



# EEZ BB3 SCPI Reference guide

## Modular T&M solution

Firmware version: 1.3

Document version: 1.2 – 09/2020

[www.envox.hr](http://www.envox.hr) – [github.com/eez-open](https://github.com/eez-open)



Free as in Freedom



# Table of Contents

1.Introduction.....	11
1.1.About SCPI.....	11
2.Syntax and style.....	13
2.1.Root Specifier.....	13
2.2.Command termination.....	13
2.3.Command separators.....	13
2.4.Querying parameter settings.....	13
2.5.Using the MIN, MAX, and DEF Parameters .....	13
2.6.Command and message types.....	14
2.7.Required Commands.....	14
2.7.1.Base functionality for the Power supply instrument class.....	14
2.8.Multiple Commands in a Message.....	15
2.9.Moving Among Subsystems.....	15
2.10.SCPI parameter types.....	16
3.Registers and queues.....	17
3.1.Standard Event Status Register.....	17
3.2.Status Byte Register.....	19
3.3.OPERation Status Register.....	20
3.3.1.Operation INSTRument Status register.....	21
3.3.2.Operation Instrument SUMmary status register.....	22
3.4.QUESTionable Status Register.....	22
3.4.1.Questionable INSTRument Status register.....	24
3.4.2.Questionable Instrument SUMmary status register.....	24
3.5.Error queue.....	25
4.Common command reference.....	27
4.1.*CLS.....	27
4.2.*ESE.....	27
4.3.*ESR?.....	28
4.4.*IDN?.....	28
4.5.*OPC.....	28
4.6.*RCL.....	29
4.7.*RST.....	29
4.8.*SAV.....	30
4.9.*SRE.....	31
4.10.*STB?.....	31
4.11.*TRG.....	31
4.12.*TST?.....	32
4.13.*WAI.....	32
5.Subsystem command reference.....	33
5.1.ABORT.....	35
5.1.1.ABORt .....	35
5.1.2.ABORt:DLOG.....	35
5.2.CALibration.....	37
5.2.1.CALibration[:MODE].....	37
5.2.2.CALibration:CLear.....	38
5.2.3.CALibration:CURRent[:DATA].....	39
5.2.4.CALibration:CURRent:LEVel {<point>}, {<level>}.....	39
5.2.5.CALibration:CURRent:RANGe.....	39
5.2.6.CALibration:PASSword:NEW.....	40
5.2.7.CALibration:REMark.....	40

5.2.8.CALibration:SAVE.....	41
5.2.9.CALibration:SCReen:INIT.....	41
5.2.10.CALibration:STATe.....	41
5.2.11.CALibration:VOLTage[:DATA].....	41
5.2.12.CALibration:VOLTage:LEVel {<point>}, {<level>}.....	42
5.3.DIAGnostic.....	43
5.3.1.DIAGnostic[:INFOrmation]:ADC?.....	43
5.3.2.DIAGnostic[:INFOrmation]:CALibration?.....	43
5.3.3.DIAGnostic[:INFOrmation]:PROTection?.....	44
5.3.4.DIAGnostic[:INFOrmation]:TEST?.....	44
5.4.DISPlay.....	47
5.4.1.DISPlay:BRIGhtness.....	47
5.4.2.DISPlay:DATA?.....	47
5.4.3.DISPlay:VIEW.....	48
5.4.4.DISPlay[:WINdow][:STATe].....	48
5.4.5.DISPlay[:WINdow]:DIALog[:OPEN].....	48
5.4.6.DISPlay[:WINdow]:DIALog:ACTIon.....	49
5.4.7.DISPlay[:WINdow]:DIALog:CLOSe.....	49
5.4.8.DISPlay[:WINdow]:DIALog:DATA.....	49
5.4.9.DISPlay[:WINdow]:DLOG.....	50
5.4.10.DISPlay[:WINdow]:ERRor.....	50
5.4.11.DISPlay[:WINdow]:INPut.....	50
5.4.12.DISPlay[:WINdow]:SELEct?.....	51
5.4.13.DISPlay[:WINdow]:TEXT.....	51
5.4.14.DISPlay[:WINdow]:TEXT:CLEar.....	51
5.5.FETCh.....	53
5.5.1.FETCh:AHOuR?.....	53
5.5.2.FETCh:WHOUr?.....	53
5.6.HCOPy.....	55
5.6.1.HCOPy:DESTination.....	55
5.6.2.HCOPy[:IMMediate].....	55
5.6.3.HCOPy:SDUMp[:IMMediate].....	55
5.7.INITiate.....	57
5.7.1.INITiate.....	57
5.7.2.INITiate:DLOG.....	57
5.7.3.INITiate:DLOG:TRACe.....	60
5.7.4.INITiate:CONTInuous.....	61
5.8.INSTrument.....	63
5.8.1.INSTrument[:SELEct].....	63
5.8.2.INSTrument:CATalog?.....	64
5.8.3.INSTrument:CATalog:FULL?.....	64
5.8.4.INSTrument:COUPle:TRACKing.....	64
5.8.5.INSTrument:DISPlay:TRACe[<n>].....	66
5.8.6.INSTrument:DISPlay:TRACe:SWAP.....	66
5.8.7.INSTrument:DISPlay:YT:RATE.....	67
5.8.8.INSTrument:COUPle:TRIGGer.....	67
5.8.9.INSTrument:NSELEct.....	67
5.9.MEASure.....	69
5.9.1.MEASure[:SCALar]:CURRent[:DC].....	69
5.9.2.MEASure[:SCALar]:POWEr[:DC].....	69
5.9.3.MEASure[:SCALar][:VOLTage][:DC].....	69
5.10.MEMory.....	71

5.10.1.MEMory:NSTates.....	71
5.10.2.MEMory:STATe:CATalog.....	71
5.10.3.MEMory:STATe:DElete.....	72
5.10.4.MEMory:STATe:FREEze.....	72
5.10.5.MEMory:STATe:NAME.....	72
5.10.6.MEMory:STATe:RECall:AUTO.....	73
5.10.7.MEMory:STATe:RECall:SElect.....	73
5.10.8.MEMory:STATe:VALid.....	73
5.11.MMEMory.....	75
5.11.1.MMEMory:CATalog.....	76
5.11.2.MMEMory:CATalog:LENgth.....	76
5.11.3.MMEMory:CDIRectory.....	77
5.11.4.MMEMory:COpy.....	77
5.11.5.MMEMory:DATE.....	78
5.11.6.MMEMory:DElete.....	78
5.11.7.MMEMory:DOWNload:ABORt.....	78
5.11.8.MMEMory:DOWNload:DATA.....	78
5.11.9.MMEMory:DOWNload:FNAME.....	79
5.11.10.MMEMory:DOWNload:SIZE.....	79
5.11.11.MMEMory:INFOrmation.....	80
5.11.12.MMEMory:LOAD:LIST.....	80
5.11.13.MMEMory:LOAD:PROFile.....	80
5.11.14.MMEMory:LOCK.....	80
5.11.15.MMEMory:MDIRectory.....	81
5.11.16.MMEMory:MOVE.....	81
5.11.17.MMEMory:MDIRectory.....	82
5.11.18.MMEMory:STORe:LIST.....	82
5.11.19.MMEMory:STORe:PROFile.....	82
5.11.20.MMEMory:TIME.....	83
5.11.21.MMEMory:UNLock.....	83
5.11.22.MMEMory:UPLoad.....	83
5.12.OUTPut.....	85
5.12.1.OUTPut[:STATe].....	85
5.12.2.OUTPut[:STATe]:TRIGgered.....	86
5.12.3.OUTPut:DELay:DURation.....	86
5.12.4.OUTPut:DPRog.....	87
5.12.5.OUTPut:MODE?.....	87
5.12.6.OUTPut:PROTection:CLEar.....	88
5.12.7.OUTPut:PROTection:COUPle.....	88
5.12.8.OUTPut:TRACK[:STATe].....	88
5.13.ROUTE.....	91
5.13.1.ROUTE:CLOSe.....	91
5.13.2.ROUTE:CHANnel:LABel.....	91
5.13.3.ROUTE:OPEN.....	92
5.14.SENSE.....	93
5.14.1.SENSE:CURRent[:DC]:RANGe[:UPPer].....	93
5.14.2.SENSE:DLOG:FUNCTion:CURRent.....	94
5.14.3.SENSE:DLOG:FUNCTion:POWer.....	94
5.14.4.SENSE:DLOG:FUNCTion:VOLTage.....	94
5.14.5.SENSE:DLOG:PERiod.....	95
5.14.6.SENSE:DLOG:TIME.....	95
5.14.7.SENSE:DLOG:TRACe[:DATA].....	95

5.14.8.SENSE:DLOG:TRACe:X:LABel.....	96
5.14.9.SENSE:DLOG:TRACe:REMark.....	96
5.14.10.SENSE:DLOG:TRACe:X:UNIT.....	96
5.14.11.SENSE:DLOG:TRACe:X:STEP.....	97
5.14.12.SENSE:DLOG:TRACe:X[:RANGe]:MIN.....	97
5.14.13.SENSE:DLOG:TRACe:X[:RANGe]:MAX.....	97
5.14.14.SENSE:DLOG:TRACe:X:SCALE.....	97
5.14.15.SENSE:DLOG:TRACe:Y<n>:LABel.....	98
5.14.16.SENSE:DLOG:TRACe:Y<n>[:RANGe]:MIN.....	98
5.14.17.SENSE:DLOG:TRACe:Y<n>[:RANGe]:MAX.....	98
5.14.18.SENSE:DLOG:TRACe:Y:SCALE.....	99
5.14.19.SENSE:DLOG:TRACe:Y<n>:UNIT.....	99
5.15.SOURce.....	101
5.15.1.[SOURce[<n>]]:CURRent.....	102
5.15.2.[SOURce[<n>]]:CURRent:STEP.....	103
5.15.3.[SOURce[<n>]]:CURRent:TRIGgered.....	104
5.15.4.[SOURce[<n>]]:CURRent:LIMit[:POSitive][:IMMediate][:AMPLitude].....	104
5.15.5.[SOURce[<n>]]:CURRent:MODE.....	105
5.15.6.[SOURce[<n>]]:CURRent:PROTection:DELAy[:TIME].....	105
5.15.7.[SOURce[<n>]]:CURRent:PROTection:STATe.....	106
5.15.8.[SOURce[<n>]]:CURRent:PROTection:TRIPped?.....	106
5.15.9.[SOURce[<n>]]:CURRent:RAMP:DURation.....	106
5.15.10.[SOURce[<n>]]:LIST:COUNT.....	107
5.15.11.[SOURce[<n>]]:LIST:CURRent[:LEVel].....	107
5.15.12.[SOURce[<n>]]:LIST:DWELl.....	108
5.15.13.[SOURce[<n>]]:LIST:VOLTagE[:LEVel].....	108
5.15.14.[SOURce[<n>]]:POWER:LIMit.....	109
5.15.15.[SOURce[<n>]]:POWER:PROTection[:LEVel].....	109
5.15.16.[SOURce[<n>]]:POWER:PROTection:DELAy[:TIME].....	110
5.15.17.[SOURce[<n>]]:POWER:PROTection:STATe.....	110
5.15.18.[SOURce[<n>]]:POWER:PROTection:TRIPped?.....	111
5.15.19.[SOURce[<n>]]:VOLTagE.....	111
5.15.20.[SOURce[<n>]]:VOLTagE:LIMit[:POSitive][:IMMediate][:AMPLitude].....	112
5.15.21.[SOURce[<n>]]:VOLTagE:STEP.....	113
5.15.22.[SOURce[<n>]]:VOLTagE:TRIGgered.....	113
5.15.23.[SOURce[<n>]]:VOLTagE:MODE.....	114
5.15.24.[SOURce[<n>]]:VOLTagE:PROGram[:SOURce].....	114
5.15.25.[SOURce[<n>]]:VOLTagE:PROTection[:LEVel].....	115
5.15.26.[SOURce[<n>]]:VOLTagE:PROTection:DELAy[:TIME].....	116
5.15.27.[SOURce[<n>]]:VOLTagE:PROTection:STATe.....	116
5.15.28.[SOURce[<n>]]:VOLTagE:PROTection:TRIPped?.....	117
5.15.29.[SOURce[<n>]]:VOLTagE:PROTection:TYPE.....	117
5.15.30.[SOURce[<n>]]:VOLTagE:RAMP:DURation.....	118
5.15.31.[SOURce[<n>]]:VOLTagE:SENSe[:SOURce].....	118
5.15.32.[SOURce[<n>]]:VOLTagE:SLEW:FALLing.....	118
5.15.33.[SOURce[<n>]]:VOLTagE:SLEW:RISing.....	118
5.16.STATUS.....	119
5.16.1.STATUS:OPERation[:EVENT]?	120
5.16.2.STATUS:OPERation:CONDition?	120
5.16.3.STATUS:OPERation:ENABLE.....	120
5.16.4.STATUS:OPERation:INSTrument[:EVENT]?	121
5.16.5.STATUS:OPERation:INSTrument:CONDition?	121

5.16.6.STATUS:OPERation:INSTrument:ENABle.....	121
5.16.7.STATUS:OPERation:INSTrument:ISUMmary[<n>][:EVENT]?.....	122
5.16.8.STATUS:OPERation:INSTrument:ISUMmary[<n>]:CONDition?.....	122
5.16.9.STATUS:OPERation:INSTrument:ISUMmary<n>:ENABle.....	122
5.16.10.STATUS:PREset.....	123
5.16.11.STATUS:QUEStionable[:EVENT]?.....	123
5.16.12.STATUS:QUEStionable:CONDition?.....	123
5.16.13.STATUS:QUEStionable:ENABle.....	124
5.16.14.STATUS:QUEStionable:INSTrument[:EVENT]?.....	124
5.16.15.STATUS:QUEStionable:INSTrument:CONDition?.....	124
5.16.16.STATUS:QUEStionable:INSTrument:ENABle.....	124
5.16.17.STATUS:QUEStionable:INSTrument:ISUMmary[<n>][:EVENT]?.....	125
5.16.18.STATUS:QUEStionable:INSTrument:ISUMmary[<n>]:CONDition?.....	125
5.16.19.STATUS:QUEStionable:INSTrument:ISUMmary[<n>]:ENABle.....	126
5.17.SYSTem.....	127
5.17.1.SYSTem:BEEPer.....	129
5.17.2.SYSTem:BEEPer:KEY:STATe.....	130
5.17.3.SYSTem:BEEPer:STATe.....	130
5.17.4.SYSTem:CAPability?.....	130
5.17.5.SYSTem:CHANnel[:COUNT]?.....	130
5.17.6.SYSTem:CHANnel:INfOrmation:CURRent?.....	131
5.17.7.SYSTem:CHANnel:INfOrmation:ONTime:LAST?.....	131
5.17.8.SYSTem:CHANnel:INfOrmation:ONTime:TOTal?.....	131
5.17.9.SYSTem:CHANnel:INfOrmation:POWer?.....	131
5.17.10.SYSTem:CHANnel:INfOrmation:VOLTage?.....	132
5.17.11.SYSTem:CHANnel:MODEl?.....	132
5.17.12.SYSTem:CHANnel:OPTion?.....	132
5.17.13.SYSTem:CHANnel:SLOT?.....	133
5.17.14.SYSTem:CHANnel:SNO?.....	133
5.17.15.SYSTem:CHANnel:VERSion?.....	133
5.17.16.SYSTem:COMMunicate:ENABle.....	134
5.17.17.SYSTem:COMMunicate:ETHernet:ADDResS.....	134
5.17.18.SYSTem:COMMunicate:ETHernet:DHCP.....	134
5.17.19.SYSTem:COMMunicate:ETHernet:DNS.....	135
5.17.20.SYSTem:COMMunicate:ETHernet:GATEway.....	135
5.17.21.SYSTem:COMMunicate:ETHernet:HOSTname.....	135
5.17.22.SYSTem:COMMunicate:ETHernet:MAC.....	136
5.17.23.SYSTem:COMMunicate:ETHernet:PORT.....	136
5.17.24.SYSTem:COMMunicate:ETHernet:SMASk.....	136
5.17.25.SYSTem:COMMunicate:MQTT:SETTings.....	136
5.17.26.SYSTem:COMMunicate:MQTT:STATe.....	137
5.17.27.SYSTem:COMMunicate:NTP.....	137
5.17.28.SYSTem:COMMunicate:USB:CLAss.....	138
5.17.29.SYSTem:COMMunicate:USB:MODE.....	138
5.17.30.SYSTem:COMMunicate:RLState.....	138
5.17.31.SYSTem:CPU:FIRMware?.....	139
5.17.32.SYSTem:CPU:INfOrmation:ONTime:LAST?.....	139
5.17.33.SYSTem:CPU:INfOrmation:ONTime:TOTal?.....	139
5.17.34.SYSTem:CPU:MODEl?.....	139
5.17.35.SYSTem:CPU:SNO?.....	140
5.17.36.SYSTem:CPU:VERSion?.....	140
5.17.37.SYSTem:DATE.....	140

5.17.38.SYSTem:DELAy.....	140
5.17.39.SYSTem:DIGital:INPut:DATA.....	141
5.17.40.SYSTem:DIGital:OUTPut:DATA.....	141
5.17.41.SYSTem:DIGital:OUTPut:PWM:DUTY.....	141
5.17.42.SYSTem:DIGital:OUTPut:PWM:FREQuency.....	142
5.17.43.SYSTem:DIGital:PIN<n>:FUNctIon.....	142
5.17.44.SYSTem:DIGital:PIN<n>:POLarity.....	143
5.17.45.SYSTem:ERRor.....	143
5.17.46.SYSTem:ERRor:COUNt?.....	144
5.17.47.SYSTem:FAN:SPEEd?.....	144
5.17.48.SYSTem:FAN:STATus?.....	144
5.17.49.SYSTem:FORMat:DATE.....	144
5.17.50.SYSTem:FORMat:TIME.....	145
5.17.51.SYSTem:INHibit?.....	145
5.17.52.SYSTem:KLOCK.....	145
5.17.53.SYSTem:LOCal.....	145
5.17.54.SYSTem:MEASure[:SCALar]:TEMPerature[:THERmistor][:DC].....	146
5.17.55.SYSTem:MEASure[:SCALar][:VOLTage][:DC]?.....	147
5.17.56.SYSTem:PASSword:CALibration:RESet.....	147
5.17.57.SYSTem:PASSword:FPANel:RESet.....	147
5.17.58.SYSTem:PASSword:NEW.....	147
5.17.59.SYSTem:PON:OUTPut:DISable.....	148
5.17.60.SYSTem:POWer.....	148
5.17.61.SYSTem:POWer:PROTection:TRIP.....	148
5.17.62.SYSTem:RELAy:CYCLes?.....	149
5.17.63.SYSTem:REMote.....	149
5.17.64.SYSTem:REStart.....	149
5.17.65.SYSTem:RWLock.....	150
5.17.66.SYSTem:SLOT[:COUNt]?.....	150
5.17.67.SYSTem:SLOT:MODEl?.....	150
5.17.68.SYSTem:SLOT:STATe.....	150
5.17.69.SYSTem:SLOT:VERSion?.....	151
5.17.70.SYSTem:TEMPerature:PROTection[:HIGH][:LEVel].....	151
5.17.71.SYSTem:TEMPerature:PROTection[:HIGH]:CLEAr.....	151
5.17.72.SYSTem:TEMPerature:PROTection[:HIGH]:DELAy[:TIME].....	152
5.17.73.SYSTem:TEMPerature:PROTection[:HIGH]:STATe.....	152
5.17.74.SYSTem:TEMPerature:PROTection[:HIGH]:TRIPped?.....	152
5.17.75.SYSTem:TIME.....	153
5.17.76.SYSTem:TIME:DST.....	153
5.17.77.SYSTem:TIME:ZONE.....	154
5.17.78.SYSTem:VERSion?.....	154
5.18.TRIGger.....	155
5.18.1.TRIGger[:SEQuence][:IMMediate].....	155
5.18.2.TRIGger[:SEQuence]:DELAy.....	155
5.18.3.TRIGger[:SEQuence]:EXIT:CONDition.....	156
5.18.4.TRIGger[:SEQuence]:SOURce.....	156
5.18.5.TRIGger:DLOG[:IMMediate].....	157
5.18.6.TRIGger:DLOG:SOURce.....	157
6.Device-specific (unclassified) commands.....	159
6.1.1.APPLy.....	159
6.1.2.DEBUg.....	159
7.Error messages.....	161



7.1.Command Error [-199, -100].....	161
7.2.Execution Error [-299, -200].....	162
7.3.Device-Specific Error [-399, -300], [1, 32767] .....	164
7.3.1.Self-Test Error Messages.....	167
8.Parameters and settings.....	171
8.1.Programming parameters.....	171
8.1.1.Voltage.....	171
8.1.2.Current.....	171
8.1.3.Power.....	171
8.2.Reset Settings (*RST).....	172
8.3.Special modes of operation.....	174
8.4.Default settings.....	174
9.Software simulator.....	177
9.1.SIMUlator.....	178
9.1.1.SIMUlator:EXIT.....	178
9.1.2.SIMUlator:GUI.....	178
9.1.3.SIMUlator:LOAD.....	178
9.1.4.SIMUlator:LOAD:STaTe.....	179
9.1.5.SIMUlator:PIN1.....	179
9.1.6.SIMUlator:PIN2.....	180
9.1.7.SIMUlator:PWRGood.....	180
9.1.8.SIMUlator:QUIT.....	180
9.1.9.SIMUlator:RPOL.....	181
9.1.10.SIMUlator:TEMPerature.....	181
9.1.11.SIMUlator:VOLTagE:PROGram:EXTeRnal.....	181
10.Programming examples.....	183
10.1.Set channel output values and working with the OCP.....	183
10.2.Voltage and current calibration.....	185
10.3.Working with profiles.....	186
10.4.Get identification info and self-test results.....	187
10.5.Programming output voltage using the list of values.....	188
11.SCPi commands scheduled for upcoming releases.....	191
12.SCPi commands summary.....	193

## 1. Introduction

This manual contains reference information for programming the open hardware/open source EEZ Bench Box 3 (BB3) that includes [STM32F7 MCU board](#) and multiple peripheral modules such as [DCP405](#) or [DCM220](#) power modules over the remote interface using the SCPI programming language.

The SCPI (*Standard Commands for Programmable Instruments*, often pronounced “skippy”) is an open standard freely available on the [IVI Foundation](#) web pages. The current version is SCPI 1999.0.

SCPI is a pure software standard, and can be used over many communication interfaces. SCPI communications are ASCII text, and therefore can be supported in programs written in almost any computer language, such as C, C++, etc.

The physical communications link is not defined by SCPI. It was originally created with the IEEE 488 (GPIB) environment in mind, but it can also be used with RS-232 (serial), Ethernet, USB, VXIbus, HiSLIP, etc. The BB3 supports Serial (via USB) and Ethernet communication.

The application software that uses SCPI commands is called a *Controller* and that in a SCPI enabled device – such as the BB3 – is called an *Instrument*.

*Please note that IEEE 488 standard documents are not freely available, and when it's mentioned in this manual we do so only for reference purposes. Those who wish to research the GPIB for better understanding or possible modification/improvement of the BB3 remote control may wish to purchase standards documents from the IEEE.*

### 1.1. About SCPI

The SCPI 1999.0 standard document says (Section 1.3) the goal of SCPI is to reduce Automatic Test Equipment (ATE) program development time. SCPI does this goal by providing a consistent programming environment for instrument control and data usage. This is achieved by use of defined program messages, instrument responses, and data formats across all SCPI instruments, regardless of manufacturer.

A consistent program environment uses the same commands and parameters to control instruments that have the same function.

SCPI programming consistency is both vertical and horizontal. Vertical programming consistency defines program messages within an instrument class. An example of vertical consistency is using the same command for reading DC voltage from different multimeters supporting SCPI. Horizontal consistency uses the same command to control similar functions across instrument classes. For example, the trigger command would be the same for trigger functions found in conforming counters, oscilloscopes, function generators, etc.

A key to consistent programming is the reduction of multiple ways to control similar instrument functions. The philosophy of SCPI is that the same instrument functions are to be controlled by the same SCPI commands. To simplify learning, SCPI uses industry-standard names, and terms that are manufacturer and customer supported.

SCPI is designed to be expanded with new defined commands in the future without causing programming problems. As new instruments are introduced, the intent is to maintain program compatibility with existing SCPI instruments.

Additional links:

- Wikipedia [SCPI](#)
- Technopedia [Standard Commands For Programmable Instruments \(SCPI\)](#)
- Wikipedia [IEEE-488](#)
- Keysight (ex. Agilent) [Developing a SCPI command set](#)
- NI (National Instrument) [GPIB Hardware and Software Specifications](#)

Implementation links:

- [Open source SCPI device](#) library
- Keysight (ex. Agilent) [Application Note 1465-29](#)
- Keysight (ex. Agilent) [Command Expert](#)



## 2. Syntax and style

Throughout this document, the following conventions are used for the SCPI command syntax:

- Square brackets ([]) indicate optional keywords or parameters. The braces are not sent with the command string.
- Braces ({} ) enclose parameters within a command string.
- Triangle brackets (<>) indicate that you must substitute a value or a code for the enclosed parameter.
- A vertical bar (|) separates one of two or more alternative parameters.

### 2.1. Root Specifier

When it precedes the first header of a message unit, the colon becomes the root specifier. It tells the command parser that this is the root or the top node of the command tree.

### 2.2. Command termination

A command string sent to the BB3 must terminate with a <new line>character. A <carriage return> followed by a <new line>is also accepted. Command string termination will always reset the current SCPI command path to the root level.

### 2.3. Command separators

A colon (:) is used to separate a command keyword from a lower-level keyword as shown below:

```
SOURce1:CURRent:PROTection:STATe
```

A semicolon (;) is used to separate two commands within the same subsystem, and can also minimize typing. For example, sending the following command string,

```
SOURce1:VOLTagE 20;CURRent 300mA
```

is the same as sending the following two commands:

```
SOURce1:VOLTagE 20  
SOURce1:CURRent 1.5
```

Use a colon and a semicolon to link commands from different subsystems. For example, in the following command string, an error is generated if you do not use the colon and semicolon:

```
SYSTem:BEEP;:SOURce1:CURRent 2.5
```

### 2.4. Querying parameter settings

You can query the value of most parameters by adding a question mark (?) to the command. For example, the following command sets the output voltage to 25.5 V:

```
VOLTagE 25.5
```

You can query the value by executing:

```
VOLTagE?
```

If error is occurred use [SYSTem:ERRor\[:NEXT\]?](#) to get more information about error.

### 2.5. Using the MIN, MAX, and DEF Parameters

For many commands, you can substitute "MIN" or "MAX" in place of a parameter. In some cases you may also substitute "DEF". For example, consider the following command:

```
[SOURce[<n>]]:VOLTagE[:LEVel][:IMMediate][:AMPLitude] {<voltage>|MIN|DEF|MAX|UP|DOWN}
```

Instead of selecting a specific value for the <voltage> parameter, you can substitute MIN to set the voltage to its minimum value, MAX to set the voltage to its maximum value, or DEF to set the voltage to its default value. For list of parameter values see [Section 8.1](#)

## 2.6. Command and message types

SCPI commands can be divided to **common** and **subsystem** commands.

- Common commands are defined by the IEEE 488.2 standard to perform common interface functions. They begin with an \* and consist of three letters (command) or three letters and a ? (query). Description of supported common commands can be found in [Section 4](#)
- Subsystem commands are specific to instrument functions. They can be a single command or a group of commands. The groups are comprised of commands that extend one or more levels below the root. See [Section 5](#) for commands that is created in accordance to the SCPI 1999.0 standard. Commands that is not defined by SCPI 1999.0 is labeled "unclassified" and are presented in [Section 6](#)

There are two types of SCPI messages, **program** and **response**.

- A program message consists of one or more properly formatted SCPI commands sent from the controller to the instrument. The message, which may be sent at any time, requests the instrument to perform some action.
- A response message consists of data in a specific SCPI format sent from the instrument to the controller. The instrument sends the message only in response to a **query** header.

## 2.7. Required Commands

The following commands are required in all SCPI instruments (see SCPI 1999.0 Section 4.2.1):

Mnemonic	SCPI 1999.0 Command Reference Section	SCPI 1999.0 Syntax and Style Section
:SYSTem		
:ERRor	21.8	
[:NEXT]?	21.8.3e	1996
:VERSion?	19.16	1991
:STATus	18	5
:OPERation		
[:EVENT]?		
:CONDition?		
:ENABle		
:ENABle?		
:QUESTionable		
[:EVENT]?		
:CONDition?		
:ENABle		
:ENABle?		
:PRESet		

### 2.7.1. Base functionality for the Power supply instrument class

SCPI Command	Description
OUTPut	
[:STATe] <bool>	Enables the specified output channel(s)
[SOURce[<n>]]	
CURRent	

```

[:LEVel]
    [:IMMediate][:AMPLitude] <current>    Sets the output current
VOLTage
    [:LEVel]
    [:IMMediate][:AMPLitude] <voltage>    Sets the output voltage

```

All SCPI power supplies shall implement the status reporting structure. STATUS Subsystem defines the commands which shall be used to control the status reporting structure. For a power supply, the bits of interest in the QUESTIONable status structure are VOLTage and CURRENT. When a power supply is operating as a voltage source, bit 1 (CURRENT) shall be set. When a power supply is operating as a current source, bit 0 (VOLTage) shall be set. When the output is unregulated, both bits shall be set (for example, while the output is changing to a new programmed value).

## 2.8. Multiple Commands in a Message

Multiple SCPI commands can be combined and sent as a single message with one message terminator. There are two important considerations when sending several commands within a single message:

- Use a semicolon to separate commands within a message.
- There is an implied header path that affects how commands are interpreted by the BB3.

The header path can be thought of as a string that gets inserted before each command within a message. For the first command in a message, the header path is a null string. For each subsequent command the header path is defined as the characters that make up the headers of the previous command in the message up to and including the last colon separator. An example of a message with two commands is:

```
OUTPut:STATe ON,CH1;PROTection:CLEAr CH1
```

which shows the use of the semicolon separating the two commands, and also illustrates the header path concept. Note that with the second command, the leading header OUTPut was omitted because after the OUTPut:STATe ON command, the header path became defined as OUTPut and thus the instrument interpreted the second command as:

```
OUTPut:PROTection:CLEAr CH1
```

In fact, it would have been syntactically incorrect to include the OUTPut explicitly in the second command, since the result after combining it with the header path would be:

```
OUTPut:OUTPut:PROTection:CLEAr CH1
```

which is incorrect.

You can combine common commands (IEEE488) with subsystem commands in the same message. Treat the common command as a message unit by separating it with a semicolon (the message unit separator). Common commands do not affect the header path; you may insert them anywhere in the message.

```
*TST?;SYSTem:ERRor?
```

## 2.9. Moving Among Subsystems

In order to combine commands from different subsystems, you need to be able to reset the header path to a null string within a message. You do this by beginning the command with a colon (:), which discards any previous header path. For example, you could clear the output protection and check the status of the Operation Condition register in one message by using a root specifier as follows:

```
OUTPut:PROTection:CLEAr CH1;:STATus:OPERation:CONDition?
```

The following message shows how to combine commands from different subsystems as well as within the same subsystem:

```
SOURce1:VOLTage:LEVel 7.5;:VOLTage:PROTection:DELay 10;:CURRENT:LEVel 0.5
```

Note the use of the optional header LEVel to maintain the correct path within the subsystems, and the

use of the root specifier to move between subsystems.

## 2.10. SCPI parameter types

The SCPI language defines several different data formats to be used in program messages and response messages:

**Numeric** Commands that require numeric parameters will accept all commonly used representations of numbers like integer (also known as NR1 format specified in ANSI X3.42-1990) or decimal representations of numbers including optional signs, decimal points (NR2 format), and scientific notation (i.e. 10E3 or NR3 format). Special values for numeric parameters like MINimum, MAXimum, and DEFault are also accepted. You can also send engineering unit suffixes (V, A, or SEC) with numeric parameters. If only specific numeric values are accepted, the BB3 will automatically round the input numeric parameters. The following command uses a numeric parameter:

```
VOLT:STEP {<step>}
```

**Discrete** Used to program settings that have a limited number of values such as BUS and IMM or CH1 and CH2. Query responses will always return the short form in all uppercase letters. The following command uses discrete parameters:

```
CAL:CURREN:LEV {MIN|MID|MAX}
```

**Boolean** Represent a single binary condition that is either true or false. For a false condition, the BB3 will accept OFF or 0. For a true condition, the BB3 will accept ON or any nonzero value (i.e. 1 but also 2.34 or -3). When you query a Boolean setting, the BB3 will always return 0 or 1. The following command uses a Boolean parameter:

```
OUTP {OFF|ON}
```

**String** Can contain virtually any set of ASCII characters. A string must begin and end with matching quotes, either with a single quote or with a double quote. You can include the quote delimiter as part of the string by typing it twice without any characters in between. The following command uses a string parameter:

```
CAL:REM <quoted string>
```

**Data block** #<length-digits><length><block>

<length-digits>	NR1	<length> number of digits (e.g. 2 for two digit number)
<length>	NR1	Limited only with available space on SD card
<block>	Discrete	Only exact size is allowed as defined with <length>

Downloads text *Hello world* and store into the file "test file" in the current directory. Digit 2 denotes two digits of data length (11).

```
MMEM:DOWN:FNAM "test file"
MMEM:DOWN:DATA #211Hello world
MMEM:DOWN:FNAM ""
```

**Channel list** The channel list parameter is used for identifying the channel number as well as the numeric suffix. A channel list always starts with an @ and is enclosed in parentheses.

The notation (@1,2) specifies a channel list that includes channels 1 and 2.

