter is described in Section 9.1.

**Return values:**

A string value indicating the set pulse threshold method, see »[:TRIGger]:RADar:THMethod <Method>[<MPMeter>]« for details.

### 9.8.9 [:TRIGger]:RADar:ATHold <Threshold>[<MPMeter>]

Set the absolute threshold for pulse detection

Parameter:

The mandatory floating-point value Threshold sets the threshold for detecting pulses in dBm.

The optional second unsigned integer parameter MPMeter is described in Section 9.1.

### 9.8.10 [:TRIGger]:RADar:ATHold? [<MPMeter>]

Query the absolute threshold for pulse detection

Parameter:

The optional second unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

A floating-point value indicating the absolute pulse detection threshold in dBm.

### 9.8.11 [:TRIGger]:RADar:RTHold <Threshold>[<MPMeter>]

Set the relative threshold for pulse detection

Parameter:

The mandatory floating-point value Threshold, sets the threshold for detecting pulses relative to the maximum power value found in the waveform, expressed in dB relative to the maximum value. Values must be greater than zero. E.g., a Threshold value of 10 will set the detection threshold 10 dB below the highest power value in the waveform.

The optional second unsigned integer parameter MPMeter is described in Section 9.1.

### 9.8.12 [:TRIGger]:RADar:RTHold? [<MPMeter>]

Query the relative threshold for pulse detection

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

A floating-point value indicating the pulse detection threshold expressed in dB relative to the maximum power value found in the waveform, see »[:TRIGger]:RADar:RTHold <Threshold>[<MPMeter>]« for details.

### 9.8.13 `[:TRIGger]:RADar:CLEARance <Clearance>[<MPMeter>]`

Set the clearance of smallest probability in the distribution to its neighboring probability peaks.

Parameter:

The mandatory floating-point value `Clearance` sets the clearance for automatic pulse threshold setting in dB. The default value is 6 dB, see »`[:TRIGger]:RADar:THMethod <Method>[<MPMeter>]`« for details.

The optional second unsigned integer parameter `MPMeter` is described in Section 9.1.

### 9.8.14 `[:TRIGger]:RADar:CLEARance? [<MPMeter>]`

Query the clearance of smallest probability in the distribution to its neighboring probability peaks.

Parameter:

The optional unsigned integer parameter `MPMeter` is described in Section 9.1.

Return values:

A floating-point value indicating the minimum clearance of the smallest probability in the distribution relative to its neighboring probability peaks.

### 9.8.15 `[:TRIGger]:RADar:THold:P[1]/P2/P3/ALL? [<MPMeter>]`

Query threshold value for pulse detection, via `:P[1]?/:P2?/:P3?` return results for channel 1, 2 or 3, via `:ALL?` return results for channel 1, 2 and 3.

Parameter:

The optional unsigned integer parameter `MPMeter` is described in Section 9.1.

Return value:

A floating-point value indicating the respective channel's pulse detection threshold in dBm. With the ALL variant, return list of three values. NAN will be returned, if there is no valid threshold value.

### 9.8.16 `[:TRIGger]:RADar:MAVG <Count>[<MPMeter>]`

Set number of samples for moving average filter.

Parameter:

The mandatory integer value `Count`, sets the number of samples to use for calculating a moving average over `Count` samples. The filter is off if `Count` is set to one.

The optional second unsigned integer parameter `MPMeter` is described in Section 9.1.

### 9.8.17 [:TRIGger]:RADar:MAVG? [<MPMeter>]

Query number of samples for moving average filter.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

An integer value indicating the number of samples used for calculating a moving average of power values.

### 9.8.18 [:TRIGger]:RADar:P[1]/P2/P3? [<MPMeter>]

Query number of pulses, pulse positions and averaged pulse power for channel 1, 2 or 3 power waveform for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

The command returns a comma-separated list of values, starting with an unsigned integer value giving the number of detected pulses, followed by index/power value pairs giving the sample position and power value of each radar radar pulse. The first value of each pair gives the sample index after crossing the threshold value in a rising fashion relative to the start of the waveform. The second value of each pair gives the power value of the respective pulse. The number of pulses will be returned as zero and no index/power value pairs be returned if no pulses could be detected, there is no valid calibration data or the trigger system is not in the DONE state

### 9.8.19 [:TRIGger]:RADar[:APOWer]:P[1]/:P2/:P3/:ALL? [<MPMeter>]

Query average power of all pulses found in the waveform, via :P[1]?/:P2?/:P3? return results for channel 1, 2 or 3, via :ALL? return results for channel 1, 2 and 3.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

A floating-point value indicating the arithmetic mean of all power values in the waveform of the respective channel which exceed the set threshold value and minimum pulse length. Using the ALL variant, return a list of all three values.

### 9.8.20 [:TRIGger]:RADar:MPOWer:P[1]/:P2/:P3/:ALL? [<MPMeter>]

Query power of pulse with the highest averaged power, via :P[1]?/:P2?/:P3? return results for channel 1, 2 or 3, via :ALL? return results for channel 1, 2 and 3.



**Parameter:**

The optional unsigned integer parameter `MPMeter` is described in Section 9.1.

**Return values:**

A floating-point value indicating the arithmetic mean of all power values of the largest pulse in the waveform for the respective channel. Using the ALL variant, return a list of all three values.

**9.8.21 [:TRIGger]:RADar:COUnt:P[1]/:P2/:P3/:ALL? [<MPMeter>]**

Query number of detected channel 1, 2, 3 or all channels' power pulses in the present power waveform, for one or multiple power meters.

**Parameter:**

The optional unsigned integer parameter `MPMeter` is described in Section 9.1.

**Return values:**

The command returns an unsigned integer value giving the number of detected pulses for the respective channel or a list of three integer values for all channels. NAN will be returned if there is no valid calibration data or the trigger system state is not equal to DONE.

**9.8.22 [:TRIGger]:RADar:PULses:STArt:P[1]/:P2/:P3? [<MPMeter>]**

Query start of all pulses found in the waveform. via `:P[1]?/:P2?/:P3?` return results for channel 1, 2 or 3.

**Parameter:**

The optional unsigned integer parameter `MPMeter` is described in Section 9.1.

**Return values:**

The command returns a list of integer values indicating the start of the respective channel's pulses expressed in samples relative to the beginning of the waveform. NAN will be returned if no pulses are detected.

**9.8.23 [:TRIGger]:RADar:PULses:LENGth:P[1]/:P2/:P3? [<MPMeter>]**

Query length of all pulses found in the waveform, via `:P[1]?/:P2?/:P3?` return results for channel 1, 2 or 3.

**Parameter:**

The optional unsigned integer parameter `MPMeter` is described in Section 9.1.

**Return values:**

The command returns a list of integer values indicating the length of the respective axis' pulses expressed as samples. NAN will be returned if no pulses are detected.

#### 9.8.24 [:TRIGger]:RADar:PULses[:APOWer]:P[1]/:P2/:P3? [<MPMeter>]

Query average power of all pulses found in the waveform, via :P[1]?/:P2?/:P3? return results for for channel 1, 2 or 3.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

The command returns a list of single-precision, floating-point values giving the arithmetic mean of each pulse's power. NAN will be returned if no pulses are detected.

#### 9.8.25 [:TRIGger]:RADar:DUTY:P[1]/:P2/:P3/:ALL? [<MPMeter>]

Query duty cycle of power values, via :P[1]?/:P2?/:P3? return results for channel 1, 2 or 3, via :ALL? return results for channel 1, 2 and 3.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

A floating-point value indicating the ratio of the number of the respective channel's samples, which exceed the set threshold value and minimum pulse length, and the total number of samples in the waveform. Return a list of three values if the ALL variant is used.

#### 9.8.26 [:TRIGger]:RADar:WPower:P[1]/:P2/:P3/:ALL? [<MPMeter>]

Query radar power waveform, via :P[1]?/:P2?/:P3? return results for channel 1, 2 or 3, via :ALL? return results for channel 1, 2 and 3.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

The command returns a comma separated list of float-valued power values of the current waveform in dBm of the respective channel, after applying the moving average filter set via see »[:TRIGger]:RADar:MAVG <Count>[<MPMeter>]«. Using the ALL variant, returns a concatenation of all three lists. NAN will be returned if there is no valid calibration data or if the trigger system state is not equal to DONE.

#### 9.8.27 [:TRIGger]:RADar:BINary? <Wave>[,<MPMeter>]

Query details of all channels' pulses found in the waveform in binary format for one or multiple E-field probes.

**Parameter:**

The first mandatory integer-valued parameter Wave controls the output of sweep corrected power waveform values. If set to 1 waveform output is enabled, if set to 0 waveform is disabled.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

**Return values:**

Binary data block followed by a carriage return, newline sequence. The first four bytes specify the number of bytes of the binary data block which will be returned following the first four bytes.

For each power meter P a chunk of binary data will be sent. All values are encoded in little endian format. If the power meter P is not defined the serial number, and sample count are set to zero and the binary data block ends. Data are ordered as follows:

**serial number**

32 bit unsigned integer value giving the serial number of the corresponding power meter.

**sample count**

32 bit unsigned integer value giving the number of samples in the waveform of the corresponding power meter.

**minimum pulse duration**

32 bit unsigned integer value giving the minimum pulse duration in samples used for pulse detection, see »[:TRIGger]:RADar:MINSamples? [<MPMeter>]«.

**moving average filter**

32 bit unsigned integer value giving the number of samples used for calculating a moving average for pulse detection, see »[:TRIGger]:RADar:MAVG? [<MPMeter>]«.

**channel 1 threshold value**

32 bit single-precision, floating-point value giving the channel 1 threshold value for pulse detection, see »[:TRIGger]:RADar:THold:P[1]/P2/P3/ALL? [<MPMeter>]«. If there is no valid calibration data, if the trigger system state is not equal to DONE or if there are no pulses detected above the value is set to NAN.

**channel 2 threshold value**

See description of channel 1 threshold value above.

**channel 3 threshold value**

See description of channel 1 threshold value above.

**channel 1 averaged power**

32 bit single-precision, floating-point value giving the arithmetic mean of all power values of the waveform which belong to pulses, see »[:TRIGger]:RADar[:APOWer]:P[1]/:P2/:P3/:ALL? [<MPMeter>]«. If there is no valid calibration data, if the trigger system state is not equal to DONE or if there are no pulses detected above the value is set to NAN.

channel 2 averaged power

See description of channel 1 averaged power above.

channel 3 averaged power

See description of channel 1 averaged power above.

channel 1 maximum pulse power

32 bit single-precision, floating-point value giving the arithmetic mean of the power of the largest channel 1 pulse in the waveform, see »[:TRIGger]:RADar:MPOWER:P[1]/:P2/:P3/:ALL? [<MPMeter>]«. If there is no valid calibration data, if the trigger system state is not equal to DONE or if there are no pulses detected above the value is set to NAN.

channel 2 maximum pulse

See description of channel 1 maximum pulse power above.

channel 3 maximum pulse

See description of channel 1 maximum pulse power above.

channel 1 duty cycle

32 bit single-precision, floating-point value giving the duty cycle of channel 1 power values, see »[:TRIGger]:RADar:DUTY:P[1]/:P2/:P3/:ALL? [<MPMeter>]«. If there is no valid calibration data, if the trigger system state is not equal to DONE or if there are no pulses detected above the value is set to NAN.

channel 2 duty cycle

See description of channel 1 duty cycle above.

channel 3 duty cycle

See description of channel 1 duty cycle above.

channel 1 pulse count, P1

32 bit unsigned integer value giving the number of detected channel 1 radar power pulses P1, see »[:TRIGger]:RADar:COUnt:P[1]/:P2/:P3/:ALL? [<MPMeter>]«. If there is no valid calibration data, if the trigger system state is not equal to DONE or if there are no channel 1 radar pulses detected above the set threshold R1 is set to zero.

channel 2 pulse count, P2

See description of channel 1 pulse count above.

channel 3 pulse count, P3

See description of channel 1 pulse count above.

channel 1 pulse start indexes

P1 32 bit unsigned integer values giving a list of start indexes of all channel 1 pulses expressed as samples relative to the beginning of the waveform, see »[:TRIGger]:RADar:PULses:STArt:P[1]/:P2/:P3? [<MPMeter>]«.

channel 1 pulse lengths

P1 32 bit unsigned integer values giving a list of sample counts of all channel 1 pulses expressed as samples, see »[:TRIGger]:RADar:PULses:LENgth:P[1]/:P2/:P3? [<MPMeter>]«.

channel 1 pulse powers arithmetic mean

P1 32 bit single-precision, floating-point values giving a list of the arithmetic mean values of the channel 1 power values belonging to each pulse, see »[:TRIGGER]:RADar:PULses[:APOWer]:P[1]/:P2/:P3? [<MPMeter>]«.

channel 2 pulse start indexes

P2 32 bit unsigned integer values, as described for channel 1 pulse start indexes above.

channel 2 pulse lengths

P2 32 bit unsigned integer values, as described for channel 1 pulse lengths above.

channel 2 pulse powers arithmetic mean

P2 32 bit single-precision, floating-point values, as described for channel 1 pulse powers arithmetic mean above.

channel 3 pulse start indexes

P3 32 bit unsigned integer values, as described for channel 1 pulse start indexes above.

channel 3 pulse lengths

P3 32 bit unsigned integer values, as described for channel 1 pulse lengths above.

channel 3 pulse powers arithmetic mean

P3 32 bit single-precision, floating-point values, as described for channel 1 pulse powers arithmetic mean above.

waveform sample count

32 bit unsigned integer value S giving the number of samples for waveform output of the corresponding power meter. If waveform output is disabled, this value is set to zero.

channel 1 power waveform

S 32 bit single-precision, floating-point values giving a list of channel 1 power values in dBm of the power waveform after applying the moving average filter, see »[:TRIGGER]:RADar:WPower:P[1]/:P2/:P3/:ALL? [<MPMeter>]« for details.

channel 2 power waveform

S 32 bit single-precision, floating-point values as described for channel 1 power waveform above.

channel 3 power waveform

S 32 bit single-precision, floating-point values as described for channel 1 power waveform above.

## 9.9 [:TRIGger]:SWEEP, Commands

### 9.9.1 [:TRIGger]:SWEEP:TStep <TStep>[,<MPMeter>]

Set number of samples per sweep step for one or multiple power meters.

**Parameters:**

The unsigned, non-zero integer-valued parameter of the command specifies the number of samples per sweep step within the power waveform, dividing it into as many sections as will fit into the waveform, starting with the first sample of the waveform.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

**9.9.2 [:TRIGger]:SWEEP:TStep? [<MPMeter>]**

Query number of samples per sweep step for one or multiple power meters.

**Parameter:**

The optional unsigned integer parameter MPMeter is described in Section 9.1.

**Return value:**

The command returns the unsigned integer-valued number of samples per sweep step, corresponding to the value set via »[:TRIGger]:SWEEP:TStep <TStep>[,<MPMeter>]«, the default value is 1000 samples. If executed for multiple power meters the command returns a list of values for each power meter of the respective list.

**9.9.3 [:TRIGger]:SWEEP:TCNT? [<MPMeter>]**

Query number of sweep step for present power waveform for one or multiple power meters.

**Parameter:**

The optional unsigned integer parameter MPMeter is described in Section 9.1.

**Return value:**

The command returns the unsigned integer-valued number of sweep steps for the present power waveform, corresponding to the length of the full waveform queried via »[:TRIGger]:FLENght? [<MPMeter>]« divided by the sweep step length set via »[:TRIGger]:SWEEP:TStep <TStep>[,<MPMeter>]«. If executed for multiple LSPM 1.0 Power Meters the command returns a list of values for each power meter of the respective list.

**9.9.4 [:TRIGger]:SWEEP:TBegin <TBegin>[,<MPMeter>]**

Set the index of the first sample of the averaged portion of each sweep step for one or multiple power meters.

**Parameters:**

The unsigned integer-valued parameter of the command sets the index of the first value used for averaging in each sweep step.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.9.5 `[:TRIGger]:SWEEP:TBegin? [<MPMeter>]`

Query the index of the first sample of the averaged portion of each sweep step for one or multiple power meters.

Parameter:

The optional unsigned integer parameter `MPMeter` is described in Section 9.1.

Return value:

The command returns the unsigned integer-valued index of the first sample used for averaging in each sweep step, corresponding to the length set via `»[:TRIGger]:SWEEP:TBegin <TBegin>[,<MPMeter>]«`, the default value is 500. If executed for multiple power meters the command returns a list of indexes for each power meter of the respective list.

### 9.9.6 `[:TRIGger]:SWEEP:TEnd <TEnd>[,<MPMeter>]`

Set the index of the last sample of the averaged portion of each sweep step for one or multiple power meters.

Parameters:

The unsigned integer-valued parameter of the command sets the index of the last value used for averaging in each sweep step.

The second, optional unsigned integer parameter `MPMeter` is described in Section 9.1.

### 9.9.7 `[:TRIGger]:SWEEP:TEnd? [<MPMeter>]`

Query the index of the last sample of the averaged portion of each sweep step for one or multiple power meters.

Parameter:

The optional unsigned integer parameter `MPMeter` is described in Section 9.1.

Return value:

The command returns the unsigned integer-valued index of the last sample used for averaging in each sweep step, corresponding to the length set via `»[:TRIGger]:SWEEP:TEnd <TEnd>[,<MPMeter>]«`, the default value is 899. If executed for multiple power meters the command returns a list of indexes for each power meter of the respective list.

### 9.9.8 `[:TRIGger]:SWEEP:MODE <Mode>[,<MPMeter>]`

Set frequency sweep mode for power waveform evaluation for one or multiple power meters.

**Parameter:**

String parameter without quotes specifying the sweep mode, valid values are FIXED, LIN, LOG and LIST. When set to LIN or LOG the sweep must be parametrized via the SCPI commands »[:TRIGger]:SWeep:BEgin <Freq>[,<MPMeter>]«, »[:TRIGger]:SWeep:STEP <Step>[,<MPMeter>]« and »[:TRIGger]:SWeep:COUnt <Count>[,<MPMeter>]«. When set to LIST an arbitrary frequency list with a non-zero number of frequencies must be set via the SCPI command »[:TRIGger]:SWeep:ARBAdd <Freq>[,<MPMeter>]«. When set to FIXED the frequency set via »:SYSTem:FREQuency <Frequency>[,<MPMeter>]« is used for all sweep steps.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.9.9 [:TRIGger]:SWeep:MODE? [<MPMeter>]

Query frequency sweep mode for power waveform evaluation for one or multiple power meters.

**Parameter:**

The optional unsigned integer parameter MPMeter is described in Section 9.1.

**Return value:**

The command returns a string value without quotes specifying the sweep mode, see »[:TRIGger]:SWeep:MODE <Mode>[,<MPMeter>]« for more details. If executed for multiple power meters the command returns a list of modes of all power meters of the respective list.

### 9.9.10 [:TRIGger]:SWeep:BEgin <Freq>[,<MPMeter>]

Set frequency of first sweep step for linear and logarithmic frequency sweeps for one or multiple power meters.

**Parameters:**

The double-precision, floating-point valued parameter of the command specifies the frequency of the first sweep step for linear and logarithmic sweeps in hertz.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.9.11 [:TRIGger]:SWeep:BEgin? [<MPMeter>]

Query frequency of first sweep step for linear and logarithmic frequency sweeps for one or multiple power meters.

**Parameter:**

The optional unsigned integer parameter MPMeter is described in Section 9.1.

**Return value:**

The command returns the double-precision, floating-point valued frequency of the first sweep



step in hertz for linear and logarithmic frequency sweeps, corresponding to the value set by »[:TRIGger]:SWeep:BEgin <Freq>[,<MPMeter>]«. The default value is 100 MHz. If executed for multiple power meters the command returns a list of floating-point values indicating the frequency of the first sweep step for each power meters.

#### 9.9.12 [:TRIGger]:SWeep:COUnt <Count>[,<MPMeter>]

Set number of frequency steps for linear and logarithmic frequency sweeps for one or multiple power meters.

Parameters:

The unsigned integer-valued parameter of the command specifies the number of frequency steps of the frequency sweep.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

#### 9.9.13 [:TRIGger]:SWeep:COUnt? [<MPMeter>]

Query number of frequency steps for linear and logarithmic frequency sweeps for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns the unsigned integer-valued number of frequency steps set via »[:TRIGger]:SWeep:COUnt <Count>[,<MPMeter>]«. The default is 10 steps. If the sweep mode is set to neither LIN nor LOG, NAN will be returned. If executed for multiple power meters the command returns a list of unsigned integer-valued number of frequency steps for each power meters.

#### 9.9.14 [:TRIGger]:SWeep:STEP <Step>[,<MPMeter>]

Set the incremental frequency step for linear and logarithmic frequency sweeps for one or multiple power meters.

Parameters:

The double-precision, floating-point valued parameter of the command specifies the frequency increment. For linear frequency sweeps the parameter gives the frequency increment from one sweep step to the next in hertz. For logarithmic frequency sweeps the parameter specifies the incremental factor from one sweep step to the next.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.9.15 [:TRIGger]:SWeep:STEP? [<MPMeter>]

Query the incremental frequency step for linear and logarithmic frequency sweeps for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns the double-precision, floating-point valued linear frequency increment in hertz with up to three decimal places, or factor from one sweep step to the next by »[:TRIGger]:SWeep:STEP <Step>[,<MPMeter>]«, the default value is 1.1. If the sweep mode is set to neither LIN nor LOG, NAN will be returned. If executed for multiple power meters the command returns a list of floating-point valued numbers for each power meter.

### 9.9.16 [:TRIGger]:SWeep:ARBAdd <Freq>[,<MPMeter>]

Append single frequency to the list of arbitrary sweep frequencies for one or multiple power meters.

Parameters:

The double-precision, floating-point valued parameter of the command specifies the frequency to be appended to the arbitrary frequency list in hertz.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.9.17 [:TRIGger]:SWeep:ARBClear [<MPMeter>]

Clear the list of arbitrary sweep frequencies for one or multiple power meters.

Parameters:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.9.18 [:TRIGger]:SWeep:ARBitrary? [<MPMeter>]

Query the arbitrary list of frequencies used for the sweep evaluation in LIST mode of a power waveform for one or multiple power meters.

Parameters:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

The command returns a double-precision, floating-point valued list of frequency steps used for sweep evaluation in list mode. The number of frequency steps is the number of frequencies added via the SCPI command »[:TRIGger]:SWeep:ARBAdd <Freq>[,<MPMeter>]«. If the number of arbitrary list frequencies is zero the query will return NAN. If executed for multiple power meters the command returns a list of frequencies for each E-field probe.

### 9.9.19 [:TRIGger]:SWEEP:LIST? [<MPMeter>]

Query the list of frequencies used for the sweep evaluation of a power waveform for one or multiple power meters.

Parameters:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

The command returns a double-precision, floating-point valued list of frequency steps in hertz, with up to three decimal places used for sweep evaluation. The number of frequency steps is the number of samples of the waveform divided by the number of samples per sweep step, rounded down to the nearest integer number, see »:TRIGger:FLENght? [<MPMeter>]« and »[:TRIGger]:SWEEP:TStep <TStep>[,<MPMeter>]«. Frequency values are applied according to the set sweep mode and its parameters beginning with the start frequency for linear and logarithmic sweeps. For arbitrary list sweeps the list's values will be applied. A constant frequency is applied in fixed sweep mode. If the number of frequency steps that fit into the waveform exceeds the number of sweep steps the list of frequency steps will be applied repeatedly until a frequency value has been assigned to each sweep step. If the number of sweep steps is zero the query will return NAN. If executed for multiple power meters the command returns a list of frequencies for each power meter.

### 9.9.20 [:TRIGger]:SWEEP:IDX? [<MPMeter>]

Query the center indices of the averaged portions of each sweep step for one or multiple power meters.

Parameters:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

The command returns a comma separated, integer-valued list of index values of the power waveform. The indices give the arithmetic mean of the first and last index used for averaging of each sweep step, see »[:TRIGger]:SWEEP:TBegin <TBegin>[,<MPMeter>]« and »[:TRIGger]:SWEEP:TEnd <TEnd>[,<MPMeter>]«. The index is especially useful for overlaying power waveforms and averaged sweep values. If executed for multiple power meters the command returns a list of indices for each power meter.

### 9.9.21 [:TRIGger]:SWEEP[:P1]/:P2/:P3/[:ALL]ower]:P? [<MPMeter>]

Query averaged channel power value for each sweep step of the power waveform for one or multiple power meters, for one of:

:P[1]?/:P2?/:P3?

averaged channel 1, 2 or 3 power for all sweep steps

:ALL?

averaged channel 1, 2 and 3 power for all sweep steps

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

The command returns a comma-separated list of float-valued averaged channel 1, 2 or 3 power values for each sweep step of the power waveform in dBm. If ALL values are queried, a multiple of three values is returned, consisting of the averaged channel 1, 2 and 3 power values for each sweep step. NAN will be returned if the trigger system's state is not equal to DONE, if there are no valid sweep frequencies or if there is no valid calibration data for this frequency step.