The notation (@2:4) specifies a channel list that includes channels 2 to 4.

The notation (@1,3:4) specifies a channel list that includes channels 1, 3 and 4. No space is allowed after or before comma separator.

*The channel list parameter is only available on certain commands.*

### 3. Registers and queues

SCPI requires the status mechanism described in Section 11 of IEEE 488.2, including full implementation of the status register structure. Summary of implemented registers structure for the BB3 is shown on Fig. 1. (commands used to access registers are written in parentheses).

All SCPI instruments have to implement status registers in the same way. The status system records various instrument conditions in the following register groups:

- the Status Byte register,
- the Standard Event register,
- the QUEStionable Status register group, and
- the OPERation Status register group.

The Status Byte register records high-level summary information reported in the other register groups. Message interchanging between Controller and Instrument is accomplished by using input buffer and Output queue and Error queue. The length of the Input buffer is 48 characters. Both Output and Error queue can handle up to 20 messages.

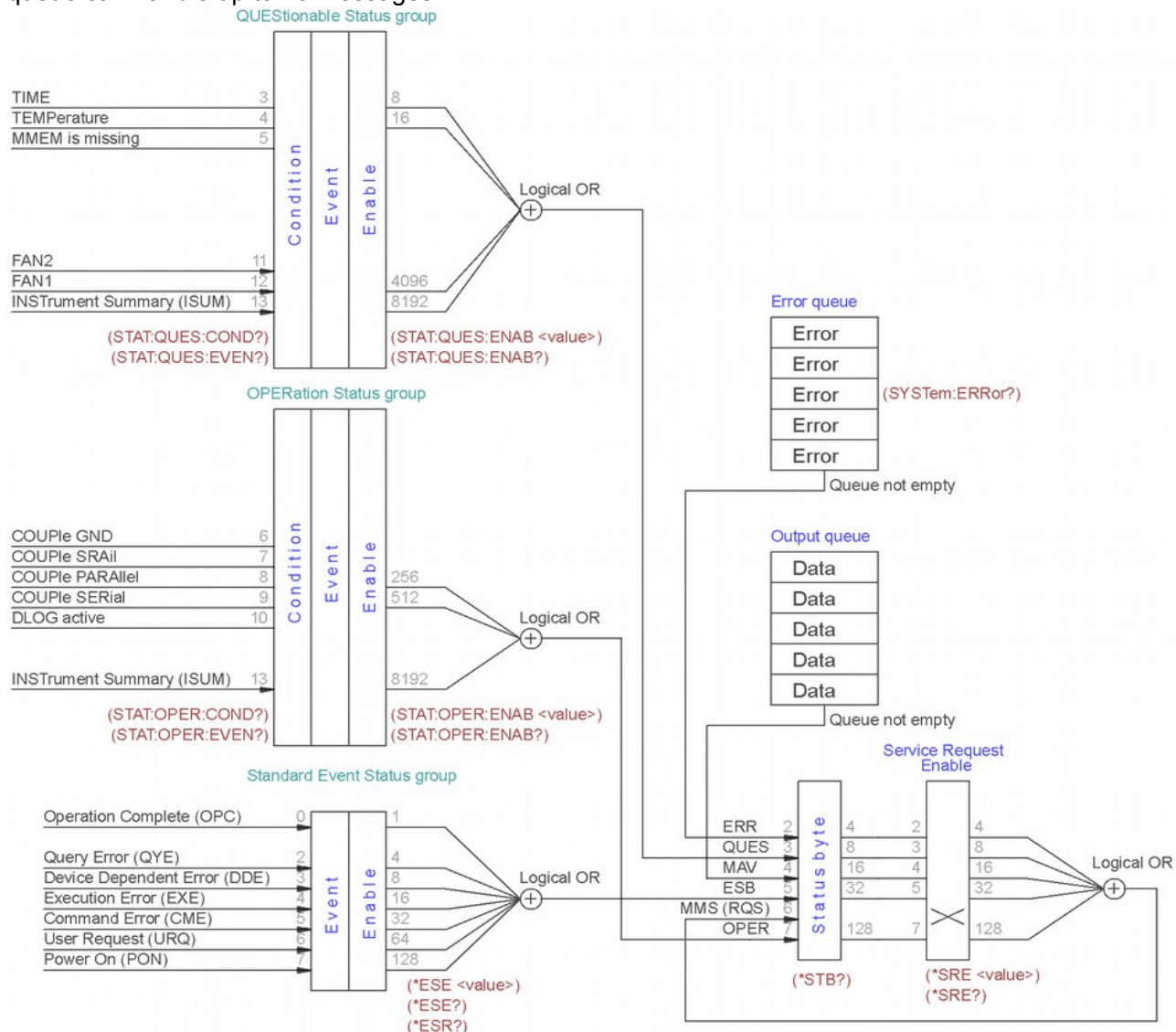


Fig. 1: Summary of status structure registers

#### 3.1. Standard Event Status Register

An status register group is consist of Condition, Event and Enable registers (see Fig. 1):

- The Condition register is a read-only register, which holds the live (unlatched) operational status of the instrument. Reading the Condition register does not clear it.
- The Event register is a read-only that reports defined conditions within the BB3. Bits in an event register are latched. Once an event bit is set, subsequent state changes are ignored. Bits in the



Event register are automatically cleared by a query of that register (such as \*ESR? or STATus:QUESTionable:EVENT?) or by sending the \*CLS (clear status) command. A reset (\*RST) or device clear will *not* clear bits in event registers (See [Section 8.2](#)). Querying an event register returns a decimal value which corresponds to the binary-weighted sum of all bits set in the register.

- The ENABLE register is used to define which bits of the Event Status register will latch ESB (bit 5) of the Status byte register.

An error status (bit 2, 3, 4 or 5) records one or more errors in the BB3 error queue. The SYSTem:ER-Ror? command can be used to read the error queue.

Implementation of the Standard Event Status register follows IEEE 488.2 Section 11.5.1.1:

Bit	Decimal value	Description
0	1	Operation Complete (OPC) – This event bit is generated in response to the *OPC command. It indicates that the BB3 has completed all selected pending operations (including *OPC).
1	2	Not used
2	4	Query ERROR (QYE) – Query Errors are detected by the Output Queue Control. This event bit indicates that either <ul style="list-style-type: none"> <li>• An attempt is being made to read data from the Output Queue when no output is either present or pending, or</li> <li>• Data in the Output Queue has been lost.</li> </ul> Events that generate Query Errors do not generate Execution Errors, Command Errors, or Device-Specific Errors.
3	8	Device-Specific ERROR (DDE) – This event bit indicates that an error has occurred that is neither a Command Error, a Query Error, nor an Execution Error. A Device-Specific Error is any executed device operation that did not properly complete due to some condition, such as over-range, a self-test or calibration error. Following a Device-Specific Error, the BB3 will continue to process the input stream. Events that generate Device-Specific Errors do not generate Command Errors, Query Errors, or Execution Errors.
4	16	Execution ERROR (ERR) – This event bit indicates that: <ul style="list-style-type: none"> <li>• A &lt;PROGRAM DATA&gt; element following a header was evaluated by the BB3 as outside of its legal input range or is otherwise inconsistent with the BB3's module capabilities.</li> <li>• A valid program message could not be properly executed due to some BB3 condition.</li> </ul> Following an Execution Error, the BB3 will continue parsing the input stream. Execution Errors will be reported by the BB3 after rounding and expression evaluation operations have taken place. Rounding a numeric data element, for example, will not be reported as an Execution Error. Events that generate Execution Errors do not generate Command Errors, Query Errors, or Device-Specific Errors.
5	32	Command ERROR (CME) – Command Errors are detected by the parser. This event bit indicates that one of the following events has occurred: <ul style="list-style-type: none"> <li>• An IEEE 488.2 syntax error has been detected by the parser. That is, a controller-to-device message was received that is in violation of this standard. Possible violations include a data element that violates the device listening formats or whose type is unacceptable to the device (see also IEEE 488.2 Section 7.1.2.2).</li> </ul>

- A semantic error has occurred indicating that an unrecognized header was received. Unrecognized headers include incorrect device-specific headers and incorrect or unimplemented common commands described in [Section 4](#)

When the BB3 detects a Command Error, parser synchronization may be lost. *When a Command Error is detected, any prior parsable elements of the same <PROGRAM MESSAGE> will be executed. That is also true for all parsable elements that follows after detected Command Error.*

The Command Error bit not be set to report any other device-specific condition. Events that are reported as Command Errors cannot be reported as Execution Errors, Query Errors, or Device-Specific Errors.

6	64	User Request (URQ) – This event bit indicates that the BB3 input device (TFT Touch screen) has been for any reason activated. The setting of this event-bit occur regardless of the IEEE 488.1 Remote/Local state of the device ( <i>not implemented yet</i> )
7	128	Power On (PON) – This event bit indicates that an off-to-on transition has occurred in the device's power supply (i.e. AUX PS module). See also SYSTem:POWer.
8 – 15	–	Not used, always zero

### 3.2. Status Byte Register

The Status Byte summary register reports conditions from the other status registers (see Fig. 1). Query data that is waiting in the BB3's output buffer is immediately reported through the "Message Available" (MAV) bit (bit 4) of the Status Byte register. Bits in the summary register are NOT latched. Clearing an event register will clear the corresponding bits in the Status Byte summary register. Reading all messages in the output buffer, including any pending queries, will clear the message available bit (MAV). The Status Byte summary register is cleared when the \*CLS (clear status) command has been executed.

The Status Byte enable register (request service) is cleared when the \*SRE 0 command has been executed.

Querying the Standard Event register (\*ESR? command) will clear only bit 5 (ESR) in the Status Byte summary register. For example, 24 (8 + 16) is returned when you have queried the status of the Status Byte register, QUES and MAV conditions have occurred.

Bit	Decimal value	Description
0 – 1	–	Not used, always zero
2	4	ERR – Error queue bit indicates that one or more errors have been stored in the Error queue.
3	8	QUES – One or more bits are set in the QUEStionable Status register (bits must be enabled in the enable register).
4	16	MAV – The Message Available bit indicates whether or not the Output Queue is empty. Whenever the device is ready to accept a request by the controller to output data bytes, the MAV is TRUE. The MAV is FALSE when the Output Queue is empty. This bit is used to synchronize information exchange with the controller. The controller can, for example, send a query command to the device and then wait for MAV to become TRUE.
5	32	ESB – One or more bits are set in the Standard Event register (bits must be enabled in the enable register, see *ESE command).
6	64	MMS – Master Status summary bit indicates that one or more bits are set in the Status Byte Register (bits must be enabled, see *SRE command). Also used to

indicate a request for service (RQS).

7            128      OPER – One or more bits are set in the OPERation Status register.

### 3.3. OPERation Status Register

The OPERation status register contains conditions which are part of the instrument's normal operation. Each channel of the BB3 is considered as separate "instrument". Up to six logical outputs (channels) of the BB3 include an INSTRument summary status register and an individual instrument ISUMmary register for each logical output.

The bit definition of OPERation Status register shown on Fig. 1.:

Bit	Decimal value	Modules	Description
0 – 5	–		Not used, always zero
6	64		COUPle CGND indicates that all channels Vout- terminals are coupled together.
7	128	<b>DCP405</b>	COUPle SRail indicates that two modules are connected as split rail.
8	256	<b>DCP405</b>	COUPle PARAllel indicate that two modules are connected in parallel.
9	512	<b>DCP405</b>	COUPle SERial indicate that two modules are connected in serial.
10	1024		Internal data logging is in progress.
11 – 12	–		Not used, always zero
13	8192		INSTRument Summary Bit – One of n multiple logical instruments is reporting OPERational status.
14 – 15	–		Not used, always zero

The Event Status Enable register is cleared when the STAT:EVEN:ENAB 0 command is executed. The \*CLS command can be also used to clear the register.

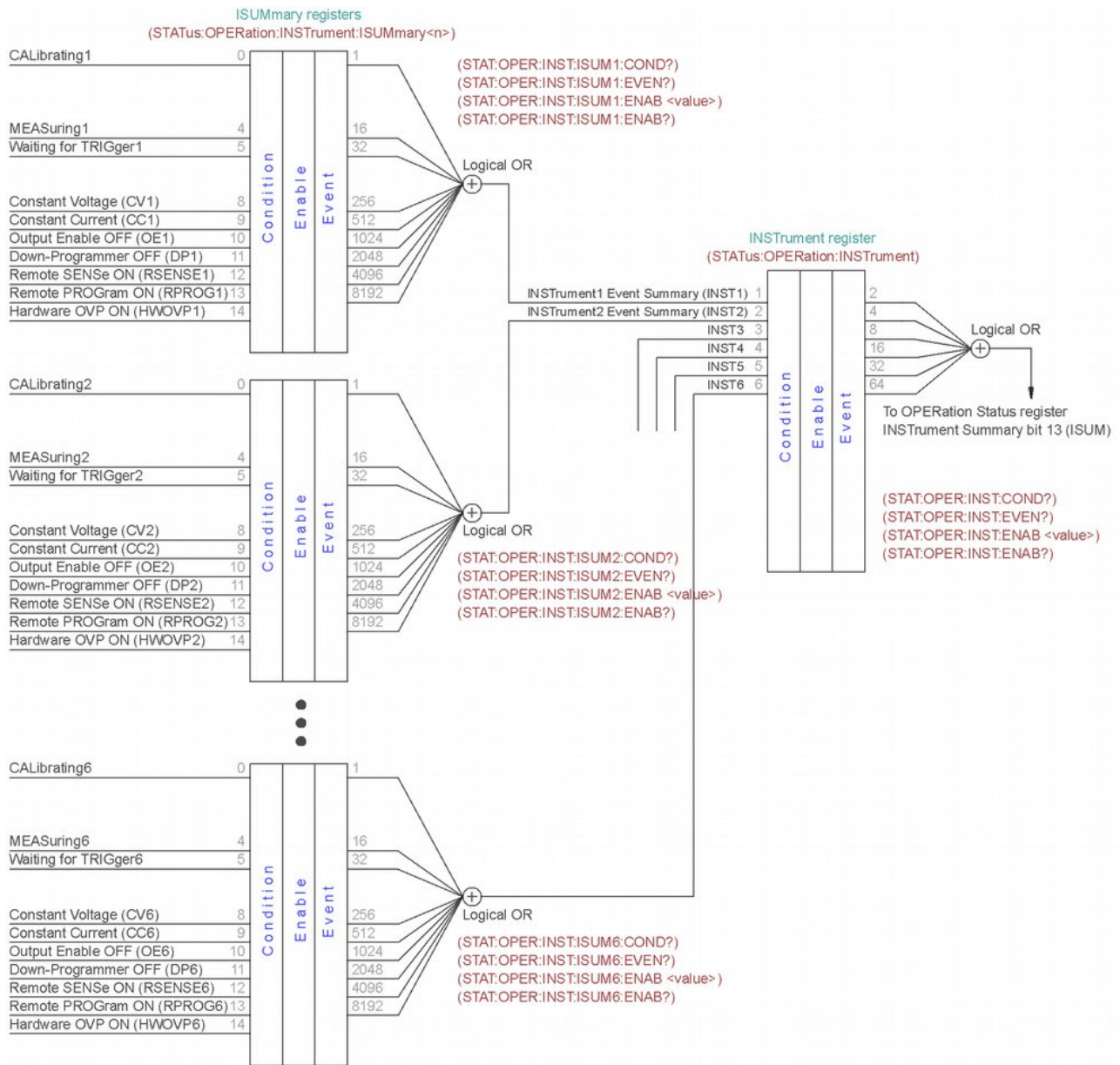


Fig. 2: OPERATION Status registers summary

### 3.3.1. Operation INSTRument Status register

The bit definition of OPERATION INSTRument Status register shown on Fig.2.:

Bit	Decimal value	Description
0	–	Not used, always zero
1	2	INST1 – Instrument1 summary bit indicate that one or more bits are changed in the Channel 1 OPERATION INSTRument Summary register
2	4	INST2 – Instrument2 summary bit indicate that one or more bits are changed in the Channel 2 OPERATION INSTRument Summary register
3	8	INST3 – Instrument3 summary bit indicate that one or more bits are changed in the Channel 3 OPERATION INSTRument Summary register
4	16	INST4 – Instrument4 summary bit indicate that one or more bits are changed in the Channel 4 OPERATION INSTRument Summary register
5	32	INST5 – Instrument5 summary bit indicate that one or more bits are changed in the Channel 5 OPERATION INSTRument Summary register
6	64	INST6 – Instrument6 summary bit indicate that one or more bits are changed in

		the Channel 6 OPERation INSTRument Summary register
7 – 15	–	Not used, always zero

### 3.3.2. Operation Instrument SUMmary status register

The ISUMmary registers report to the INSTRument register, which in turn reports to bit 13 of the Operation Status register. This is illustrated on Fig. 2. Using such a status register configuration allows a status event to be cross-referenced by output channel and type of event. The INSTRument register indicates which channel(s) have generated an event. The ISUMmary register represent a pseudo-operation Status register for a particular logical output.

The bit definition of OPERation INSTRument ISUMmary Status register shown on Fig.2.:

Bit	Decimal value	Modules	Description
0	1		CALibrating – Channel is performing calibration
1 – 3	–		Not used, always zero
4	16		MEASuring – Channel is performing measurement ( <i>not implemented yet</i> )
5	32		Waiting for TRIGger – Channel is waiting for the trigger event
6 – 7	–		Not used, always zero
8	256		CV – Channel is entered CV operation mode
9	512		CC – Channel is entered CC operation mode
10	1024		OE – Output is switched off
11	2048	DCP405	DP – Down-programmer is switched off
12	4096	DCP405	RSense – Remote voltage sense is switched on
13	8192	DCP405	RPROG – Remote voltage programming is switched on
14	16384	DCP405	HWOVP – Hardware OVP is switched on
15	–		Not used, always zero

### 3.4. QUEStionable Status Register

The Questionable Status register provides information about unexpected operations of the BB3. Each channel of the BB3 is considered as separate "instrument". Up to six logical outputs (channels) of the BB3 include an INSTRument summary status register and an individual instrument ISUMmary register for each logical output.

The ISUMmary registers report to the INSTRument register, which in turn reports to bit 13 of the Questionable Status register. This is illustrated on Fig. 3. Using such a status register configuration allows a status event to be cross-referenced by output channel and type of event. The INSTRument register indicates which channel(s) have generated an event. The ISUMmary register represent a pseudo-Questionable Status register for a particular logical output.



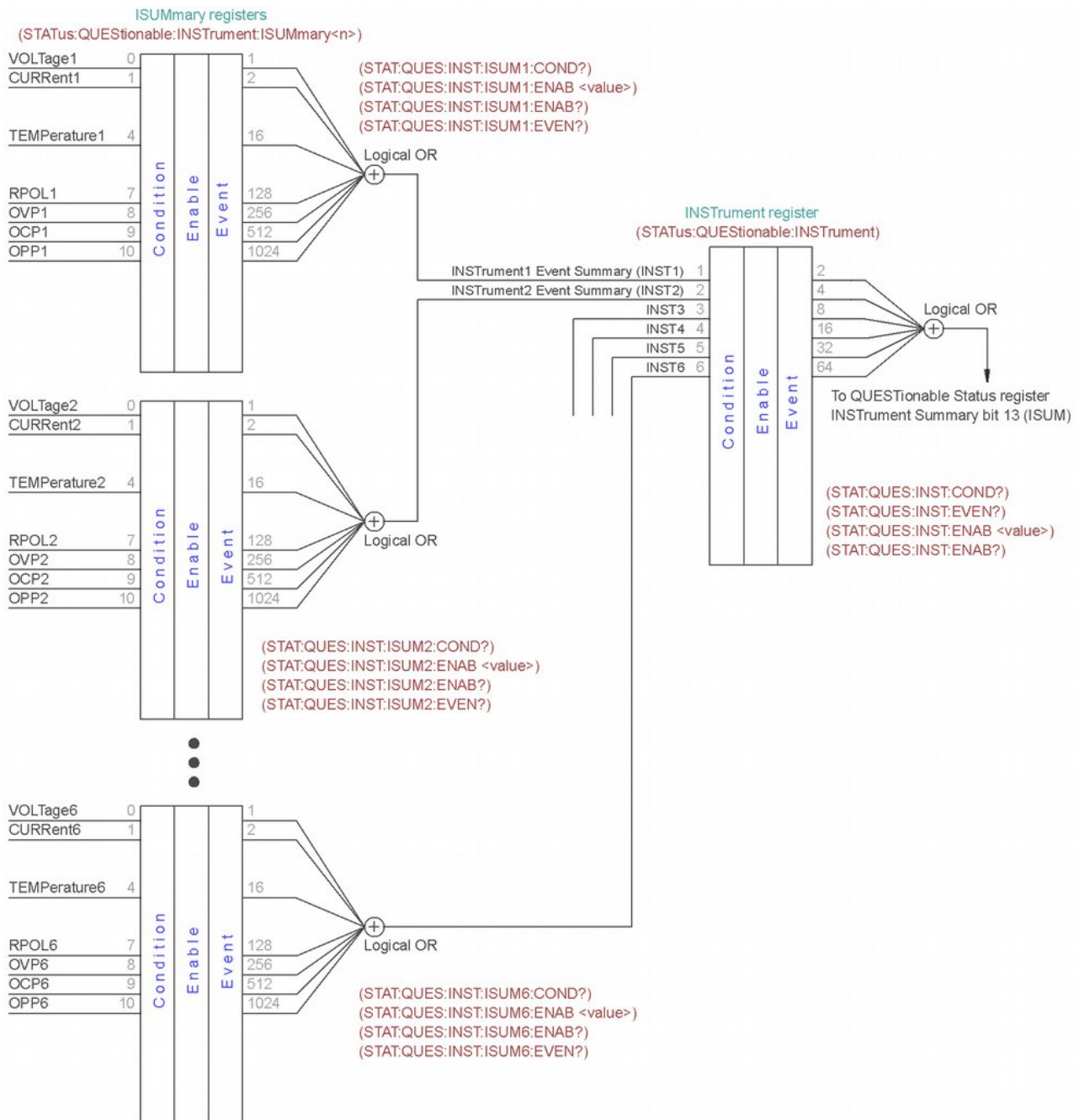


Fig. 3: QUEStionable INSTrument registers summary

For example, if one of the two channels is in constant voltage (CV) mode and due to an overload loses regulation, bit 13 is set (latched). To read the register, the command STAT:QUES:INST:ENAB 6 (2 + 4) has to be send to enable the Questionable instrument register, followed by the command STAT:QUES:INST:ISUM<n>:ENAB 19 for each channel to enable the QUESTIONABLE INSTRUMENT SUMMARY register, where n is 1 or 2.

Bit definition for QUEStionable Status register shown on Fig. 1.:

Bit	Decimal value	Description
0 – 2	–	Not used, always zero
3	8	TIME – indicate abnormal time/date situation due to RTC failure or conflict between current and time/date retrieved from the stored configuration.
4	16	TEMPerature – temperature measurement that use the AUX temperature sensor on the AUX PS module require attention (i.e. over-temperature condition is

detected, sensor is not functional, etc.).

*Do not confuse this sensor with that is connected to a power module.*

5	32	MMEM – SD card is not inserted
6 – 10	–	Not used, always zero
11	2048	FAN2 – auxiliary fan failure is detected (if installed)
12	4096	FAN1 – cooling fan failure is detected
13	8192	INSTrument summary, is described later in this chapter in association with multiple logical instruments.
14 – 15	–	Not used, always zero

The Questionable Status Enable register is cleared when the STAT:QUES:ENAB 0 command is executed. The \*CLS command can be also used to clear the register.

### 3.4.1. Questionable INSTrument Status register

Bit definition for QUESTionable INSTrument register:

Bit	Decimal value	Description
0	–	Not used, always zero
1	2	INST1 – Instrument1 summary bit indicate that one or more bits are changed in the Channel 1 OPERation INSTrument Summary register.
2	4	INST2 – Instrument2 summary bit indicate that one or more bits are changed in the Channel 2 OPERation INSTrument Summary register.
3	8	INST3 – Instrument3 summary bit indicate that one or more bits are changed in the Channel 3 OPERation INSTrument Summary register.
4	16	INST4 – Instrument4 summary bit indicate that one or more bits are changed in the Channel 4 OPERation INSTrument Summary register.
5	32	INST5 – Instrument5 summary bit indicate that one or more bits are changed in the Channel 5 OPERation INSTrument Summary register.
6	64	INST6 – Instrument6 summary bit indicate that one or more bits are changed in the Channel 6 OPERation INSTrument Summary register.
7 – 15	–	Not used, always zero

### 3.4.2. Questionable Instrument SUMmary status register

There are three questionable instrument summary registers, one for each power module output. These registers provide information about voltage and current regulation.

Bit definition for QUESTionable INSTrument SUMmary register:

Bit	Decimal value	Modules	Description
0	1		VOLTage – This bit is set when the voltage becomes unregulated, therefore a channel enters CC operation mode. <i>If the over-voltage protection (OVP) is activated (see VOLTage:PRO-Tection:STATe) channel output will be switched off.</i>
1	2		CURRent – This bit is set when the current becomes unregulated, therefore a channel enters CV operation mode. <i>If the over-current protection (OCP) is activated (see CURRent:PRO-Tection:STATe) channel output will be switched off.</i>
2 – 3	–		Not used, always zero
4	16		TEMPerature – the temperature sensor on the module requires atten-

tion (i.e. over-temperature condition is detected, sensor is not functional, etc.).

*Do not confuse this sensor with the AUX sensor available on the AUX PS module.*

5 – 6	–		Not used, always zero
7	128	<b>DCP405</b>	RPOL – Remote sense reverse polarity is detected.
8	256		OVP – Over-voltage protection is activated. The query VOLT:PROT:TRIP? returns value of this bit. See also STAT:QUES.
9	512		OCP – Over-current protection is activated. The query CURR:PROT:TRIP? returns value of this bit. See also STAT:QUES.
10	1024		OPP – Over-power protection is activated. The query POW:PROT:TRIP? returns value of this bit. See also STAT:QUES.
11 – 15	–		Not used, always zero

*Please note here that CURRent bit is use for questionable Voltage operating mode and vice versa.*

*If 0 and 1 bits is true that indicate neither the voltage nor the current is regulated (so-called unregulated or UR mode), and both bits false indicate the BB3 channel are off.*

To read the register for each BB3 channel, the command STAT:QUES:INST:ISUM[<n>]? has to be send, where [<n>] is 1, 2 or 3. If [<n>] is not specified the currently selected channel is used.

Use STAT:QUES:INST:ISUM<n>:COND? to determine operating mode (CV or CC) for the BB3 channel (where n is 1, 2 or 3 depending on the output).

The Questionable Status event register is cleared with:

- the \*CLS (clear status) command or
- the event register is queried using the STAT:QUES? (status questionable event register) command.

### 3.5. Error queue

The error queue contains items that include a numerical and textual description of the error or event.

The <Error/event\_number> is a unique integer in the range [-32 768, 32 767]. All positive numbers are instrument-dependent. All negative numbers are reserved by the SCPI standard with certain standard error/event codes. The value, zero, is also reserved to indicate that no error or event has occurred.

The second parameter of the full response is a quoted string containing an <Error/event\_description>. Each <Error/event\_number> has a unique and fixed <Error/event\_description> associated with it. An example:

```
-113,"Undefined header"
```

The maximum string length of <Error/event\_description> plus <Device-dependent\_info> is 255 characters. List of all error/event messages can be found in [Section 7](#) of this document.

As errors and events are detected, they are placed in a queue. This queue is first in, first out. If the queue overflows, the last error/event in the queue is replaced with error:

```
-350,"Queue overflow"
```

Any time the queue overflows, the least recent errors/events remain in the queue, and the most recent error/event is discarded. Reading an error/event from the head of the queue removes that error/event from the queue, and opens a position at the tail of the queue for a new error/event, if one is subsequently detected.

If the error queue is not empty, bit 2 of the Instrument Summary Status Register is set. A query returns only the oldest error code and associated error description information from the error queue. To return all error codes and associated description information, use repetitive queries until an error value of zero is returned, or until bit 2 of the status register is 0.

The error queue is cleared when any of the following occur (IEEE 488.2, section 11.4.3.4):



### *EEZ BB3 SCPI reference*

- Upon power up
- Upon receipt of a \*CLS command
- Upon reading the last error message from the queue

## 4. Common command reference

This section summarizes the mandatory subset of IEEE 488.2 commands required for any SCPI compliant instrument.

Common command	Description
<a href="#">*CLS</a>	Clears all status data structures
<a href="#">*ESE {&lt;value&gt;}</a>	Programs the Standard Event Status Enable register bits
<a href="#">*ESR?</a>	Reads the Standard Event Status Register
<a href="#">*IDN?</a>	Returns the UNIQUE identification of the BB3
<a href="#">*OPC</a>	Operation Complete Command used for program synchronization
<a href="#">*RCL {&lt;profile&gt;}</a>	Recalls the BB3 state stored in the specified storage location
<a href="#">*RST</a>	Reset BB3 to the initial state
<a href="#">*SAV {&lt;profile&gt;}</a>	Stores the current BB3 state in the specified storage location
<a href="#">*SRE</a>	Enables bits in the Status Byte enable register.
<a href="#">*STB?</a>	Reads the Status Byte register
<a href="#">*TRG</a>	Generates a software trigger
<a href="#">*TST?</a>	Returns Self-Test results
<a href="#">*WAI</a>	Waits until all pending commands are completed

### 4.1. \*CLS

**Syntax** [\\*CLS](#)

**Description** Clear Status Command. This command clears all status data structures in the BB3:

- Standard Event Status Register
- OPERation Event Status Register
- QUEStionable Event Status Register
- Error/Event Queue

The corresponding condition and enable registers are unaffected. If \*CLS immediately follows a program message terminator (<NL>), then the output queue and the MAV bit are also cleared.

**Return** None

**Related Commands** [\\*ESR?](#)  
STATus:OPERation[:EVENT]  
STATus:OPERation:INSTrument[:EVENT]  
STATus:OPERation:INSTrument:ISUMmary[<n>][:EVENT]  
STATus:QUEStionable[:EVENT]  
STATus:QUEStionable:INSTrument[:EVENT]  
STATus:QUEStionable:INSTrument:ISUMmary[<n>][:EVENT]  
SYSTem:ERRor

### 4.2. \*ESE

**Syntax** [\\*ESE {<value>}](#)  
[\\*ESE?](#)

**Description** Standard Event Status Enable Command. This command sets the Standard Event Status Enable register bits in the BB3. Those settings determine which events of the Standard Event Status Event register (see [\\*ESR?](#)) are allowed to set the ESB (Event Summary Bit) of the Status Byte register. A 1 in the bit position enables the corresponding event. All of the enabled events of the Standard Event Status Event Register are logically ORed sets the Event Summary Bit (ESB) of the Status Byte Register.

A STATUS:PRESet command does not clear the bits in the Status Byte register. See also [Section 3.1](#) in this document.

Parameters	Name	Type	Range	Default
	<value>	NR1	0 – 255 (A decimal value which corresponds to the binary-weighted sum of the bits in the register. See also table in <a href="#">Section 3.1</a> ).	–
<b>Return</b>	The query reads the enable register and returns a decimal value which corresponds to the binary-weighted sum of all bits set in the register.			
<b>Usage example</b>	To enable bit 2 (decimal value = 4), bit 3 (decimal value = 8), and bit 7 (decimal value = 128), the corresponding decimal value would be 140 (4 + 8 + 128):  *ESE 140  Read value of the Standard Event Status Enable register:  *ESE? 140			
<b>Errors</b>	-200, "Execution error"			
<b>Related Commands</b>	*CLS *ESR *RST STATUS:PRESet			

#### 4.3. \*ESR?

<b>Syntax</b>	*ESR?
<b>Description</b>	Standard Event Status Register (see <a href="#">Section 3.2</a> ) Query. Reading the Standard Event Status Event register clears it.
<b>Return</b>	The BB3 returns a decimal value which corresponds to the binary-weighted sum of all bits in the register.
<b>Usage example</b>	If ERRor (bit 2) is set:  ESR? 4

#### 4.4. \*IDN?

<b>Syntax</b>	*IDN?
<b>Description</b>	Identification query for the UNIQUE identification of the BB3. (see also IEEE 488.2 10.14).
<b>Return</b>	The following system parameters will be displayed: <vendor>, <model>, <serial number>, <firmware>. The <model> include information about the CPU in use in brackets and could be <i>STM32F7</i> or <i>Simulator</i> . More information about the simulator can be found in <a href="#">Section 9</a> .
<b>Usage example</b>	*IDN?  Envox,EEZ BB3 (STM32F7),00001,v0.1

#### 4.5. \*OPC

<b>Syntax</b>	*OPC *OPC?
<b>Description</b>	Operation Complete Command. The command is mainly used for program synchronization. It causes the BB3 to set the OPC bit (bit 0) of the Standard Event Status register when the BB3 has completed all pending operations *OPC. Pending operations

are complete when:

- All commands sent before \*OPC is received, including paralleled commands, have been completed. Most commands are sequential and are completed before the next command is executed. Commands that affect output voltage, current, or state, relays, and trigger actions are executed in parallel with subsequent commands. \*OPC provides notification that all parallel commands have completed.
- All triggered actions are completed.

Query whether the current operation is completed and the query returns 1.  
See also IEEE 488.2 Section 12.5 – 12.8.

**Return** Query causes the BB3 to place a 1 in the output buffer when all pending operations are completed. \*OPC? does not suspend processing of commands.

**Usage example** \*OPC?  
1

if current operation is not completed:

\*OPC?  
0

#### 4.6. \*RCL

**Syntax** \*RCL {<profile>}

**Description** This command recalls the BB3 state stored in the specified storage location. The BB3 has ten storage locations in non-volatile memory to store BB3 states. It is not possible to recall the BB3 state from a storage location that is empty or was deleted (Error 400 will be generated). When the firmware is started for the first time, storage locations 1 through 9 are empty (location 0 has the power-on state).