### 9.9.22 [:TRIGger]:SWEEP:RSSI:P[1]/:P2/:P3/[:ALL]? [<MPMeter>]

Query averaged channel RSSI value for each sweep step of the power waveform for one or multiple power meters, for one of:

:P[1]?/:P2?/:P3?

averaged channel 1, 2 or 3 RSSI value for all sweep steps

:ALL?

averaged channel 1, 2 and 3 RSSI values for all sweep steps

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

The command returns a comma-separated list of integer-valued averaged channel 1, 2 or 3 RSSI values for each sweep step of the power waveform in LSB. If ALL values are queried, a multiple of three values is returned, consisting of the averaged channel 1, 2 and 3 RSSI values for each sweep step. NAN will be returned if the trigger system's state is not equal to DONE or if there are no valid sweep frequencies.

### 9.9.23 [:TRIGger]:SWEEP:WPOWER:P[1]/:P2/:P3/[:ALL]? [<MPMeter>]

Query power values of frequency corrected power waveform using the configured sweep frequencies for one or multiple power meters.

Return results for one of:

:P[1]?/:P2?/:P3?

channel 1, 2 or 3 power waveform values

:ALL?

channel 1, 2 and 3 power waveform values

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

The command returns a list of float-valued channel 1, 2 or 3 power values of the sweep frequency corrected power waveform in dBm. If ALL values are queried, a multiple of three values is returned, consisting of the channel 1, 2 and 3 power values for each sample in the waveform. NAN will be returned if there is no valid calibration data, if the trigger system state is not the state DONE or if there are no valid sweep frequencies.

### 9.9.24 [:TRIGger]:SWEEP:BINary?

Query averaged and unaveraged sweep corrected channel 1, 2 and 3 power values, and center indices of averaged sweep values in binary format for one or multiple power meters.

Parameters:

The first mandatory integer-valued parameter Wave controls the output of sweep corrected power waveform values. If set to 1 waveform output is enabled, if set to 0 waveform is disabled.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

Return values:

Binary data block followed by a carriage return, newline sequence. The first four bytes specify the number of bytes of the binary data block which will be returned following the first four bytes.

For each power meter a chunk of binary data will be sent. All values are encoded in little endian format. If the power meter is not defined the power meter serial number, and sweep step count are set to zero and the binary data block ends. Data are ordered as follows:

power meter number

32 bit unsigned integer value giving the serial number of the corresponding power meter.

sweep step count

32 bit unsigned integer value giving the number of sweep steps S1 in the power waveform of the corresponding power meter. If there is no valid calibration data, if the trigger system state is not equal to DONE or if there are no valid sweep frequencies S1 is set to zero and the binary data block ends.

sample count

32 bit unsigned integer value giving the number of samples S2 in the sweep corrected power waveform of the corresponding power meter.

## index

S1 32 bit unsigned integer values giving a list of center index values for the averaged portion of each sweep step of the power waveform as described in »[:TRIGger]:SWeep:IDX? [<MPMeter>]«.

## Frequencies

S1 64 bit double-precision, floating-point values giving a list of the sweep frequency values for the averaged portion of each sweep step of the power waveform as described in »[:TRIGger]:SWeep:LIST? [<MPMeter>]«.

## channel 1 averaged power

S1 32 bit single-precision, floating-point values giving a list of averaged channel 1 power values in dBm of each sweep step within the power waveform, see »[:TRIGger]:SWeep:P[1]/:P2/:P3/[:ALL]ower]:P? [<MPMeter>]«.

## channel 2 averaged power

S1 32 bit single-precision, floating-point values giving a list of averaged channel 2 power values in dBm of each sweep step within the power waveform, see »[:TRIGger]:SWeep:P[1]/:P2/:P3/[:ALL]ower]:P? [<MPMeter>]«.

## channel 3 averaged power

S1 32 bit single-precision, floating-point values giving a list of averaged channel 3 power values in dBm of each sweep step within the power waveform, see »[:TRIGger]:SWeep:P[1]/:P2/:P3/[:ALL]ower]:P? [<MPMeter>]«.

## channel 1 averaged RSSI

S1 32 bit single-precision, floating-point values, giving a list of averaged channel 1 RSSI values in LSB of each sweep step of the power waveform, see »[:TRIGger]:SWeep:RSsi:P[1]/:P2/:P3/[:ALL]? [<MPMeter>]«.

## channel 2 averaged RSSI

S1 32 bit single-precision, floating-point values, giving a list of averaged channel 2 RSSI values in LSB of each sweep step of the power waveform, see »[:TRIGger]:SWeep:RSsi:P[1]/:P2/:P3/[:ALL]? [<MPMeter>]«.

## channel 3 averaged RSSI

S1 32 bit single-precision, floating-point values, giving a list of averaged channel 3 RSSI values in LSB of each sweep step of the power waveform, see »[:TRIGger]:SWeep:RSsi:P[1]/:P2/:P3/[:ALL]? [<MPMeter>]«.

## channel 1 power waveform

S2 32 bit single-precision, floating-point values giving a list of sweep frequency corrected channel 1 power values in dBm, see »[:TRIGger]:SWeep:WPower:P[1]/:P2/:P3/[:ALL]? [<MPMeter>]«.

## channel 2 power waveform

S2 32 bit single-precision, floating-point values giving a list of sweep frequency corrected channel 2 power values in dBm, see »[:TRIGger]:SWeep:WPower:P[1]/:P2/:P3/[:ALL]? [<MPMeter>]x.

channel 3 power waveform

S2 32 bit single-precision, floating-point values giving a list of sweep frequency corrected channel 3 power values in dBm, see »[:TRIGger]:SWEEP:WPower:P[1]/:P2/:P3/[:ALL]? [<MPMeter>]«.

## 9.10 :STATistics Commands

### 9.10.1 :STATistics:MAster <State>

Set currently selected power meter to be the master or a slave power meter for continuous statistics collection. The active power meter is set using »:SYSTem:SERial <Value>«. By default the first enumerated power meter automatically becomes the continuous statistics master power meter.

Parameter:

Setting State to 1 makes the current power meter the master of the continuous statistics subsystem. A State of 0 makes the power meter a slave of the continuous statistics subsystem, i.e., continuous statistics will be controlled by a different power meter.

### 9.10.2 :STATistics:MAster? [<MPMeter>]

Query statistics subsystem master/slave status of the currently active power meter. By default the first enumerated power meter automatically becomes the continuous statistics master power meter.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns an unsigned integer value containing the master/slave status of the currently active power meter. Slaves return 0, the master returns 1.

### 9.10.3 :STATistics:ENable <State>[,<MPMeter>]

Enable or disable statistics acquisition for statistics subsystem master power meter. This command is only effective for power meters configured as the statistics subsystem master power meter, see »:STATistics:MAster <State>«. Enabling statistics acquisition resets the snapshot counter queried via »:STATistics:COUnT? [<MPMeter>]«.

Parameters:

Setting State to 1 activates statistics acquisition, setting State to 0 disables statistics acquisition for one or multiple power meters. Changing the state from disabled to enabled will reset and start statistics collection. Changing the state from enabled to disabled will trigger an automatic snapshot identical to issuing »:STATistics:SNAPshot [<Triggered>][,<MPMeter>]« and stop statistics collection, see also »:STATistics:SNAPshot [<Triggered>][,<MPMeter>]«.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

#### 9.10.4 :STATistics:ENable? [<MPMeter>]

Query status of statistics acquisition for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns an unsigned integer value containing the status of statistics acquisition. A value of 1 is returned when statistics acquisition is enabled, 0 is returned if statistics acquisition is disabled. The enable state of the statistics subsystem is controlled by the statistics subsystem master, see »:STATistics:MAster <State>« and »:STATistics:ENable <State>[,<MPMeter>]«. All statistics subsystem slave power meters are controlled by the statistics master power meter, their return value is thus always be identical to the master power meter if connected correctly.

#### 9.10.5 :STATistics:SNAPshot [<Triggered>][,<MPMeter>]

Create a snapshot of either continuously collected statistics, or of waveforms recorded by the trigger subsystem for one or multiple power meters.

Parameters:

The optional integer-valued parameter Triggered selects the source for the statistics snapshot. If the parameter is omitted or set to 0 a snapshot of the continuously acquired statistics is created for subsequent analysis. This type of statistics snapshot is triggered by the power meter configured as the statistics subsystem master, see »:STATistics:MAster <State>«. Additionally, statistics acquisition must be enabled using »:STATistics:ENable <State>[,<MPMeter>]« before creating a snapshot.

If the parameter Triggered is set to 1 the most recently acquired triggered waveforms are analyzed to obtain a statistics snapshot for subsequent analysis. This kind of snapshot can only be created for one power meter or multiple power meters at a time, see the description of the second parameter below.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1. If the parameter MPMeter is set the parameter Triggered is mandatory. For continuous statistics only power meters configured as the statistics master power meter will output a snapshot trigger signal.

#### 9.10.6 :STATistics:COUnt? [<MPMeter>]

Return continuous statistics snapshot counter for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.



Return value:

The command returns an unsigned integer value giving the number of snapshots taken for the selected power meter since the last enabling of statistics acquisition.

### 9.10.7 :STATistics:RESolution <Resolution>[,<MPMeter>]

Set resolution for histograms and distribution functions for one or multiple power meters.

Parameters:

The float-valued parameter Resolution specifies the power value resolution in dB for all statistics query commands returning histograms and distribution functions. E.g., a value of 1.0 will output histograms with a bin size of 1 dB. Bins are aligned relative to and centered around 0 dBm. The smallest permissible value for Resolution is 0.005 dB.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.10.8 :STATistics:RESolution? [<MPMeter>]

Query resolution in dB for histograms and distribution functions for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns a float value giving the power resolution in dB for all statistics query commands returning histograms and distribution functions.

### 9.10.9 :STATistics:HISTogram:SIZE? [<Triggered>][,<MPMeter>]

Query number of bins for histograms and distribution functions for one or multiple power meters.

Parameters:

If no parameter is provided or if Triggered is set to 0 the number of bins for the most recently created snapshot histograms and distribution functions based on continuous statistics acquisition is returned. If the parameter Triggered is set to 1 the number of bins for the most recently created snapshot histograms and distribution functions based on triggered waveforms is returned.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1. If the parameter MPMeter is set the parameter Triggered is mandatory.

Return value:

The command returns an unsigned integer value giving the number of histogram bins for the set resolution, see »:STATistics:RESolution <Resolution>[,<MPMeter>]«. NAN will be returned if there is no valid statistics snapshot data or triggered waveform data.

### 9.10.10 :STATistics:HISTogram:OFFset? [<Triggered>][,<MPMeter>]

Query offset of the first bin of all histograms and distribution functions, expressed as a multiple of the histogram's resolution, i.e., bin size, relative to 0 dBm, for one or multiple power meters.

Parameters:

If no parameter is provided or if Triggered is set to 0 the offset for the most recently created snapshot histograms and distribution functions based on continuous statistics acquisition is returned. If the parameter Triggered is set to 1 the offset for the most recently created snapshot histograms and distribution functions based on triggered waveforms is returned.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1. If the parameter MPMeter is set the parameter Triggered is mandatory.

Return value:

The command returns an integer value for the presently set resolution, see »:STATistics:RESolution <Resolution>[,<MPMeter>]«. E.g., when the resolution is set to 1 dB, a return value of -20 indicates that the first bin of the histogram covers -20 dBm  $\pm 0.5$  dB. NAN will be returned if there is no valid statistics snapshot data or triggered waveform data.

### 9.10.11 :STATistics:SAMples? [<Triggered>][,<MPMeter>]

Query number of sample values used for statistics acquisition for one or multiple power meters.

Parameters:

If no parameter is provided or if Triggered is set to 0 the number of samples per axis used for the most recently created statistics snapshot based on continuous statistics acquisition is returned. If the parameter Triggered is set to 1 the number of samples per axis used for the most recently created statistics snapshot based on triggered waveforms is returned, i.e., the number of samples returned by »:TRIGger:LENgth <Length>[,<MPMeter>]«.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1. If the parameter MPMeter is set the parameter Triggered is mandatory.

Return value:

The command returns an unsigned 64 bit integer value giving the number of samples used to build the respective histogram. NAN will be returned if there is no valid statistics snapshot data or triggered waveform data.

### 9.10.12 :STATistics:MINimum:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]

Query minimum power of the most recent statistics snapshot or triggered waveform for one or multiple power meters.

Return results for one of:

:P[1]?/:P2?:P3?

channel 1, 2 or 3 power waveform values

:ALL?

channel 1, 2 and 3 power waveform values as a list

Parameters:

If no parameter is provided or if Triggered is set to 0 the channel 1 minimum power for the most recently created statistics snapshot based on continuous statistics acquisition is returned. If the parameter Triggered is set to 1 the channel 1 minimum power of the most recently created statistics snapshot based on triggered waveforms is returned.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1. If the parameter MPMeter is set the parameter Triggered is mandatory.

Return value:

The command returns a float-valued channel 1 minimum power in dBm. NAN will be returned if there is no valid statistics snapshot data or triggered waveform data.

#### 9.10.13 :STATistics:MAXimum:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]

Query maximum power of the most recent statistics snapshot or triggered waveform for one or multiple power meters, see »:STATistics:MINimum:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]« for description of parameters and return values.

#### 9.10.14 :STATistics:MEAN:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]

Query arithmetic mean power of the most recent statistics snapshot or triggered waveform for one or multiple power meters, see »:STATistics:MINimum:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]« for description of parameters and return values.

#### 9.10.15 :STATistics:RMS:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]

Query root mean square power of the most recent statistics snapshot or triggered waveform for one or multiple power meters, see »:STATistics:MINimum:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]« for description of parameters and return values.

#### 9.10.16 :STATistics:SDEVIation:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]

Query standard deviation power of the most recent statistics snapshot or triggered waveform for one or multiple power meters, see »:STATistics:MINimum:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]« for description of parameters and return values.

### 9.10.17 :STATistics:Power? [<Triggered>][,<MPMeter>]

Query center power value of bins used by histograms and distribution functions for one or multiple power meters.

Parameters:

If no parameter is provided or if Triggered is set to 0 the center power values of all bins for the most recently created statistics snapshot based on continuous statistics acquisition are returned. If the parameter Triggered is set to 1 the center power values of the most recently created statistics snapshot based on triggered waveforms are returned.

The second, optional unsigned integer parameter MPMeter is describe in Section 9.1. If the parameter MPMeter is set the parameter Triggered is mandatory.

Return values:

The command returns a comma-separated list of float-valued power values in dBm. NAN will be returned if there is no valid statistics snapshot data or triggered waveform data.

### 9.10.18 :STATistics:HISTogram:P[1]/:P2/:P3? [<Triggered>][,<MPMeter>]

Query channel 1, 2 or 3 histogram for one or multiple power meters.

Parameters:

If no parameter is provided or if Triggered is set to 0 the channel 1 histogram for the most recently created statistics snapshot based on continuous statistics acquisition is returned. If the parameter Triggered is set to 1 the channel 1 histogram of the most recently created statistics snapshot based on triggered waveforms is returned.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1. If the parameter MPMeter is set the parameter Triggered is mandatory.

Return values:

The command returns a comma-separated list of unsigned 64 bit integer values specifying the number of samples of a power value falling into the associated power value bins returned by »:STATistics:Power? [<Triggered>][,<MPMeter>]«. The bin size is specified by »:STATistics:RESolution <Resolution>[,<MPMeter>]«. NAN will be returned if there is no valid statistics snapshot data or triggered waveform data.

### 9.10.19 :STATistics:P[1]/:P2/:P3DF:P? [<Triggered>][,<MPMeter>]

Query channel 1, 2 or 3 discrete relative probability distribution of the most recent statistics snapshot or triggered waveform for one or multiple power meters.

Parameters:

See »:STATistics:HISTogram:P[1]/:P2/:P3? [<Triggered>][,<MPMeter>]« parameter description for details.

Return values:

The command returns a list of float-valued discrete relative probabilities of channel 1, 2 or 3 power values. Each value is associated with a power value bin returned by »:STATistics:Power? [<Triggered>][,<MPMeter>]«. NAN will be returned if there is no valid statistics snapshot data or triggered waveform data.

#### 9.10.20 :STATistics:CDF:P[1]/:P2/:P3? [<Triggered>][,<MPMeter>]

Query channel 1, 2 or 3 discrete cumulative probability distribution of the most recent statistics snapshot or triggered waveform for one or multiple power meters, see »:STATistics:P[1]/:P2/:P3DF:P? [<Triggered>][,<MPMeter>]« for description of parameters and return values.

#### 9.10.21 :STATistics:CCDF:P[1]/:P2/:P3? [<Triggered>][,<MPMeter>]

Query channel 1, 2 or 3 discrete complementary cumulative probability distribution of the most recent statistics snapshot or triggered waveform for one or multiple power meters, see »:STATistics:P[1]/:P2/:P3DF:P? [<Triggered>][,<MPMeter>]« for description of parameters and return values.

#### 9.10.22 :STATistics:BINary? [<Triggered>][,<MPMeter>]

Query all statistical values of the most recent statistics snapshot or triggered waveform in binary format for one or multiple power meters. This command can be used to reduce communications overhead when polling statistical values via software.

Parameters:

See »:STATistics:MINimum:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]« parameter description for details.

Return values:

Binary data block followed by a carriage return, newline sequence. The first four bytes specify the number of bytes of the binary data block which will be returned following the first four bytes.

For each power meter P a chunk of binary data will be sent. All values are encoded in little endian format. Data are ordered as follows:

power meter serial number

32 bit unsigned integer value giving the serial number of the corresponding power meter.

If the power meter P is not defined the power meter serial number and bin count are set to zero and the binary data block ends.

#### bin count

Four bytes specifying the number of bins  $N$  contained in the following binary data, the value is a 32 bit unsigned integer value, as described in »:STATistics:HISTogram:SIZE? [<Triggered>][,<MPMeter>]«. If there is no valid statistics snapshot data  $N$  will have a value of zero and no further data will be returned for the binary data chunk.

#### histogram offset

32 bit signed integer value as described in »:STATistics:HISTogram:OFFSET? [<Triggered>][,<MPMeter>]«.

#### sample count

64 bit unsigned integer value as described in »:STATistics:SAMPLES? [<Triggered>][,<MPMeter>]«.

#### resolution

32 bit single-precision, floating-point value as described in »:STATistics:RESOLUTION? [<MPMeter>]«.

#### minimum

Three 32 bit single-precision, floating-point values as described in »:STATistics:MINimum:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]«.

#### maximum

Three 32 bit single-precision, floating-point values as described in »:STATistics:MAXimum:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]«.

#### arithmetic mean

Three 32 bit single-precision, floating-point values as described in »:STATistics:MEAN:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]«.

#### root mean square

Three 32 bit single-precision, floating-point values as described in »:STATistics:RMS:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]«.

#### standard deviation

Three 32 bit single-precision, floating-point values as described in »:STATistics:SDEViation:P[1]/:P2/:P3/[:ALL]? [<Triggered>][,<MPMeter>]«.

#### bins

$N$  32 bit single-precision, floating-point values as described in »:STATistics:Power? [<Triggered>][,<MPMeter>]«.

#### histogram, channel 1

$N$  64 bit unsigned integer values as described in »:STATistics:HISTogram:P[1]/:P2/:P3? [<Triggered>][,<MPMeter>]«.

#### histogram, channel 2

$N$  64 bit unsigned integer values as described in »:STATistics:HISTogram:P[1]/:P2/:P3? [<Triggered>][,<MPMeter>]«.

histogram, channel 3

N 64 bit unsigned integer values as described in »:STATistics:HISTogram:P[1]/:P2/:P3? [<Triggered>][,<MPMeter>]«.

relative probability, channel 1

N 32 bit single-precision, floating-point values as described in »:STATistics:P[1]/:P2/:P3DF:P? [<Triggered>][,<MPMeter>]«.

relative probability, channel 2

N 32 bit single-precision, floating-point values as described in »:STATistics:P[1]/:P2/:P3DF:P? [<Triggered>][,<MPMeter>]«.

relative probability, channel 3

N 32 bit single-precision, floating-point values as described in »:STATistics:P[1]/:P2/:P3DF:P? [<Triggered>][,<MPMeter>]«.

relative cumulative probability, channel 1

N 32 bit single-precision, floating-point values as described in »:STATistics:CDF:P[1]/:P2/:P3? [<Triggered>][,<MPMeter>]«.

relative cumulative probability, channel 2

N 32 bit single-precision, floating-point values as described in »:STATistics:CDF:P[1]/:P2/:P3? [<Triggered>][,<MPMeter>]«.

relative cumulative probability, channel 3

N 32 bit single-precision, floating-point values as described in »:STATistics:CDF:P[1]/:P2/:P3? [<Triggered>][,<MPMeter>]«.

relative complementary cumulative probability, channel 1

N 32 bit single-precision, floating-point values as described in »:STATistics:CCDF:P[1]/:P2/:P3? [<Triggered>][,<MPMeter>]«.

relative complementary cumulative probability, channel 2

N 32 bit single-precision, floating-point values as described in »:STATistics:CCDF:P[1]/:P2/:P3? [<Triggered>][,<MPMeter>]«.

relative complementary cumulative probability, channel 3

N 32 bit single-precision, floating-point values as described in »:STATistics:CCDF:P[1]/:P2/:P3? [<Triggered>][,<MPMeter>]«.

## 9.11 :MProbe Commands

### 9.11.1 :MPMeter:SERial <MPMeter>,<SN1>[,<SN2>,...,<SNN>]

Define a LSPM 1.0 Multi-Power Meter System by specifying one or multiple LSPM 1.0 Power Meter serial numbers.

#### Parameters:

The first unsigned integer parameter MPMeter sets the LSPM 1.0 Multi-Power Meter System number for a LSPM 1.0 Multi-Power Meter System setup. MPMeter must be greater than zero. MPMeter does not specify the number of power meters in a LSPM 1.0 Multi-Power Meter System.

The following unsigned integer parameters SN1 through SNN specify the LSPM 1.0 Power Meter serial numbers of a LSPM 1.0 Multi-Power Meter System setup. Individual power meters may be referenced multiple times by one or more LSPM 1.0 Multi-Power Meter Systems. Power meter serial numbers must be set to one of the enumerated power meters that can be queried via »:SYSTEM:SERial? [<MPMeter>]«, unknown power meter serial numbers will cause empty output of multi power meter commands.

### 9.11.2 :MPMeter:SERial? [<MPMeter>]

Query power meter serial number(s) for LSPM 1.0 Multi-Power Meter Systems. This command is an alias of »:SYSTEM:SERial? [<MPMeter>]« when the latter is used with the same MPMeter parameter.

#### Parameters:

The unsigned integer parameter MPMeter specifies LSPM 1.0 Multi-Power Meter System number defined either automatically, as for setup number 0, or via »:MPMeter:SERial [<MPMeter>,<SN1>,<SN2>,...,<SNN>]« for setup numbers greater than zero.

#### Return values:

Comma-separated list of unsigned integers indicating the LSPM 1.0 Power Meter serial numbers for the LSPM 1.0 Multi-Power Meter System specified by the parameter MPMeter. NAN will be returned instead of the power meter serial number if the specified LSPM 1.0 Multi-Power Meter System does not exist. NAN will be returned if the specified LSPM 1.0 Multi-Power Meter System has not been configured.

## 9.12 :VIRTual Power Meter Commands

### 9.12.1 :VIRTual:SERial?

Query serial numbers of connected virtual power meters.

#### Return values:

Unsigned integer-valued comma-separated list of all connected virtual power meter serial numbers. If no virtual power meters are connected the command will return NAN.

### 9.12.2 :VIRTual:CONnect [<SER>]

Connect a new virtual power meter.



Parameter:

The optional unsigned integer parameter power meter specifies the serial number of the virtual power meter. If omitted the default serial number is set to 1.

### 9.12.3 :VIRTual:DISConnect

Disconnect currently active power meter if it is a virtual power meter.

### 9.12.4 :VIRTual:CW <RSSI1>,<RSSI2>,<RSSI3>

Set channel 1, 2 and 3 CW RSSI values of the currently selected virtual power meter.

Parameters:

The three unsigned integer parameters for the channel 1, 2 and 3 RSSI values set the signal strength indicated by the virtual power meter's ADCs. The default values are 0.

### 9.12.5 :VIRTual:CW?

Query channel 1, 2 and 3 RSSI values of the currently selected virtual power meter.

Return values:

The command returns a comma-separated list of three unsigned integer values giving the channel 1, 2 and 3 RSSI values. NAN will be returned if the active power meter is not virtual.

### 9.12.6 :VIRTual:NOIse <NOISE1>,<NOISE2>,<NOISE3>

Set the added noise amplitude of the currently selected virtual power meter.

Parameters:

The three unsigned integer parameters for channel 1, 2 and 3 NOISE set the maximum added RSSI value noise amplitude. The time-average of the values is zero. Ranges are distributed evenly between -1 times the given amplitudes and +1 times the given amplitudes. The default values are 0.

### 9.12.7 :VIRTual:NOIse?

Query channel 1, 2 and 3 noise amplitude of the currently selected virtual power meter.

Return values:

The command returns a comma-separated list of three unsigned integer values giving the maximum amplitude of added channel 1, channel 2 and channel 3 noise in LSB. NAN will be returned if the active power meter is not virtual.

### 9.12.8 :VIRTual:PULse [<RSSI1>],[<RSSI2>],[<RSSI3>],[<T>],[<Ton>]

Set the parameters of the virtual pulse signal for currently selected virtual power meter.