*The BB3 uses location 0 to automatically save the state of the BB3 at power down.*

Parameters	Name	Type	Range	Default
	<profile>	NR1	0 – 9	–
<b>Return</b>	None			
<b>Usage example</b>	*RCL 2			
<b>Errors</b>	400, "Cannot load empty profile"			
<b>Related Commands</b>	*SAV MEMory:STATe:DELeTe MEMory:STATe:RECall:AUTO MEMory:STATe:RECall:SELEct SYSTem:POWeR			

#### 4.7. \*RST

**Syntax** \*RST

**Description** Reset Command. Restores the BB3 to its initial state (as predefined in the BB3 firmware, see [Section 8.2](#)) and clears the error queue. The reset command does NOT affect calibration data, nor any of saved configuration profiles (0 to 9). When \*RST is issued, all outputs are set to OFF, and voltage and current are programmed to 0. The power up sequence is started. All SPI peripherals are reinitialize except the controller if an active Ethernet connection exists.

**Return** None

**Usage example** \*RST  
MEMory:RECall:AUTO

SYSTem:POWer

**Related Commands** \*RST  
\*SAV  
MEMory:STAtE:CATalog?

#### 4.8. \*SAV

**Syntax** \*SAV {<profile>}

**Description** This command stores the current instrument state in the specified storage location. Any state previously stored in the same location is overwritten without generating any errors. The BB3 has nine storage locations in non-volatile memory which are available to the user for storing BB3 states. The following channel and system parameters are stored in the non-volatile memory:

- Calibration status ([CALibration:STAtE](#))
- Output enable state ([OUTPut\[:STAtE\]](#))
- Output track state ([OUTPut:TRACk\[:STAtE\]](#))
- Channel coupling state ([INSTrument:COUPle:TRACking](#))
- Remote sense state ([\[SOURce\[<n>\]\]:VOLTage:SENSe\[:SOURce\]](#))
- Output voltage ([\[SOURce\[<n>\]\]:VOLTage](#))
- Output voltage limit ([\[SOURce\[<n>\]\]:VOLTage:LIMit](#))
- Output voltage step ([\[SOURce\[<n>\]\]:VOLTage:STEP](#))
- OVP status ([\[SOURce\[<n>\]\]:VOLTage:PROTection:STAtE](#))
- OVP delay ([\[SOURce\[<n>\]\]:VOLTage:PROTection:DELay](#))
- Output current ([\[SOURce\[<n>\]\]:CURRent](#))
- Output current limit ([\[SOURce\[<n>\]\]:CURRent:LIMit](#))
- Output current step ([\[SOURce\[<n>\]\]:CURRent:STEP](#))
- OCP status ([\[SOURce\[<n>\]\]:CURRent:PROTection:STAtE](#))
- OCP delay ([\[SOURce\[<n>\]\]:CURRent:PROTection:DELay](#))
- Output power limit ([\[SOURce\[<n>\]\]:POWer:LIMit](#))
- OPP level ([\[SOURce\[<n>\]\]:POWer:PROTection\[:LEVel\]](#))
- OPP status ([\[SOURce\[<n>\]\]:POWer:PROTection:STAtE](#))
- OPP delay ([\[SOURce\[<n>\]\]:POWer:PROTection:DELay](#))
- OTP level ([SYSTem:TEMPerature:PROTection\[:HIGH\]\[:LEVel\]](#))
- OTP status ([SYSTem:TEMPerature:PROTection\[:HIGH\]:STAtE](#))
- OTP delay ([SYSTem:TEMPerature:PROTection\[:HIGH\]:DELay](#))
- Power on state ([SYSTem:POWer](#))
- Simulator load value ([SIMUlator:LOAD](#))
- Simulator load connection ([SIMUlator:LOAD:STAtE](#))

Users can assign an arbitrary name to each of locations 1 through 9 using the [MEMory:STAtE:NAME](#) command.

A reset ([\\*RST](#) command) does not affect the configurations stored in memory. Once a state is stored, it remains constant until it is overwritten using this command or specifically deleted using the [MEMory:STAtE:DELeTe](#) command.

*The BB3 uses location 0 to automatically hold the state of the BB3 at power down.*

Parameters	Name	Type	Range	Default
	<profile>	NR1	1 – 9	–
<b>Return</b>	None			
<b>Usage example</b>	*SAV 2			
<b>Related Commands</b>	*RCL *RST			

MEMory:STATe:CATalog?  
 MEMory:STATe:NAME  
 MEMory:STATe:DELeTe

#### 4.9. \*SRE

**Syntax** \*SRE {<value>}  
 \*SRE?

**Description** Enable bits in the Status Byte enable register (see [Section 3.2](#)).

Parameters	Name	Type	Range	Default
	<value>	NR1	0 – 255 (A decimal value which corresponds to the binary-weighted sum of the bits in the register. See also table in <a href="#">Section 3.1</a> ).	–

**Return** Query the Status Byte enable register. The BB3 returns a decimal value which corresponds to the binary-weighted sum of all bits set in the enable register.

**Usage example** \*SRE 32

**Related Commands** \*STB

#### 4.10. \*STB?

**Syntax** \*STB?

**Description** Read Status Byte Query. This query reads the Status Byte register (see [Section 3.2](#)), which contains the status summary bits and the Output Queue MAV bit. The Status Byte is a read-only register and its bits are not cleared when it is read.

A serial poll also returns the value of the Status Byte register, except that bit 6 returns Request for Service (RQS) instead of Master Status Summary (MSS). A serial poll clears RQS, but not MSS. When the MSS bit is set, the BB3 has one or more reasons for requesting service.

**Return** The BB3 returns a decimal value which corresponds to the binary-weighted sum of all bits in the register.

**Usage example** If OPER (bit 7) is set:  
 \*STB?  
 128

**Related Commands** \*SRE

#### 4.11. \*TRG

**Syntax** \*TRG

**Description** This command generates a trigger to the trigger subsystem which has selected a bus (software) trigger as its source (TRIGger[:SEquence]:SOURce BUS).

**Return** None

**Usage example** Generate a trigger operation after 5 seconds:

```
TRIG:SOUR BUS
TRIG:DEL 5
INIT
*TRG
```

**Errors** -211, "Trigger ignored"

```
304,"Incompatible transient modes"
307,"List lengths are not equivalent"
```

**Related Commands**

```
*WAI
ABORt
INITiate
TRIGger[:SEQuence]:DELay
TRIGger[:SEQuence]:SOURce
```

#### 4.12. \*TST?

**Syntax** [\\*TST?](#)

**Description** Self-Test Query. The self-test query causes an internal self-test, and places a response into the Output Queue indicating whether or not the BB3 completed the self-test without detected errors.

*Note: All all terminal connections must be removed while the internal self-test is being performed.*

If an active Ethernet connection exists, testing of the Ethernet controller will be skipped. You can use [DIAGnostic\[:INFormation\]:TEST?](#) for to produce a detailed report of the latest self-test.

If a fan is installed, and not running, this command will start it for the short time to obtain speed information.

**Return** 0 or 1 depends of the self-test results. See also [DIAGnostic\[:INFormation\]:TEST?](#).

**Usage example** If all tests passed:

```
*TST?
0
```

If one or more tests failed:

```
*TST?
1
```

**Related Commands**

```
DIAGnostic[:INFormation]:TEST?
SYSTem:BEEP:STATE
```

#### 4.13. \*WAI

**Syntax** [\\*WAI](#)

**Description** *Not implemented yet*

The Wait-to-Continue Command causes the BB3 to wait until all pending commands are completed before executing any other commands.

Pending operations are as defined under the \*OPC command.

**Return** None

**Usage example** For example, the \*WAI command can be used to make a voltage measurement after an OUTPut ON command has completed:

```
OUTPut ON;*WAI;:MEASure:VOLTagE?
```

**Related Commands**

```
*OPC
```

## 5. Subsystem command reference

This section summarizes the Standard Commands for Programmable Instruments (SCPI) available to program the BB3 remotely.

- [ABORt](#)
- [CALibration](#)
- [DIAGnostic](#)
- [DISPlay](#)
- [FEtCh](#) (*not implemented yet*)
- [HCOPY](#) (*not implemented yet*)
- [INITiate](#)
- [INSTrument](#)
- [MEASure](#)
- [MEMory](#)
- [MMEMemory](#)
- [OUTPut](#)
- [ROUTe](#)
- [SENSe](#)
- [SOURce](#)
- [STATus](#)
- [SYSTem](#)
- [TRIGger](#)



## 5.1. ABORT

Abort commands cancel any triggered actions.

SCPI command	Description
<a href="#">ABORt</a>	Resets the trigger system to the Idle state
<a href="#">:DLOG</a>	Stops the internal data logging session

### 5.1.1. ABORt

**Syntax** [ABORt](#)

**Description** The ABORt command resets the trigger system and places all trigger sequences in the IDLE state. Any actions related to the trigger system that are in progress will be also aborted as quickly as possible. As a result, subsequent triggers have no effect on the input level.

*ABORt is also executed at power-on and upon execution of the \*RST command.*

**Usage example**

ABOR

**Related Commands** \*RST  
INITiate  
[SOURce[<n>]]:CURRent:TRIGgered  
[SOURce[<n>]]:VOLTage:TRIGgered  
[SOURce[<n>]]:LIST:COUNt

### 5.1.2. ABORt:DLOG

**Syntax** [ABORt:DLOG](#)

**Description** This command stops the internal data logging session.

*ABORt:DLOG is also executed at power-on and upon execution of the \*RST command.*

**Usage example**

ABOR:DLOG

**Related Commands** \*RST

## 5.2. CALibration

This subsystem provides commands for the module output calibration. Only one channel can be calibrated at a time. If calibration mode has not been enabled with CALibration:STATe, the calibration commands will generate an error. Use CALibration:SAVE to save any changes, otherwise all changes will be lost on exit from calibration mode. Within the same calibration session both output voltage and current can be calibrated for the currently selected channel.

Calibration cannot start if channel output is not enabled (OUTPut[:STATe] ON).

During calibration process up to 20 measurement points can be entered within range for selected module and value (e.g. 0 to 40 V for DCP405 power module). The minimum recommended number of measurement points is two to compensate slope and offset errors.

SCPI Command	Description
<a href="#">CALibration[:MODE] {&lt;bool&gt;}, {&lt;password&gt;}</a>	Enables/disables calibration mode
<a href="#">:CLEar {&lt;password&gt;}</a>	Clears all calibration parameters
<a href="#">:CURRent</a>	
<a href="#">[:DATA] {&lt;value&gt;}</a>	Enters the calibration value for set point
<a href="#">:LEVel {&lt;point&gt;}, {&lt;level&gt;}</a>	Calibrates the output current programming
<a href="#">:RANGe {range}</a>	Sets current range for multiple current range module
<a href="#">:PASSword</a>	
<a href="#">:NEW {&lt;old&gt;}, {&lt;new&gt;}</a>	Changes calibration password
<a href="#">:REMark {&lt;string&gt;}</a>	Saves calibration information
<a href="#">:SAVE</a>	Saves the new cal constants in non-volatile memory
<a href="#">:SCReen:INIT</a>	Initiates touchscreen calibration procedure
<a href="#">:STATe {&lt;bool&gt;}, {&lt;password&gt;}</a>	Enables calibration parameters
<a href="#">:VOLTage</a>	
<a href="#">[:DATA] {&lt;value&gt;}</a>	Enters the calibration value for set point
<a href="#">:LEVel {&lt;point&gt;}, {&lt;level&gt;}</a>	Calibrates the output voltage programming

### 5.2.1. CALibration[:MODE]

**Syntax**      [CALibration\[:MODE\] {<bool>}, {<password>}](#)  
[CALibration\[:MODE\]?](#)

**Modules**     **DCP** **DCM** **SMX**

**Description** This command enables or disables calibration mode. Calibration mode must be enabled for the channel to accept any calibration commands. The first parameter specifies the ON (1) or OFF (0) state. The second parameter is the password. Successful execution of this command set both output VOLTage and CURRent of the selected channel to the MINimum value (see [Section 8.1](#)). Execution of this command also affects bit 0 (CALibrating) of the Operation Instrument Isummary register (see [Section 3.3.2](#)).

*If both voltage and current calibration parameters exists on calibration mode exit (CALibration[:MODE] OFF) the CALibration:STATe ON command will automatically follows.*

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	OFF
	<password>	Quoted string	4 to 16 characters	"eezbb3"

**Return**      The returned parameter is 0 (OFF) or 1 (ON).

**Usage example**      See [Section 10.2](#)

**Errors**        102, "Invalid cal password"

104,"Bad sequence of calibration commands"  
 312,"Cannot execute when the channels are coupled"

**Related Commands** CALibration:STAt  
 DIAGnostic[:INfOrmatIOn]:OTIme?  
 INSTrument:COUPle:TRACking

### 5.2.2. CALibration:CLEar

**Syntax** CALibration:CLEar {<password>}

**Modules** DCP DCM SMX

**Description** Clear all calibration parameters stored in the module non-volatile memory for the currently selected channel. After successful execution of this command CALibration:STAt will be set to OFF (0) and further usage of the calibration data will be disabled. This command will be also filled calibration remark with the date and note that calibration data has been cleared.

Parameters	Name	Type	Range	Default
	<password>	Quoted string	4 to 16 characters	"eezbb3"

**Return** None

**Usage example**

```

DIAG:CAL?
"remark=2020-04-28 new cal",
"u_cal_params_exists=1",
"u_point1_dac=0.150000",
"u_point1_data=0.145000",
"u_point1_adc=0.178900",
"u_point2_dac=38.000000",
"u_point2_data=39.292000",
"u_point2_adc=38.032799",
"i_5A_cal_params_exists=1",
"i_5A_point1_dac=0.050000",
"i_5A_point1_data=0.060100",
"i_5A_point1_adc=0.059840",
"i_5A_point2_dac=4.800000",
"i_5A_point2_data=5.072900",
"i_5A_point2_adc=4.810040",
"i_50mA_cal_params_exists=1",
"i_50mA_point1_dac=0.000500",
"i_50mA_point1_data=0.000591",
"i_50mA_point1_adc=0.000600",
"i_50mA_point2_dac=0.048000",
"i_50mA_point2_data=0.049897",
"i_50mA_point2_adc=0.048100"

CAL:STAT?
1

CAL:CLE "eezbb3"
CAL:STAT?
0

DIAG:CAL?
"remark= Not calibrated",
"u_cal_params_exists=0",
"i_cal_params_exists=0"
```

**Errors** 102,"Invalid cal password"

**Related Commands** CALibration:STAt  
 DIAGnostic[:INfOrmatIOn]:CALibration?

### 5.2.3. CALibration:CURRent[:DATA]

**Syntax** CALibration:CURRent[:DATA] {<value>}

**Modules** **DCP** **DCM**

**Description** This command can only be used when calibration is enabled and the output state of the currently selected channel is ON. It enters a current value that is obtained by reading an external meter. The calibration level (CALibration:CURRent:LEVel {<point>, <level>}) has to be selected first for the value being entered. The BB3 then computes new current calibration constants. These constants has to be stored in non-volatile memory with CALibration:SAVE command.

Parameters	Name	Type	Range	Default
	<value>	NR2	-0.5 A to MAX +0.5 A The maximum value is dependent on the power module current rating. See <a href="#">Section 8.1</a>	–

**Return** None

**Usage example** See [Section 10.2](#)

**Errors** 104, "Bad sequence of calibration commands"  
107, "Cal value out of range"

**Related Commands** CALibration:CURRent:LEVel

### 5.2.4. CALibration:CURRent:LEVel {<point>}, {<level>}

**Syntax** CALibration:CURRent:LEVel {<point>}, {<level>}

**Modules** **DCP** **DCM**

**Description** This command can only be used when calibration is enabled and the output state of the currently selected channel is ON. It sets the power module to a calibration point that is entered with the CAL:CURR[:DATA] command.

Parameters	Name	Type	Range	Default
	<point>	NR1	1 – 20	–
	<level>	NR2	0 to MAX (see also <a href="#">Section 8.1</a> )	–

**Return** None

**Usage example** See [Section 10.2](#)

**Errors** 101, "Calibration state is off"  
104, "Bad sequence of calibration commands"

**Related Commands** CALibration:CURRent[:DATA]  
CALibration:STAtE  
INSTrument:NSElect  
INSTrument[:SElect]  
[SOURce[<n>]]:CURRent

### 5.2.5. CALibration:CURRent:RANGe

**Syntax** CALibration:CURRent:RANGe {<range>}

**Modules** **DCP**

**Description** When BB3 is equipped with power modules that has multiple current range (e.g. DCP405 that can be find out with the SYSTem:CHANnel:MODEl? command) it's recommended to perform calibration of all ranges. Use this command to select current range

on which calibration will be accomplished.

Parameters	Name	Type	Range	Default
	<range>	Discrete NR2	LOW HIGH 0.05 5	HIGH 5
<b>Return</b>	None			
<b>Usage example</b>	See <a href="#">Section 10.2</a>			
<b>Errors</b>	101, "Calibration state is off" -241, "Hardware missing"			
<b>Related Commands</b>	CALibration:CURRent:LEVel CALibration:STATe SYSTem:CHANnel:MODEl?			

### 5.2.6. CALibration:PASSword:NEW

**Syntax** CALibration:PASSword:NEW {<old>}, {<new>}

**Modules** DCP DCM SMX

**Description** Enter a new calibration password. To change the password, first unsecure the BB3 using the old password. Then, the new code has to be entered. The calibration code may contain up to 16 characters over the remote interface. Minimum length is 4 characters. The new password is automatically stored in non-volatile memory and does not have to be stored with CALibration:SAVE.

Parameters	Name	Type	Range	Default
	<old>	Quoted string	4 to 16 characters	eezbb3
	<new>	Quoted string	4 to 16 characters	–
<b>Return</b>	None			
<b>Usage example</b>	CAL:PASS:NEW "eezbb3", "mycal1234"			
<b>Errors</b>	102, "Invalid cal password" 105, "Cal password too long" 106, "Cal password too short"			

### 5.2.7. CALibration:REMark

**Syntax** CALibration:REMark {<user remark>}  
CALibration:REMark?

**Modules** DCP DCM SMX

**Description** Record calibration information about the power module. The BB3 should be in calibration mode before sending a calibration message.

Parameters	Name	Type	Range	Default
	<user remark>	Quoted string	0 to 32 characters	"Calibration passed"
<b>Return</b>	Query the calibration message.			
<b>Usage example</b>	CAL:REM "Calibrated by EEZ" CAL:REM? "Calibrated by EEZ"  See also <a href="#">Section 10.2</a>			
<b>Errors</b>	The following errors could be generated by command but not query:  101, "Calibration state is off" 104, "Bad sequence of calibration commands"			

**Related Commands** CALibration:STATe

### 5.2.8. CALibration:SAVE

**Syntax** CALibration:SAVE

**Modules** DCP DCM SMX

**Description** This command saves calibration constants in non-volatile memory after the calibration procedure has been completed. If calibration mode is exited by programming CALibration:STATe OFF without first saving the new constants, the previous constants are re-stored. Execution of this command also affects bit 0 (CALibrating) of the Operation Instrument Isummary register (see [Section 3.3.2](#)).

**Return** None

**Usage example** See [Section 10.2](#)

**Errors**

- 340, "Calibration failed"
- 101, "Calibration state is off"
- 104, "Bad sequence of calibration commands"
- 111, "No new cal data exists"

### 5.2.9. CALibration:SCReen:INIT

**Syntax** CALibration:SCReen:INIT

**Description** Use this command to initiate touchscreen calibration procedure when calibration data are lost or corrupted. Calibration has to be performed locally on the BB3.

*New calibration procedure can be initiated also by touching the screen and hold for more then 15 seconds.*

**Return** None

**Usage example** CAL:SCR:INIT

### 5.2.10. CALibration:STATe

**Syntax** CALibration:STATe {<bool>}  
CALibration:STATe?

**Modules** DCP DCM SMX

**Description** This command enables or disables usage of calibration parameters if they exists.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	OFF ON 0 1	ON

**Return** The returned parameter is 0 (OFF) or 1 (ON).

**Usage example** CAL:STAT OFF

**Errors** 110, "Cal params missing or corrupted"

**Related Commands** DIAGnostic[:INFormation]:OTIME?

### 5.2.11. CALibration:VOLTage[:DATA]

**Syntax** CALibration:VOLTage[:DATA] {<value>}

**Modules** DCP DCM SMX

**Description** This command can only be used when calibration is enabled and the output state of the

currently selected channel is ON. It enters a voltage value that you obtained by reading an external meter. The calibration level (see CALibration:VOLTage:LEVel {<point>, <level>}) for the value being entered. The BB3 then computes new voltage calibration constants. These constants has to be stored in non-volatile memory with CALibration:SAVE command.

Parameters	Name	Type	Range	Default
	<value>	NR2	-1 V to MAX +1 V The maximum value is dependent on the power module voltage rating. See <a href="#">Section 8.1</a>	–
<b>Return</b>	None			
<b>Usage example</b>	See <a href="#">Section 10.2</a>			
<b>Errors</b>	104, "Bad sequence of calibration commands" 107, "Cal value out of range"			
<b>Related Commands</b>	CALibration:SAVE CALibration:STAtE CALibration:VOLTage:LEVel INSTrument:NSElect INSTrument[:SElect]			

#### 5.2.12. CALibration:VOLTage:LEVel {<point>}, {<level>}

**Syntax** CALibration:VOLTage:LEVel {<point>}, {<level>}

**Modules** DCP DCM SMX

**Description** This command can only be used when calibration is enabled and the output state of the currently selected channel is ON. It sets the power module to a calibration point that is entered with the CAL:VOLT[:DATA] command.

Parameters	Name	Type	Range	Default
	<point>	NR1	1 – 20	–
	<level>	NR2	0 to MAX (see also <a href="#">Section 8.1</a> )	–
<b>Return</b>	None			
<b>Usage example</b>	See Section <a href="#">Section 10.2</a>			
<b>Errors</b>	101, "Calibration state is off" 104, "Bad sequence of calibration commands"			
<b>Related Commands</b>	CALibration:STAtE CALibration:VOLTage[:DATA] INSTrument:NSElect INSTrument[:SElect] [SOURce[<n>]]:VOLTage			

### 5.3. DIAGnostic

The purpose of the DIAGnostic subsystem is to provide a tree node for all of the BB3 service and diagnostic routines used in routine maintenance and repair.

SCPI command	Description
DIAGnostic	
[:INFOrmation]	
:ADC?	Returns the latest values acquired by ADC
:CALibration?	Returns a list of the calibration parameters
:PROTection?	Returns the information about all protections.
:TEST?	Returns results of the most recent self-test

#### 5.3.1. DIAGnostic[:INFOrmation]:ADC?

**Syntax**      `DIAGnostic[:INFOrmation]:ADC? [<channel>]`

**Description** This query returns the latest values acquired by ADC (Analog-to-Digital Converter) of the currently selected channel.

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4 CH5 CH6	n/a

**Return** Return a list of quoted strings. The U\_SET and I\_SET are values measure on DAC outputs, and U\_MON and I\_SET on the channel output terminals.

**Usage example**      `DIAG:ADC? CH2`

"U\_SET=10.1202", "U\_MON=10.12", "I\_SET=3.00", "I\_MON=1.23"

**Related Commands**      `MEASure[:SCALar]:CURRent[:DC]`  
`MEASure[:SCALar][:VOLTage][:DC]`  
`[SOURce<n>]:CURRent[:LEVel][:IMMediate][:AMPLitude]`  
`[SOURce<n>]:VOLTage[:LEVel][:IMMediate][:AMPLitude]`

#### 5.3.2. DIAGnostic[:INFOrmation]:CALibration?

**Syntax**      `DIAGnostic[:INFOrmation]:CALibration? [<channel>]`

**Description** This query returns a list of calibration parameters for the currently selected channel. If the selected channel is in the calibration mode (CALibration[:MODE] ON) then all calibration information collected to the current calibration step will be returned. Otherwise the calibration data stored in non-volatile memory will be returned.

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4 CH5 CH6	n/a

**Return** The information will be returned as a list of quoted strings.

**Usage example**      Calibration parameters for BB3 when the channel 2 is not in the calibration mode:

`DIAG:INFO:CAL? CH2`

"remark=20200123 cert", "u\_cal\_params\_exists=1",  
"u\_min\_level=0.15V", "u\_min\_data=0.08V", "u\_min\_adc=0.114V",  
"u\_mid\_level=20V", "u\_mid\_data=20.597V", "u\_mid\_adc=19.9685V",  
"u\_max\_level=38V", "u\_max\_data=39.192V", "u\_max\_adc=37.973V",  
"u\_min\_range=0V", "u\_max\_range=40V", "i\_5A\_cal\_params\_exists=1",  
"i\_5A\_min\_level=0.05A", "i\_5A\_min\_data=0.0405A",  
"i\_5A\_min\_adc=0.03695A", "i\_5A\_mid\_level=2.425A",  
"i\_5A\_mid\_data=2.4757A",



```
"i_5A_mid_adc=2.4125A", "i_5A_max_level=4.8A",
"i_5A_max_data=4.9122A", "i_5A_max_adc=4.78855A",
"i_5A_min_range=0A", "i_5A_max_range=5A",
"i_50mA_cal_params_exists=1", "i_50mA_min_level=0.0005A",
"i_50mA_min_data=0.00041A", "i_50mA_min_adc=0.000371A",
"i_50mA_mid_level=0.02425A", "i_50mA_mid_data=0.02545A",
"i_50mA_mid_adc=0.0240945A", "i_50mA_max_level=0.048A",
"i_50mA_max_data=0.05052A", "i_50mA_max_adc=0.047887A",
"i_50mA_min_range=0A", "i_50mA_max_range=0A"
```

The query results when a channel is just entered the calibration mode:

```
DIAG:INFO:CAL?
```

```
"u_level=none", "i_level=none"
```

The query results when a channel is at the step MIDdle of the voltage calibration:

```
DIAG:INFO:CAL?
```

```
"u_min=0.11V", "u_level=mid", "u_level_value=24.05V",
"u_adc=24.14V", "i_level=none"
```

**Related Commands** CALibration:REMark  
CALibration:SAVE

### 5.3.3. DIAGnostic[:INFormation]:PROTection?

**Syntax** [DIAGnostic\[:INFormation\]:PROTection?](#)

**Description** This query returns information about all supported output protection mechanisms.

**Return** The information will be returned as a list of quoted strings.

**Usage example** DIAG:PROT?

```
"CH1 u_tripped=0", "CH1 u_state=0", "CH1 u_type=1", "CH1
u_delay=5 ms", "CH1 u_level=38V", "CH1 i_tripped=0", "CH1
i_state=0", "CH1 i_delay=20 ms", "CH1 p_tripped=0", "CH1
p_state=0", "CH1 p_delay=10 s", "CH1 p_level=155W", "CH2
u_tripped=0", "CH2 u_state=0", "CH2 u_type=0", "CH2 u_delay=5
ms", "CH2 u_level=40V", "CH2 i_tripped=0", "CH2 i_state=0", "CH2
i_delay=20 ms", "CH2 p_tripped=0", "CH2 p_state=1", "CH2
p_delay=10 s", "CH2 p_level=155W"
```

**Related Commands** [SOURce[<n>]]:CURRent:PROTection:DELAy[:TIME]  
[SOURce[<n>]]:CURRent:PROTection:STATE  
[SOURce[<n>]]:CURRent:PROTection:TRIPped?  
[SOURce[<n>]]:POWEr:PROTection  
[SOURce[<n>]]:POWEr:PROTection:DELAy[:TIME]  
[SOURce[<n>]]:POWEr:PROTection:STATE  
[SOURce[<n>]]:POWEr:PROTection:TRIPped?  
[SOURce[<n>]]:VOLTage:PROTection:DELAy[:TIME]  
[SOURce[<n>]]:VOLTage:PROTection:STATE  
[SOURce[<n>]]:VOLTage:PROTection:TRIPped?  
SYSTem:TEMPerature:PROTection[:HIGH][:LEVel]  
SYSTem:TEMPerature:PROTection[:HIGH]:DELAy[:TIME]  
SYSTem:TEMPerature:PROTection[:HIGH]:STATE  
SYSTem:TEMPerature:PROTection[:HIGH]:TRIPped?

### 5.3.4. DIAGnostic[:INFormation]:TEST?

**Syntax** [DIAGnostic\[:INFormation\]:TEST? \[<resource>\]](#)

**Description** This query returns results of the most recent self-test (see [\\*TST?](#) Command) for all or specified resource.

Parameters	Name	Type	Range	Default
------------	------	------	-------	---------

<code>&lt;resource&gt;</code>	Discrete	EEProm SDCard Ethernet RTC  DATEtime FAN AUXTemp  CH1Temp CH2Temp CH3Temp  CH4Temp CH5Temp CH6Temp  CH1 CH2 CH3 CH4 CH5 CH6  SLOT1 SLOT2 SLOT3	–
-------------------------------	----------	---	---

**Return** The information will be returned in the following format: "<return code, device name, installed, return message>" where the return code could be one of the following values:

- 0 – not installed or recognized
- 1 – failed
- 2 – passed
- 3 – testing
- 4 – skipped
- 5 – warning

This information format will repeat with as many iterations as the number of devices found in configuration parameters of the BB3.

**Usage example** Return self-test results when SD card is not found:

```
DIAG:TEST?
```

```
"2, EEPROM, installed, passed", "2, SD card, installed, passed",
"2, Ethernet, installed, passed", "2, RTC, installed, passed",
"2, DateTime, installed, passed", "2, Fan, installed, passed",
"2, AUX temp, installed, passed", "2, CH1 temp, installed,
passed", "2, CH2 temp, installed, passed", "2, CH3 temp,
installed, passed", "2, CH4 temp, installed, passed", "2, SLOT1,
installed, passed", "2, SLOT2, installed, passed",
"2, SLOT3, installed, passed"
```

Check status of specified device, e.g. channel 2:

```
DIAG:TEST? CH2
```

```
2
```

**Related Commands** \*TST?  
SYSTem:CHANnel:OPTion?

## 5.4. DISPlay

The DISPlay commands are used to set the display mode, turn on or off the front panel TFT display, select main page appearance, display and clear the text sent using a controller application. DISPlay is independent of, and does not modify, how data is returned to the controller application.

SCPI command	Description
DISPlay	
<a href="#">:BRIGhtness {&lt;value&gt;}</a>	Sets the intensity of the front panel TFT display
<a href="#">:DATA?</a>	Reads screen image data
<a href="#">:VIEW {&lt;mode&gt;}</a>	Sets the front panel TFT display main page appearance
[[:WINDow]	
<a href="#">[:STATe] {&lt;bool&gt;}</a>	Sets the front panel TFT display state
:DIALog	
<a href="#">[:OPEN] {&lt;filename&gt;}</a>	Displays content of the resource file
<a href="#">:ACTIon? [ &lt;timeout&gt;]</a>	Returns information about selected item
<a href="#">:CLOSE</a>	Closes last opened dialog window
<a href="#">:DATA {&lt;name&gt;}, {&lt;type&gt;}, [{&lt;unit&gt;}], {&lt;value&gt;}</a>	Sets the value of the dialog data item
<a href="#">:DLOG [ &lt;filename&gt;]</a>	Opens DLOG viewer
<a href="#">:ERRor {&lt;message&gt;}</a>	Displays message box and generates error beep tone
<a href="#">:INPUt? {&lt;label&gt;}, {&lt;type&gt;} [, &lt;min&gt;, &lt;max&gt;, &lt;value&gt;]</a>	Displays entry form and waits for input on the front panel TFT display
<a href="#">:SElect? {&lt;defaultSelection&gt;}, {&lt;optionText&gt;}[,...]</a>	Displays pop-up display with multiple radio buttons
<a href="#">:TEXT {&lt;message&gt;}</a>	Displays a message on the front panel TFT display
<a href="#">:CLEar</a>	Clear a message on the front panel TFT display

### 5.4.1. DISPlay:BRIGhtness

**Syntax**      [DISPlay:BRIGhtness {<value>}](#)  
[DISPlay:BRIGhtness?](#)

**Description** Controls the intensity of the front panel TFT display. The range of the parameter is 1 to 20, where 20 is full intensity and 1 is fully blanked.

Parameters	Name	Type	Range	Default
	<value>	NR1	1 – 20	20

**Return** This query returns set front panel's TFT display brightness value.

**Usage example**      `DISP:BRIG?`  
20

**Related Commands**      \*RST

### 5.4.2. DISPlay:DATA?

**Syntax**      [DISPlay:DATA?](#)

**Description** This query reads screen image data. The image is formatted as a .png file. Use the `HCOPY[:IMMediate]` or `HCOPY:SDUMp[:IMMediate]` command to capture screen image and save it as a file on the SD card.