Parameters:

RSSI1

unsigned integer value setting the channel 1 RSSI pulse value

RSSI2

unsigned integer value setting the channel 2 RSSI pulse value

RSSI3

unsigned integer value setting the channel 3 RSSI pulse value

T

unsigned integer value setting the pulse period expressed as a number of samples

Ton

unsigned integer value setting the ON-time at the beginning of reach pulse period expressed as a number of samples

### 9.12.9 :VIRTual:PULse?

Query the pulse parameters of the currently selected virtual power meter.

Return values:

The command returns a comma-separated list of five unsigned integer values as described in the parameter's description of »:VIRTual:PULse [<RSSI1>],[<RSSI2>],[<RSSI3>],[<T>],[<Ton>]«. NAN will be returned if the active power meter is not virtual.

### 9.12.10 :VIRTual:PLIST <P1\_1>,<P2\_1>,<P3\_1>[,...,<P1\_N>,<P2\_N>,<P3\_N>]

Append sets of channel 1, channel 2 and channel 3 power values to list of the currently selected virtual power meter.

Parameters:

Multiples of three float-valued power values, specifying channel 1, 2 and 3 power values. The command accepts up to tree times 256 values. The power values are converted to RSSI values using the currently set mode and frequency. RSSI values will not be adjusted when mode or frequency are changed. Power values exceeding the calibrated signal range will be limited to the maximum or minimum calibrated value.

### 9.12.11 :VIRTual:PLIST?

Query the list of arbitrary power values of the currently selected virtual power meter.

Return values:

The command returns a comma separated, floating-point valued list of all channel 1, 2, and 3 power values of the arbitrary power meter value list. NAN will be returned if the active power meter is not virtual or if the list is empty.

### 9.12.12 :VIRTual:LIST <RSSI1\_1>,<RSSI2\_1>,<RSSI3\_1>[,...,<RSSI2\_N>,<RSSI3\_N>]

Append sets of channel 1, 2 and 3 RSSI values to list of the currently selected virtual power meter.

Parameters:

Multiples of three integer-valued RSSI, specifying the channel 1, channel 2 and channel 3 RSSI values. The command accepts up to three times 256 values.

### 9.12.13 :VIRTual:LIST?

Query the list of arbitrary RSSI values the currently selected virtual power meter.

Return values:

The command returns a comma separated, unsigned integer list of all channel 1, 2 and 3 RSSI values of the arbitrary power value list. NAN will be returned if the active power meter is not virtual or the list is empty.

### 9.12.14 :VIRTual:LCNt?

Query number of samples in arbitrary power value list of the currently selected virtual power meter.

Return value:

Unsigned integer-valued number of power samples in arbitrary power value list. NAN will be returned if the active power meter is not virtual.

### 9.12.15 :VIRTual:LClear

Clear arbitrary power value list of the currently selected virtual power meter.

## 9.13 :STReam Recording Commands

### 9.13.1 :STReam:MAster <State>

Set currently selected power meter to be the master or a slave power meter for stream recording.

Parameter:

Setting State to 1 makes the current power meter the master during stream recording. A State of 0 makes the power meter a slave during stream recording, i.e., stream synchronization will be controlled by a different power meter.

### 9.13.2 :STReam:MAster? [<MPMeter>]

Query master/slave status of the currently active power meter during.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns an unsigned integer value giving the stream recording master/slave status of the currently active power meter. Slaves return 0, the master returns 1.

### 9.13.3 :STReam:LENgth <Length>[,<MPMeter>]

Set number of samples to be recorded during stream recording for one or multiple power meters. The parameter can only be set when stream recording is inactive.

Parameters:

The unsigned integer-valued parameter of the command specifies the number of consecutive samples to be streamed. E.g., »:STReam:len 100« will record 100 consecutive samples. Setting Length to zero configures indefinite streaming, i.e., streaming needs to be terminated by issuing a »:STReam:ENable? [<MPMeter>]« command.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.13.4 :STReam:LENgth? [<MPMeter>]

Query number of stream samples to be recorded for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns the unsigned integer-valued number of samples to be streamed as specified using the »:STReam:LENgth <Length>[,<MPMeter>]« command. A value of zero indicates indefinite streaming. If executed for multiple power meters the command returns a list of unsigned integer-valued lengths of the number of streaming samples for each power meter of the list specified by the MPMeter parameter.

### 9.13.5 :STReam:ENable <State>[,<MPMeter>]

Enable or disable stream recording for one or multiple power meters.

Parameters:

Setting State to 1 activates stream recording, creating a new stream file for each addressed power meter. Setting State to 0 disables stream recording and closes the associated stream file(s). See Section 10.1.6 for details about the stream file format.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.13.6 :STReam:ENable? [<MPMeter>]

Query status of stream recording for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns an unsigned integer value containing the stream recording status. A value of 1 is returned when stream data acquisition is enabled, 0 is returned if stream data acquisition is disabled.

### 9.13.7 :STReam:PROgress? [<MPMeter>]

Query number of samples in current stream recording for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns an unsigned 64 bit integer value giving the number of samples that have been recorded for the selected power meter since the start of stream recording.

### 9.13.8 :STReam:SKIp <SkipCnt>[,<MPMeter>]

Set number of stream samples to be skipped for one or multiple power meters. Parameter may only be set if stream data acquisition is disabled.

Parameters:

The unsigned integer-valued parameter SkipCnt specifies the number of samples to be skipped after recording a sample during stream recording. A SkipCnt of 99 will reduce the data rate by a factor of 100. A SkipCnt of 0 will skip no samples.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.13.9 :STReam:SKIp? [<MPMeter>]

Query number of stream samples to be skipped for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns the unsigned integer-valued number of samples to be skipped after recording a sample during stream recording, it corresponds to the value set using »:STReam:SKIp <SkipCnt>[,<MPMeter>]«. If executed for multiple power meters the command returns a list of unsigned integer-valued numbers for values to skip for each power meter of the respective list.

### 9.13.10 :STReam:PREfix <String>[,<MPMeter>]

Set file prefix for stream recording for one or multiple power meters. The parameter can only be set when stream recording is disabled.

Parameters:

String parameter with quotes specifying the stream log file prefix. The string may not exceed 127 characters. The default value is set to "stream". E.g., »:str:pre "streamFile"« will result in a log files named "streamFile\_PP\_YYYYMMDD\_hhmmss\_msmsms.csv" to be saved when enabling stream recording. See Section 10.1.6 for a detailed description of stream file naming conventions and the stream file format.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

### 9.13.11 :STReam:PREfix? [<MPMeter>]

Query file prefix for stream recording for one or multiple power meters.

Parameter:

The optional unsigned integer parameter MPMeter is described in Section 9.1.

Return value:

The command returns a string value without quotes specifying the set stream log file prefix, see »:STReam:PREfix <String>[,<MPMeter>]« for more details. If executed for multiple power meters the command returns a list of string values containing the value for each power meter of the respective list.

### 9.13.12 :STReam:SYNC <Sync>[,<MPMeter>]

Set synchronization source for stream recording for one or multiple power meters.

**Parameters:**

String parameter without quotes specifying the synchronization source, valid values are OFF, EXT and EXT2. When set to OFF stream recording will start immediately upon enabling using the »:STReam:ENable <State>[,<MPMeter>]« command. When set to EXT the power meter's BNC connector will be used to synchronize stream recording slaves with the stream recording master. When set to EXT2 the power meter's RJ45 socket will be used to synchronize stream recording slaves with the stream recording master. When set to EXT or EXT2 the slave/master status set using the »:STReam:MAster <State>« command determines the input/output configuration of the respective connector.

The second, optional unsigned integer parameter MPMeter is described in Section 9.1.

**9.13.13 :STReam:SYNC? [<MPMeter>]**

Query synchronization source for stream recording for one or multiple power meters.

**Parameter:**

The optional unsigned integer parameter MPMeter is described in Section 9.1.


**Return value:**

The command returns a string value without quotes specifying the set stream synchronization source, see »:STReam:SYNC <Sync>[,<MPMeter>]« for more details. If executed for multiple power meters the command returns a list of string values containing the value for each power meter of the respective list.

## 10 File Formats

All data used by the LSPM 1.0 TCP Server and GUI are stored in the form of tabulator-separated plain ASCII text files. The uniform file extension is `.csv`. Lines are separated by newline characters (ASCII code `0xa`), columns are separated by tabulators.

Numbers are expressed as plain decimal integers, as floating-point numbers using "." as the decimal separator, or in exponential format using "." as the decimal separator and "e" as the exponential separator, e.g., "1.2e3" encoding a value of 1,200.

In all examples given below "-" denotes a tabulator,  a newline character, and "↵" a line wrap indicating that the contents of the next line in this document belong to the same line of the `.csv` file.

### 10.1 LSPM 1.0 GUI Log Files

The creation time and power meter serial number of all log files are saved by appending a time string formatted as "\_N\_YYYYMMDD\_hhmmss\_SSS" to a file's base name. "N", "YYYY", "MM", "DD", "hh",

“mm”, “ss’ and “SSS” denote power meter serial number, year, month, day, hour, minute, second and millisecond of log file creation with their respective number of digits. The number of digits used for the power meter serial number depends on the numeric value of the serial number.

The first line of all log files contains a header starting with a hash mark (#), which describes the contents of each column in the remainder of the file.

### 10.1.1 Basic Data Logger

The file format for all basic log files contains at least 7 columns described with their column headers and unit values in the table below.

Continuous logging will add one line for for every newly polled set of values. If more than one power meter are present one log file will be created for every power meter.

If a log file is created using the GUI’s “Quick Save” button all presently displayed values of the active power meter will be saved to a newly created file.

Column	Header	Unit	Description
1	t	s	Timestamp expressed as a float number of seconds since 1904/01/01 00:00:00 UTC
2	Mode		Measurement mode, see Table 1, page 15
3	f	Hz	Compensation frequency
4	P1	dBm	Channel 1 power
5	P2	dBm	Channel 2 power
6	P3	dBm	Channel 3 power
7	fLpP	Hz	Power value low pass filter frequency
8	RSSI1	LSB	Channel 1 RSSI value, optional
9	RSSI2	LSB	Channel 2 RSSI value, optional
10	RSSI3	LSB	Channel 3 RSSI value, optional

Example of basic log file:

```
#t →Mode →f→P1 →P2 →P3 →fLpP →RSSI1→RSSI2→RSSI3↵
3606197249.102 →1→100000000→-42.45547→-41.116783 →-41.568943 →0→↵
6074 →6074 →6074↵
3606197249.245 →1→100000000→-38.207194 →-36.87114→-37.288121 →0→↵
6702 →6702 →6702↵
```



### 10.1.2 Power Scope Data Logger

The file format for all field scope log files contains at least five columns described with their column headers and unit values in the table below.

Continuous logging will create a new log file for every newly recorded set of waveforms. If more than one power meter is present one log file will be created for each power meter.

If a log file is created using the GUI's "Quick Save" button all presently displayed values of the active power meter will be saved to a newly created file.

Column	Header	Unit	Description
1	Mode		Measurement mode, see Table 1, page 15
2	f	Hz	Compensation frequency
3	P1	dBm	Channel 1 power
4	P2	dBm	Channel 2 power
5	P3	dBm	Channel 3 power
6	RSSI1	LSB	Channel 1 RSSI value, optional
7	RSSI2	LSB	Channel 2 RSSI value, optional
8	RSSI3	LSB	Channel 3 RSSI value, optional

Example of field scope log file:

```
#Mode f P1 P2 P3 RSSI1 RSSI2 RSSI3 f P1 P2 P3 ↵
1 100000000 -82.007195 -83.007195 -75.007301 30 30 30 ↵
  100000000 -82.007195 -83.007195 -75.007301 ↵
1 100000000 -82.007195 -83.007195 -75.007301 1030 1030 1030 ↵
  200000000 -82.007195 -83.007195 -75.007301 ↵
```

### 10.1.3 Radar Data Logger

The file format for all radar log files contains at least 13 columns described with their column headers and unit values in the table below. The number of columns is dependent on the number of detected pulses for all channels. For every pulse a triple of columns consisting of the sample index of the start of the pulse, of the length of the pulse in samples and maximum power value for the pulse will be added to the radar log file. First, N1 value pairs for the channel 1 values will be added, followed by N2 and N3 value pairs for the other channels' values. The pulse counts N1, N2 and N3 are given in columns 11 through 13.

Continuous logging will create a new line in the log file for every newly recorded set of waveforms. If more than one power meter are present one log file will be created for every power meter.

If a log file is created using the GUI's "Quick Save" button all presently displayed values of the active power meter will be saved to a newly created file.