**Return** Screen image data is returned in the IEEE-488.2 # data block format (see [Section 2.10](#)).

**Usage example**     `DISP:DATA?`  
                       `#<length-digits><length><block>`

**Related Commands**   `HCOPY[:IMMediate]`  
                           `HCOPY:SDUMp[:IMMediate]`  
                           `MMEMory:FEED`

### 5.4.3. DISPlay:VIEW

**Syntax**            `DISPlay:VIEW {<mode>}`  
                       `DISPlay:VIEW?`

**Description**     Use this command to set front panel TFT display main page appearance. The following modes are available:

- 1 – Numerical
- 2 – Bar graph horizontal
- 3 – Bar graph vertical
- 4 – YT view (scroll)
- 5 – YT view (scan line)

Parameters	Name	Type	Range	Default
	<mode>	NR1 Discrete	1 – 5 DEFault	DEFault
<b>Return</b>	This query returns set front panel's TFT display main page appearance numeric value (NR1).			
<b>Usage example</b>	<code>DISP:VIEW 2</code>			
<b>Related Commands</b>	<code>*RST</code>			

### 5.4.4. DISPlay[:WINdow][:STATe]

**Syntax**            `DISPlay[:WINdow][:STATe] {<bool>}`  
                       `DISPlay[:WINdow][:STATe]?`

**Description**     Turn the front panel TFT display off or on. When the display is turned off, outputs are not sent to the display and all indicators are disabled except the Event view indicator. The display state is automatically turned on when you return to the local mode. Press and hold the display for about a second to return to the LOCAL from the REMote control.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	–
<b>Return</b>	<code>DISPlay?</code> query the front panel TFT display state. Returns 0 (OFF) or 1 (ON).			
<b>Usage example</b>	<code>DISP ON</code>			
<b>Related Commands</b>	<code>SYSTem:LOCal</code> <code>SYSTem:REMote</code>			

### 5.4.5. DISPlay[:WINdow]:DIALog[:OPEN]

**Syntax**            `DISPlay[:WINdow]:DIALog[:OPEN] {<filename>}`

**Description**     Use this command to draw new page defined with resource file created in EEZ Studio and contains page widgets, actions and data items. The resource file has to be located on the SD card (see MMEMory subsystem) and the full path to the file must be specified. The new displayed page can be closed (removed) by using the `DISPlay[:WINdow]:DIALog:CLOSe` to close the page.

Parameters	Name	Type	Range	Default
	<filename>	Quoted string	–	–
<b>Usage example</b>	DISP:DIAL "/Scripts/Diode Tester.res"			
<b>Related Commands</b>	DISPlay[:WINDow]:DIALog:CLOSe DISPlay[:WINDow]:DIALog:DATA			

#### 5.4.6. DISPlay[:WINDow]:DIALog:ACTIon

<b>Syntax</b>	DISPlay[:WINDow]:DIALog:ACTIon? [<timeout>]			
<b>Description</b>	This command waits for timeout (in seconds) to get information about what action item is selected on the currently displayed page as defined in resource file activated with the DISPlay[:WINDow]:DIALog[:OPEN] command. The default timeout value is 0 and command will wait until user select one of the action items.			
Parameters	Name	Type	Range	Default
	<timeout>	NR1	–	0
<b>Return</b>	Returns name of the latest action item selected by user. 0 will be returned if currently opened page does not contain any action dialog. 1 is returned when timeout parameter is used and set time expired without user action.			
<b>Usage example</b>	DISP:DIAL:ACTI? "input_diode_name"			
<b>Related Commands</b>	DISPlay[:WINDow]:DIALog[:OPEN]			

#### 5.4.7. DISPlay[:WINDow]:DIALog:CLOSe

<b>Syntax</b>	DISPlay[:WINDow]:DIALog:CLOSe			
<b>Description</b>	Use this command to close last dialog window opened with the DISPlay[:WINDow]:DIALog[:OPEN] command.			
<b>Usage example</b>	DISP:DIAL:CLOS			
<b>Related Commands</b>	DISPlay[:WINDow]:DIALog[:OPEN]			

#### 5.4.8. DISPlay[:WINDow]:DIALog:DATA

<b>Syntax</b>	DISPlay[:WINDow]:DIALog:DATA {<name>}, {<type>}, [{<unit>}], {<value>}			
<b>Description</b>	Use this command to set the value of the dialog data item defined as <name>. The list of possible data items is defined within the dialog resource file created in the EEZ Studio and specified with the DISPlay[:WINDow]:DIALog:OPEN command.  If <type> is FLOat then <unit> has to be specified, too.			
Parameters	Name	Type	Range	Default
	<name>	Quoted string		–
	<type>	NR1 NR3 Quoted string	INTeger FLOat STRing	–
	<unit>	Discrete	UNKNown VOLT  AMPEr  WATT JOULe SECOnd OHM  FARAd HERTZ	–
<b>Usage example</b>	DISP:DIAL:DATA "test", INT			

**Related Commands**    DISPlay[:WINDow]:DIALog:OPEN

#### 5.4.9. DISPlay[:WINDow]:DLOG

**Syntax**            DISPlay[:WINDow]:DLOG [<filename>]

**Description**    This command opens DLOG viewer and displays specified by filename. If filename is omitted it displays the last log file during or after recording is finished.

**Usage example**    DISP:DLOG "Recording/test\_log.dlog"

**Related Commands**    INITiate:DLOG  
INITiate:DLOG:TRACe

#### 5.4.10. DISPlay[:WINDow]:ERRor

**Syntax**            DISPlay[:WINDow]:ERRor {<message>}

**Description**    Use this command to display message box and generates error beep tone.

**Usage example**    DISP:ERR "Test message!"

**Related Commands**    DISPlay[:WINDow]:TEXT  
SYSTem:BEEPer[:IMMEDIATE]

#### 5.4.11. DISPlay[:WINDow]:INPut

**Syntax**            DISPlay[:WINDow]:INPut? {<label>}, {<type>} [, <min>, <max>, <value>]

**Description**    Use this command to define input field on the display that can returns a value of type <type> which can be TEXT | NUMBer | INTeger | MENU. The following variants are possible:

- DISPlay[:WINDow]:INPut? {<label>}, TEXT, <min\_length>, <max\_length>, <value>
- DISPlay[:WINDow]:INPut? {<label>}, NUMBer, <unit>, <min>, <max>, <value>
- DISPlay[:WINDow]:INPut? {<label>}, INTeger, <min>, <max>, <value>
- DISPlay[:WINDow]:INPut? {<label>}, MENU, <menu\_type>, <menu\_item>, [... <menu\_item>]

<min> and <max> values should be defined for TEXT, NUMBer and INTeger.

For NUMBer and INTeger this is the minimum and maximum allowable value, and for STRing it is the minimum and maximum allowable text length. NUMBer should also be set to <unit> (see above which units are allowed).

If <type> is MENU then you should set <menu\_type> (which can only be BUTTon) and up to max. 4 menu items (button texts) to be offered for selection.

A numeric keypad will be open for NUMBer and INTeger, and a full keypad will be open for TEXT. The default <label> (quoted string) will be displayed on the screen.

In all variants except MENU (where no need to be specified), the <value> parameter is the initial (or default) value.

Parameters	Name	Type	Range	Default
	<label>	Quoted string	–	–
	<type>	Discrete	TEXT NUMBer MENU	
<b>Return</b>	This query returns value of defined type.			
<b>Usage example</b>	DISP:INP? "Enter you text", TEXT, 2, 10, "Hello" "Test"			

**5.4.12. DISPlay[:WINDow]:SELEct?****Syntax** `DISPlay[:WINDow]:SELEct? [<defaultSelection>, {<optionText>}[, ...]`**Description** Display pop-up display with multiple radio buttons. The <defaultSelection> specify which of the presented option will be selected by default.

Parameters	Name	Type	Range	Default
	<defaultSelection>	NR1	1 to the total number of options	–
	<optionText>	Quoted string	–	–

**Return** The query returns NR1 value of the selected option.**Usage example** Default option is 2 and query returns 3 when "Option 3" is selected:

```
DISP:SEL? 2, "Option 1", "Option 2", "Option 3"
3
```

**Related Commands** `DISPlay[:WINDow]:DIALog:CLOSe`  
`DISPlay[:WINDow]:DIALog:DATA`  
`DISPlay[:WINDow]:INPut`**5.4.13. DISPlay[:WINDow]:TEXT****Syntax** `DISPlay[:WINDow]:TEXT {<text message>}`  
`DISPlay[:WINDow]:TEXT?`**Description** Display a message on the front panel TFT display. The BB3 will display up to 32 characters in a message.

Parameters	Name	Type	Range	Default
	<text message>	Quoted string	–	–

**Return** Query the message sent to the front panel TFT display and returns a quoted string.**Usage example** Send "Hello world" textual message:

```
DISP:TEXT "Hello world"
```

**Related Commands** `DISPlay[:WINDow]:TEXT:CLEAr`**5.4.14. DISPlay[:WINDow]:TEXT:CLEAr****Syntax** `DISPlay[:WINDow]:TEXT:CLEAr`**Description** Clear the message displayed on the front panel TFT display.**Return** None**Usage example** `DISP:TEXT:CLE`**Related Commands** `DISPlay[:WINDow]:TEXT`

## 5.5. FETCh

Fetch commands return measurement data that has been previously acquired. FETCh queries do not generate new measurements, but allow additional measurement calculations from the same acquired data.

SCPI command	Description
FETCh	
<a href="#">:AHOu? {&lt;channel&gt;}</a>	Returns the delivered energy in amp-hours
<a href="#">WHOu? {&lt;channel&gt;}</a>	Returns the delivered energy in watt-hours

### 5.5.1. FETCh:AHOu?

**Syntax** [FETCh:AHOu? {<channel>}](#)

**Description** *Not implemented yet*

Use this command to query delivered energy on the specified channel in amp-hours accumulated after last power-on or SENSE:AHOu:RESet command.

This value is measured independently of channel's total delivered energy in amp-hours stored in non-volatile memory that can be queried using the SYSTem:CHANnel:INFORmation:AHOu:TOTal? Command.

*If channels are coupled (in series or parallel) or in tracking mode, this command will return a sum of delivered energy on both channels.*

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4 CH5 CH6	–

**Usage example**  
FETCh:AHO? CH1  
1340.30

**Related Commands**  
FETCh:WHOu?  
INSTrument:COUPle:TRACking  
OUTPut:TRACk[:STATe]  
SENSE:AHOu:RESet  
SYSTem:CHANnel:INFORmation:AHOu:TOTal?

### 5.5.2. FETCh:WHOu?

**Syntax** [FETCh:WHOu? {<channel>}](#)

**Description** *Not implemented yet*

Use this command to query delivered energy on the specified channel in watt-hours accumulated after last power-on or SENSE:AHOu:RESet command.

This value is measured independently of channel's total delivered energy in watt-hours stored in non-volatile memory that can be queried using the SYSTem:CHANnel:INFORmation:WHOu:TOTal? Command.

*If channels are coupled (in series or parallel) or in tracking mode, this command will return a sum of delivered energy on both channels.*

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4 CH5 CH6	–

**Usage example**  
FETCh:WHO? CH1  
100.30  
OUTP 0



## EEZ BB3 SCPI reference

FETC:WHO:RES CH1  
FETC:WHO? CH1

0

**Related** FETCh:AHOu?  
**Commands** INSTrument:COUPle:TRACking  
OUTPut:TRACk[:STATe]  
SENSe:WHOu:RESet  
SYSTem:CHANnel:INFOrmation:WHOu:TOTal?

## 5.6. HCOPy

The Hardcopy commands are used to print the entire display to a specified file rather than “printing” to an external device.

SCPI command	Description
HCOPy	
<a href="#">[:IMMediate]</a>	Initiates hardcopy output
<a href="#">:DESTination {&lt;destination&gt;}</a>	Sets the hardcopy destination
<a href="#">:SDUMp[:IMMediate]</a>	Initiates hardcopy output

### 5.6.1. HCOPy:DESTination

**Syntax** [HCOPy:DESTination {<destination>}](#)

**Description** *Not implemented yet*

This command sets the hardcopy destination. The destination is always set to MMEMory (i.e. SD Card). This command is included only for compatibility with the SCPI standard. The destination file on the mass memory device is specified by the MMEMory:NAME command.

**Usage example** HCOPy:DEST MMEM

**Related Commands** MMEMory:CLOSe  
MMEMory:NAME  
MMEMory:OPEN

### 5.6.2. HCOPy[:IMMediate]

**Syntax** [HCOPy\[:IMMediate\]](#)

**Description** *Not implemented yet*

This command immediately initiates hardcopy output according to the current HCOPy setup parameters. This command is the same as HCOPy:SDUMp[:IMMediate].

**Usage example** MMEM:NAME "sample1.png"  
MMEM:OPEN  
HCOP:DEST "MMEM"  
HCOP  
MMEM:CLOS

**Related Commands** HCOPy:DESTination  
HCOPy:SDUMp[:IMMediate]

### 5.6.3. HCOPy:SDUMp[:IMMediate]

**Syntax** [HCOPy:SDUMp\[:IMMediate\]](#)

**Description** *Not implemented yet*

This command initiates a screen dump of the entire TFT display's screen, and is the same as the HCOPy[:IMMediate] command.

**Usage example** MMEM:NAME "sample2.png"  
MMEM:OPEN  
HCOP:DEST "MMEM"  
HCOP:SDUM  
MMEM:CLOS

**Related Commands** HCOPy:DESTination  
HCOPy[:IMMediate]



## 5.7. INITiate

INITiate commands initialize the trigger system. This enables the trigger system to receive triggers.

SCPI command	Description
INITiate	
[:IMMediate]	Completes one full trigger cycle
:DLOG {<filename>}	Initiates automated internal data log session
:TRACe {<filename>}	
:CONtinuous {<bool>}	Enables/disables continuous transient triggers

### 5.7.1. INITiate

**Syntax**      `INITiate[:IMMediate]`

**Description** The INITiate command is used to initialize the trigger system of the BB3. This command completes one full trigger cycle when the trigger source is an IMMediate and initiates the trigger subsystem when the trigger source is BUS.

For example, when the IMMediate is selected as a trigger source, an INITiate command immediately transfers the VOLTage:TRIGgered[:AMPLitude] and CURRent:TRIGgered[:AMPLitude] values to VOLTage[:LEVel][:IMMediate][:AMPLitude] and CURRent[:LEVel][:IMMediate][:AMPLitude] values. Any delay is ignored.

Execution of this command also affects bit 5 (Waiting for TRIGger) of the Operation Instrument Isummary register (see [Section 3.3.2](#)).

**Usage example**      Generate a trigger operation after 5 seconds:

```
TRIG:SOUR BUS
TRIG:DEL 5
INIT
*TRG
```

**Errors**      -213, "Init ignored"  
307, "List lengths are not equivalent"  
308, "Cannot be changed while transient trigger is initiated"  
309, "Cannot initiate while in fixed mode"

**Related Commands**      \*TRG  
INSTRument:COUPle:TRIGger

### 5.7.2. INITiate:DLOG

**Syntax**      `INITiate:DLOG {<filename>}`

**Description** The command starts automated internal data logging session. All selected measurements defined with SENSE:DLOG:FUNCTION:CURRent, SENSE:DLOG:FUNCTION:POWer and SENSE:DLOG:FUNCTION:VOLTage commands will be recorded periodically in the specified log filename that is formatted as described below. The data logging session will last until time specified with the SENSE:DLOG:TIME not expired or is not interrupted prematurely using the e.g. ABORT:DLOG, SYSTem:REStart or \*RST command.

#### Log file format

Data log structure allows recording of multiple values simultaneously that can be also viewed simultaneously on the same X-Y graph as in case of e.g. curve tracer when each Y-axis data set (column) represents one trace. This minimizes the required size of the data log file because the X-axis content is common for multiple values to be displayed on the Y-axis.

Data log file contains fixed header, flexible header and data section.

**Fixed header**

The fixed header includes the following information in little-endian format, i.e. the least significant byte (LSB) value is at the lowest address:

Position	Type	Description
0 – 7	ASCII	Always contains “EEZ-DLOG” text in ASCII format
8 – 9	UINT16	File format version (e.g. 0x00 0x02)
10 – 11	UINT16	Number of columns
12 – 15	UINT32	File offset to data rows

**Flexible header**

The flexible header contains meta information about data that follows. The number of fields defined in the flexible header is arbitrary (hence the name flexible header) formatted as follows:

```
<Field 1 length><Field 1 ID><Field 1 data>
<Field 2 length><Field 2 ID><Field 2 data>
<Field 3 length><Field 3 ID><Field 3 data>
...
<Field N length><Field N ID><Field N data>
```

Name	Length	Description
Field length	UINT16	Total length of the field.
Field ID	UINT8	Field unique identification number (see below the list of currently supported fields).
Field data	For example if the <Field ID> is 14 or "X-axis label", then the field is of type String. Text string begins with UINT16 that contains its length, followed by length * UINT8 data.	Field data as specified below in the list of currently supported fields.

List of currently supported fields:

ID	Name	Data type	Description
1	Comment	String	Data log user comment. Max. length is 128 characters.
10	Unit	UINT8	0 – Unknown 1 – Volt 3 – Ampere 6 – Watt 8 – Second 12 – Ohm 16 – Hertz 17 – Joule/Ws 21 – Farad
11	X-axis step	Float	X-axis resolution in units as specified in ID10. For example, if X-axis unit is 8 (seconds), when period of data logger is set to 20 ms this field will contains 0.02.

12	X-axis range min value	Float	X-axis minimal or starting value which is 0 in most of the cases.
13	X-axis range max value	Float	X-axis maximal or ending value.
14	X-axis label	String	X-axis data description
15	X-axis scale	UINT8	0 – Linear 1 – Logarithmic
30	Y-axis unit	UINT8 + UINT8	First UINT8 value specifies column number and second UINT8 contains Y-axis unit similar as X-axis unit specified under ID10.
32	Y-axis range min value	UINT8 + Float	UINT8 value specifies column number followed by minimum recorded value.
33	Y-axis range max value	UINT8 + Float	UINT8 value specifies column number followed by maximum recorded value. For example for uncoupled DCP405 module that will be 40 V and 5 A or for coupled in series or parallel 80 V or 10 A. This parameter is used by Auto-scale option in BB3 built-in log viewer.
34	Y-axis label	UINT8 + String	UINT8 value specifies column number followed by Y-axis data description. This field is optional and if its value is not defined, the data log viewer (e.g. BB3 built-in viewer) could generate it from the channel number and unit symbol, for example: U1 for recorded voltage on the channel 1.
35	Y-axis channel	UINT8 + UINT8	First UINT8 value specifies column number and second UINT8 contains instrument channel that is recorded
36	Y-axis scale	UINT8 + UINT8	Similar to X-axis scale: first UINT8 value specifies column number and second UINT8 can contains: 0 – Linear 1 – Logarithmic
50	Channel module type	UINT8 + UINT16	UINT8 value specifies channel number, and UINT16 contains module type ID as follows: 220 – DCM220 405 – DCP405
51	Channel revision	UINT8 + UINT16	UINT8 value specifies channel number, and UINT16 contains module version number, for example 0x0206 for “R2B6”.

**Logged data**

For the each X-axis step the series of Y values (columns) are recorded. All Y values are of type float.

```
<Y1 value><Y2 value>...<Yn value>
<Y1 value><Y2 value>...<Yn value>
...
```

This command will automatically set the following fields:

- *X-axis unit = 8*
- *X-axis step = SENSE:DLOG:PERiod?*
- *X-axis range min = 0*
- *X-axis range max = SENSE:DLOG:TIME?*

For example when voltage and current are logged for channel 1 and 2 on two DCP405 modules, additional fields will be populated as follows:

- *Y1-axis unit = <1><1>*
- *Y1-axis range min = 1 + SOUR1:VOLT? MIN*
- *Y1-axis range max = 1 + SOUR1:VOLT? MAX*
- *Y1-axis channel index = 1*
- *Y2-axis unit = <2><3>*
- *Y2-axis range min = 2 + SOUR1:CURREN? MIN*
- *Y2-axis range max = 2 + SOUR1:CURREN? MAX*
- *Y2-axis channel index = 1*
- *Y3-axis unit = <3><1>*
- *Y3-axis range min = 3 + SOUR2:VOLT? MIN*
- *Y3-axis range max = 3 + SOUR2:VOLT? MAX*
- *Y3-axis channel index = 2*
- *Y4-axis unit = <4><3>*
- *Y4-axis range min = 4 + SOUR2:CURREN? MIN*
- *Y4-axis range max = 4 + SOUR2:CURREN? MAX*
- *Y4-axis channel index = 2*
- *Channel 1 module type = "405"*
- *Channel 1 module revision = "207"*
- *Channel 2 module type = "405"*
- *Channel 2 module revision = "207"*

Parameters	Name	Type	Range	Default
	<filename>	Quoted string	File name, either / (slash) or \ (back-slash) can be used as the path separator. 1 to 255 characters	–
<b>Usage example</b>	Start data logging of channel 1 voltage and current with resolution of 0.1 seconds that lasts 60 seconds:  <pre> SENS:DLOG:PER 0.1 SENS:DLOG:TIME 60 SENS:DLOG:FUNC:VOLT ON, CH1 SENS:DLOG:FUNC:CURREN ON, CH1 INIT:DLOG "Recordings/test_log.dlog" </pre>			
<b>Related Commands</b>	*RST ABORT:DLOG DISPlay[:WINdow]:DLOG INITiate:DLOG:TRACe MMEMory:INFORmation? SENSE:DLOG:FUNCTION:CURRENt SENSE:DLOG:FUNCTION:POWER SENSE:DLOG:FUNCTION:VOLTage SENSE:DLOG:PERiod SENSE:DLOG:TIME			

### 5.7.3. INITiate:DLOG:TRACe

**Syntax**      `INITiate:DLOG:TRACe {<filename>}`

**Description** The command starts internal data logging session. Single X-axis and multiple Y-axis are allowed as in case on the INITiate:DLOG:TRACe command, but offering more flexibility

in defining recording parameters. The duration of data logging is not defined in advance and new data can be added until termination command is not executed e.g. ABORT:DLOG, SYSTem:REStart or \*RST command.

Parameters	Name	Type	Range	Default
	<filename>	Quoted string	File name, either / (slash) or \ (back-slash) can be used as the path separator. 1 to 255 characters	–
<b>Usage example</b>	<p>Data logging is a four step process as follows:</p> <ol style="list-style-type: none"> <li>1. Define X- and Y-axis parameters. Number of Y-axis is arbitrary and two is used in this example. Data log file remark is not mandatory.</li> </ol> <pre> SENS:DLOG:TRAC:X:UNIT SECO SENS:DLOG:TRAC:X:STEP 0.01 SENS:DLOG:TRAC:X:RANG:MIN 0 SENS:DLOG:TRAC:X:RANG:MAX 20 SENS:DLOG:TRAC:X:LAB "t" SENS:DLOG:TRAC:Y1:UNIT VOLT SENS:DLOG:TRAC:Y1:LAB "U" SENS:DLOG:TRAC:Y1:RANG:MIN 0 SENS:DLOG:TRAC:Y1:RANG:MAX 40 SENS:DLOG:TRAC:Y2:UNIT AMPE SENS:DLOG:TRAC:Y2:LAB "I" SENS:DLOG:TRAC:Y2:RANG:MIN 0 SENS:DLOG:TRAC:Y2:RANG:MAX 5 SENS:DLOG:TRAC:X:SCAL LIN SENS:DLOG:TRAC:Y:SCAL LIN </pre> <ol style="list-style-type: none"> <li>2. Define data log filename and comment/description.</li> </ol> <pre> SENS:DLOG:TRAC:REM "data log test" INIT:DLOG:TRAC "Recordings/test_log.dlog" </pre> <ol style="list-style-type: none"> <li>3. Add arbitrary number of recordings:</li> </ol> <pre> SENS:DLOG:TRAC:DATA 10.1,2.55 SENS:DLOG:TRAC:DATA 12.0,2.66 SENS:DLOG:TRAC:DATA 16.34,3.63 ... </pre> <ol style="list-style-type: none"> <li>4. Close data logging session.</li> </ol> <pre> ABOR:DLOG </pre>			
<b>Related Commands</b>	<pre> *RST ABORT:DLOG DISPlay[:WINdow]:DLOG INITiate:DLOG MMEMory:INFORmation? SENSe:DLOG:TRACe:REMark SYSTem:REStart </pre>			

#### 5.7.4. INITiate:CONTInuous

**Syntax**      `INITiate:CONTInuous {<bool>}`  
`INITiate:CONTInuous?`

**Description** This command is used to select whether the trigger system is continuously initiated or not. With CONTInuous set to OFF, the trigger system remain in the IDLE state until CONTInuous is set to ON or INITiate:IMMediate is received. Once CONTInuous is set to ON, the trigger system will be initiated and exit the IDLE state. On completion of each trigger cycle, with CONTInuous ON, the trigger system im-



mediately commence another trigger cycle without entering the IDLE state.

When INITiate:CONTinuous is set to OFF, the current trigger cycle will be completed before entering the IDLE state. The return to IDLE also occur as the result of an ABORt or \*RST command.

The ABORt command force the trigger system to the IDLE state; however, the value of INITiate:CONTinuous is unaffected.

If INITiate:CONTinuous was set to ON prior to receiving ABORt, it remains ON and the trigger system immediately exit the IDLE state.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	OFF ON 0 1	OFF
<b>Return</b>	The query command returns 0 if continuous transients are disabled (OFF), and 1 if continuous transients are enabled (ON).			
<b>Usage example</b>	INIT:CONT ON			
<b>Related Commands</b>	*RST ABORt			

## 5.8. INSTRument

Each channel of the BB3 is considered as separate (logical) instrument, which is required by the SCPI standard. The INSTRument subsystem provides a mechanism to identify and select instruments and establish coupling to simplify programming of more channels at once.

SCPI command	Description
INSTRument	
<a href="#">[:SElect] {&lt;channel&gt;}</a>	Selects the output to be programmed
<a href="#">:CATalog?</a>	Returns a quoted string of the list of valid choices for the instrument channels
<a href="#">:FULL?</a>	Returns a list of string – number pairs
:COUPle	
<a href="#">:TRACking {&lt;type&gt;}</a>	Selects independent, common ground-tracking, split rail-tracking, parallel-tracking, or series-tracking mode
<a href="#">:TRIGger {&lt;mode&gt;}</a>	Selects a coupling between channels trigger systems
:DISPlay	
<a href="#">:TRACe[&lt;n&gt;] {&lt;value&gt;}</a>	Selects output value on the specified display trace
<a href="#">:SWAP</a>	Swaps positions of selected output values
:YT	
<a href="#">:RATE {&lt;duration&gt;}</a>	Selects YT view sample duration
<a href="#">:NSElect {&lt;channel&gt;}</a>	Selects the output to be programmed

### 5.8.1. INSTRument[:SElect]

**Syntax**      [INSTRument\[:SElect\] {<channel>}](#)  
[INSTRument\[:SElect\]?](#)

**Description** This command selects the output to be programmed by the output identifier. The outputs of the BB3 are considered as separate logical instruments. The INSTRument command provides a mechanism to identify and select an output. When one output is selected, the other output is unavailable for programming until selected. The following commands are affected by the INSTRument command: SOURce, MEASure, and CALibration.

Parameters	Name	Type	Range	Default
	<channel>	Discrete ChannelList	CH1 CH2 CH3 CH4 CH5 CH6	–

**Return** Query returns the currently selected output by the INSTRument[:SElect] or INSTRument:NSElect command. The returned value is CH1, CH2, CH3, CH4, CH5 or CH6.

**Usage example**      INST?

CH1

INST:SEL?

CH1

Select AO1 channel on the SMX46 module installed in slot 3:

INST (@301)

**Related Commands**      INSTRument:CATalog?  
INSTRument:CATalog:FULL?  
INSTRument:NSElect

### 5.8.2. INSTRument:CATalog?

**Syntax** INSTRument:CATalog?

**Description** This query returns a comma-separated list of strings which contains the names of all logical instruments. If no logical instruments are defined, a single null string is returned.

**Usage example** INST:CAT?  
"CH1", "CH2", "CH3", "CH4"

**Related Commands** INSTRument[:SElect]  
INSTRument:CATalog:FULL?  
INSTRument:NSElect

### 5.8.3. INSTRument:CATalog:FULL?

**Syntax** INSTRument:CATalog:FULL?

**Description** Use this query to get a list of string – number pairs. The string contains the name of the logical instrument. The immediately following NR1-formatted number is its associated logical instrument number. All response data elements are comma separated. If no logical instrument is defined, a null string followed by a zero is returned.

**Usage example** INST:CAT:FULL?  
"CH1", 1, "CH2", 2

**Related Commands** INSTRument[:SElect]  
INSTRument:CATalog?  
INSTRument:NSElect

### 5.8.4. INSTRument:COUPle:TRACking

**Syntax** INSTRument:COUPle:TRACking {<type>}  
INSTRument:COUPle:TRACking?

**Description** This command selects how channel's outputs will be internally connected:

- independent (NONE)
- common ground (CGND) controls K\_CGND power relay state that connects Vout- outputs of all three modules together
- split rail (SRAil) controls K\_SRAIL power relay state that combines CH1 Out- and CH2 Out+. The CH1 Out+ becomes positive rail, and CH2 Out- negative rail (**DCP405**).
- parallel-tracking (PARallel) controls K\_PAR power relay state when max. output current is doubled, e.g. 10 A instead of 5 A (**DCP405**), or
- series-tracking (SERies) that controls K\_SER power relay state and max. output voltage is doubled, e.g. 80 V instead of 40 V (**DCP405**).

When channels are coupled, resulting output will be present on different output terminals as indicated with red module's OE LED instead of green OE LED.

Coupled channels in series or parallel will be seen as single channel. Therefore the following commands will affect the both channel regardless of which channel is currently selected using the INSTRument[:SElect] or INSTRument:NSElect command):

- OUTPut[:STATe], OUTPut:DPRog, OUTPut:PROTection:CLEAr
- MEASure[:SCALar]:CURRent[:DC], MEASure[:SCALar]:POWER[:DC], MEASure[:SCALar][:VOLTage][:DC]
- SIMULator:LOAD, SIMULator:LOAD:STATe
- [SOURce[<n>]]:CURRent, [SOURce[<n>]]:CURRent:STEP, [SOURce[<n>]]:CURRent:TRIGgered, [SOURce[<n>]]:CURRent:LIMit, [SOURce[<n>]]:CURRent:MODE, [SOURce[<n>]]:CURRent:PROTection:DELay, [SOURce[<n>]]:CURRent:PROTection:STATe, [SOURce[<n>]]:CURRent:PROTection:TRIPped?, [SOURce[<n>]]:LIST:COUNT, [SOURce[<n>]]:LIST:CURRent, [SOURce[<n>]]:LIST:DWELL,

- [SOURce[<n>]]:LIST:VOLTage[:LEVel], [SOURce[<n>]]:POWER:LIMit,
- [SOURce[<n>]]:POWER:PROTection[:LEVel],
- [SOURce[<n>]]:POWER:PROTection:DELAy[:TIME],
- [SOURce[<n>]]:POWER:PROTection:STATe,
- [SOURce[<n>]]:POWER:PROTection:TRIPped?, [SOURce[<n>]]:VOLTage,
- [SOURce[<n>]]:VOLTage:LIMit, [SOURce[<n>]]:VOLTage:STEP,
- [SOURce[<n>]]:VOLTage:TRIGgered, [SOURce[<n>]]:VOLTage:MODE,
- [SOURce[<n>]]:VOLTage:PROTection[:LEVel],
- [SOURce[<n>]]:VOLTage:PROTection:DELAy,
- [SOURce[<n>]]:VOLTage:PROTection:STATe,
- [SOURce[<n>]]:VOLTage:PROTection:TRIPped?
- TRIGger[:SEQuence][:IMMediate], TRIGger[:SEQuence]:DELAy,
- TRIGger[:SEQuence]:SLOPe, TRIGger[:SEQuence]:SOURce

The following channel specific commands will generate a device specific error 312 when channels are coupled:

- CALibrate,
- OUTPut:TRACk[:STATe],
- SIMULator:RPOL, SIMULator:VOLTage:PROGram,
- [SOURce[<n>]]:VOLTage:PROGram,
- [SOURce:]VOLTage:SENSe (series-tracking only)

Channels coupled in parallel could have only one down-programmer circuit active that is on the channel 1. State of the channel 2 down-programmer will always be off regardless of what is set with the OUTPut:DPRog command.

If channels are calibrated (CAL:STAT? 1) a max. value of the channel that has lower calibrated value will be used as a reference and multiplied by two. For example, if channel 1 max. voltage is 39.98 V and channel 2 max. voltage is 40.00 V the new max. value for the SERies-tracking will become 79.96 V instead of 80.00 V.

Two conditions requires special attention: that is entering CC mode while channels are coupled in SERies or entering CV mode when PARAllel coupling is active. For example when coupled in SERies and output voltage is set to 60 V and current to 1.7 A with connected load of 1  $\Omega$  the power module will enter the CC mode of operation (see OUTPut:MODE?) and output voltage will drop to 1.7 V. Coupling mechanism will set both channels to 30 V (initially set value divided by two) but that value cannot be maintained on any output and there is no warranty that new voltage will be equally shared between channels (i.e.  $1.7 / 2 = 1.35$  V per channel). It's even possible that outputs become unbalanced in a way that one of the channels becomes negative like -0.6 V on one channel and +2.3 V on another that still resulting in required +1.7 V limited by max. current. To avoid such situation coupling mechanism also include *balancing* to ensure that such deviation when one channel is pushed to sink instead of source power is rectified. That is accomplished by calculating and set more appropriate output voltage values during the CC mode of operation. The measured output voltage (using the MEASure[:SCALar][:VOLTage][:DC]? Command) will return that newly programmed value, that could be e.g. 30.93 V instead of 60 V. When output come back to CV mode of operation (e.g. load is disconnected), initial set 60 V will be measured again.

Execution of this command also affects bit 8 (PARAllel) or bit 9 (SERies) of the Operation status register (see [Section 3.3](#)).

At \*RST, channels will be uncoupled (NONE).