Column	Header	Unit	Description
1	t	s	Timestamp expressed as a float number of seconds since 1904/01/01 00:00:00 UTC
2	Mode		Measurement mode, see Table 1, page 15
3	f	Hz	Compensation frequency
4	Samp		Number of samples in waveforms evaluated for radar measurements
5	minS		Set minimum required pulse duration in samples for radar measurements
6	MAVG		Set number of samples for moving average filter for radar measurements
7	Th P1	dBm	Threshold value for power pulse detection for channel 1
8	Th P2	dBm	Threshold value for power pulse detection for channel 2
9	Th P3	dBm	Threshold value for power pulse detection for channel 3
10	Avg P1	dBm	Arithmetic mean of all pulses' channel 1 averaged pulse power values
11	Avg P2	dBm	Arithmetic mean of all pulses' channel 2 averaged pulse power values
12	Avg P3	dBm	Arithmetic mean of all pulses' channel 3 averaged pulse power values
13	Max P1	dBm	Power of pulse with the highest averaged power of all channel 1 pulses
14	MAx P2	dBm	Power of pulse with the highest averaged power of all channel 2 pulses
15	Max P3	dBm	Power of pulse with the highest averaged power of all channel 3 pulses
16	Duty P1		Duy cycle of channel 1 power values
17	Duty P2		Duy cycle of channel 2 power values
18	Duty P3		Duy cycle of channel 3 power values
19	CNT1		Number of pulses detected for channel 1
20	CNT2		Number of pulses detected for channel 2
21	CNT3		Number of pulses detected for channel 3
22...	IDX1N		Sample index of N <sup>th</sup> channel 1 pulse
23...	Len1N		N <sup>th</sup> channel 1 pulse's length in samples
24...	P1N	dBm	N <sup>th</sup> channel 1 pulse's maximum power value
22...	IDX2N		Sample index of N <sup>th</sup> channel 2 pulse

Column	Header	Unit	Description
23...	Len1N		N <sup>th</sup> channel 2 pulse's length in samples
24...	P2N	dBm	N <sup>th</sup> channel 2 pulse's maximum power value
22...	IDX3N		Sample index of N <sup>th</sup> channel 3 pulse
23...	Len1N		N <sup>th</sup> channel 3 pulse's length in samples
24...	P3N	dBm	N <sup>th</sup> channel 3 pulse's maximum power value

Example of radar log file:

```
#t →Mode →f→Samp →minS →MAVG →Th P1→Th P2→Th P3→Avg P1 →Avg P2 →Avg↵
P3 →Max P1 →Max P2 →Max P3 →Duty P1→Duty P2→Duty P3→CNT1 →CNT2↵
→CNT3 →IDX1N→Len1N→P1N→IDX2N→Len2N→P2N→IDX3N→Len3N→P3N↵
3686018740.515 →1→100000000→2000 →1→1→-5.822069→-13.437619 →↵
-13.163526 →-3.990641→-11.685132 →-11.351974 →-3.916312→↵
-11.432185 →-11.341158 →0.025→0.025→0.025→2→2→2→892→25 →↵
-4.066264→1892 →25 →-3.916312→892→25 →-11.953728 →1892 →25 →↵
-11.432185 →892→25 →-11.362818 →1892 →25 →-11.341158↵
3686018824.093 →1→100000000→2000 →1→1→-25→-25→-25→-4.010297→↵
-11.81766→-11.410587 →-3.973205→-11.771922 →-11.294949 →0.025→↵
0.025→0.025→2→2→2→947→25 →-3.973205→1947 →25 →-4.047709→947→25↵
→-11.771922 →1947 →25 →-11.863885 →947→25 →-11.529389 →1947 →↵
25 →-11.294949↵
```

### 10.1.4 Sweep Data Logger

The file format for all sweep log files contains at least six columns described with their column headers and unit values in the table below. One line is written for each sweep step.

Continuous logging will create a new log file for every newly recorded set of waveforms and change of a sweep parameter. If more than one power meter are present one log file will be created for every power meter.

If a log file is created using the GUI's "Quick Save" button all presently displayed values of the active power meter will be saved to a newly created file.

Column	Header	Unit	Description
1	Mode		Measurement mode, see Table 1, page 15
2	f	Hz	Compensation frequency
3	Index		Center sample index
4	P1	dBm	Averaged channel 1 power

Column	Header	Unit	Description
5	P2	dBm	Averaged channel 2 power
6	P3	dBm	Averaged channel 3 power
7	RSSI1	LSB	Averaged raw channel 1 RSSI value, optional
8	RSSI2	LSB	Averaged raw channel 2 RSSI value, optional
9	RSSI3	LSB	Averaged raw channel 3 RSSI value, optional

Example of sweep log file:

```
#Mode→f→Index→P1 →P2 →P3 →RSSI1→RSSI2→RSSI3↵↵↵↵
1→100000000→699→-52.943813 →-45.659973 →-49.853455 →4561 →5422 →↵
4896↵↵↵↵
1→450000000→1699 →-52.845222 →-45.333584 →-49.958347 →4575 →5469 →↵
4881↵↵↵↵
```

If the “Save freq. corr. waveforms” checkbox in the “Log”-tab is enabled, an additional sweep log file is saved, containing the sweep frequency corrected waveform values of the current waveform. The file format for all sweep waveform log files contains five columns described with their column headers and unit values in the table below. A power waveform log file will only be created

Continuous logging will create a new log file for every newly recorded set of waveforms. If more than one power meter is present one log file will be created for each power meter.

If a log file is created using the GUI’s “Quick Save” button all presently displayed values of the active power meter will be saved to a newly created file.

Column	Header	Unit	Description
1	Mode		Measurement mode, see Table 1, page 15
2	fSweep	Hz	Compensation frequency
3	P1	dBm	Channel 1 power for sweep
4	P2	dBm	Channel 2 power for sweep
5	P3	dBm	Channel 3 power for sweep

```
#Mode→fSweep →P1 →P2 →P3↵↵↵
1→100000000→-31.042856 →-35.417912 →-35.820896↵↵↵
1→100000000→-31.957144 →-37.623077 →-34.283581↵↵↵
```

### 10.1.5 Statistics Data Logger

The file format for all statistics log files contains at least 26 columns described with their column headers and unit values in the table below. The number of columns is dependent on the number of power value bins in the recorded histogram, there is at least one bin in the histogram. For every bin three columns consisting of the number of channel 1, 2 and 3 samples detected for the corresponding bin will be added to the radar log file. The number of bins is specified in column number 23.

Continuous logging will add a new line to the log file for every newly recorded statistics snapshot. If more than one power meter are present, one log file will be created for every power meter.

If a log file is created using the GUI's "Quick Save" button all presently displayed values of the active power meter will be saved to a newly created file.

Column	Header	Unit	Description
1	t	s	Timestamp expressed as a float number of seconds since 1904/01/01 00:00:00 UTC
2	Mode		Measurement mode, see Table 1, page 15
3	f	Hz	Compensation frequency
4	Type		Statistics type, 0 for continuous statistics, 1 for triggered statistics
5	MIN1	dBm	Channel 1 power minimum
6	MIN2	dBm	Channel 2 power minimum
7	MIN3	dBm	Channel 3 power minimum
8	MAX1	dBm	Channel 1 power maximum
9	MAX2	dBm	Channel 2 power maximum
10	MAX3	dBm	Channel 3 power maximum
11	MEAN1	dBm	Channel 1 power arithmetic mean
12	MEAN2	dBm	Channel 2 power arithmetic mean
13	MEAN3	dBm	Channel 3 power arithmetic mean
14	RMS1	dBm	Channel 1 power root mean square
15	RMS2	dBm	Channel 2 power root mean square
16	RMS3	dBm	Channel 3 power root mean square
17	SDEV1	dBm	Channel 1 power standard deviation
18	SDEV2	dBm	Channel 2 power standard deviation
19	SDEV3	dBm	Channel 3 power standard deviation
20	Samp		Number of samples used for statistics evaluation
21	Res	dB	Power resolution for histogram output
22	Offs	dBm	Power offset of minimum bin in histogram

Column	Header	Unit	Description
23	Bins		Number of bins in power value histogram
24...	CNT1N		Number of channel 1 power values in N <sup>th</sup> bin of histogram
25...	CNT2N		Number of channel 2 power values in N <sup>th</sup> bin of histogram
26...	CNT3N		Number of channel 3 power values in N <sup>th</sup> bin of histogram

Example of statistics log file:

```
#t →Mode →f→Type →MIN1 →MIN2 →MIN3 →MAX1 →MAX2 →MAX3 →MEAN1→MEAN2→↵
  MEAN3→RMS1 →RMS2 →RMS3 →SDEV1→SDEV2→SDEV3→Samp →Res→Ofss →Bins↵
  →CNT1N→CNT2N→CNT3N↵
3606271892.999 →1→100000000→0→-82.004997 →-83.004997 →-75.004997 →↵
  10.995 →10.995 →10.995 →-30.340851 →-29.428646 →-28.592386 →↵
  44.246136→43.74614 →41.94656 →32.204865→32.367878→30.691845→↵
  22726830 →5→-17→20 →0→3406472→122056 →330106 →678243 →914033 →↵
  962578 →1002801→1000027→1024993→1015284→1015284→1018058→1001650↵
  →1014628→975764 →1029706→981996 →927903 →4305248→3190100→37449→↵
  119282 →313462 →654664 →891841 →952869 →994479 →991705 →1020832↵
  →1011123→1009736→1020832→1004223→1014628→970212 →1020180→979244↵
  →919581 →4610388→0→0→3453630→325945 →651890 →890454 →951482 →↵
  984770 →980609 →1013897→1002801→1008349→1015284→995946 →1006300↵
  →959108 →1013236→975061 →911259 →4586809↵
3606271976.273 →1→100000000→0→-82.004997 →-83.004997 →-75.004997 →↵
  10.995 →10.995 →10.995 →-30.341587 →-29.429396 →-28.593102 →↵
  44.246834→43.746834→41.947224→32.205132→32.368137→30.692085→↵
  26576572 →20 →-4 →-6→4273970→3806834→4734618→4716471→4345745→↵
  4698934→4058244→3722490→4705422→4721137→4362165→5007114→4191248↵
  →3716002→4668116→4685486→4357266→4958454↵
```

### 10.1.6 Stream Files in Binary Format

The format “PREFIX\_SN\_YYYYMMDD\_hhmmss.bin” and “PREFIX\_SN\_YYYYMMDD\_hhmmss.lut” containing the following information is used for all binary stream files.

**PREFIX**

File prefix set by the user, its default value is ‘stream’,

**SN**

Serial number of the power meter which recorded the stream data.

**YYYYMMDD\_hhmmss**

Year, month, day of the month, hour, minute and second of the start of the stream file.

Stream files are optimized for small file size and low processor load. The .bin files contain the raw, unsigned integer RSSI values for all channels, including frame indicators.

The stream .lut files contain arbitrary numbers of lookup-table blocks containing three-channel power lookup-tables and auxiliary information about mode, frequency, temperature and skip count. Power value lookup-tables can be used off-line for converting RSSI values to power values. Lookup-table blocks start with a 64 bit unsigned integer value indicating the streaming sample in the .bin file as from which the new look-up table is valid. They have a size of 196,635 bytes and the following structure:

Data	Offset	Size	Description
Sample Start Index	0	8	Stream sample start index of LUT, 64 bit, little-endian unsigned integer
Serial number	8	2	Power meter serial number, 16 bit, little endian, unsigned integer
Mode	10	1	Power meter mode, 8 bit, little endian, unsigned integer
Frequency	11	8	Power meter frequency in hertz, 64 bit, double-precision, little-endian floating-point value
Temperature	19	4	Power meter temperature in °C, 32 bit, single-precision, little-endian floating-point value
Skip Count	23	4	Stream recording skip count, 32 bit, little endian, unsigned integer value
channel1[0]	27	4	Channel 1 power value for RSSI=0, 32 bit, single-precision, little-endian floating-point value
channel2[0]	31	4	Channel 2 power value for RSSI=0, 32 bit, single-precision, little-endian floating-point value
channel3[0]	35	4	Channel 3 power value for RSSI=0, 32 bit, single-precision, little-endian floating-point value
channel1[1]	39	4	Channel 1 power value for RSSI=1, 32 bit, single-precision, little-endian floating-point value
...	...	4	...
channel3[16383]	196,631	4	Channel 3 power value for RSSI=16383, 32 bit, single-precision, little-endian floating-point value

The binary measurement data files contain arbitrary number of RSSI data blocks. Each block contains simultaneous channel 1, 2 and 3 RSSI values and the frame indicator for one sampling instant. Each block has a size of seven bytes and the following structure:

Data	Size	Description
frame	1	Frame information for the current sample, alternating between zero and one, 8 bit unsigned integer
channel1 RSSI	2	Channel1 RSSI value, 16 bit unsigned integer little-endian format
channel2 RSSI	2	Channel2 RSSI value, 16 bit unsigned integer little-endian format
channel3 RSSI	2	Channel3 RSSI value, 16 bit unsigned integer little-endian format

### 10.1.7 Stream Files in CSV Format

The Bin2Csv.exe program which is part of the LSPM Installer can be used to generate CSV files from binary stream files described in the previous section. CSV output files have the following format:

Column	Header	Unit	Description
1	Mode		Measurement mode, see Table 1, page 15, optional
2	f	Hz	Compensation frequency, optional
3	P1	dBm	Channel 1 power
4	P2	dBm	Channel 2 power
5	P3	dBm	Channel 3 power
6	RSSI1	LSB	Channel 1 raw RSSI value, optional
7	RSSI2	LSB	Channel 2 raw RSSI value, optional
8	RSSI3	LSB	Channel 3 raw RSSI value, optional
9	T	°C	Power meter cold plate temperature, optional
10	Skip		Skip count used during stream recording, optional

The Bin2Csv.exe program accepts an arbitrary number of command line switches followed by the file name(s) of one or more binary stream files. One CSV file will be generated for every bin file, replacing the extension "bin" by "csv". Note that the Bin2Csv.exe program can be used for both E-field and power value stream files. The stream file type will be detected automatically. The following command line switches are supported for power stream files, redundant parameters will be ignored silently:

- h  
display usage information and quit program,
- s  
set sample index for the start of binary to CSV conversion, default is zero,
- e  
set sample index for the end of binary to CSV conversion, default is last sample in bin file,



- l  
set number of samples to be converted to CSV format, relative to start sample index if specified, defaults to all samples in bin file,
- M  
enable optional mode column,
- F  
enable optional frequency column,
- T  
enable optional temperature column,
- r  
enable optional RSSI value columns,
- S  
enable optional skip count column.

## 10.2 extCalLog TCP-Server Logger

For each TCP/IP connection with the exception of the LSPM 1.0 GUI, the SCPI command :CALibration:LOGging <Value> enables/ disables logging of power meter status information and power values every time a »:MEASure:P[1]/P2/P3/ALL? [<MPMeter>]« SCPI query is sent by a client which previously set its log flag set to 1.

The CSV file format for all external calibration log files contains sixteen columns described by their column headers and unit values in the table below.

Column	Header	Unit	Description
1	RSSI1	LSB	Channel 1 RSSI value
2	RSSI2	LSB	Channel 2 RSSI value
3	RSSI3	LSB	Channel 3 RSSI value
4	T	°C	Power meter cold plate temperature
5	f	Hz	Compensation frequency
6	P1	dBm	Channel 1 power
7	P2	dBm	Channel 2 power
8	P3	dBm	Channel 3 power
9	Serno LSPM		Power meter serial number
10	Mode		Measurement mode, see Table 1, page 15
11	FW LSPM		LSPM 1.0 FPGA firmware version
12	FW Server		LSPM 1.0 TCP Server firmware version
13	t	s	Timestamp expressed as a float number of seconds since 1904/01/01 00:00:00 UTC

Column	Header	Unit	Description
14	cal:ext		External calibration enabled state, see :CALibration:EXTernal <Value>[,<MPMeter>]
15	fLpP	Hz	Power value low pass filter frequency
16	123a		Indicator which SCPI query prompted logging of actual line

Example of extCalLog file:

```
#RSSIx →RSSIy→RSSIz→T→f→P1 →P2 →P3 →SerNo LSPM →Mode →FW LSPM→FW ↵
Server→t→cal:ext→fLpP →123a↵
```

### 10.3 Generic Calibration Result Files

External power calibration is always performed in mode 0 and 3 against the in-house power calibration data, recording the calibration power value and the indicated power value. The usage of preexisting external calibration data is prevented by issuing a »:CALibration:EXTernal <Value>[,<MPMeter>]« command with the parameter Value set to zero, this command is mandatory. Sending »:CALibration:LOGging <Value>« with the parameter Value set to one is highly recommended, see Section 7.3 on page 57 for details.

Generic calibration result files are CSV files using the formatting conventions outlined in Section 10. The following information must be recorded for each combination of mode, frequency and channel, ensuring a verifiably correct calibration procedure. Each item below corresponds to one or three columns in the CSV file, as detailed in the sample result file below. While the header line is optional, adding such a comment line at the beginning of the file is strongly recommended.

Column	Header	Unit	Description
1	Mode		Power meter mode, as queried using »:SYSTem:MODE? [<MPMeter>]«
2	fcal	Hz	Calibration frequency, as queried using »:SYSTem:FREQUency? [<MPMeter>]«
3	P1_cal	dBm	Channel 1 power value of the calibration setup
4	P1_disp	dBm	Channel 1 power value, as queried by »:MEASure:P[1]/P2/P3/ALL? [<MPMeter>]«
5	P2_cal	dBm	Channel 2 power value of the calibration setup
6	P2_disp	dBm	Channel 2 power value, as queried by »:MEASure:P[1]/P2/P3/ALL? [<MPMeter>]«
7	P3_cal	dBm	Channel 3 power value of the calibration setup

Column	Header	Unit	Description
8	P3_disp	dBm	Channel 3 power value, as queried by »:MEASure:P[1]/P2/P3/ALL? [<MPMeter>]«

Generic calibration result file:

```
#mode←fcal ←P1_cal ←P1_disp←P2_cal ←P2_disp←P3_cal ←P3_disp↵
0←100000000←0.01 ←0.14 ←-0.04←-0.21←0.04 ←-0.14↵
0←120000000←0.03 ←0.23 ←-0.02←-0.14←0.05 ←-0.07↵
```

## 10.4 Calibration Files

Each LSPM 1.0 Power Meter comes with a detailed set of in-house calibration files used for linearity compensation, frequency compensation and absolute power value calibration. Optionally, an external power calibration file can be added. Calibration files are stored in one directory per power meter, directory names consist of »sn« followed by the decimally coded power meter serial number without leading zeros. Calibration directories are stored in the directory specified via the CAL\_PATH setting of the configuration file LSPM\_1.0.ini. The CSV file conventions detailed in section 10 apply to all calibration files.

Calibration data folders can also be stored in the form of ZIP files consisting of »sn« followed by the decimally coded power meter serial number and the extension ».zip«. LSPM\_CAL\_PATH must not contain both a ZIP file and a directory for the same serial number.

The contents of all calibration files are protected against inadvertent modification by means of a decimally coded integer checksum in the last column of each calibration file's first line. The checksum is calculated by adding all ASCII code values of the calibration file starting with the first character of the second line of the respective calibration file.

### 10.4.1 In-House Linearity and Frequency Compensation Files

One in-house linearity and frequency file, or short LF file, exists for every in-house calibrated frequency and power meter mode. LF file names consist of »sn« followed by the decimally coded power meter serial number, »m« followed by the decimally coded mode number, »f« followed by the decimally coded frequency value in hertz, followed by ».csv«.

The first line of LF files starts with a hash mark character (#) and gives context information for the LF file. It does not give column names for the data in the remainder of the CSV file. The first line's columns have the following contents:

Column	Unit	Description
1		Power meter serial number as decimally coded integer value

Column	Unit	Description
2		Measurement mode, see Table 1, page 15
3	Hz	Measurement frequency
5		Time stamp of calibration for given frequency and mode
6		Checksum for rest of file as described in Section 10.4

The following lines of LF files have the following contents:

Column	Unit	Description
1	dBm	Power level used for given mode and frequency given in the first line of the file
2	LSB	Channel 1 RSSI value for given frequency, mode and power value in first column
3	LSB	Channel 2 RSSI value for given frequency, mode and power value in first column
4	LSB	Channel 3 RSSI value for given frequency, mode and power value in first column

Calibration files for power meters equipped with less than three channels contain an RSSI constant value of zero for all power levels for the unavailable power detectors.

#### 10.4.2 External Power Calibration Files

One external power calibration file or short EP file, exists for every externally calibrated power meter. An EP file contains correction factors expressed in Decibels, the correction factor will be applied to the base power value derived from the supplied internal calibration data. A value greater than zero will make the displayed power value larger, values smaller than zero lower the displayed power value. The EP file name consists of »sn« followed by the decimally coded power meter serial number followed by ».csv«.

The first line of an EP file starts with a hash mark character (#) and gives context information for the EP file, it does not give column names for the data in the remainder of the CSV file. The first line's columns have the following contents:

Column	Unit	Description
1		Power meter serial number as decimally coded integer value
2		Maximum time stamp during calibration
3		Checksum for rest of file as described in Section 10.4

The second line of EP files starts with a hash mark character (#) and contains a calibration certificate string that extends to the end of the second line. The subsequent lines of an EP file have the following contents:

---

<b>Column</b>	<b>Unit</b>	<b>Description</b>
1	Hz	Frequency for external power calibration
2	dB	Channel 1 correction value for the given frequency
3	dB	Channel 2 correction value for the given frequency
4	dB	Channel 3 correction value for the given frequency

---

## 11 Specifications

Table 21: LSPM 1.0 Power Meter specifications

Frequency Range	
Low Band	9 kHz ... 400 MHz
High Band	30 MHz ... 6 GHz (usable up to 12 GHz)
Analog Rise Time	
Low Band (Video BW 500 Hz)	1.9 ms
Low Band (Video BW 1 MHz)	770 ns
High Band (Video BW 3 MHz)	330 ns
Minimum Pulse Width	500 ns
VSWR	<1.2:1
Sampling Rate	2 MSamples/s
Measurement Range & Dynamic Range	
Low Band	<-60 dBm ... >20 dBm (>80 dB)
High Band up to 4 GHz	<-70 dBm ... >20 dBm (>90 dB)
High Band 4 ... 6 GHz	<-50 dBm ... >20 dBm (>70 dB)
High Band 6 ... 12 GHz	<0 dBm ... >20 dBm (>20 dB)
Amplitude Accuracy*	0.1 dB
Linearity Error	0.15 dB
Temperature Stability	0.1 dB
Power Resolution	<0.1 dB (see plot below)
Channel Isolation	>50 dB
Damage Level	>30 dBm
PC Interface	USB 2.0
Application Software	LSPM 1.0 TCP Server, LSPM 1.0 GUI, CallImport
Trigger Voltage	5 V
Trigger Connector	BNC
Input Voltage	5 V $\pm$ 5 %
Input Current	<3 A
Ambient Temperature	10 ... 40 °C
Dimensions (W $\times$ D $\times$ H)	165 $\times$ 142 $\times$ 61 mm <sup>3</sup>
Certifications	CE

\*) At 0 dBm, CW, accredited Calibration at Ametek CTS Europe GmbH.