Parameters	Name	Type	Range	Default
	<type>	Discrete	NONE CGND SRAI  PARAllel SERies	NONE
<b>Return</b>	Query returns the currently selected output coupling state.			
<b>Usage</b>	INST:COUP:TRAC SER			

**example**     VOLT 70  
                  VOLT?  
                  70.00  
                  INST:COUP:TRAC PAR  
                  CURR 9  
                  CURR?  
                  9.00

**Errors**       312, "Cannot execute when the channels are coupled"

**Related**       \*SAV

**Commands**    MEASure[:SCALar]:CURRent[:DC]  
                  MEASure[:SCALar][:VOLTage][:DC]  
                  OUTPut:DPRog  
                  OUTPut:MODE?  
                  OUTPut:TRACk[:STATe]

### 5.8.5. INSTRument:DISPlay:TRACe[<n>]

**Syntax**        INSTRument:DISPlay:TRACe[<n>] {<value>}  
                  INSTRument:DISPlay:TRACe[<n>]?

**Description** This command sets the output value that will be displayed on the display position (trace) defined with [<n>]. This command affects only display modes 2, 3 and 4 (see the DISPlay:VIEW command).  
 An attempt to select the same value on both positions (traces) will generate an execution error.

Parameters	Name	Type	Range	Default
	<value>	Discrete	VOLTage CURRent POWER	–
<b>Return</b>	Query returns the currently selected displayed output value on the selected display position (trace).			
<b>Usage example</b>	INST:DISP:TRAC2? CURR			
<b>Errors</b>	-200, "Execution error"			
<b>Related Commands</b>	DISPlay:VIEW INSTRument:DISPlay:TRACe:SWAP INSTRument:DISPlay:YT:RATE			

### 5.8.6. INSTRument:DISPlay:TRACe:SWAP

**Syntax**        INSTRument:DISPlay:TRACe:SWAP

**Description** Use this command to swap output values display positions.

**Usage example**    INST:DISP:TRAC1?  
                  VOLT  
                  INST:DISP:TRAC2?  
                  CURR  
                  INT:DISP:TRAC:SWAP  
                  INST:DISP:TRAC1?  
                  CURR  
                  INST:DISP:TRAC2?  
                  VOLT

**Related Commands**    DISPlay:VIEW  
                  INSTRument:DISPlay:TRACe[<n>] {<value>}

INSTrument:DISPlay:YT:RATE

**5.8.7. INSTrument:DISPlay:YT:RATE**

**Syntax**      `INSTrument:DISPlay:YT:RATE {<duration>}`  
`INSTrument:DISPlay:YT:RATE?`

**Description** This command sets the sample duration in seconds when YT (mode 4, see the DISPlay:VIEW command) display view is selected.

Parameters	Name	Type	Range	Default
	<duration>	NR2	0.02 – 300	0.1

**Return**      The query command returns the programmed sample duration in seconds.

**Usage example**      `INST:DISP:YT:RATE 10`

**Related Commands**      `DISPlay:VIEW`  
`INSTrument:DISPlay:TRACe[<n>] {<value>}`  
`INSTrument:DISPlay:TRACe:SWAP`

**5.8.8. INSTrument:COUPle:TRIGger**

**Syntax**      `INSTrument:COUPle:TRIGger {<mode>}`  
`INSTrument:COUPle:TRIGger?`

**Description** *Not implemented yet*

This command defines a coupling between channels trigger systems. Use ALL parameter to couple or NONE to remove coupling.

At \*RST, trigger systems are uncoupled.

Parameters	Name	Type	Range	Default
	<mode>	Discrete	ALL CH1 CH2 CH3  CH4 CH5 CH6 NONE	ALL

**Return**      This query returns the currently coupled output.

**Usage example**      `INST:SEL CH1`  
`VOLT:TRIG 12`  
`CURR:TRIG 1.5`  
`INST:SEL CH2`  
`VOLT:TRIG 5`  
`CURR:TRIG MAX`  
`INST:COUP:TRIG ALL`  
`TRIG:SOUR IMM`  
`INIT`

**Related Commands**      `*RST`  
`INSTrument:COUPle:TRACking`

**5.8.9. INSTrument:NSElect**

**Syntax**      `INSTrument:NSElect {<channel>}`  
`INSTrument:NSElect?`

**Description** This command is used in conjunction with the SElect command. It serves the same purpose, except that it uses a numeric value instead of the identifier used in the SElect command.

Parameters	Name	Type	Range	Default
	<channel>	NR1	1 – 6	–

**Return**      When queried it returns the logical instrument number of the currently selected BB3

channel. Note that the numbering used for logical instruments directly corresponds to the numbers used in status reporting for multiple instruments; specifically the STATUS:QUESTIONable:INSTRument and STATUS:OPERation:INSTRument commands.

**Usage  
example**

INST:NSEL 2  
INST:NSEL?  
2

**Related  
Commands**

STATUS:QUESTIONable:INSTRument  
STATUS:OPERation:INSTRument

## 5.9. MEASure

Measure commands return back the output voltage, current, power or temperature. They trigger the acquisition of new data before returning the reading. Measurements are performed by digitizing the instantaneous output voltage, current or temperature. Output power is calculated as product of measured voltage and current. Keyword [:DC] is optional since all measurement are by default of the DC level of the signal.

SCPI command	Description
MEASure	
[:SCALar]	
:CURRent	
[:DC]? [<channel>]	Takes a measurement; returns the average current
:POWer	
[:DC]? [<channel>]	Takes a measurement; returns the average power
[:VOLTage]	
[:DC]? [<channel>]	Takes a measurement; returns the average voltage

### 5.9.1. MEASure[:SCALar]:CURRent[:DC]

**Syntax** MEASure[:SCALar]:CURRent[:DC]? [<channel>]

**Description** Query the current measured across the current sense resistor inside the power module.

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4  CH5 CH6	–

**Return** Returns the average output current in amperes as decimal number (NR2).

**Usage example** Measure current on the currently selected channel (CH1) and CH2:

```
MEAS:CURR?; :MEAS:CURR? CH2  
1.23;0.12
```

**Related Commands** INSTRument:COUPle:TRACking

### 5.9.2. MEASure[:SCALar]:POWer[:DC]

**Syntax** MEASure[:SCALar]:POWer[:DC]? [<channel>]

**Description** Query the output power calculated as product of measured voltage and current value.

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4  CH5 CH6	n/a

**Return** Returns the average output power in watts as decimal number (NR2).

**Usage example** MEAS:POW? CH2  
80.44

**Related Commands**

### 5.9.3. MEASure[:SCALar][:VOLTage][:DC]

**Syntax** MEASure[:SCALar][:VOLTage][:DC]? [<channel>]

**Description** Query the voltage measured at the sense terminals of the selected channel.



Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4  CH5 CH6	n/a
<b>Return</b>	Returns the average output voltage in volts as decimal number (NR2).			
<b>Usage example</b>	<pre>MEAS:VOLT? CH1 43.25</pre> <p>Query voltage of the channel 2 that is currently selected:</p> <pre>INST CH2 MEAS? 12.40</pre>			
<b>Related Commands</b>	INSTrument:COUPle:TRACking			

## 5.10. MEMory

The MEMory subsystem works with BB3 state files that are saved to ([\\*SAV](#)) and recalled from ([\\*RCL](#)) non-volatile storage locations numbered 0 through 9. The storage location 0 named "Power down state" is used to store the current BB3 parameters.

SCPI command	Description
MEMory	
<a href="#">:NSTates?</a>	Returns total number of state storage memory locations
:STATe	
<a href="#">:CATalog?</a>	Lists the names associated with all ten state storage locations
<a href="#">:DELeTe {&lt;profile&gt;}</a>	Deletes the contents of a state storage location
:ALL	Deletes the contents of all state storage locations
<a href="#">:FREEze {&lt;bool&gt;}</a>	Freezes saving changes into profile 0
<a href="#">:NAME {&lt;profile&gt;}, {&lt;name&gt;}</a>	Assigns a custom name to a state storage locations
:RECall	
<a href="#">:AUTO {&lt;bool&gt;}</a>	Specifies whether the power-down state is recalled from location 0 on power-on
<a href="#">:SELeT {&lt;profile&gt;}</a>	Specifies which BB3 state will be used at power on
<a href="#">:VALid? {&lt;profile&gt;}</a>	Determines whether a storage location contains a valid state

### 5.10.1. MEMory:NSTates

**Syntax** [MEMory:NSTates?](#)

**Description** Returns the total number of \*SAV/\*RCL states available in the BB3.

**Return** Returns numeric value (NR1) which is one greater than the maximum that can be sent as a parameter to the \*SAV and \*RCL commands.

**Usage example** MEM:NST?  
10

### 5.10.2. MEMory:STATe:CATalog

**Syntax** [MEMory:STATe:CATalog?](#)

**Description** This query requests a list of defined names in the MEMory:STATe subsystem.

**Return** The BB3 returns a list of defined <name>'s in a comma separated list. Each <name> is returned in a quoted string.

**Usage example** MEM:STAT:CAT?  
",  
"12V/1A",  
"list V",  
"4ch",  
"tracking",  
"parallel",  
"long lists",  
"list all channels",  
"Saved at 2019-10-22 09:33:55",  
"--Empty--"

**Related Commands** MEMory:STATe:NAME

### 5.10.3. MEMory:STATe:DELeTe

**Syntax**      `MEMory:STATe:DELeTe {<profile>}`  
                  `MEMory:STATe:DELeTe:ALL`

**Description** When used with a profile number this command deletes the contents of the specified storage location. The `MEMory:STATe:DELeTe:ALL` deletes the contents of storage locations 1 through 9.

*An error is generated on an attempt to recall a deleted state.*

Parameters	Name	Type	Range	Default
	<profile>	NR1	1 – 9	–
<b>Return</b>	None			
<b>Usage example</b>	<code>MEM:STAT:DEL 2</code>			
<b>Related Commands</b>	*RCL *SAV			

### 5.10.4. MEMory:STATe:FREEze

**Syntax**      `MEMory:STATe:FREEze {<bool>}`  
                  `MEMory:STATe:FREEze?`

**Description** This command prohibits saving any further changes to profile 0. It can be used, for example, to freeze program states before running the MicroPython script because the script will change some of the parameters. Once the script is complete, saving changes caused by further user activity can be re-enabled.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	–
<b>Return</b>	The command returns 0 if saving changes to profiles 0 is enabled, or returns 1 if saving changes to profiles 0 is disabled.			
<b>Usage example</b>	<code>MEM:STAT:FREE 1</code>			
<b>Related Commands</b>	*SAV			

### 5.10.5. MEMory:STATe:NAME

**Syntax**      `MEMory:STATe:NAME {<profile>}, {<name>}`  
                  `MEMory:STATe:NAME? {<profile>}`

**Description** This command associates a <name> with a \*SAV/\*RCL register number. May assign same name to different locations and state names are unaffected by [\\*RST](#). Deleting a storage location's contents `MEMory:STATe:DELeTe` resets associated name to "--Empty--"

Parameters	Name	Type	Range	Default
	<profile>	NR1	1 – 9	–
	<name>	Quoted string	0 to 32 characters	–
<b>Return</b>	Returns a *SAV/*RCL register number associates with profile number.			
<b>Usage example</b>	<code>MEM:STAT:DEF, 2, "All outputs on"</code> <code>MEM:STAT:DEF? 2</code> "All outputs on"			
<b>Related Commands</b>	<code>MEMory:STATe:DELeTe</code>			

### 5.10.6. MEMory:STATe:RECall:AUTO

**Syntax**      `MEMory:STATe:RECall:AUTO {<bool>}`  
`MEMory:STATe:RECall:AUTO?`

**Description** This command disables or enables the automatic recall of a specific stored BB3 state selected using the MEMory:STATe:RECall:SElect command when power is turned on. Select ON to automatically recall one of the ten stored states or the “power-down” state (location 0) when power is turned on. Select OFF to issue a reset ([\\*RST](#)) when power is turned on.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	–

**Return** The query command returns 0 (OFF) or 1 (ON).

**Usage example** `MEM:STAT:REC:AUTO?`  
 1

**Related Commands** \*SAV  
 MEMory:STATe:RECall:SElect  
 SYSTem:POWer

### 5.10.7. MEMory:STATe:RECall:SElect

**Syntax**      `MEMory:STATe:RECall:SElect {<profile>}`  
`MEMory:STATe:RECall:SElect?`

**Description** This command selects which BB3 state will be used at power on if the automatic recall mode is enabled (see MEMory:STATe:RECall:AUTO ON command). If the automatic recall mode is disabled (MEMory:STATe:RECall:AUTO OFF), then a “factory reset” (return to the default values) is issued when power is turned on.

Parameters	Name	Type	Range	Default
	<profile>	NR1	0 – 9	–

**Return** The query command returns numeric value (NR1) for 0 to 9 indicating which instrument state will be used at power on.

**Usage example** `MEM:STAT:REC:SEL?`  
 2

**Related Commands** \*SAV  
 MEMory:STATe:RECall:AUTO  
 SYSTem:POWer

### 5.10.8. MEMory:STATe:VALid

**Syntax**      `MEMory:STATe:VALid? {<profile>}`

**Description** This command queries the specified storage location to determine if a valid state is currently stored in this location.

*Use this command before sending the [\\*RCL](#) command to determine if a valid state has been previously stored on queried location.*

Parameters	Name	Type	Range	Default
	<profile>	NR1	0 – 9	–

**Return** Returns 0 if no state has been stored or if it has been deleted. It returns 1 if a valid state is stored in this location.

**Usage example** `MEM:STAT:VAL? 2`  
 1

**Related** \*RCL

**Commands** \*SAV

## 5.11. MMEMemory

The MMEMemory commands are used to store, read or delete file in the BB3's SD card. It can also query SD card information. In addition it is used for storing and recalling values used by [SOURCE[<n>]]:LIST subsystem. File and directory (folder) names cannot contain the following characters: \ / : \* ? " < > |

SCPI command	Description
MMEMemory	
<a href="#">:CATalog [&lt;directory&gt;]</a>	Returns a list of items in the specified directory (folder)
<a href="#">:LENgth [&lt;directory&gt;]</a>	Returns the number of items in the specified directory
<a href="#">:CDIRectory {&lt;directory&gt;}</a>	Changes the current directory
<a href="#">:CLOSe</a>	Closes the file specified in NAME
<a href="#">:COPY {&lt;source&gt;}, {&lt;destination&gt;}</a>	Copies <source> to <destination>
<a href="#">:DATE? {&lt;filename&gt;}</a>	Returns date that the specified file was last saved
<a href="#">:DELeTe {&lt;filename&gt;}</a>	Deletes an existing file
:DOWNload	
<a href="#">:ABORt</a>	Aborts current download session
<a href="#">:DATA {&lt;block&gt;}</a>	Downloads data from the host computer
<a href="#">:FNAME {&lt;filename&gt;}</a>	Creates or opens the specified filename for download data
<a href="#">:SIZE {&lt;filesize&gt;}</a>	Sets information about file size used for progress bar
<a href="#">:FEED</a>	Sets data handle used to feed data into the file
<a href="#">:INFOrmation?</a>	Returns used and free space
:LOAD	
<a href="#">:LIST&lt;n&gt; {&lt;filename&gt;}</a>	Loads stored LIST to the specified channel
<a href="#">:PROFile {&lt;filename&gt;}</a>	Loads stored user profile
<a href="#">:LOCK {&lt;password&gt;}</a>	Sets write protection
<a href="#">:MDIRectory {&lt;directory&gt;}</a>	Makes a new directory
<a href="#">:MOVE {&lt;source&gt;}, {&lt;destination&gt;}</a>	Moves or renames <source> to <destination>
<a href="#">:NAME {&lt;filename&gt;}</a>	Sets the file name to be opened or closed
<a href="#">:OPEN</a>	Opens the file specified in NAME
<a href="#">:RDIRectory {&lt;directory&gt;}</a>	Removes the specified directory
:STORe	
<a href="#">:LIST&lt;n&gt; {&lt;filename&gt;}</a>	Saves specified channel LIST
<a href="#">:PROFile {&lt;filename&gt;}</a>	Saves specified user profile
<a href="#">:TIME? {&lt;filename&gt;}</a>	Returns time that the specified file was last saved
<a href="#">:UNLock {&lt;password&gt;}</a>	Clears write protection
<a href="#">:UPLoad? {&lt;filename&gt;}</a>	Uploads data to the host computer

### 5.11.1. MMEMory:CATalog

**Syntax** `MMEMory:CATalog? [<directory>]`

**Description** Returns the list of files and directories (folders) names, types and sizes in the current or specified directory. Number of items (files/directories) corresponds to the value returned by the MMEMory:CATalog:LENgth? command.

To read out the information in the root directory (folder), specify "\" (backslash) or "/". If <directory> is not set, this function is applied to the current directory. If directory=<path>, this function is applied to <current directory>\<path>.

Error occurs if the specified directory does not exist.

Parameters	Name	Type	Range	Default
	<directory>	Quoted string	Directory name, either / (slash) or \ (backslash) can be used as the path separator. 1 to 255 characters	–

**Return** Returns all files in the directory as list of comma delimited quoted strings of <filename>, <filetype> and <filesize>. The following file types are supported:

- BIN – binary data
- CSV – textual data (comma separated)
- FOLD – directory (folder)
- LIST – LIST program data
- LOG – trace (display) or logged data (file extension .log)
- PROF – user profile data (file extension .profile)
- STAT – instrument (setting) state or user profiles (file extension .conf)

**Usage example** `MMEM:CAT?`

`"USER,FOLD,0","SCPI.PDF,BIN,1274844","SCH5B13A.PDF,BIN,296589",  
"Documents,FOLD,0","Lists,FOLD,0","Videos,FOLD,0",  
"profile0.profile,PROF,264"`

`MMEM:CAT? "USER"`

`"LST_2_3.CSV,BIN,88","FERY2.PDF,BIN,2443"`

**Errors**

- 250,"Mass storage error"
- 251,"Missing mass storage"
- 252,"Missing media"
- 256,"File name not found"

**Related Commands**

- MMEMory:CATalog:LENgth?
- MMEMory:INFOrmation?
- MMEMory:DATE?
- MMEMory:TIME?

### 5.11.2. MMEMory:CATalog:LENgth

**Syntax** `MMEMory:CATalog:LENgth? [<directory>]`

**Description** This command returns the number of items in the current or specified directory (folder). The result corresponds to the number of files returned by the MMEMory:CATalog? command.

Parameters	Name	Type	Range	Default
	<directory>	Quoted string	Directory name, either / (slash) or \ (backslash) can be used as the path separator. 1 to 255 characters	–

**Return** The query command returns number (NR1) of items (files and directories).

**Usage example** `MMEM:CAT:LEN? "USER"`  
`2`

**Errors** `-256,"File name not found"`

**Related Commands** `MMEMory:CATalog?`

### 5.11.3. MMEMory:CDIRectory

**Syntax** `MMEMory:CDIRectory {<directory>}`  
`MMEMory:CDIRectory?`

**Description** Changes the current directory to the specified directory (folder). This directory must exist otherwise an error will be generated.

At \*RST, this value is set to the root path.

Parameters	Name	Type	Range	Default
	<directory>	Quoted string	Directory name, either / (slash) or \ (backslash) can be used as the path separator. 1 to 255 characters	–

**Return** This query returns the full path of the current directory.

**Usage example** `MMEM:CDIR "TEST/Test folder2"`  
`MMEM:CDIR?`  
`"TEST/Test folder2"`

**Errors** `-256,"File name not found"`

**Related Commands** `*RST`  
`MMEMory:CATalog?`  
`MMEMory:MDIRectory`  
`MMEMory:RDIRectory`

### 5.11.4. MMEMory:COPY

**Syntax** `MMEMory:COPY {<source>}, {<destination>}`

**Description** Makes a copy of an existing file in the current directory. The file names must include any file extension.

If <destination> is a file name, the copy file is created in the current directory. When <destination> is a <path> (e.g. "test lists/test022") the source file is duplicated in <current directory>/<path>.

Parameters	Name	Type	Range	Default
	<source>	Quoted string	Source file name, 1 to 255 characters	–
	<destination>	Quoted string	Copy file name or directory name, either / (slash) or \ (backslash) can be used as the path separator. 1 to 255 characters	–

**Usage example** `MMEM:COPY "test.bin", "new2/test_new.bin"`

**Errors** `-253,"Corrupt media"`  
`-254,"Media full"`  
`-256,"File name not found"`  
`-258,"Media protected"`

**Related Commands** `MMEMory:CATalog?`  
`MMEMory:CDIRectory`  
`MMEMory:LOCK`



MMEMory:MDIRectory  
MMEMory:MOVE

### 5.11.5. MMEMory:DATE

**Syntax** `MMEMory:DATE? {<filename>}`

**Description** Returns the (year, month, day) that the specified file was last saved.

Parameters	Name	Type	Range	Default
	<filename>	Quoted string	File name or directory name, either / (slash) or \ (backslash) can be used as the path separator. 1 to 255 characters	—
<b>Return</b>	Comma-separated numbers (NR1) of year, month, day formatted as yyyy, mm, dd			
<b>Usage example</b>	MMEM:DATE? "test.002" 2017, 10, 1			
<b>Errors</b>	-256,"File name not found"			
<b>Related Commands</b>	MMEMory:TIME			

### 5.11.6. MMEMory:DELeTe

**Syntax** `MMEMory:DELeTe {<filename>}`

**Description** Use this command to delete a file in the current directory. If SD card is locked using the MMEMory:LOCK command, an error -258 will be generated.

Parameters	Name	Type	Range	Default
	<filename>	Quoted string	File name, either / (slash) or \ (backslash) can be used as the path separator. 1 to 255 characters	—
<b>Usage example</b>	MMEM:DEL "test.002"			
<b>Errors</b>	-256,"File name not found" -258,"Media protected"			
<b>Related Commands</b>	MMEMory:CATalog? MMEMory:CDIRectory MMEMory:RDIRectory MMEMory:LOCK			

### 5.11.7. MMEMory:DOWNload:ABORt

**Syntax** `MMEMory:DOWNload:ABORt`

**Description** Use this command to abort currently running file transfer from the host initiated with the MMEMory:DOWNload:DATA command. If not active file transfer exists, the command will be ignored without generating any error.

**Usage example** MMEM:ABOR

**Related Commands** MMEMory:DOWNload:DATA

### 5.11.8. MMEMory:DOWNload:DATA

**Syntax** `MMEMory:DOWNload:DATA {<block>}`

**Description** Downloads data from the host computer to a file in the SD card. This is a multiple steps

process:

- The filename must have been previously specified by MMEMory:DOWNload:FNAME.
- The data can be transferred in single or more blocks. Receiving of first block will erase all previously stored, and each consecutive block will be appended to the end
- Download is finished when MMEMory:DOWNload:FNAME with empty name was sent

Use MMEMory:INFOrmation? command first to check available space.

Parameters	Name	Type	Range	Default
	<block>	Data block	–	–
<b>Usage example</b>	Downloads text <i>Hello world</i> and store into the file "test file" in the current directory. Digit 2 denotes two digits of data length (11).  MMEM:DOWN:FNAME "test file" MMEM:DOWN:DATA #211Hello world MMEM:DOWN:FNAME ""			
<b>Errors</b>	-253, "Corrupt media" -254, "Media full" -258, "Media protected"			
<b>Related Commands</b>	MMEMory:DOWNload:FNAME MMEMory:INFOrmation? MMEMory:LOCK			

#### 5.11.9. MMEMory:DOWNload:FNAME

<b>Syntax</b>	MMEMory:DOWNload:FNAME {<filename>}			
<b>Description</b>	Creates or opens the specified filename prior to writing data to that file with MMEMory:DOWNload:DATA.			
Parameters	Name	Type	Range	Default
	<filename>	Quoted string	File name, either / (slash) or \ (backslash) can be used as the path separator. 1 to 255 characters	–
<b>Usage example</b>	MMEMory:DOWNload:FNAME "new_list.bin"			
<b>Errors</b>	-258, "Media protected"			
<b>Related Commands</b>	MMEMory:DOWNload:DATA			

#### 5.11.10. MMEMory:DOWNload:SIZE

<b>Syntax</b>	MMEMory:DOWNload:SIZE {<filesize>}			
<b>Description</b>	This command define filesize used by progress bar displayed on the local console during the file transfer. If filesize is not provided, progress bar will not be displayed.			
Parameters	Name	Type	Range	Default
	<filesize>	NR1	0 to 2.147.483.648 (2 GiB) as limited by SD Card FAT	0
<b>Usage example</b>	MMEMory:DOWNload:SIZE 124000			
<b>Related Commands</b>	MMEMory:DOWNload:DATA MMEMory:DOWNload:FNAME			

**5.11.11. MMEMory:INFOrmation****Syntax** `MMEMory:INFOrmation?`**Description** Use this command to find out total amount of storage currently used and storage available on the SD card. The sum of that two amounts represents SD card capacity in bytes.**Return** The command returns used space and free space as two comma separated integers.**Usage example**  
`MMEM:INFO?`  
`3932160,7732461568`**Related Commands** `MMEMory:DOWNload:DATA`**5.11.12. MMEMory:LOAD:LIST****Syntax** `MMEMory:LOAD:LIST<n> {<filename>}`**Description** Load stored LIST program from SD card location defined by <filename> to the channel defined with <n>.

Parameters	Name	Type	Range	Default
	<filename>	Quoted string	File name, either / (slash) or \ (backslash) can be used as the path separator. 1 to 255 characters	–
<b>Usage example</b>	<code>MMEM:LOAD:LIST1 "DC_DC conv testing.list"</code> <code>MMEM:LOAD:LIST2 "DC_DC conv testing.list"</code> <code>TRIG:SOUR BUS</code> <code>INIT</code> <code>*TRG</code>			
<b>Errors</b>	<code>-200,"Execution error"</code> <code>-256,"File name not found"</code>			
<b>Related Commands</b>	<code>MMEMory:STORe:LIST[&lt;n&gt;]</code> <code>[SOURce[&lt;n&gt;]]:LIST:COUNT</code> <code>[SOURce[&lt;n&gt;]]:LIST:CURRENT[:LEVel]</code> <code>[SOURce[&lt;n&gt;]]:LIST:DWELl</code> <code>[SOURce[&lt;n&gt;]]:LIST:VOLTage[:LEVel]</code>			

**5.11.13. MMEMory:LOAD:PROFile****Syntax** `MMEMory:LOAD:PROFile {<filename>}`**Description** This command loads data from user profile file to the user profile 0.

Parameters	Name	Type	Range	Default
	<filename>	Quoted string	File name, either / (slash) or \ (backslash) can be used as the path separator. 1 to 255 characters	–
<b>Usage example</b>	<code>MMEM:LOAD:PROF "old settings.profile"</code>			
<b>Errors</b>	<code>-256,"File name not found"</code>			
<b>Related Commands</b>	<code>*RCL</code> <code>*SAV</code> <code>MEMory:STATe:RECall:AUTO</code> <code>MMEMory:STORe:PROFile</code>			

**5.11.14. MMEMory:LOCK****Syntax** `MMEMory:LOCK {<password>}`  
`MMEMory:LOCK?`

**Description** Use this command to enable write protection of SD card. All writing, deleting or modifying attempts on files or directories will generate an error if SD card is locked.

Parameters	Name	Type	Range	Default
	<password>	Quoted string	System password (4 to 16 characters)	–

**Return** This query returns 0 if SD card is unlocked, or 1 if SD card is locked.

**Usage example**  
 MMEM:LOCK "test123"  
 MMEM:LOCK?  
 1

**Errors** 122, "Invalid sys password"

**Related Commands** MMEemory:UNLock  
 SYSTem:PASSword:NEW

#### 5.11.15. MMEemory:MDIRectory

**Syntax** MMEemory:MDIRectory {<directory>}

**Description** This command creates a new directory. If directory=<path>, this command creates a <current directory>/<path> directory.

Parameters	Name	Type	Range	Default
	<directory>	Quoted string	Directory name, either / (slash) or \ (backslash) can be used as the path separator. 1 to 255 characters	–

**Usage example** MMEM:MDIR "test folder"

**Errors** -258, "Media protected"

**Related Commands** MMEemory:CDIRectory  
 MMEemory:RDIRectory

#### 5.11.16. MMEemory:MOVE

**Syntax** MMEemory:MOVE {<source>}, {<destination>}

**Description** This command moves or renames an existing file. If destination is a file name, the source file is renamed to the new file name in the current directory. When destination=<path>, the source file is moved to <current directory>/<path>.

Error occurs if the source file does not exist or the destination file already exists.

Parameters	Name	Type	Range	Default
	<source>	Quoted string	Source file name, 1 to 255 characters	–
	<destination>	Quoted string	New file name or directory name, either / (slash) or \ (backslash) can be used as the path separator. 1 to 255 characters	–

**Usage example** Rename file:  
 MMEM:MOVE "old name", "new name"  
 Move file from current directory to "/Test" directory:  
 MMEM:MOVE "new name", "/Test/new name"  
 Move and rename file:  
 MMEM:MOVE "/Test/new name", "/Documents/new doc"

**Errors** -256, "File name not found"  
 -258, "Media protected"

**Related Commands** MMEMory:CDIRectory  
 MMEMory:COPY

#### 5.11.17. MMEMory:MDIRectory

**Syntax** MMEMory:MDIRectory {<directory>}

**Description** This command creates a new directory. If directory=<path>, this command creates a <current directory>/<path> directory.

Parameters	Name	Type	Range	Default
	<directory>	Quoted string	Directory name, either / (slash) or \ (backslash) can be used as the path separator. 1 to 255 characters	–

**Usage example** MMEM:MDIR "test folder"

**Errors** -258, "Media protected"

**Related Commands** MMEMory:CDIRectory  
 MMEMory:RDIRectory

#### 5.11.18. MMEMory:STORe:LIST

**Syntax** MMEMory:STORe:LIST<n> {<filename>}

**Description** Store LIST program of channel defined with <n> to SD card location defined by <filename>. Stored file type will be TRAC. Default file extension is .list

Parameters	Name	Type	Range	Default
	<filename>	Quoted string	File name, either / (slash) or \ (backslash) can be used as the path separator. 1 to 255 characters	–

**Usage example** MMEM:STOR:LIST1 "DC\_DC conv testing.list"

**Errors** -258, "Media protected"

**Related Commands** MMEMory:LOAD:STATe  
 [SOURce[<n>]]:LIST:COUNT  
 [SOURce[<n>]]:LIST:CURREnt[:LEVel]  
 [SOURce[<n>]]:LIST:DWELl  
 [SOURce[<n>]]:LIST:VOLTage[:LEVel]

#### 5.11.19. MMEMory:STORe:PROFile

**Syntax** MMEMory:STORe:PROFile {<filename>}

**Description** This command stores the BB3 state (i.e. user profile 0) to SD card location defined by <filename>. Stored file type will be STAT. Default file extension is .conf

Parameters	Name	Type	Range	Default
	<filename>	Quoted string	File name, either / (slash) or \ (backslash) can be used as the path separator. 1 to 255 characters	–

**Usage example** MMEM:STOR:PROF "Both channels 5V\_3A.profile"

**Errors** -258, "Media protected"

**Related** \*RCL

**Commands** \*SAV  
 MEMory:STATe:RECall:AUTO  
 MMEMory:LOAD:PROFile

#### 5.11.20. MMEMory:TIME

**Syntax** MMEMory:TIME? {<filename>}

**Description** Returns the (hours, minute, seconds) that the specified file was last saved.

Parameters	Name	Type	Range	Default
	<filename>	Quoted string	File name or directory name, either / (slash) or \ (backslash) can be used as the path separator. 1 to 255 characters	—
<b>Return</b>	Comma-separated numbers (NR1) of hours, minute, seconds formatted as hh, mm, ss			
<b>Usage example</b>	MMEM:TIME? "test.002"  22, 10, 14			
<b>Errors</b>	-256, "File name not found"			
<b>Related Commands</b>	MMEMory:DATE			

#### 5.11.21. MMEMory:UNLock

**Syntax** MMEMory:UNLock {<password>}

**Description** Disable write protection of SD card activated by the MMEMory:LOCK command.

Parameters	Name	Type	Range	Default
	<password>	Quoted string	System password (4 to 16 characters)	—
<b>Usage example</b>	MMEM:LOCK?  1  MMEM:UNL "test123" MMEM:LOCK?  0			
<b>Errors</b>	122, "Invalid sys password"			
<b>Related Commands</b>	MMEMory:LOCK SYSTem:PASSword:NEW			

#### 5.11.22. MMEMory:UPLoad

**Syntax** MMEMory:UPLoad? {<filename>}

**Description** This command uploads the contents of a file from the instrument to the host computer. The format for <file> is "[<path>]<file\_name>", where <path> must be an absolute folder path. If <path> is omitted, the folder specified by the MMEMory:CDIRectory command is used. Absolute paths begin with a "\" or "/" and start at the root folder of SD card.

Parameters	Name	Type	Range	Default
	<filename>	Quoted string	File name or directory name, either / (slash) or \ (backslash) can be used as the path separator. 1 to 255 characters	—
<b>Return</b>	The query returns the file contents are returned as an IEEE 488.2 definite-length block.			
<b>Usage</b>	MMEM:UPL? "test file"			

## *EEZ BB3 SCPI reference*

**example**      #211Hello world  
**Errors**        -257,"File name error"  
**Related**       MMEMory:DOWNload:DATA  
**Commands**

## 5.12. OUTPut

The OUTPut subsystem controls the output state, coupling outputs and protections, protection clear and tracking state.

SCPI command	Description
OUTPut	
<a href="#">[:STATe] {&lt;bool&gt;}</a>	Controls the specified channel output state
<a href="#">TRIGGered {&lt;bool&gt;} [, &lt;chanlist&gt;]</a>	Controls channel output state with trigger
:DELay	
<a href="#">:DURation {&lt;duration&gt;} [, &lt;channel&gt;]</a>	Sets the output start delay duration
<a href="#">:DPRog {&lt;DprogState&gt;}</a>	Controls down-programmer circuit
<a href="#">:MODE?</a>	Returns the channel mode of operation
:PROTection	
<a href="#">:CLEar</a>	Resets latched protection
<a href="#">:COUPle {&lt;bool&gt;}</a>	Enables channel coupling for protection faults
<a href="#">:MEASure {&lt;bool&gt;}</a>	Enables measuring of output voltage before channel output is turned on
<a href="#">:TRACK[:STATe] {&lt;chanlist&gt;}</a>	Enables channels to operate in the track mode

### 5.12.1. OUTPut[:STATe]

**Syntax**     [OUTPut\[:STATe\] {<bool>} \[, <chanlist>\]](#)  
[OUTPut\[:STATe\]? \[<channel>\]](#)

**Description** This command enables or disables the specified output channel(s). The enabled state is ON (1); the disabled state is OFF (0). The state of a disabled output is a condition of zero output voltage and zero source current.  
Execution of this command also affects bit 10 (OE) and bit 11 (DP) of the Operation Instrument Summary register (see [Section 3.3.2](#)).  
Self-test operation initiated by [\\*TST?](#) command will put all BB3 modules into disable state.  
When channels are not coupled together (INSTrument:COUPle:TRACking) this command activate green OE LED indicator between output terminals.

Execution of the OUTP ON command on the channel which has one or more protection tripped (OCP, OVP, OPP or OTP) will generate error 201.  
Use OUTPut:PROTection:CLEar command to clear all tripped protections.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	–
	<chanlist>	Discrete ChannelList	ALL CH1 CH2 CH3 CH4 CH5 CH6	–

**Return**     The query command returns 0 if the output is OFF, and 1 if the output is ON.

**Usage example**     Set and check output on channel 1:

```
OUTP ON, CH1
OUTP? CH1
1
```

Set outputs on channels 1,2 and 4, and check all outputs state:

```
OUTP ON, (@1:2,4)
OUTP? ALL
```



	1,1,0,1
<b>Errors</b>	108,"Cal output disabled" 201,"Cannot execute before clearing protection"
<b>Related Commands</b>	*TST INSTrument:COUPle:TRACking OUTPut:PROTection:CLEar [SOURce:]VOLTage:SENSe[:SOURce]

### 5.12.2. OUTPut[:STATe]:TRIGgered

Syntax

OUTPut[:STATe]:TRIGgered {<bool>} [, <channel>]  
OUTPut[:STATe]:TRIGgered? [<channel>]

Description

This command programs the pending triggered channel's output state. The pending triggered output state is a stored value that is transferred to the output terminals when a trigger occurs.

Parameters

Name	Type	Range	Default
<bool>	Boolean	ON OFF 0 1	—
<channel>	Discrete	CH1 CH2 CH3 CH4 CH5 CH6	—

Return

Query the triggered output state presently programmed. If no triggered level is programmed, the OUTPut? state is returned.

Usage example

OUTP?  
0  
  
OUTP:TRIG ON  
TRIG:SOUR IMM  
INIT  
OUTP?  
1

Errors

108,"Cal output disabled"  
201,"Cannot execute before clearing protection"

Related Commands

\*TST  
OUTPut:PROTection:CLEar  
[SOURce:]VOLTage:SENSe[:SOURce]

### 5.12.3. OUTPut:DELAy:DURation

Syntax

OUTPut:DELAy:DURation {<duration>} [, <channel>]  
OUTPut:DELAy:DURation? [<channel>]

Description

This command sets the power output turn on delay in seconds on the specified output channel.

Parameters

Name	Type	Range	Default
<duration>	NR2	0.002 – 10	–
<channel>	Discrete	CH1 CH2 CH3 CH4 CH5 CH6	–

Return

The query command returns the programmed output turn on delay in seconds.

Usage example

OUTP:DEL:DUR 0.1

Related Commands

OUTPut:DELAy[:STATe]

### 5.12.4. OUTPut:DPRog

**Syntax**      `OUTPut:DPRog {<DprogState>}`  
`OUTPut:DPRog?`

**Modules**      **DCP**

**Description** A down-programmer is a circuit built into the channel's output of a power module that actively pulls the output voltage down when the programmed output voltage value is moving from a higher setting to a lower setting.  
 The down-programmer circuit is active by default and only rare situation requires to be turned off. One such situation is when battery is connected as a load. Another one is connecting two channel in parallel when only one down-programmer circuit is enough for actively pulls the output voltage down while lower voltage is set.  
 Execution of this command also affects bit 11 (DP) of the Operation Instrument Isummary register (see Section [Section 3.3.2](#)).

*Despite of the down-programmer state programmed by this command, it will be deactivated when the channel output is turned off (i.e. `OUTPut[:STATe] OFF`) after `DP_OFF_DELAY_PERIOD` in seconds). When the channel output is turned on again, down-programmer will be set back to the state programmed with this command.*

If negative output power (`DP_NEG_LEV`) is measured and last more then `DP_NEG_DELAY` seconds the down-programmer will be switched off and an error 500 will be generated.

Parameters	Name	Type	Range	Default
	<DprogState>	Discrete	ON OFF	ON
<b>Return</b>	This query returns 0 (OFF) if down-programmer is disabled, or 1 (ON) when down-programmer is enabled.			
<b>Usage example</b>	<code>OUTPut:DPR?</code> "ON"			
<b>Errors</b>	500, "Down-programmer on CH1 switched off" 501, "Down-programmer on CH2 switched off"			
<b>Related Commands</b>	<code>OUTPut[:STATe]</code>			

### 5.12.5. OUTPut:MODE?

**Syntax**      `OUTPut:MODE? [<channel>]`

**Description** This command simplify resolving a results that can be obtained reading the bit 8 (CV) and 9 (CC) of the read-only Instrument Isummary Operation Status register for a specific channel (see table in the [Section 3.3.2](#)). The power module can works in one of the three output modes:

- CV (Constant Voltage), when the output voltage equals the voltage setting value and the output current is determined by the load
- CC (Constant Current), when the output current equals the current setting value and the output voltage is determined by the load and
- UR (Unregulated) that is critical mode when neither CV nor CC mode is active

*The UR mode is not supported by DCM220 module and in [software simulator](#).*

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4  CH5 CH6	–
<b>Return</b>	The query returns CV, CC or UR.			
<b>Usage example</b>	Set output voltage to 20 V and max. current, check that output voltage is as defined that indicate the constant voltage mode of operation:			

```
VOLT 20; CURR MAX
MEAS:VOLT?
20.0
OUTP:MODE?
CV
```

**Related Commands** STATus:OPERation:INSTrument:ISUMmary[<n>][:EVENT]  
 STATus:OPERation:INSTrument:ISUMmary[<n>]:CONDition  
 STATus:OPERation:INSTrument:ISUMmary[<n>]:ENABle

### 5.12.6. OUTPut:PROTection:CLEar

**Syntax** OUTPut:PROTection:CLEar [<channel>]

**Description** This command clears the latched protection status that disables the output when an over-voltage, over-current or a power-limit condition is detected. All conditions that generate the fault must be removed before the latched status can be cleared. The output is restored to the state it was in before the fault condition occurred.

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4 CH5 CH6	–

**Usage example** The following command clears the latched protection status on all channels:  
 OUTP:PROT:CLE

**Related Commands** [SOURce[<n>]]:CURRent:PROTection:STATe  
 [SOURce[<n>]]:POWEr:PROTection:STATe  
 [SOURce[<n>]]:VOLTage:PROTection:STATe

### 5.12.7. OUTPut:PROTection:COUPle

**Syntax** OUTPut:PROTection:COUPle {<bool>}  
 OUTPut:PROTection:COUPle?

**Description** This command enables or disables output coupling for protection faults. When enabled, all output channels are disabled when a protection fault occurs on any output channel. When disabled, only the affected output channel is disabled when a protection fault is triggered.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	OFF

**Return** The query command returns 0 if the protection coupling is OFF (disabled), and 1 if the protection coupling is ON (enabled).

**Usage example** OUTP:PROT:COUP ON  
 OUTP:PROT:COUP?  
 1

**Related Commands** OUTPut:PROTection:CLEar

### 5.12.8. OUTPut:TRACk[:STATe]

**Syntax** OUTPut:TRACk[:STATe] {<chanlist>}  
 OUTPut:TRACk[:STATe]?

**Description** This command enables or disables two or more channels to operate in the track mode defined with the channel list. Once enabled, any change on any of the channel from the channel list will be applied to the whole channel list.

The OUTPut:TRACk OFF command returns all affected channels to the non-track mode.

A device-specific error 312 will be generated if this command is tried to executed when channels are in any of coupled mode.

The following commands will affect the both channel regardless of which channel is currently selected using the INSTrument[:SElect] or INSTrument:NSElect command):

- OUTPut:PROTection:CLEar
- [SOURce[<n>]]:CURRent, [SOURce[<n>]]:CURRent:STEP, [SOURce[<n>]]:CURRent:TRIGgered, [SOURce[<n>]]:CURRent:LIMit, [SOURce[<n>]]:CURRent:PROTection:DELay, [SOURce[<n>]]:CURRent:PROTection:STATE, [SOURce[<n>]]:CURRent:PROTection:TRIPped?, [SOURce[<n>]]:LIST:COUNT, [SOURce[<n>]]:LIST:CURRent, [SOURce[<n>]]:LIST:DWELI, [SOURce[<n>]]:LIST:VOLTage[:LEVel], [SOURce[<n>]]:POWER:LIMit, [SOURce[<n>]]:POWER:PROTection[:LEVel], [SOURce[<n>]]:POWER:PROTection:DELay[:TIME], [SOURce[<n>]]:POWER:PROTection:STATE, [SOURce[<n>]]:POWER:PROTection:TRIPped?, [SOURce[<n>]]:VOLTage, [SOURce[<n>]]:VOLTage:LIMit, [SOURce[<n>]]:VOLTage:STEP, [SOURce[<n>]]:VOLTage:TRIGgered, [SOURce[<n>]]:VOLTage:MODE, [SOURce[<n>]]:VOLTage:PROTection[:LEVel], [SOURce[<n>]]:VOLTage:PROTection:DELay, [SOURce[<n>]]:VOLTage:PROTection:STATE, [SOURce[<n>]]:VOLTage:PROTection:TRIPped?
- TRIGger[:SEquence][:IMMediate], TRIGger[:SEquence]:DELay, TRIGger[:SEquence]:SLOPe, TRIGger[:SEquence]:SOURce

The following channel specific commands will generate a device-specific error 313 when channels are in tracking mode:

- CALibrate,
- INSTrument:COUPle:TRACking,
- SIMULator:RPOL, SIMULator:VOLTage:PROGram,
- [SOURce[<n>]]:VOLTage:PROGram,

At \*RST, the tracking mode is disabled.

Parameters	Name	Type	Range	Default
	<chanlist>	Boolean ChannelList	List of channels, OFF 0	OFF
<b>Return</b>	Query the tracking mode state of the power modules. The returned value is 0 (OFF) or 1 (ON).			
<b>Usage example</b>	Define track group that contains channel 1, 3 and 4:			
	<pre>OUTP:TRAC (@1,3:4) VOLT 12 MEAS? 12.00 MEAS? CH3 12.00</pre>			
<b>Errors</b>	312,"Cannot execute when the channels are coupled" 313,"Cannot execute in tracking mode"			
<b>Related Commands</b>	*RST INSTrument:COUPle:TRACking			

### 5.13. ROUTe

The ROUTe command subsystem controls switching operations for relay modules such as SMX46 and PREL6 and defines channel labels.

*Please note that when list of channels separated with comma is specified no space is allowed. For example (@111,121) instead of (@111, 121) or (@111 , 121).*

SCPI command	Description
ROUTe	
:CHANnel	
:LABel {<label>},{<chanlist>}	Sets labels for relay matrix channels
:CLOSe {<chanlist>}	Closes (turns on) specified relay matrix channels
:OPeN {<chanlist>}	Opens (turns off) specified relay matrix channels

#### 5.13.1. ROUTe:CLOSe

**Syntax** ROUTe:CLOSe {<chanlist>  
ROUTe:CLOSe? {<chanlist>}

**Modules** SMX PREL

**Description** This command closes (turns it on) the relay matrix channels specified by <chanlist>. <chanlist> has the form (@nrc) where *n* is module slot number, *r* is channel row number and *c* is column number.

Parameters	Name	Type	Range	Default
	<chanlist>	ChannelList	SMX: (@n03) and (@n11) to (@n46), PREL: (@n01) to (@n05) where n is a slot number	–

**Return** Returns the current state of the channel(s) queried. The command returns "1" if channel(s) are closed (relay is energized) or returns "0" if channel(s) are open (relay is idle).

**Usage example** Close (activate) four crosspoints (X1Y1, X2Y1, X3Y1, X4Y1) at once on the SMX46 inserted into slot 1:

```
ROUT:CLOS (@111,112,113,114)
```

Close all crosspoints in row 3 (X3Y1 to X3Y6) at once on the SMX46 inserted into slot 1:

```
ROUT:CLOS (@113:163)
```

Close power relay on SMX46 inserted in slot 2:

```
ROUT:CLOS (@203)
```

**Errors** -200, "Execution error"

**Related Commands** ROUTe:OPeN  
SYSTem:RELAy:CYCLes?

#### 5.13.2. ROUTe:CHANnel:LABel

**Syntax** ROUTe:CHANnel:LABel: {label}, {<chanlist>  
ROUTe:CHANnel:LABel? {<chanlist>}

**Modules** SMX PREL

**Description** Use this command to define the column label of the relay matrix channels specified by <row>.

Parameters	Name	Type	Range	Default
------------	------	------	-------	---------

	<label>	Quoted string	Max. 5 characters	–
	<chanlist>	ChannelList	SMX: (@n03) and (@n11) to (@n46), PREL: (@n01) to (@n05) where n is a slot number	–
<b>Return</b>	Returns the label of the channel queried.			
<b>Usage example</b>	<p>Set label “AC in” for the switch matrix row 1 on the SMX46 inserted into slot 1:</p> <pre>ROUT:CHAN:LAB "AC in", (@111)</pre> <p>Set label “Master” for the power relay on the SMX46 inserted into slot 1:</p> <pre>ROUT:CHAN:LAB "Master", (@103)</pre>			
<b>Errors</b>	-200, "Execution error"			
<b>Related Commands</b>	ROUTe:CLOSe ROUTe:OPEN			

### 5.13.3. ROUTe:OPEN

<b>Syntax</b>	ROUTe:OPEN {<chanlist>} ROUTe:OPEN? {<chanlist>}			
<b>Modules</b>	SMX PREL			
<b>Description</b>	<p>This command closes (turns it off) the relay matrix channels specified by &lt;chanlist&gt;. &lt;chanlist&gt; has the form (@nrc) where <i>n</i> is module slot number, <i>r</i> is channel row number and <i>c</i> is column number.</p> <p>The *RST command will open all channels of all modules.</p>			
<b>Parameters</b>	Name	Type	Range	Default
	<chanlist>	ChannelList	SMX: (@n03) and (@n11) to (@n46), PREL: (@n01) to (@n05) where n is a slot number	–
<b>Return</b>	Returns the current state of the channel(s) queried. The command returns "1" if channel(s) are open (relay is idle) or returns "0" if channel(s) are closed (relay is energized).			
<b>Usage example</b>	<p>Open (deactivate) two crosspoints (X1Y3 and X4Y2) at once on the SMX46 inserted into slot 1:</p> <pre>ROUT:OPEN (@131,124)</pre> <p>Open two sequence of crosspoints (X2Y2 to X4Y2 and X2Y4 to X4Y4, i.e. six crosspoints in total) at once on the SMX46 inserted into slot 1:</p> <pre>ROUT:OPEN (@122:124,142:144)</pre> <p>Open power relay on SMX46 inserted in slot 2:</p> <pre>ROUT:OPEN (@203)</pre>			
<b>Errors</b>	-200, "Execution error"			
<b>Related Commands</b>	*RST ROUTe:CLOSe			

## 5.14. SENSE

The SENSE control the current measurement range, energy counting/window, the data acquisition sequence and DLOG viewer parameters.

SCPI command	Description
SENSe	
:CURRent	
[:DC]	
RANGe[:UPPer] {<range>}	Selects a DC current measurement range
:DLOG	
:FUNCTion	
:CURRent {<bool>}, {<channel>}	Enables/disables output current internal data logging
:POWer {<bool>}, {<channel>}	Enables/disables output power internal data logging
:VOLTagE {<bool>}, {<channel>}	Enables/disables output voltage internal data logging
:PERiod {<time>}	Sets the sample period for internal data logging
:TIME {<time>}	Sets the sample duration for internal data logging
:TRACe	
[:DATA] {<value>}[, ...]	Adds new data into log file
:REMark {<user remark>}	Adds user description to the log file
:X	
:LABel {<label>}	Sets X-axis label
[:RANGe]:MIN {<min>}	Sets X-axis min. value
[:RANGe]:MAX {<max>}	Sets X-axis max. value
SCALE	
:STEP {<step>}	Sets X-axis step value
:UNIT {<unit>}	Sets X-axis units
:Y<n>	
:LABel {<label>}	Sets Y-axis<n> label
[:RANGe]:MIN {<min>}	Sets Y-axis<n> min. value
[:RANGe]:MAX {<max>}	Sets Y-axis<n> max. value
:SCALE	
:UNIT {<unit>}	Sets Y-axis<n> units

### 5.14.1. SENSE:CURRent[:DC]:RANGe[:UPPer]

**Syntax** SENSE:CURRent[:DC]:RANGe[:UPPer] {<range>}  
SENSe:CURRent[:DC]:RANGe[:UPPer]?

**Modules** DCP

**Description** This command selects a DC current measurement range when Power board with multiple current ranges is installed (e.g. DCP405 that can be find out with the SYSTem:CHANnel:MODEl? command). The entered value must be higher than the maximum current that you expect to measure. Units are in amperes.

Parameters	Name	Type	Range	Default
	<range>	NR2 Discrete	0.5, 5, LOW HIGH BEST	BEST

**Usage example**     `SENS:CURR:RANG?`  
                          `0.05`

**Errors**            `-241,"Hardware missing"`

**Related Commands**   `SYSTem:CHANnel:MODEl?`

#### 5.14.2. SENSE:DLOG:FUNCTION:CURRENT

**Syntax**            `SENSe:DLOG:FUNCTION:CURRENT {<bool>}, {<channel>}`  
                          `SENSe:DLOG:FUNCTION:CURRENT? {<channel>}`

**Description**     Use this command to enable or disable output current internal data logging on the specified channel.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	OFF
	<channel>	Discrete	CH1 CH2 CH3 CH4  CH5 CH6	–

**Return**            The query command returns the status (0 or 1) of the output current internal data logging enabled on the specified channel

**Usage example**     `SENS:DLOG:FUNC:CURR ON, CH1`

**Related Commands**   `INIT:DLOG`  
                          `SENSe:DLOG:FUNCTION:POWer`  
                          `SENSe:DLOG:FUNCTION:VOLTage`  
                          `SENSe:DLOG:PERiod`  
                          `SENSe:DLOG:TIME`

#### 5.14.3. SENSE:DLOG:FUNCTION:POWER

**Syntax**            `SENSe:DLOG:FUNCTION:POWER {<bool>}, {<channel>}`  
                          `SENSe:DLOG:FUNCTION:POWER? {<channel>}`

**Description**     Use this command to enable or disable output power internal data logging on the specified channel.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	OFF
	<channel>	Discrete	CH1 CH2 CH3 CH4  CH5 CH6	–

**Return**            The query command returns the status (0 or 1) of the output power internal data logging enabled on the specified channel

**Usage example**     `SENS:DLOG:FUNC:VOLT OFF, CH2`

**Related Commands**   `INIT:DLOG`  
                          `SENSe:DLOG:FUNCTION:CURRENT`  
                          `SENSe:DLOG:FUNCTION:VOLTage`  
                          `SENSe:DLOG:PERiod`  
                          `SENSe:DLOG:TIME`

#### 5.14.4. SENSE:DLOG:FUNCTION:VOLTAGE

**Syntax**            `SENSe:DLOG:FUNCTION:VOLTage {<bool>}, {<channel>}`  
                          `SENSe:DLOG:FUNCTION:VOLTage? {<channel>}`

**Description**     Use this command to enable or disable output voltage internal data logging on the specified channel.



Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	OFF
	<channel>	Discrete	CH1 CH2 CH3 CH4 CH5 CH6	–
<b>Return</b>	The query command returns the status (0 or 1) of the output voltage internal data logging enabled on the specified channel			
<b>Usage example</b>	SENS:DLOG:FUNC:VOLT OFF, CH2			
<b>Related Commands</b>	INIT:DLOG SENSe:DLOG:FUNCtion:CURRent SENSe:DLOG:FUNCtion:POWEr SENSe:DLOG:PERiod SENSe:DLOG:TIME			

#### 5.14.5. SENSe:DLOG:PERiod

<b>Syntax</b>	SENSe:DLOG:PERiod {<time>} SENSe:DLOG:PERiod?			
<b>Description</b>	Use this command to set the data logging sample period in seconds.			
Parameters	Name	Type	Range	Default
	<time>	NR2	0.005 – 120	0.02
<b>Return</b>	The query command returns the sample period in seconds.			
<b>Usage example</b>	SENS:DLOG:PER 1			
<b>Related Commands</b>	INITiate:DLOG SENSe:DLOG:TIME			

#### 5.14.6. SENSe:DLOG:TIME

<b>Syntax</b>	SENSe:DLOG:TIME {<time>} SENSe:DLOG:TIME?			
<b>Description</b>	This command sets the data logging sample duration, the entered value is in seconds. For sample duration, the maximum is 86 400 000 seconds or 24 000 hours or 1000 days (depending of the available space on the installed SD Card, see the MMEMory:INFORmation? command).			
Parameters	Name	Type	Range	Default
	<time>	NR1	1 – 86400000	60
<b>Return</b>	The query command returns the sample duration in seconds.			
<b>Usage example</b>	SENS:DLOG:TIME 3600			
<b>Related Commands</b>	INITiate:DLOG MMEMory:INFORmation? SENSe:DLOG:PERiod			

#### 5.14.7. SENSe:DLOG:TRACe[:DATA]

<b>Syntax</b>	SENSe:DLOG:TRACe[:DATA] {<value>}[, ...]			
<b>Description</b>	When the X-axis and all Y-axes is defined, and when the DLOG file name is specified, it remains to add points (data) into DLOG. Use this command to do this as many times as needed. The first time this command is called, the Y-axis values are set for the first point (as many <value> parameters as the Y-axis should be set), the second time for the second point, etc.			

When the values of all points are defined, execute the ABORT:DLOG command and the creation of the DLOG file is complete.

Parameters	Name	Type	Range	Default
	<value>	NR2	Y:RANGe:MINimum to Y:RANGe:MAXimum	–
<b>Usage example</b>	SENS:DLOG:TRAC 10.2, 33.34, 0.022			
<b>Related Commands</b>	ABORT:DLOG			

#### 5.14.8. SENSE:DLOG:TRACe:X:LABel

**Syntax** SENSE:DLOG:TRACe:X:LABel {<label>}  
SENSe:DLOG:TRACe:X:LABel?

**Description** Use this command to set DLOG X-axis label name.

Parameters	Name	Type	Range	Default
	<label>	Quoted string	0 to 32 characters	–
<b>Return</b>	The query returns X-axis label name as quoted string.			
<b>Usage example</b>	SENS:DLOG:TRAC:X:LAB "Imon"			
<b>Related Commands</b>	SENSe:DLOG:TRACe:X:STEP SENSe:DLOG:TRACe:X:UNIT			

#### 5.14.9. SENSE:DLOG:TRACe:REMark

**Syntax** SENSE:DLOG:TRACe:REMark {<user remark>}  
SENSe:DLOG:TRACe:REMark?

**Description** Use this command to add description to the log file.

Parameters	Name	Type	Range	Default
	<user remark>	Quoted string	0 to 128 characters	–
<b>Return</b>	The query command returns log file description as a quoted string.			
<b>Usage example</b>	SENS:DLOG:TRAC:REM "Test data log"			
<b>Related Commands</b>	INITiate:DLOG INITiate:DLOG:TRACe SENSe:DLOG:TRACe[:DATA]			

#### 5.14.10. SENSE:DLOG:TRACe:X:UNIT

**Syntax** SENSE:DLOG:TRACe:X:UNIT {<unit>}  
SENSe:DLOG:TRACe:X:UNIT?

**Description** This command specifies DLOG X-axis unit, typically this is the time, or SEConD.

Parameters	Name	Type	Range	Default
	<unit>	Discrete	VOLT AMPEr WATT  JOULe SECOnd  OHM FARAd HERTZ	–
<b>Return</b>	The query command returns log file X-axis units name as quoted string.			
<b>Usage example</b>	SENS:DLOG:TRAC:X:UNIT VOLT			

**Related Commands** SENSE:DLOG:TRACe:X:STEP  
SENSe:DLOG:TRACe:X:LABeI

#### 5.14.11. SENSE:DLOG:TRACe:X:STEP

**Syntax** SENSE:DLOG:TRACe:X:STEP {<step>}  
SENSe:DLOG:TRACe:X:STEP?

**Description** Use this command to set step or  $\Delta x$ , i.e. the distance between two points on the X-axis of the DLOG.

Parameters	Name	Type	Range	Default
	<step>	NR2	Greater then zero	—

**Return** The query command returns X-axis step value.

**Usage example** SENS:DLOG:TRAC:X:STEP 0.1

**Related Commands** SENSE:DLOG:TRACe:X:LABeI  
SENSe:DLOG:TRACe:X:UNIT

#### 5.14.12. SENSE:DLOG:TRACe:X[:RANGe]:MIN

**Syntax** SENSE:DLOG:TRACe:X[:RANGe]:MIN {<min>}  
SENSe:DLOG:TRACe:X[:RANGe]:MIN?

**Description** Use this command to set the first point of the DLOG X-axis. The next point has the value MIN + STEP, then MIN + 2 \* STEP, etc.

Parameters	Name	Type	Range	Default
	<min>	NR2	—	—

**Return** The query command returns set X-axis minimal value.

**Usage example** SENS:DLOG:TRAC:X:MIN 0

**Related Commands** SENSE:DLOG:TRACe:X:UNIT

#### 5.14.13. SENSE:DLOG:TRACe:X[:RANGe]:MAX

**Syntax** SENSE:DLOG:TRACe:X[:RANGe]:MAX {<max>}  
SENSe:DLOG:TRACe:X[:RANGe]:MAX?

**Description** Optional command to set the value of the last point on the DLOG X-axis.

Parameters	Name	Type	Range	Default
	<max>	NR2	—	—

**Return** The query command returns set X-axis maximum value.

**Usage example** SENS:DLOG:TRAC:X:MAX 20

**Related Commands** SENSE:DLOG:TRACe:X:UNIT

#### 5.14.14. SENSE:DLOG:TRACe:X:SCALE

**Syntax** SENSE:DLOG:TRACe:X:SCALE {<scale>}  
SENSe:DLOG:TRACe:X:SCALE?

**Description** Use this command to set the X-axis scale.

*Currently, only linear (LInear) scale is supported.*

Parameters	Name	Type	Range	Default
	<scale>	Discrete	LINear LOGarithmic	–
<b>Return</b>	The query command returns X-axis scale type.			
<b>Usage example</b>	SENS:DLOG:TRAC:X:SCAL LIN			
<b>Related Commands</b>	SENSe:DLOG:TRACe:Y:SCALe			

#### 5.14.15. SENSE:DLOG:TRACe:Y<n>:LABel

<b>Syntax</b>	SENSe:DLOG:TRACe:Y<n>:LABel {<label>} SENSe:DLOG:TRACe:Y<n>:LABel?			
<b>Description</b>	Use this command to set DLOG Y-axis label name. DLOG file could contains up to 18 Y-axis defined by <n>.			
Parameters	Name	Type	Range	Default
	<label>	Quoted string	0 to 32 characters	–
<b>Return</b>	The query returns label name as quoted string of the selected Y-axis.			
<b>Usage example</b>	SENS:DLOG:TRAC:Y1:LAB "Umon"			
<b>Related Commands</b>	SENSe:DLOG:TRACe:Y<n>:UNIT			

#### 5.14.16. SENSE:DLOG:TRACe:Y<n>[:RANGe]:MIN

<b>Syntax</b>	SENSe:DLOG:TRACe:Y<n>[:RANGe]:MIN {<min>} SENSe:DLOG:TRACe:Y<n>[:RANGe]:MIN?			
<b>Description</b>	Use this command to set the first point of the <n> Y-axis. The next point has the value MIN + STEP, then MIN + 2 * STEP, etc.			
Parameters	Name	Type	Range	Default
	<min>	NR2	–	–
<b>Return</b>	The query command returns set <n> Y-axis minimal value.			
<b>Usage example</b>	SENS:DLOG:TRAC:Y1:MIN 1			
<b>Related Commands</b>	SENSe:DLOG:TRACe:Y<n>[:RANGe]:MAX SENSe:DLOG:TRACe:Y<n>:UNIT			

#### 5.14.17. SENSE:DLOG:TRACe:Y<n>[:RANGe]:MAX

<b>Syntax</b>	SENSe:DLOG:TRACe:Y<n>[:RANGe]:MAX {<max>} SENSe:DLOG:TRACe:Y<n>[:RANGe]:MAX?			
<b>Description</b>	Optional command to set the value of the last point on the <n> Y-axis.			
Parameters	Name	Type	Range	Default
	<max>	NR2	–	–
<b>Return</b>	The query command returns set <n> Y-axis maximum value.			
<b>Usage example</b>	SENS:DLOG:TRAC:Y1:MAX 50			
<b>Related Commands</b>	SENSe:DLOG:TRACe:Y<n>[:RANGe]:MIN SENSe:DLOG:TRACe:Y<n>:UNIT			

**5.14.18.   SENSe:DLOG:TRACe:Y:SCALe**

**Syntax**       SENSe:DLOG:TRACe:Y:SCALe {<scale>}  
 SENSe:DLOG:TRACe:Y:SCALe?

**Description** Use this command to set the Y-axis scale.

Parameters	Name	Type	Range	Default
	<scale>	Discrete	LINear LOGarithmic	—

**Return**       The query command returns Y-axis scale type.

**Usage example**   SENS:DLOG:TRAC:Y:SCAL LIN

**Related Commands**   SENSe:DLOG:TRACe:X:SCALe

**5.14.19.   SENSe:DLOG:TRACe:Y<n>:UNIT**

**Syntax**       SENSe:DLOG:TRACe:Y<n>:UNIT {<unit>}  
 SENSe:DLOG:TRACe:Y<n>:UNIT?

**Description** This command specifies DLOG <n> Y-axis unit.

Parameters	Name	Type	Range	Default
	<unit>	Discrete	VOLT AMPER WATT  JOULE SECOnd  OHM FARAd HERTZ	—

**Return**       The query command returns <n> Y-axis units name as quoted string.

**Usage example**   SENS:DLOG:TRAC:Y1:UNIT "AMPE"

**Related Commands**   SENSe:DLOG:TRACe:Y<n>:LABeI  
 SENSe:DLOG:TRACe:Y<n>:UNIT

## 5.15. SOURce

The SOURce commands are used to set the output voltage and current values, remote voltage sensing, and implemented protection mechanisms on the specified channel. Although the [APPLy](#) command provides the most straightforward method to program the BB3 over the remote interfaces, the SOURce commands give you more flexibility to change individual parameters.

SCPI command	Description
[SOURce[<n>]]	
:CURRent	
[:LEVel]	
[:IMMediate][:AMPLitude] {<current>}	Sets the output current
:STEP[:INCRement] {<step>}	Sets the step of the current change
:TRIGgered[:AMPLitude] {<current>}	Sets the triggered output current
:LIMit	
[:POSitive][:IMMediate][:AMPLitude] {<current>}	Sets the output current limit
:MODE {<mode>}	Sets the current trigger mode
:PROTection	
:DELay	
[:TIME] {<time>}	Sets the over-current protection (OCP) programming delay
:STATe {<bool>}	Enables/disables over-current protection on the selected channel
:TRIPped?	Returns status of over-current protection activation
:RAMP:DURation {<duration>}	Sets output current ramp duration
:LIST	
:COUNT	Sets the number of times that the list is executed
:CURRent[:LEVel]	Specifies the current setting for each list step
:DWELl	Specifies the dwell time for each list step
:VOLTage[:LEVel]	Specifies the voltage setting for each list step
:POWeR	
:LIMit {<power>}	Sets the output power limit
:PROTection[:LEVel]	Sets the over-power protection (OPP) level
:DELay	
[:TIME] {<time>}	Sets the over-power protection programming delay
:STATe {<bool>}	Enables/disables over-power protection on the selected channel
:TRIPped?	Returns status of over-power protection activation
:VOLTage	
[:LEVel]	
[:IMMediate][:AMPLitude] {<voltage>}	Sets the output voltage
:STEP[:INCRement] {<step>}	Sets the step of the voltage change
:TRIGgered[:AMPLitude] {<voltage>}	Sets the triggered output voltage
:LIMit	
[:POSitive][:IMMediate][:AMPLitude]	Sets the output voltage limit

<a href="#">{&lt;voltage&gt;}</a>	
<a href="#">:MODE {&lt;mode&gt;}</a>	Sets the voltage trigger mode
<a href="#">:PROGram[:SOURce] {&lt;source&gt;}</a>	Sets voltage programming source
<a href="#">:PROtection[:LEVel]</a>	Sets the over-voltage protection (OVP) level
<a href="#">:DELay</a>	
<a href="#">[:TIME] {&lt;time&gt;}</a>	Sets the over-voltage protection (OVP) programming delay
<a href="#">:STATe {&lt;bool&gt;}</a>	Enables/disables over-voltage protection on the selected channel
<a href="#">:TRIPped?</a>	Returns status of over-voltage protection activation
<a href="#">:TYPE {&lt;type&gt;}</a>	Selects the over-voltage protection (OVP) type
<a href="#">:RAMP:DURation {&lt;duration&gt;}</a>	Sets output voltage ramp duration
<a href="#">:SENSe[:SOURce] {&lt;source&gt;}</a>	Sets voltage sense inputs source

### 5.15.1. [SOURce[<n>]]:CURRent

**Syntax** [\[SOURce\[<n>\]\]:CURRent\[:LEVel\]\[:IMMediate\]\[:AMPLitude\] {<current>}](#)  
[\[SOURce\[<n>\]\]:CURRent\[:LEVel\]\[:IMMediate\]\[:AMPLitude\]? \[<query current>\]](#)

**Modules** **DCP** **DCM**

**Description** This command sets the immediate current level of the channel. Units are in amperes. When [:SOURce[<n>]] or [<n>] is omitted, the currently selected channel will be affected by this command.