### 11.0.1 Typical Dynamic Range

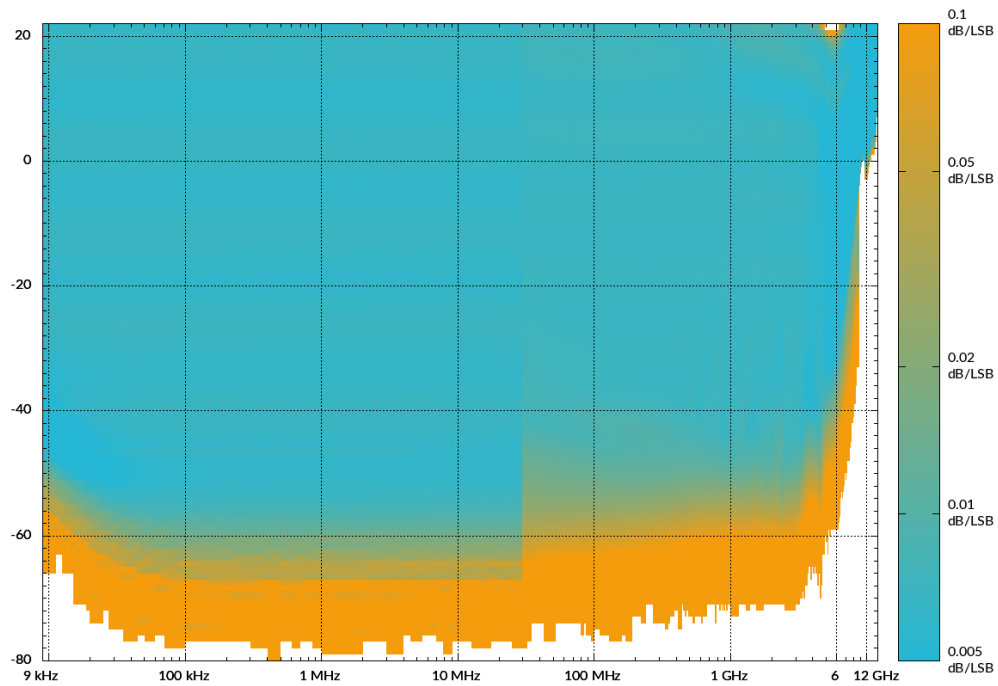


Figure 47: Typical dynamic range, low and high band

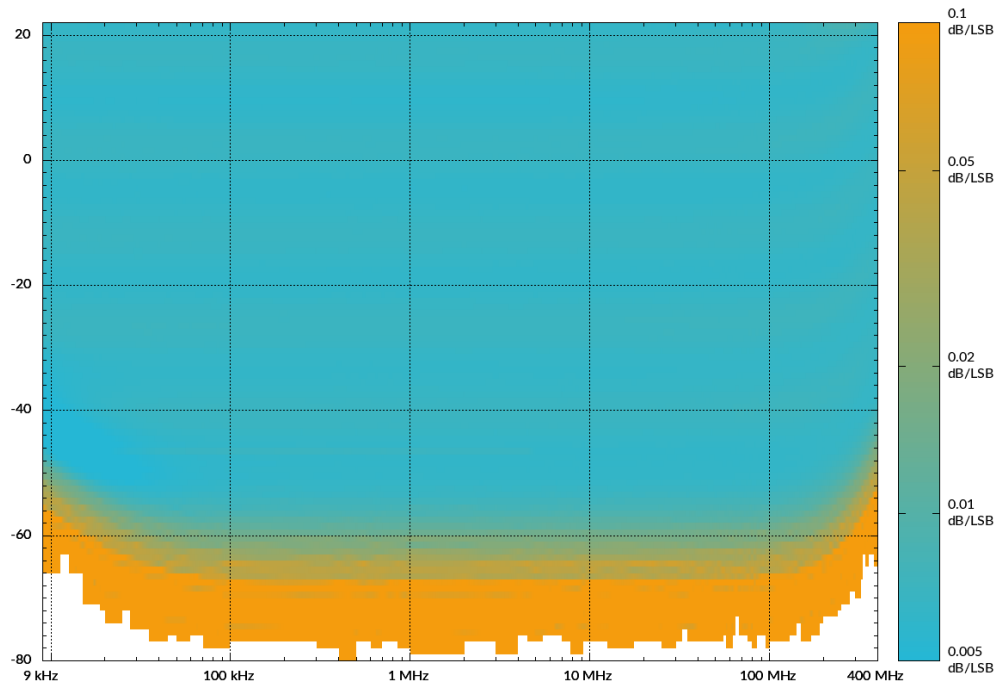


Figure 48: Typical dynamic range, low band

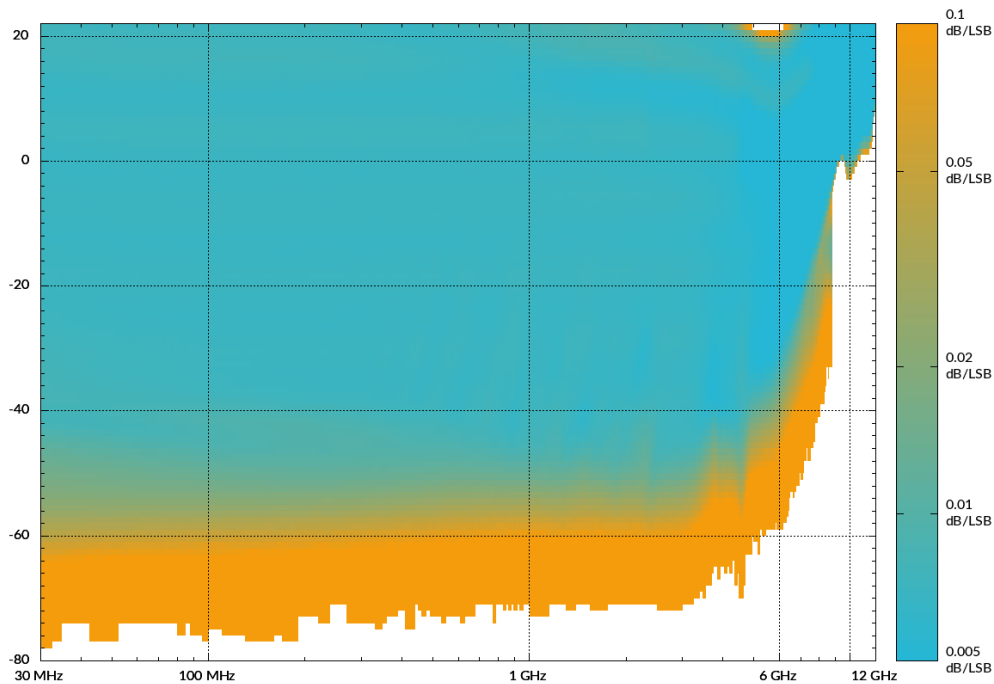


Figure 49: Typical dynamic range, high band

## 12 Warranty Conditions

1. The period of warranty shall start from the date of delivery of the product to the customer and shall cover a period of 24 months.
2. These warranty conditions apply to devices purchased in Germany. These conditions of warranty also apply if these devices are exported abroad and meet the technical requirements (e.g., voltage, frequency) for the respective country and which are suitable for the respective climatic and environmental conditions.
3. Every and all parts of the product are under LUMILOOP's warranty coverage against any defect that may occur during production, assembly and/or defective parts.
4. In case of repair within warranty period, the time spent on the repair work is added to the warranty period. Repair time of the product is maximum 20 (twenty) working days. A warranty event does not lead to a new warranty period. The warranty period for built-in spare parts ends with the warranty period for the entire device.
5. In case of failure of the product during warranty period, the producer or reseller company has to assign another product to the customer with similar features until completion of repair of the product.
6. Within the warranty period, if the product fails because of general material and workmanship, or mounting faults, it will be repaired without demanding any charge.
7. In case of any failure in the product, occurring at least four times in one year or six times within the warranty period, product replacement or refund is mandatory depending on the choice of the customer.



8. Free repair and product exchange obligations will be annulled under the following conditions:
  - a) If the product becomes faulty due to use contrary to the terms or conditions stated in the user's manual,
  - b) If the product has been opened, used, or previously repaired by unauthorized persons,
  - c) Use of the product by plugging into inappropriate voltages or with faulty electric installation,
  - d) If the product serial number has been altered or removed,
  - e) If the fault or damage to the product occurred during the transportation outside of the responsibility of LUMILOOP GmbH,
  - f) A break or scratch to the product's exterior while in the customer's possession,
  - g) Damage from chemical and electrochemical effects of water
  - h) When our product is damaged due to use with spare parts, accessories or devices purchased from other companies which are not original parts.
  - i) Those damages caused by natural disasters such as fire, lightning, flood, earthquake, etc.
  
9. A short report prepared by the LUMILOOP GmbH will determine whether the damage was caused by improper use.
10. Customers are required to initially report any conflicts between themselves and an authorized reseller to the address below:

Gostritzer Str. 63  
01217 Dresden, Germany  
Phone: +49 (0)351 85097870  
E-mail: info@lumiloop.de

# EC DECLARATION OF CONFORMITY

We, **LUMILOOP GmbH**,  
Gostritzer Str. 63,  
01217 Dresden,  
GERMANY,

declare under sole responsibility that the:

**Model / Part Name:** LSPM 1.0 Power Meters  
**Model / Part Numbers:** 2101, 2102, 2103  
**Date of Declaration:** January 8, 2021

to which this declaration relates, meets the requirements and is in conformity with the relevant EC Directives listed below using the relevant section(s) of the following EC harmonized standards and other normative documents:

## Applicable Directives:

2014/35/EU (Low Voltage Directive)  
2014/30/EU (EMC Directive)  
2011/65/EU (RoHS), Directive (EU) 2015/863 (RoHS3)

## Applicable harmonized standards and/or other normative documents:

EN 61010-1:2010 Safety requirements for electrical equipment for measurement, control, and laboratory use  
Part 1: General requirements

EN 61326-1:2013 Electrical equipment for measurement, control and laboratory use – EMC requirements  
Part 1: General requirements

## Authorized Signatories:

---

LUMILOOP GmbH  
Eike Suthau, Technical Director

This declaration attests the compliance with the stated directives. It does not imply any assurance of characteristics.

## 14 Revision History

### 2018/10/16

- Initial release.

### 2018/11/27

- Update of dynamic range plots.
- Fix spelling.
- Expected format of external calibration files by TCP-Server fixed.
- LSPM 1.0 GUI indicates external calibration enabled/disabled state via date string.

### 2020/11/24

- Bug fix: TCP server crashed when sending data to closed TCP client socket.
- Bug fix: statistics data was returned for a few  $\mu$ s after »stat:en 0« and statistics because of mutex error.
- Bug fix: use average of power values instead of RSSI value average for all averaging queries.
- Bug fix: »stat:pdf:all?«, »stat:cdf:all?«, »stat:ccdf:all?« -> wrong order of return values.
- Bug fix: TCP server crash on »stat:snap 1« without waveform (state not done).
- Bug fix: prevent bind to already used TCP port.
- Bug fix: »trig:arm« race condition fixed by handshake flag in CI and modified trigger state transitions.
- Bug fix: fix monotonous range error for snXmYfZ.csv calibration file parsing.
- Bug fix: do not re-use old LUT for new waveforms.
- Bug fix: return correct number of commas for more than 64k return values.
- Firmware fix: prevent self-triggering on external trigger polarity change.
- Command line prompt with history and shortcuts added for TCP server.
- Loop command for all SCPI commands added for TCP server.
- »[:TRIGger]:RADar:BINary? <Wave>[,<MPMeter>]« added.
- Stricter sanity checks for all calibration data files.
- Support multi-level and multi-mode external calibration data using snX\_Y.csv files.
- »:SYSTEM:WAIT <Sec>« added to wait for a number of milliseconds.
- »:SYSTEM:AUTOCONnect <State>« added to reduce interference with other FTDI USB devices.
- CI polling interval set to 1 s, only call CreateDeviceInfoList() if number of connected USB devices has changed.
- »:TRIGger:POINT <Points>[,<MPMeter>]« added for point triggering.
- Simplify SCPI syntax for MEAS and TRIG subsystems.
- »[:TRIGger]:RADar:MINSamples <MinS>[,<MPMeter>]«, »[:TRIGger]:RADar:MINTime <MinT>[,<MPMeter>]«, »[:TRIGger]:RADar:THMethod <Method>[,<MPMeter>]«, »[:TRIGger]:RADar:THold:P[1]/P2/P3/ALL? [<MPMeter>]«, »[:TRIGger]:RADar:MAVG

- <Count>[<MPMeter>]«, »[:TRIGger]:RADar:MPOWER:P[1]/:P2/:P3/:ALL? [<MP-Meter>]«, »[:TRIGger]:RADar:DUTY:P[1]/:P2/:P3/:ALL? [<MPMeter>]«, »[:TRIGger]:RADar:WPower:P[1]/:P2/:P3/:ALL? [<MPMeter>]« added to radar pulse subsystem.
- »[:TRIGger]:RADar[:APOWER]:P[1]/:P2/:P3/:ALL? [<MPMeter>]«, »[:TRIGger]:RADar:PULses:STArt:P[1]/:P2/:P3? [<MPMeter>]«, »[:TRIGger]:RADar:PULses:LENGth:P[1]/:P2/:P3? [<MPMeter>]« added for individual pulse queries.
  - »[:TRIGger]:RADar:TRIM <State>[,<MProbe>]« to control behavior at pulses' edges.
  - »[:TRIGger]:SWEEP:ARbitrary? [<MPMeter>]« added.
  - Ask for confirmation before closing TCP server due to error.
  - »:SYSTem:ERRor[:NEXT]?« and »:SYSTem:ERRor:COUNT?« extended for detailed error messages.
  - »:MEASure:VBW <Frequency>[,<MPMeter>]« added to support low frequency operation with high video bandwidth detector and software-based video bandwidth setting.
  - Extend operating frequency range for mode 2 to low frequencies, add forbidden frequency range to avoid aliasing..
  - »:SYSTem:DFLags?« added for textual debug flag setting and query.
  - Create separate waveform log files for sweep waveforms in GUI.
  - Display pulse average values in GUI's waveform.
  - Install 32 bit or 64 bit version of TCP server and support libraries depending on host system.
  - Include LabView RTE in installer.
  - Cleanup of TCP server startup messages and summary tables.
  - »:TRIGger:FLENGht? [<MPMeter>]« added for query of waveform length, including all trigger points.
  - »:TRIGger:PTTimes? [<MPMeter>]« added to query trigger timing.
  - »[:TRIGger]:SWEEP:TCNT? [<MPMeter>]« added to query number of sweep steps that fit into waveform.
  - Bin2CSV: LSPM streaming file support added.
  - New streaming files with separate LUT and data files, also supported by Bin2CSV.
  - Stricter sanity checks for pulse modulation of virtual probes added.
  - Added reduced data rate mode for slow USB connections.