This command also increases or decreases the immediate current level using the 'UP' or DOWN parameter by a predetermined amount. The command CURRent:STEP sets the amount of increase or decrease. A new increment setting will *not* cause an execution error -222, "Data out of range" when the maximum or the minimum rated current is exceeded – the output value will be set to the maximum or the minimum value instead.

At [\\*RST](#), the signal being sourced will be set to a "safe" condition. This is achieved by setting the amplitude to its MINimum value (see [Section 8.1](#)).

Parameters	Name	Type	Range	Default
	<current>	NR2 Discrete	0 to MAXimum, MIN DEF MAX UP DOWN The maximum value is dependent on the module current rating. See <a href="#">Section 8.1</a>	–
	<query current>	Discrete	MIN DEF MAX	–

**Return** The query command returns the programmed current level. CURR? MIN, CURR? DEF and CURR? MAX can be used to obtain minimum, default and maximum current level on the currently selected channel. For actual output current value use MEASure:CURRent command.

**Usage example** A 10  $\Omega$  load is connected and voltage is set to 20 V. With MAX current set measured current will be 2 A. When new current value is set to 1.2 A, voltage will drop to 12 V (the channel enters the CC mode of operation):

```
INST CH1
VOLT 20
CURR MAX
MEAS:VOLT?
20.00
```

```
CURR 1.2
MEAS:VOLT?
12.00
```

Query that returns maximum current of the currently selected channel:

```
CURR? MAX
5.00
```

**Errors** 150, "Power limit exceeded"  
-222, "Data out of range"

**Related Commands** \*SAV  
\*RST  
APPLY  
MEASure[:SCALar]:CURRent[:DC]?  
[SOURce[<n>]]:CURRent[:LEVel][:IMMediate]:STEP[:INCRement]

### 5.15.2. [SOURce[<n>]]:CURRent:STEP

**Syntax** [SOURce[<n>]]:CURRent[:LEVel][:IMMediate]:STEP[:INCRement] {<step>}  
[SOURce[<n>]]:CURRent[:LEVel][:IMMediate]:STEP[:INCRement]? [<query step>]

**Description** Set the step of the current change of the channel. When [SOURce[<n>]] or [<n>] is omitted, the currently selected channel will be affected by this command. Step change is performed by using UP and DOWN as parameter for the [SOURce[<n>]]:CURRent command.

Parameters	Name	Type	Range	Default
	<step>	NR2 Discrete	0.01 to 1 DEFault	0.05
	<query step>	Discrete	DEFault	–

**Return** The query returns the step of the current change of the specified channel.

**Usage example** Return default step value:

```
CURR:STEP? DEF
0.05
```

When a 10  $\Omega$  load is connected with voltage set to 20 V and current to 1 A the first channel enters CC mode of operation. Current is then increased from 1 A in two steps to 1.2 A:

```
APPL CH1, 20, 1
MEAS:VOLT?
10.00
CURR:STEP 0.1
CURR UP
MEAS:CURR?
1.10
CURR UP
MEAS:CURR?
1.20
MEAS:VOLT?
12.00
```

**Related Commands** \*SAV  
[SOURce[<n>]]:CURRent



### 5.15.3. [SOURce[<n>]]:CURRent:TRIGgered

**Syntax** [SOURce[<n>]]:CURRent[:LEVel]:TRIGgered[:AMPLitude] {<current>}  
[SOURce[<n>]]:CURRent[:LEVel]:TRIGgered[:AMPLitude]? [<query current>]

**Modules** **DCP**

**Description** This command programs the pending triggered current level. The pending triggered current level is a stored value that is transferred to the output terminals when a trigger occurs.

*A pending triggered level is not affected by subsequent CURRent commands.*

Parameters	Name	Type	Range	Default
	<current>	NR2 Discrete	0 to maximum, MIN DEF MAX The MAXimum value is dependent on the module current rating. See <a href="#">Section 8.1</a>	0.00
	<query current>	Discrete	MIN DEF MAX	—

**Return** Query the triggered current level presently programmed. If no triggered level is programmed, the CURRent level is returned. CURR:TRIG? MIN and CURR:TRIG? MAX return the lowest and highest programmable triggered current levels.

**Usage example** On the currently selected channel voltage will be set to 3.3 V and current to 1 A when INITiate command is executed:

```
VOLT:TRIG 3.3
CURR:TRIG 1
TRIG:SOUR IMM
INIT
```

**Related Commands** ABORt

### 5.15.4. [SOURce[<n>]]:CURRent:LIMit[:POSitive][:IMMediate][:AMPLitude]

**Syntax** [SOURce[<n>]]:CURRent:LIMit[:POSitive][:IMMediate][:AMPLitude] {<current>}  
[SOURce[<n>]]:CURRent:LIMit[:POSitive][:IMMediate][:AMPLitude]? [<query current>]

**Description** This command sets the channel's output current limit. Units are in amperes. When [:SOURce[<n>]] or [<n>] is omitted, the currently selected channel will be affected by this command. This command could be used as mean of protection against incidental programming of too high output current that can be set for example with [SOURce[<n>]]:CURRent:STEP command.

*Maximum limit value will be affected with detected failure of cooling fan or channel's temperature sensor. Use CURR:LIM? MAX to query actual value.*

Parameters	Name	Type	Range	Default
	<current>	NR2 Discrete	0 to MAXimum, MIN DEF MAX The MAXimum value is dependent on the module current rating. See <a href="#">Section 8.1</a>	MAXimum
	<query current>	Discrete	MIN DEF MAX	—

**Return** The query returns the max. allowed output current of the specified channel. Querying MAX on an output channel returns the maximum current limit.

**Usage example** Normal mode of operation for 5 A module:

```
CURR:LIM MAX?
```

5.00

Max. allowable current with faulty cooling fan or temperature sensor:

CURR:LIM MAX?

1.00

**Related Commands** \*SAV  
 [SOURce[<n>]]:CURRent:STEP  
 [SOURce[<n>]]:POWer:LIMit  
 [SOURce[<n>]]:VOLTagE:LIMit[:POSitive][:IMMediate][:AMPLitude]

#### 5.15.5. [SOURce[<n>]]:CURRent:MODE

**Syntax** [SOURce[<n>]]:CURRent:MODE {<mode>}  
 [SOURce[<n>]]:CURRent:MODE?

**Modules** **DCP**

**Description** This command determines what happens to the output current when the transient system is initiated and triggered:

- FIXed – the output current remains at the immediate value
- LIST – the output follows the list values when a trigger occurs.
- STEP – the output goes to the triggered level when a trigger occurs.

Parameters	Name	Type	Range	Default
	<mode>	Discrete	FIXed LIST STEP	FIXed
<b>Return</b>	The query command returns the current mode of the currently selected channel.			
<b>Usage example</b>	CURR:MODE? FIX			
<b>Related Commands</b>	[SOURce[<n>]]:LIST:CURRent[:LEVel] [SOURce[<n>]]:VOLTagE:MODE			

#### 5.15.6. [SOURce[<n>]]:CURRent:PROTection:DELaY[:TIME]

**Syntax** [SOURce[<n>]]:CURRent:PROTection:DELaY[:TIME] {<time>}  
 [SOURce[<n>]]:CURRent:PROTection:DELaY[:TIME]? [<query time>]

**Description** This command sets the over-current protection delay. The over-current protection function will not be triggered on the selected output channel during the delay time. After the delay time has expired, the over-current protection function will be active. This prevents momentary changes in output status from triggering the over-current protection function. Programmed values can range from 0 to 10 seconds. See also [Section 8.1](#)

Parameters	Name	Type	Range	Default
	<time>	NR2 Discrete	0 – 10 DEFAult	20 ms
	<query time>	Discrete	DEFAult	–
<b>Return</b>	The query command returns the programmed delay time.			
<b>Usage example</b>	Get default OCP delay of 20 milliseconds: CURR:PROT:DEL? DEF 0.02			
<b>Related Commands</b>	*SAV OUTPut:PROTection:CLEar			

**5.15.7. [SOURCE[<n>]]:CURRENT:PROTECTION:STATE**

**Syntax** [SOURCE[<n>]]:CURRENT:PROTECTION:STATE {<bool>}  
[SOURCE[<n>]]:CURRENT:PROTECTION:STATE?

**Description** This command enables or disables the over-current protection (OCP) function. The enabled state is ON (1); the disabled state is OFF (0).  
Since the power modules do not have a dedicated over-current protection circuit that can be programmed independently of output current level, entering the CC (constant current) mode of operation is used as a trigger to start OCP sequence. When delay time specified with the [SOURCE[<n>]]:CURRENT:PROTECTION:DELAY[:TIME] command expired the output turns off and the Questionable Condition status register OCP bit 9 is set. An error tone will also follow if beeper is enabled (see SYSTem:BEEPer:STATE).  
[SOURCE[<n>]]:CURRENT:PROTECTION:TRIPPed? command can be used to query whether over-current protection occurred on the selected channel.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	OFF
<b>Return</b>	The query command returns 0 if the current protection state is OFF, and 1 if the current protection state is ON.			
<b>Usage example</b>	CURR:PROT:STAT? 0			
<b>Related Commands</b>	*SAV OUTPut:PROTECTION:CLEar OUTPut:PROTECTION:COUPLe [SOURCE[<n>]]:CURRENT:PROTECTION:DELAY[:TIME] [SOURCE[<n>]]:CURRENT:PROTECTION:TRIPPed SYSTem:BEEPer:STATE			

**5.15.8. [SOURCE[<n>]]:CURRENT:PROTECTION:TRIPPed?**

**Syntax** [SOURCE[<n>]]:CURRENT:PROTECTION:TRIPPed?

**Description** Query whether OCP occurred on the currently selected channel. When protection is tripped bit 9 (OCP) of the Questionable Instrument Isummary register will be set (see [Section 3.4.2](#)).  
The [OUTPut:PROTECTION:CLEar](#) command can be send to clear OCP condition on the selected channel.

**Return** This query returns a 1 if the protection circuit is tripped and a 0 if it is untripped.

**Usage example** CURR:PROT:TRIP?  
1

**Related Commands** OUTPut:PROTECTION:CLEar

**5.15.9. [SOURCE[<n>]]:CURRENT:RAMP:DURATION**

**Syntax** [SOURCE[<n>]]:CURRENT:RAMP:DURATION {<duration>}  
[SOURCE[<n>]]:CURRENT:RAMP:DURATION?

**Description** Use this command to configures a constant rise of output current limit on the channel <n> within a set duration. The output current limit can be increased continuously within a 0 to 10 seconds with 1 ms step size.

Parameters	Name	Type	Range	Default
	<duration>	NR2	0 – 10	–
<b>Return</b>	The query command returns set duration in seconds of the output current start ramp.			
<b>Usage example</b>	CURR:RAMP:DUR 0.5			

**Related Commands** [SOURce[<n>]]:VOLTage:RAMP:DURation

#### 5.15.10. [SOURce[<n>]]:LIST:COUNT

**Syntax** [SOURce[<n>]]:LIST:COUNT {<count>}  
[SOURce[<n>]]:LIST:COUNT?

**Description** This command sets the number of times that the list is executed before it is completed. The list count range is 1 through 65535. Use the INFinity parameter or 0 to execute a list continuously.

Use ABORt to stop the list at any time. When the list is aborted, the output returns to the settings that were in effect before the list started.

Parameters	Name	Type	Range	Default
	<count>	NR1 Discrete	0 1 – 65535 INFinity	1
<b>Return</b>	The query command returns the list count. Multiple responses are separated by commas. If a repeat count of 0 is returned, it means the list is set to repeat continuously.			
<b>Usage example</b>	LIST:COUNT 10			
<b>Related Commands</b>	ABORt INITiate[:IMMediate] MMEMory:LOAD:LIST[<n>] MMEMory:STORe:LIST[<n>]			

#### 5.15.11. [SOURce[<n>]]:LIST:CURRENT[:LEVel]

**Syntax** [SOURce[<n>]]:LIST:CURRENT[:LEVel] {<current>}, ...]  
[SOURce[<n>]]:LIST:CURRENT[:LEVel]?

**Description** This command specifies the current setting for each list step in amperes. A comma-delimited list of up to 256 steps may be programmed. The order in which the current values are entered determines the sequence when the list executes. To create a valid list, the VOLTage, CURRent and DWELl lists must either all be the same length, or have a length of 1, which is interpreted as having the same length as the list with the maximum length. This command overwrites any previously programmed current list; it does not append to the previous list.

Parameters	Name	Type	Range	Default
	<current>	NR2 Discrete	0 to maximum, MIN DEF MAX UP DOWN The MAXimum value is dependent on the power module current rating. See <a href="#">Section 8.1</a>	–
<b>Return</b>	The query command returns the programmed current level. Multiple responses are separated by commas.			
<b>Usage example</b>	LIST:CURR 0.25			
<b>Errors</b>	306, "Too many list points"			
<b>Related Commands</b>	MMEMory:LOAD:LIST[<n>] MMEMory:STORe:LIST[<n>] [SOURce[<n>]]:CURRent:MODE [SOURce[<n>]]:LIST:COUNT			

**5.15.12. [SOURce[<n>]]:LIST:DWELI**

**Syntax** [SOURce[<n>]]:LIST:DWELI {<time>}[, ...]  
[SOURce[<n>]]:LIST:DWELI?

**Description** This command specifies the dwell time for each list step. A comma-delimited list of up to 256 steps may be programmed. Dwell time is the time that the output will remain at a specific step. Dwell times can be programmed from 0 through 65535 seconds.

*Note that min. dwell time that can be achieved during the list execution depends of MCU activity and waveform shape. Therefore one have to find that out experimentally while list is executed on desired number of channels (one or two). Usage of oscilloscope is recommended for fast transitions since e.g. YT view resolution cannot be set to less then 20 ms. The expected usable min. dwell time goes well below 10 ms (down to 1 ms).*

Parameters	Name	Type	Range	Default
	<time>	NR2	0 – 65535	–
<b>Return</b>	The query command returns the programmed dwell times. Multiple responses are separated by commas.			
<b>Usage example</b>	LIST:DWEL 20ms,10ms,10ms,50ms			
<b>Errors</b>	306, "Too many list points"			
<b>Related Commands</b>	MMEMory:LOAD:LIST[<n>] MMEMory:STORe:LIST[<n>] [SOURce[<n>]]:LIST:COUNt			

**5.15.13. [SOURce[<n>]]:LIST:VOLTage[:LEVel]**

**Syntax** [SOURce[<n>]]:LIST:VOLTage[:LEVel] {<voltage>}[, ...]  
[SOURce[<n>]]:LIST:VOLTage[:LEVel]?

**Description** This command specifies the voltage setting for each list step in volts. A comma-delimited list of up to 256 steps may be programmed.  
The order in which the voltage values are entered determines the sequence when the list executes.  
To create a valid list, the VOLTage, CURRent and DWELI lists must either all be the same length, or have a length of 1, which is interpreted as having the same length as the list with the maximum length.  
This command overwrites any previously programmed voltage list; it does not append to the previous list.

Parameters	Name	Type	Range	Default
	<voltage>	NR2 Discrete	0 to maximum, MIN DEF MAX UP DOWN The maximum value is dependent on the power module voltage rating. See <a href="#">Section 8.1</a>	–
<b>Return</b>	The query command returns the programmed voltage level. Multiple responses are separated by commas.			
<b>Usage example</b>	Programming the list that contain 4 steps and will be executed 20 times on the channel 2. Execution will start by receiving remote command (*TRG) since BUS is selected as a trigger source:  INST CH2 LIST:COUN 20 LIST:VOLT 0,1.5,3,4.5 LIST:CURRE 0.25			

```

LIST:DWEL 20ms,10ms,10ms,50ms
OUTP ON
*OPC?

0

TRIG:SOUR BUS
INIT
*TRG

```

**Errors** 306, "Too many list points"

**Related Commands** MMEMory:LOAD:LIST[<n>]  
MMEMory:STORe:LIST[<n>]  
[SOURce[<n>]]:LIST:COUNT  
[SOURce[<n>]]:VOLTage:MODE

#### 5.15.14. [SOURce[<n>]]:POWER:LIMit

**Syntax** [SOURce[<n>]]:POWER:LIMit {<power>}  
[SOURce[<n>]]:POWER:LIMit? [<query power>]

**Modules** **DCP** **DCM**

**Description** This command sets the channel's output power limit. Units are in Watts. Such limitation is required if AC/DC power module *cannot* provides the same power as connected channel power module. For example, if power module can deliver 200 W but AC/DC power module offers only 155 W then MAXimum allowable continuous power is only 155 W.

Parameters	Name	Type	Range	Default
	<power>	NR2 Discrete	0 to MAXimum, MIN DEF MAX The MAXimum value is dependent on the power module power rating. See <a href="#">Section 8.1</a>	MAXimum
	<query power>	Discrete	MIN MAX	—

**Return** The query returns the max. allowed output power of the specified channel. Querying MAX on an output channel returns the maximum rated power limit.

**Usage example**  
 POW:LIM DEF  
 POW:LIM?  
 150.00

**Related Commands** \*SAV  
 [SOURce[<n>]]:CURRent:LIMit[:POSitive][:IMMediate][:AMPLitude]  
 [SOURce[<n>]]:VOLTage:LIMit[:POSitive][:IMMediate][:AMPLitude]

#### 5.15.15. [SOURce[<n>]]:POWER:PROTection[:LEVel]

**Syntax** [SOURce[<n>]]:POWER:PROTection[:LEVel] {<power>}  
[SOURce[<n>]]:POWER:PROTection[:LEVel]?

**Description** Set the over-power protection (OPP) value of the channel. When [:SOURce[<n>]] or [<n>] is omitted, the currently selected channel will be affected by this command. When the over-power protection function of the specified channel is enabled ([SOURce[<n>]]:POWER:PROTection:STATe), the output turns off automatically when the output power exceeds the over-power protection value currently set. [SOURce[<n>]]:POWER:PROTection:TRIPped? command can be used to query whether over-power protection occurred on the selected channel.

Parameters	Name	Type	Range	Default
	<power>	NR2 Discrete	0 to MAXimum, MIN	DEFault

DEF|MAX  
The maximum value is dependent on the power module power rating. See [Section 8.1](#)

**Return** Query the over-power protection (OPP) value of the selected channel.

**Usage example** Set power protection to 50 W on the channel 2:

```
SOUR2:POW:PROT 50
```

**Related Commands** \*SAV  
OUTPut:PROTection:COUPle  
[SOURce[<n>]]:POWer:PROTection:TRIPped?  
[SOURce[<n>]]:POWer:PROTection:STATe

#### 5.15.16. [SOURce[<n>]]:POWer:PROTection:DELay[:TIME]

**Syntax** [SOURce[<n>]]:POWer:PROTection:DELay[:TIME] {<time>}  
[SOURce[<n>]]:POWer:PROTection:DELay[:TIME]? [<query time>]

**Description** This command sets the over-power protection (OPP) delay. The over-power protection function will not be triggered on the selected output channel during the delay time. After the delay time has expired, the over-power protection function will be active. This prevents momentary changes in output status from triggering the over-power protection function. Programmed values can range from 0 to 300 seconds. See also [Section 8.1](#)

Parameters	Name	Type	Range	Default
	<time>	NR2 Discrete	0 – 300 DEFault	10
	<query time>	Discrete	DEFault	–

**Return** The query command returns the programmed delay time.

**Usage example** Get default OPP delay of 10 seconds:

```
POW:PROT:DEL? DEF
10
```

**Related Commands** \*SAV  
OUTPut:PROTection:CLEAr

#### 5.15.17. [SOURce[<n>]]:POWer:PROTection:STATe

**Syntax** [SOURce[<n>]]:POWer:PROTection:STATe {<bool>}  
[SOURce[<n>]]:POWer:PROTection:STATe?

**Description** This command enables or disables the over-power protection (OPP) function. The enabled state is ON (1); the disabled state is OFF (0). If the over-power protection function is enabled and the measure output power reach value set by [SOURce[<n>]]:POWer:PROTection[:LEVel] the output is disabled and the Questionable Condition status register OPP bit 10 is set.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	OFF

**Return** The query command returns 0 if the current protection state is OFF, and 1 if the current protection state is ON.

**Usage example** POW:PROT:STAT ON  
POW:PROT:STAT?  
1

**Related** \*SAV



**Commands** OUTPut:PROTection:CLEar  
[SOURce[<n>]]:POWEr:PROTection[:LEVel]

#### 5.15.18. [SOURce[<n>]]:POWEr:PROTection:TRIPped?

**Syntax** [SOURce[<n>]]:POWEr:PROTection:TRIPped?

**Description** Query whether OPP occurred on the currently selected channel. When protection is tripped bit 10 (OPP) of the Questionable Instrument Isummary register will be set (see [Section 3.4.2](#)).

The [OUTPut:PROTection:CLEar](#) command can be send to clear OPP condition on the selected channel.

**Return** This query returns a 1 if the protection circuit is tripped and a 0 if it is untripped.

**Usage example**  
POW:PROT:TRIP?  
0

**Related Commands** OUTPut:PROTection:CLEar

#### 5.15.19. [SOURce[<n>]]:VOLTage

**Syntax** [SOURce[<n>]]:VOLTage[:LEVel][:IMMediate][:AMPLitude] {<voltage>}  
[SOURce[<n>]]:VOLTage[:LEVel][:IMMediate][:AMPLitude]? [<query voltage>]

**Modules** DCP DCM SMX

**Description** This command sets the immediate voltage level of the output channel. Units are in volts. When [:SOURce[<n>]] or [<n>] is omitted, the currently selected channel will be affected by this command.

This command also increases or decreases the immediate voltage level using the 'UP' or DOWN parameter by a predetermined amount. The command VOLTage:STEP sets the amount of increase or decrease. A new increment setting will *not* cause an execution error -222,"Data out of range" when the maximum or the minimum rated current is exceeded – the output value will be set to the maximum or the minimum value instead.

At \*RST, the signal being sourced will be set to a "safe" condition. This is achieved by setting the amplitude to its MINimum value (see [Section 8.1](#)).

The SMX46 has the following two channels that can be programmed with VOLTage command (n is slot number):

- (@n01) – Analog output #1, 0 – 10 V, 10 mV set resolution
- (@n02) – Analog output #2, 0 – 10 V, 10 mV set resolution

Parameters	Name	Type	Range	Default
	<voltage>	NR2 Discrete	0 to MAXimum, MIN DEF MAX UP DOWN The maximum value is dependent on the power module voltage rating. See <a href="#">Section 8.1</a>	–
	<query voltage>	Discrete	MIN DEF MAX	–
<b>Return</b>	The query command returns the programmed voltage level. VOLT? MIN, VOLT? DEF and VOLT? MAX can be used to obtain minimum, default and maximum voltage level on the currently selected channel. For actual output voltage value use MEASure:VOLTage? command.			
<b>Usage</b>	A 10 $\Omega$ load is connected and current is set to 1 A. With MAX voltage set measured volt-			



**example** age will be 10 V. When new voltage value is set to 5 V, current will drop to 0.5 A (the channel enters the CV mode of operation):

```
INST CH1
VOLT MAX
CURR 1
MEAS:CURR?
```

```
1.00
```

```
VOLT 5
MEAS:CURR?
```

```
0.50
```

Query that returns maximum voltage of the currently selected channel:

```
VOLT? MAX
40.00
```

Set 2 V and 5 V respectively for AO1 and AO2 channels of SMX46 installed in slot 2:

```
INST (@201)
VOLT 2
INST (@202)
VOLT5
```

**Errors** 150, "Power limit exceeded"  
-222, "Data out of range"

**Related Commands** \*SAV  
\*RST  
APPLy  
MEASure[:SCALar]:VOLTage[:DC]?  
[SOURce[<n>]]:VOLTage[:LEVel][:IMMediate]:STEP[:INCRement]

#### 5.15.20. [SOURce[<n>]]:VOLTage:LIMit[:POSitive][:IMMediate][:AMPLitude]

**Syntax** [SOURce[<n>]]:VOLTage:LIMit[:POSitive][:IMMediate][:AMPLitude] {<voltage>}  
[SOURce[<n>]]:VOLTage:LIMit[:POSitive][:IMMediate][:AMPLitude]? [<query voltage>]

**Description** This command sets the channel's output voltage limit. Units are in volts. When [:SOURce[<n>]] or [<n>] is omitted, the currently selected channel will be affected by this command. This command could be used as mean of protection against incidental programming of too high output voltage that can be set for example with [SOURce[<n>]]:VOLTage:STEP command.

Parameters	Name	Type	Range	Default
	<voltage>	NR2 Discrete	0 to MAXimum, MIN DEF MAX The MAXimum value is dependent on the power module current rating. See <a href="#">Section 8.1</a>	MAXimum
	<query voltage>	Discrete	MIN MAX	—

**Return** The query returns the max. allowed output voltage of the specified channel. Querying MAX on an output channel returns the maximum voltage limit.

**Usage example** VOLT:LIM 20

**Related Commands** \*SAV  
[SOURce[<n>]]:CURRent:STEP

[SOURce[<n>]]:POWER:LIMit  
[SOURce[<n>]]:VOLTage:LIMit[:POSitive][:IMMediate][:AMPLitude]

### 5.15.21. [SOURce[<n>]]:VOLTage:STEP

**Syntax** [SOURce[<n>]]:VOLTage[:LEVel][:IMMediate]:STEP[:INCRement] {<step>}  
[SOURce[<n>]]:VOLTage[:LEVel][:IMMediate]:STEP[:INCRement]? [<query step>]

**Description** Set the step of the voltage change of the channel. When [SOURce[<n>]] or [<n>] is omitted, the currently selected channel will be affected by this command. Step change is performed by using UP and DOWN as parameter for the [SOURce[<n>]]:VOLTage command.

Parameters	Name	Type	Range	Default
	<step>	NR2 Discrete	0.01 to 10 DEFault	0.1
	<query step>	Discrete	DEFault	—

**Return** The query returns the step of the voltage change of the specified channel.

**Usage example** Return default step value:

```
VOLT:STEP? DEF
0.10
```

By connecting a 10  $\Omega$  load and current set to 2 A and voltage to 10 V the first channel enters the CV mode of operation. Voltage is then decreased from 10 V in two steps to 6 V:

```
APPL CH1, 10, 2
MEAS:CURR?
1.0
VOLT:STEP 2
VOLT DOWN
VOLT DOWN
MEAS:VOLT?
6.0
MEAS:CURR?
0.60
```

**Related Commands** \*SAV  
[SOURce[<n>]]:VOLTage

### 5.15.22. [SOURce[<n>]]:VOLTage:TRIGgered

**Syntax** [SOURce[<n>]]:VOLTage[:LEVel]:TRIGgered[:AMPLitude] {<voltage>}  
[SOURce[<n>]]:VOLTage[:LEVel]:TRIGgered[:AMPLitude]? [<query voltage>]

**Description** This command programs the pending triggered voltage level. The pending triggered current level is a stored value that is transferred to the output terminals when a trigger occurs.

*A pending triggered level is not affected by subsequent VOLTage commands.*

Parameters	Name	Type	Range	Default
	<voltage>	NR2 Discrete	0 to maximum, MIN DEF MAX The maximum value is dependent on the power module voltage rating. See <a href="#">Section 8.1</a>	0.00
	<query voltage>	Discrete	MIN MAX	

<b>Return</b>	Query the triggered voltage level presently programmed. If no triggered level is programmed, the VOLTage level is returned. VOLT:TRIG? MIN and VOLT:TRIG? MAX return the lowest and highest programmable triggered voltage levels.
<b>Usage example</b>	On the currently selected channel voltage will be set to 3.3V and current to 1A when INITiate command is executed:  <pre>VOLT:TRIG 3.3 CURR:TRIG 1 TRIG:SOUR IMM INIT</pre>
<b>Related Commands</b>	ABORT

### 5.15.23. [SOURce[<n>]]:VOLTage:MODE

Syntax	[SOURce[<n>]]:VOLTage:MODE {<mode>} [SOURce[<n>]]:VOLTage:MODE?			
Description	This command determines what happens to the output voltage when the transient system is initiated and triggered: <ul style="list-style-type: none"><li>• FIXed – the output current remains at the immediate value</li><li>• LIST – the output follows the list values when a trigger occurs.</li><li>• STEP – the output goes to the triggered level when a trigger occurs.</li></ul>			
Parameters	Name	Type	Range	Default
	<mode>	Discrete	FIXed LIST STEP	FIXed
Return	The query command returns the voltage mode of the currently selected channel.			
Usage example	VOLT:MODE?  LIST			
Related Commands	[SOURce[<n>]]:LIST:VOLTage[:LEVel] [SOURce[<n>]]:CURRent:MODE			

### 5.15.24. [SOURce[<n>]]:VOLTage:PROGram[:SOURce]

<b>Syntax</b>	[SOURce[<n>]]:VOLTage:PROGram[:SOURce] {<source>} [SOURce[<n>]]:VOLTage:PROGram[:SOURce]?
<b>Modules</b>	<b>DCP</b>
<b>Description</b>	<p>Use this command to define source for output voltage programming if channel support this option (use the SYSTem:CHANnel[:INfOrMation]:PROGram? to query channel programming capability).</p> <p>A channel's D/A converter controlled by CPU is used by default for voltage output programming. That source can be calibrated (see the CALibrate subsystem) and provide output within safe limits.</p> <p>The external voltage programming could be used when fast interaction with an external process is required. For example, if tracking output of the connected D.U.T. (i.e. a power supply) is needed the BB3's power module like DCP405 effectively becomes a pre-regulator keeping its output voltage in relation with changes of the D.U.T. output keeping constant difference between connected D.U.T input and output and in that way its max. power dissipation.</p> <p><i>Max. D/A converter programmed voltage of 2.5 V would results with MAXimal voltage output regardless of the channel's voltage range (see <a href="#">Section 8.1</a>). Therefore if EXTer-nal programming source is selected, any voltage value higher then 2.5 V could produce unexpected results and eventually damage the power module and/or connected load.</i></p> <p><i>To limit possible damage when programming source is set to EXTer-nal, the over-voltage protection (OVP) will be activated and set to MAXimum ([SOURce[&lt;n&gt;]]:VOLTage:PRO-Tection[:LEVel]) with protection delay time set to zero ([SOURce[&lt;n&gt;]]:VOLTage:PRO-</i></p>

*Tection:DElay[:TIME]). Further adjustment of OVP level and delay is also allowed (e.g. to decrease OVP level or increase delay time).*

The enabled state is EXTERNAL (1); the disabled state is INTERNAL (0). Execution of this command also affects bit 13 (RPROG) of the Operation Instrument Isummary register (see [Section 3.3.2](#)) and activate dedicated LED indicator (*Rprog*) on the module's front panel.

Self-test operation initiated by [\\*TST?](#) command will reset voltage programming on all channels to the internal/local source.

If external programming source is selected the DIAGnostic[:INfOrmation]:ADC? Query returns 0.00 value for U\_SET.

Parameters	Name	Type	Range	Default
	<source>	Discrete	INTERNAL EXTERNAL	INTERNAL
<b>Return</b>	The query command returns 0 if the local (internal) voltage programming is selected, and 1 if the remote (external) sense is selected.			
<b>Usage example</b>	VOLT:PROG EXT VOLT:PROG? 1			
<b>Errors</b>	-241, "Hardware missing" 312, "Cannot execute when the channels are coupled"			
<b>Related Commands</b>	*TST DIAGnostic[:INfOrmation]:ADC? INSTRument:COUPle:TRACkING OUTPut[:STATe] [SOURce[<n>]]:VOLTage:PROTection[:LEVel] [SOURce[<n>]]:VOLTage:PROTection:DElay[:TIME] [SOURce[<n>]]:VOLTage:PROTection:TRIPped? SYSTem:CHANnel[:INfOrmation]:PROGram?			

#### 5.15.25. [\[SOURce\[<n>\]\]:VOLTage:PROTection\[:LEVel\]](#)

**Syntax** [\[SOURce\[<n>\]\]:VOLTage:PROTection\[:LEVel\] {<voltage>}](#)  
[\[SOURce\[<n>\]\]:VOLTage:PROTection\[:LEVel\]?](#)

**Description** Set the over-voltage protection (OVP) value of the channel. When [:SOURce[<n>]] or [<n>] is omitted, the currently selected channel will be affected by this command. When the over-voltage protection function of the specified channel is enabled ([SOURce[<n>]]:VOLTage:PROTection:STATe), the output turns off automatically when the output voltage exceeds the over-voltage protection value currently set. [SOURce[<n>]]:VOLTage:PROTection:TRIPped? command can be used to query whether over-voltage protection occurred on the selected channel.

An execution error -222, "Data out of range" will be generated when output voltage is controlled internally ([SOURce[<n>]]:VOLTage:PROGram[:SOURce]) and OVP value is set below programmed output voltage ([SOURce[<n>]]:VOLTage).

Parameters	Name	Type	Range	Default
	<voltage>	NR2 Discrete	0 to maximum, MIN DEF MAX The maximum value is dependent on the power module voltage rating. See <a href="#">Section 8.1</a>	—

**Return** Query the over-voltage protection (OVP) value of the selected channel.

**Usage example** Set voltage protection to 10.2 V on the channel 1:

SOUR1:VOLT:PROT 10.2

**Errors** -222, "Data out of range"

**Related Commands** \*SAV  
 OUTPut:PROTection:COUPle  
 [SOURce[<n>]]:VOLTage  
 [SOURce[<n>]]:VOLTage:PROGram[:SOURce]  
 [SOURce[<n>]]:VOLTage:PROTection:TRIPped?  
 [SOURce[<n>]]:VOLTage:PROTection:STATe

#### 5.15.26. [SOURce[<n>]]:VOLTage:PROTection:DELay[:TIME]

**Syntax** [SOURce[<n>]]:VOLTage:PROTection:DELay[:TIME] {<time>}  
 [SOURce[<n>]]:VOLTage:PROTection:DELay[:TIME]? [<query time>]

**Description** This command sets the over-voltage protection delay. The over-voltage protection function will not be triggered on the selected output channel during the delay time. After the delay time has expired, the over-voltage protection function will be active. This prevents momentary changes in output status from triggering the over-voltage protection function. Programmed values can range from 0 to 10 seconds. See also [Section 8.1](#)

Parameters	Name	Type	Range	Default
	<time>	NR2 Discrete	0 – 10 DEFault	5 ms
	<query time>	Discrete	DEFault	–

**Return** The query command returns the programmed delay time.

**Usage example** Get default OVP delay of 50 milliseconds:

```
VOLT:PROT:DEL? DEF
0.050
```

**Related Commands** \*SAV  
 OUTPut:PROTection:CLEAr

#### 5.15.27. [SOURce[<n>]]:VOLTage:PROTection:STATe

**Syntax** [SOURce[<n>]]:VOLTage:PROTection:STATe {<bool>}  
 [SOURce[<n>]]:VOLTage:PROTection:STATe?

**Description** This command enables or disables the over-voltage protection (OVP) function. The enabled state is ON (1); the disabled state is OFF (0). Power modules that do not have a dedicated over-voltage protection circuit can be programmed independently of output current level, entering the CV (constant voltage) mode of operation is used as a trigger to start OVP sequence. When delay time specified with the [SOURce[<n>]]:VOLTage:PROTection:DELay[:TIME] command expired the output turns off and the Questionable Condition status register OCP bit 8 is set. An error tone will also follow if beeper is enabled (see SYSTem:BEEPer:STATe). [SOURce[<n>]]:VOLTage:PROTection:TRIPped? command can be used to query whether over-voltage protection occurred on the selected channel.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	OFF

**Return** The query command returns 0 if the voltage protection state is OFF, and 1 if the voltage protection state is ON.

**Usage example** VOLT:PROT:STAT?  
 0

**Related Commands** \*SAV  
 OUTPut:PROTection:CLEAr

[SOURce[<n>]]:VOLTage:PROTection:DELAy[:TIME]  
 [SOURce[<n>]]:VOLTage:PROTection:TRIPped  
 SYSTem:BEEPer:STATe

#### 5.15.28. [SOURce[<n>]]:VOLTage:PROTection:TRIPped?

**Syntax** [SOURce[<n>]]:VOLTage:PROTection:TRIPped?

**Description** Query whether OVP occurred on the currently selected channel. When protection is tripped bit 8 (OVP) of the Questionable Instrument Isummary register will be set (see [Section 3.4.2](#)).  
 When channel's output voltage is controlled remotely ([SOURce[<n>]]:VOLTage:PROGram[:SOURce]) this protection will change voltage control back to INTERNAL source.  
 The [OUTPut:PROTection:CLEar](#) command can be send to clear OVP condition on the selected channel.

**Return** This query returns a 1 if the protection circuit is tripped and a 0 if it is untripped.

**Usage example**  
 VOLT:PROT:TRIP?  
 0  
 VOLT:PROG?  
 0

**Related Commands** OUTPut:PROTection:CLEar  
 [SOURce[<n>]]:VOLTage:PROGram[:SOURce]

#### 5.15.29. [SOURce[<n>]]:VOLTage:PROTection:TYPE

**Syntax** [SOURce[<n>]]:VOLTage:PROTection:TYPE {<type>}  
 [SOURce[<n>]]:VOLTage:PROTection:TYPE?

**Modules** **DCP**

**Description** Use this command to select one of the following types of over-voltage protection mechanism:

- HW – on-board OVP triac crow-bar circuit is active with trip level set to about 3 % over output voltage set using the [SOURce[<n>]]:VOLTage[:LEVel][:IMMEDIATE][:AMPLitude] {<voltage>} command.
- SW – MCU is monitoring output voltage and compare it with trip level set using the [SOURce[<n>]]:POWer:PROTection[:LEVel] command. If trip level is reached OVP will disable output when time set by [SOURce[<n>]]:POWer:PROTection:DELAy[:TIME] command passed.

*The HW OVP circuit will cause that output terminals be shorted, and that could damage one or both of the on-board fuses. If that happened be sure to use new fuse with the same current rating.*

Parameters	Name	Type	Range	Default
	<type>	Discrete	HW SW	SW
<b>Return</b>	The query command returns type of currently active OVP mechanism.			
<b>Usage example</b>	VOLT:PROT:TYPE HW			
<b>Errors</b>	-241, "Hardware missing"			
<b>Related Commands</b>	*RST [SOURce[<n>]]:VOLTage[:LEVel][:IMMEDIATE][:AMPLitude] {<voltage>} [SOURce[<n>]]:POWer:PROTection[:LEVel] [SOURce[<n>]]:VOLTage:PROTection:STATe			

**5.15.30. [SOURce[<n>]]:VOLTage:RAMP:DURation**

**Syntax** [SOURce[<n>]]:VOLTage:RAMP:DURation {<duration>}  
[SOURce[<n>]]:VOLTage:RAMP:DURation?

**Description** Use this command to configures a constant rise of output voltage on the channel <n> within a set duration. The output voltage can be increased continuously within a 0 to 10 seconds with 1 ms step size.

Parameters	Name	Type	Range	Default
	<duration>	NR2	0 – 10	–

**Return** The query command returns set duration in seconds of the output voltage start ramp.

**Usage example** CURR:RAMP:DUR 0.5

**Related Commands** [SOURce[<n>]]:CURRent:RAMP:DURation

**5.15.31. [SOURce[<n>]]:VOLTage:SENSe[:SOURce]**

**Syntax** [SOURce[<n>]]:VOLTage:SENSe[:SOURce] {<source>}  
[SOURce[<n>]]:VOLTage:SENSe[:SOURce]?

**Modules** **DCP**

**Description** This command enables or disables remote sensing. The enabled state is EXTERNAL (1); the disabled state is INTERNAL (0). Execution of this command also affects bit 12 (RSENSE) of the Operation Instrument Summary register (see [Section 3.3.2](#)). Self-test operation initiated by \*TST? command will put remote sense on all channels into disable state.

When channels are not coupled together (INSTrument:COUPle:TRACking) this command turn on a dedicated LED indicator (*Rsense*) mounted on the power module's front panel.

*Remote sensing has no effect during CC (Constant Current) operation. Rsense indicator will not be affected if output state is off (OUTPut OFF command).*

Parameters	Name	Type	Range	Default
	<source>	Discrete	INTERNAL EXTERNAL	INTERNAL

**Return** The query command returns 0 if the internal sense is selected, and 1 if the remote (external) sense is selected.

**Usage example** VOLT:SENS EXT  
VOLT:SENS?  
1

**Errors** -241, "Hardware missing"  
312, "Cannot execute when the channels are coupled"

**Related Commands** \*SAV  
\*TST  
INSTrument:COUPle:TRACking  
OUTPut[:STATe]  
SYSTem:CHANnel[:INFORMation]:PROGram?

**5.15.32. [SOURce[<n>]]:VOLTage:SLEW:FALLing**

*Not implemented yet*

**5.15.33. [SOURce[<n>]]:VOLTage:SLEW:RISing**

*Not implemented yet*



## 5.16. STATus

Status register programming lets you determine the operating condition of the instrument at any time. This subsystem controls the SCPI-defined status-reporting structures. SCPI defines, in addition to those in IEEE 488.2, QUEStionable, OPERation, INSTRument SUMmary and INSTRument registers. These registers conform to the IEEE 488.2 specification and each may be comprised of a condition register, an event register, an enable register. The purpose and definition of the SCPI-defined registers is described in "Volume 1: Syntax and Style". SCPI also defines an IEEE 488.2 queue for status. The queue provides a human readable record of instrument events. The application programmer may individually enable events into the queue.

STATus:PRESet enables errors and disables all other events.

SCPI command	Description
STATus	
:OPERation	
[:EVENT]?	Returns the value of the Operation Event register
:CONDition?	Returns the value of the Operation Instrument Condition register
:ENABLE {<value>}	Enables specific bits in the Operation Event register
:INSTRument[<n>]	
[:EVENT]?	Returns the value of the Operation Instrument Event register
:CONDition?	Returns the value of the Operation Instrument Condition register
:ENABLE {<value>}	Enables specific bits in the Operation Instrument Event register
:ISUMmary<n>	
[:EVENT]?	Returns the value of the Operation Instrument Isummary Event register
:CONDition?	Returns the value of the Operation Instrument Isummary Condition register
:ENABLE {<value>}	Enables specific bits in the Operation Instrument Isummary Event register
:PRESet	Presets all enable registers to power-on state
:QUEStionable	
[:EVENT]?	Returns the value of the Questionable Event register
:CONDition?	Returns the value of the Questionable Condition register
:ENABLE {<value>}	Enables specific bits in the Questionable Event register
:INSTRument[<n>]	
[:EVENT]?	Returns the value of the Questionable Instrument Event register
:CONDition?	Returns the value of the Questionable Instrument Condition register
:ENABLE {<value>}	Enables specific bits in the Questionable Instrument Event register
:ISUMmary<n>	
[:EVENT]?	Returns the value of the Questionable Instrument Isummary Event register
:CONDition?	Returns the value of the Questionable Instrument Isummary Condition register



**:ENABLE {<value>}**

Enables specific bits in the Questionable Instrument Summary Event register

**5.16.1. STATus:OPERation[:EVENT]?****Syntax**      **STATus:OPERation[:EVENT]?****Description** This query returns the value of the read-only Operation Status Event register. The bits are latched and reading the register will clear it. The \*CLS command can be also used to clear the register.**Return** The value returned is the binary-weighted sum of all bits set in the register. For example, if bit 9 (decimal value = 512) and bit 13 (decimal value = 8192) are set, this command will return 8704. See table in the [Section 3.3](#) for bits description.**Usage example** If GROUp PARAllel (bit 8) is set (next query returns 0 since the first query clears the event register):

STAT:OPER?

256

STAT:OPER?

0

**Related**      \*CLS**Commands**    \*STB?

STATus:OPERation:ENABLE

**5.16.2. STATus:OPERation:CONDition?****Syntax**      **STATus:OPERation:CONDition?****Description** This query returns the value of the read-only Operation Status Condition register.**Return** The value returned is the binary-weighted sum of all bits set in the register. For example, if bit 9 (decimal value = 512) and bit 13 (decimal value = 8192) are set, this command will return 8704. See table in the [Section 3.3](#) for bits description.**Usage example** If GROUp PARAllel (bit 8) is set:

STAT:OPER:COND?

256

**Related**      STATus:OPERation:ENABLE  
**Commands****5.16.3. STATus:OPERation:ENABLE****Syntax**      **STATus:OPERation:ENABLE {<value>}**  
**STATus:OPERation:ENABLE?****Description** This command and its query set and read the value of the Operation Status Enable register. The Enable register is a mask for enabling specific bits from the Operation Event register to set the operation summary bit 7 (OPER) of the Status Byte register. This bit is the logical OR of all the Operational Event register bits that are enabled by the Operation Status Enable register.

Parameters	Name	Type	Range	Default
	<value>	NR1	A decimal value which corresponds to the binary-weighted sum of the bits in the register (see the table in <a href="#">Section 3.3</a> )	PREset=0

**Return** Query the Operation Status Enable register. The BB3 returns a binary-weighted decimal

representing the bits set in the enable register.

**Usage example** Enable ISUM (bit 13):  
`STAT:OPER:ENAB 8192`

**Related Commands** \*CLS  
 \*STB?  
 STATus:OPERation[:EVENT]?

#### 5.16.4. STATus:OPERation:INSTrument[:EVENT]?

**Syntax** `STATus:OPERation:INSTrument[:EVENT]?`

**Description** This query returns the value of the read-only Instrument Operation Status Event register. The bits are latched and reading the register will clear it. The \*CLS command can be also used to clear the register.

**Return** The value returned is the binary-weighted sum of all bits set in the register. For example, if bit 1 (decimal value = 2) and bit 2 (decimal value = 4) are set, this command will return 6. See table in the [Section 3.3.1](#) for bits description.

**Usage example** If bit 2 (INST2) is set:  
`STAT:OPER:INST?`  
 4

**Related Commands** \*CLS  
 STATus:PREset

#### 5.16.5. STATus:OPERation:INSTrument:CONDition?

**Syntax** `STATus:OPERation:INSTrument:CONDition?`

**Description** This query returns the value of the read-only Instrument Operation Status Condition register.

**Return** The value returned is the binary-weighted sum of all bits set in the register. For example, if bit 1 (decimal value = 2) and bit 2 (decimal value = 4) are set, this command will return 6. See table in the [Section 3.3.1](#) for bits description.

**Usage example** If bit 2 (INST2) is set:  
`STAT:OPER:INST:COND?`  
 4

**Related Commands** STATus:PREset

#### 5.16.6. STATus:OPERation:INSTrument:ENABLE

**Syntax** `STATus:OPERation:INSTrument:ENABLE {<value>}`  
`STATus:OPERation:INSTrument:ENABLE?`

**Description** Enable bits in the Instrument Operation Status Enable register. The selected bits are then reported to the Operation Status Event register.

Parameters	Name	Type	Range	Default
	<value>	NR1	A decimal value which corresponds to the binary-weighted sum of the bits in the register (see the table in <a href="#">Section 3.3.1</a> )	PREset=0

**Return** Query the Instrument Operation Status Enable register. The BB3 returns a binary-weighted decimal representing the bits set in the enable register.

**Usage example** Enable INST1 (bit 1) and INST2 (bit 2):  
 STAT:OPER:INST:ENAB 6

**Related Commands** \*CLS  
 STATus:PREset

#### 5.16.7. STATus:OPERation:INSTrument:ISUMmary[<n>][:EVENT]?

**Syntax** STATus:OPERation:INSTrument:ISUMmary[<n>][:EVENT]?

**Description** This query returns the value of the read-only Instrument Isummary Operation Status Event register for a specific channel of the BB3 represented by numeric value [<n>]. When [<n>] is omitted, the system queries the Instrument Isummary Operation Status Event register of the current channel. The bits are latched and reading the register will clear it. The \*CLS command can be also used to clear the register.

**Return** The value returned is the binary-weighted sum of all bits set in the register. See table in the [Section 3.3.2](#) for bits description.

**Usage example** If bit 8 (CV1) and bit 10 (OE1) on the channel 1 are set ( $256 + 1024 = 1280$ ):  
 STAT:OPER:INST:ISUM1?  
 1280

**Related Commands** \*CLS  
 OUTPut:MODE?

#### 5.16.8. STATus:OPERation:INSTrument:ISUMmary[<n>]:CONDition?

**Syntax** STATus:OPERation:INSTrument:ISUMmary[<n>]:CONDition?

**Description** This query returns the value of the read-only Instrument Isummary Operation Status Condition register for a specific channel of the BB3 represented by numeric value [<n>]. When [<n>] is omitted, the system queries the Instrument Isummary Operation Status Condition register of the current channel.

**Return** The value returned is the binary-weighted sum of all bits set in the register. See table in the [Section 3.3.2](#) for bits description.

**Usage example** If bit 8 (CV1) and bit 10 (OE1) on the channel 1 are set ( $256 + 1024 = 1280$ ):  
 STAT:OPER:INST:ISUM1:COND?  
 1280

**Related Commands** OUTPut:MODE?

#### 5.16.9. STATus:OPERation:INSTrument:ISUMmary<n>:ENABLE

**Syntax** STATus:OPERation:INSTrument:ISUMmary[<n>]:ENABLE {<value>}  
 STATus:OPERation:INSTrument:ISUMmary[<n>]:ENABLE?

**Description** Enable bits in the Instrument Isummary Operation Status Enable register for a specific channel of the BB3 represented by numeric value [<n>]. When [<n>] is omitted, the system queries the Instrument Isummary Operation Status Enable register of the current channel. The selected bits are then reported to the Status Byte.

This command and its query set and read the value of the Operation Status Enable register. The Enable register is a mask for enabling specific bits from the Operation Event register to set the operation summary bit (OPER) of the Status Byte register. This bit (bit 7) is the logical OR of all the Operational Event register bits that are enabled by the Operation Status Enable register

Parameters	Name	Type	Range	Default
	<value>	NR1	A decimal value	PREset=0

which corresponds to the binary-weighted sum of the bits in the register (see the table in [Section 3.3.2](#))

<b>Return</b>	Query the Instrument Isummary Operation Status Enable register. The BB3 returns a binary-weighted decimal representing the bits set in the enable register.
<b>Usage example</b>	The query returns that VOLT1 (bit 0), CURR1 (bit 1) and TEMP1 (bit 4) are set (1 + 2 + 16 = 19):  <pre>INST? CH2  INST CH1 STAT:OPER:INST:ISUM:ENABLE?  19</pre>
<b>Related Commands</b>	*CLS STATus:PREset

#### 5.16.10. STATus:PREset

<b>Syntax</b>	STATus:PREset
<b>Description</b>	This command clears all bits in the Enable registers.
<b>Return</b>	None
<b>Usage example</b>	STAT:PRE
<b>Related Commands</b>	*CLS

#### 5.16.11. STATus:QUEStionable[:EVENT]?

<b>Syntax</b>	STATus:QUEStionable[:EVENT]?
<b>Description</b>	Query the Questionable Status event register. The bits are latched and reading the register will clear it. The *CLS command can be also used to clear the register.
<b>Return</b>	The BB3 returns a decimal value which corresponds to the binary-weighted sum of all bits in the register. See table in the <a href="#">Section 3.4</a> for bits description.
<b>Usage example</b>	If the error is detected in RTC (Real-time clock) circuit, the bit 3 (TIME) is set and this command returns 8:  <pre>STAT:QUES?  8</pre>
<b>Related Commands</b>	*CLS

#### 5.16.12. STATus:QUEStionable:CONDition?

<b>Syntax</b>	STATus:QUEStionable:CONDition?
<b>Description</b>	Query the Questionable Status condition register.
<b>Return</b>	The BB3 returns a decimal value which corresponds to the binary-weighted sum of all bits in the register. See table in the <a href="#">Section 3.4</a> for bits description.
<b>Usage example</b>	If the error is detected in RTC (Real-time clock) circuit, the bit 3 (TIME) is set and this command returns 8:  <pre>STAT:QUES:COND?  8</pre>

**5.16.13.    STATus:QUEStionable:ENABle**

**Syntax**        `STATus:QUEStionable:ENABle {<value>}`  
                  `STATus:QUEStionable:ENABle?`

**Description** Enable bits in the Questionable Status Enable register. The selected bits are then reported to the Status Byte.  
 When <enable value> is set to 0, executing this command will clear the Questionable Status Enable register.

Parameters	Name	Type	Range	Default
	<value>	NR1	A decimal value which corresponds to the binary-weighted sum of the bits in the register (see table in <a href="#">Section 3.4</a> )	PREset=0

**Return**        Query the Questionable Status Enable register. The BB3 returns a binary-weighted decimal representing the bits set in the enable register.

**Usage example**    The query returns that TIME (bit 3), TEMPerature (bit 4) and ISUM (bit 13) are enabled ( $8 + 16 + 8192 = 8216$ ):

```
STAT:QUES:ENAB?
```

```
8216
```

**Related Commands**    \*CLS  
                  STATus:PREset

**5.16.14.    STATus:QUEStionable:INSTrument[:EVENT]?**

**Syntax**        `STATus:QUEStionable:INSTrument[:EVENT]?`

**Description** Query the questionable instrument event register. The bits are latched and reading the register will clear it. The \*CLS command can be also used to clear the register.

**Return**        The BB3 returns a decimal value which corresponds to the binary-weighted sum of all bits in the register and clears the register. See table in the [Section 3.4.1](#) for bits description.

**Usage example**    Result of the query when INST1 (bit 1) and INST2 (bit 2) are set ( $2 + 4 = 6$ ):

```
STAT:QUES:INST?
```

```
6
```

**Related Commands**    \*CLS

**5.16.15.    STATus:QUEStionable:INSTrument:CONDition?**

**Syntax**        `STATus:QUEStionable:INSTrument:CONDition?`

**Description** Query the questionable instrument condition register.

**Return**        The BB3 returns a decimal value which corresponds to the binary-weighted sum of all bits in the register and clears the register. See table in the [Section 3.4.1](#) for bits description.

**Usage example**    Result of the query when INST1 (bit 1) and INST2 (bit 2) are set ( $2 + 4 = 6$ ):

```
STAT:QUES:INST:COND?
```

```
6
```

**5.16.16.    STATus:QUEStionable:INSTrument:ENABle**

**Syntax**        `STATus:QUEStionable:INSTrument:ENABle {<value>}`

**STATus:QUEStionable:INSTrument:ENABle?**

**Description** Set the value of the questionable instrument enable register. This register is a mask for enabling specific bits from the questionable instrument event register to set the instrument summary bit 13 (ISUM) of the Questionable Status register. The ISUM bit of the Questionable Status register is the logical OR of all the questionable instrument event register bits that are enabled by the questionable instrument enable register.

Parameters	Name	Type	Range	Default
	<value>	NR1	A decimal value which corresponds to the binary-weighted sum of the bits in the register (see table in <a href="#">Section 3.4.1</a> )	PREset=0

**Return** Query the Questionable Instrument Enable register. The BB3 returns a binary-weighted decimal representing the bits set in the enable register.

**Usage example** Set INST1 (bit 1) and INST2 (bit 2):  
 STAT:QUES:INST:ENAB 6

**Related Commands** \*CLS

**5.16.17. STATus:QUEStionable:INSTrument:ISUMmary[<n>][:EVENT]?** 

**Syntax** STATus:QUEStionable:INSTrument:ISUMmary[<n>][:EVENT]?

**Description** Return the value of the Questionable Instrument Isummary Event register for a specific channel of the BB3 represented by numeric value [<n>]. When [<n>] is omitted, the system queries the questionable instrument Isummary enable register of the current channel. The event register is a read-only register which holds (latches) all events. Reading the Questionable Instrument Isummary Event register clears it. The \*CLS command can be also used to clear the register.

When the power module is operating as a voltage source, bit 1 (CURRENT) is set. When the power module is operating as a current source, bit 0 (VOLTage) is set. When the output is unregulated (UR), both bits are set (for example, while the output is changing to a new programmed value or when the power module is sinking instead of sourcing because down-programmer is active with battery with higher voltage then set output is connected).

**Return** The BB3 returns a binary-weighted decimal representing the bits set in the enable register. See table in the [Section 3.4.2](#) for bits description.

**Usage example** Result of the query when over-current protection (OCP) condition is detected (bit 9):  
 STAT:QUES:INST:ISUM1?  
 512

**Related Commands** \*CLS

**5.16.18. STATus:QUEStionable:INSTrument:ISUMmary[<n>]:CONDition?** 

**Syntax** STATus:QUEStionable:INSTrument:ISUMmary[<n>]:CONDition?

**Description** Return the value of the Questionable Instrument Isummary Condition register for a specific channel of the BB3 represented by numeric value [<n>]. When [<n>] is omitted, the system queries the questionable instrument Isummary enable register of the current channel.

When the power module is operating as a voltage source, bit 1 (CURRENT) is set. When the power module is operating as a current source, bit 0 (VOLTage) is set. When the out-

put is unregulated (UR), both bits are set (for example, while the output is changing to a new programmed value or when the power module is sinking instead of sourcing because down-programmer is active with battery with higher voltage then set output is connected).

**Return** The BB3 returns a binary-weighted decimal representing the bits set in the enable register. See table in the [Section 3.4.2](#) for bits description.

**Usage example** Result of the query when over-current protection (OCP) condition is detected (bit 9):

```
STAT:QUES:INST:ISUM1:COND?
```

```
512
```

#### 5.16.19. STATus:QUESTionable:INSTrument:ISUMmary[<n>]:ENABLE

**Syntax** STATus:QUESTionable:INSTrument:ISUMmary[<n>]:ENABLE {<value>}  
STATus:QUESTionable:INSTrument:ISUMmary[<n>]:ENABLE?

**Description** Set the value of the Questionable Instrument Isummary Enable register for a specific channel of the BB3 represented by numeric value [<n>]. When [<n>] is omitted, the system queries the Questionable Instrument Isummary Enable register of the current channel. The \*CLS command can be used to clear the register.

This register is a mask for enabling specific bits from the Questionable Instrument Isummary Event register to set the Instrument Summary bit (bits 1 and 2) of the Questionable Instrument register. These bits are the logical OR of all the Questionable Instrument Isummary Event register bits that are enabled by the Questionable Instrument Isummary Enable register.

Parameters	Name	Type	Range	Default
	<value>	NR1	A decimal value which corresponds to the binary-weighted sum of the bits in the register (see table in <a href="#">Section 3.4.2</a> )	PREset=0

**Return** Query the value of the Questionable Instrument Isummary Enable register.

**Usage example** Enable bits for all events on channel 2 – VOLT2 (bit 0, value=1), CURR (bit 1, decimal value=2), TEMP2 (bit 4, value 16), OVP1 (bit 8, value=256), OCP2 (bit 9, value=512), OPP2 (bit 10, value=1024), therefore the enable value is  $1 + 2 + 16 + 256 + 512 + 1024 = 1811$ :

```
STAT:QUES:INST:ISUM2:ENAB 1811
```

**Related Commands** \*CLS  
STATus:PREset

## 5.17. SYSTem

System commands control system functions that are not directly related to output control, measurement, or status functions.

SCPI command	Description
SYSTem	
<a href="#">:BEEPer[:IMMediate]</a>	Issues a single beep immediately
<a href="#">:KEY</a>	
<a href="#">:STATe {&lt;bool&gt;}</a>	Enables click tone for local control
<a href="#">:STATe {&lt;bool&gt;}</a>	Enables beeper function
<a href="#">:CAPability?</a>	Returns an <instrument_specifier>
CHANnel	
<a href="#">[:COUNT]?</a>	Returns the number of output channels
INFOrmation	
<a href="#">:CURRent?</a>	Returns output current capability
<a href="#">:ONTime</a>	
<a href="#">LAST?</a>	Returns time passed after last output enable
<a href="#">TOTal?</a>	Returns channel's total active time
<a href="#">:POWER?</a>	Returns output power capability
<a href="#">:VOLTage?</a>	Returns output voltage capability
<a href="#">:MODEl?</a>	Returns the channel model and version name
<a href="#">:OPTion? [{&lt;channel&gt;}]</a>	Returns names of all channel resources
<a href="#">:SLOT?</a>	Returns the channel slot number
<a href="#">:SNO?</a>	Returns the channel serial number
<a href="#">:VERSion?</a>	Returns the channel version number
COMMunicate	
<a href="#">:ENABLE {&lt;bool&gt;}, {&lt;interface&gt;}</a>	Enables the remote interface
ETHernet	
<a href="#">:ADDRess {&lt;ip_address&gt;}</a>	Sets the static LAN (IP) address
<a href="#">:CONTRol?</a>	Returns the control connection port
<a href="#">:DHCP {&lt;bool&gt;}</a>	Enables the use of the Dynamic Host Configuration Protocol (DHCP)
<a href="#">:DNS {&lt;ip_address&gt;}</a>	Sets the IP address of the DNS server.
<a href="#">:GATEway {&lt;ip_address&gt;}</a>	Sets the IP address of the default gateway
<a href="#">:HOSTname {&lt;name&gt;}</a>	Sets the Ethernet communication host name
<a href="#">:MAC?</a>	Returns the MAC address
<a href="#">:PORT {&lt;number&gt;}</a>	Sets the port number
<a href="#">:SMASk {&lt;mask&gt;}</a>	Sets the static subnet mask
<a href="#">:NTP {&lt;server&gt;}</a>	Set s NTP service server address
MQTT	
<a href="#">:SETTings {&lt;address&gt;, &lt;port&gt;, &lt;user&gt;, &lt;password&gt;, &lt;period&gt;}</a>	Sets MQTT connection parameters
<a href="#">:STATe?</a>	Returns MQTT connection status



:USB	
:CLAss {<usb_class>}	Sets USB class
:MODE {<usb_mode>}	Sets USB mode
:RLState {<state>}	Places the instrument in remote or local mode
:CPU	
:FIRMWare?	Returns BB3 firmware version
:INFOrmation	
:ONtime	
LAST?	Returns time passed after last power on
TOTal?	Returns the BB3 total active time
:MODEl?	Returns the control board model name
:SNO?	Returns the BB3 serial number
:VERSIon?	Returns the BB3 version number
:DATE {<year>}, {<month>}, {<day>}	Sets the date of the system clock
DELAy {<time>}	Sets a pause in SCPI execution
:DIGital	
:INPut:DATA? {<pin>}	Reads the state of the digital port pins
:OUTPut	
:DATA {<pin>}, {<state>}	Sets the state of the digital port pins
:PWM	
:DUTY {<pin>}, {<duty>}	Sets square wave generator duty cycle
:FREQuency {<pin>}, {<frequency>}	Sets square wave generator frequency
:PIN<n>	
:FUNCTion {<function>}	Sets the selected pin's function
:POLarity {<polarity>}	Sets the selected pin's polarity
:ERRor	
[:NEXT]?	Queries and clears errors from the error queue
:COUNT?	Queries the error/event queue for the number of unread items
:FAN	
:SPEEd	Returns speed of the cooling fan
:STATus	Returns status of the cooling fan
:FORMat	
:DATE	Sets format for displaying date
:TIME	Sets 12h or 24h clock format
:INHibit?	Queries system inhibit state
:KEY	
:KLOCK	Disables front panel [lock/unlock] icon
:LOCAl	Places the BB3 in the local mode
:MEASure	
[:SCALar]	

:TEMPerature	
[:DC]? {<sensor>}	Takes a measurement; returns the average temperature
[:VOLTage]	
[:DC]? {<device>}	Takes a voltage measurement of the RTC battery
:PASSword	
:CALibration	
:RESet	Resets the calibration password to initial value
:FPANel	
:RESet	Resets the front panel lock password to initial value
:NEW {<old>}, {<new>}	Changes system password
:PON:OUTPut:DISable {<bool>}	Sets output state on power up
:POWer {<bool>}	Enters the BB3 into the standby mode
:PROtection:TRIP {<bool>}	Enters the BB3 into the standby mode in case of protection trip
:RElay:CYCLes? {<chanlist>}	Returns the cycle count on the specified channels
:REMOte	Places the BB3 in the remote mode
:REStart	Initiate low-level (hardware) reset
:RWLock	Places the BB3 in the remote mode and disables front panel [lock/unlock] icon
:SLOT	
[:COUNT]?	Returns number of available slots
:MODEl? {<slot>}	Returns module model name
:STATe {<slot>}, {<bool>}	Enables/disables module on selected slot
:VERSion? {<slot>}	Returns module version number
:TEMPerature	
:PROtection	
[:HIGH]	
[:LEVel] {<temperature>} [, <sensor>]	Sets the OTP value
:CLEar {<sensor>}	Clears the latched protection status of the over-temperature protection (OTP)
:DElay	
[:TIME] {<delay>} [, <sensor>]	Sets the OTP programming delay
:STATe {<bool>} [, <sensor>]	Enables/disables OTP on the selected temperature sensor
:TRIPped? {<sensor>}	Returns status of OTP activation
:TIME {<hours>}, {<minutes>}, {<seconds>}	Sets the time of the system clock
:DTS {rules}	Defines daylight saving time (DST) rules
:ZONE {zone}	Defines time zone
:VERSion?	Returns the SCPI version number

### 5.17.1. SYSTem:BEEPer

**Syntax**      SYSTem:BEEPer[:IMMediate]

**Description** This command issues a single beep immediately.

**Usage example** SYST:BEEP

### 5.17.2. SYSTem:BEEPer:KEY:STATe

**Syntax** SYSTem:BEEPer:KEY:STATe {<bool>}  
SYSTem:BEEPer:KEY:STATe?

**Description** Use this command to enable or disable generation of audible “click” sound when front panel option is selected.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	OFF

**Usage example** SYST:BEEP:KEY:STAT ON

**Related Commands** SYSTem:BEEPer[:IMMediate]  
SYSTem:BEEPer:STATe

### 5.17.3. SYSTem:BEEPer:STATe

**Syntax** SYSTem:BEEPer:STATe {<bool>}  
SYSTem:BEEPer:STATe?

**Description** When the beeper is enabled, the BB3 generates audible sound in any of the following situations:

- the power is turns on or off (see SYSTem:POWer),
- when error occurs during front panel operation or remote operation (see [Section 7](#) for the list of error messages),
- self-test is failed and
- any of the protection function is “tripped”

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	ON

**Usage example** SYST:BEEP:STAT ON  
SYST:BEEP:STAT?  
1

**Related Commands** SYSTem:BEEPer[:IMMediate]  
SYSTem:POWer

### 5.17.4. SYSTem:CAPability?

**Syntax** SYSTem:CAPability?

**Description** This query returns the BB3’s capabilities and outputs the appropriate specifiers. See also SCPI Volume 4: Section 7.1, 1.4.1, 7.2.1, 7.2.2, and 7.2.3

**Usage example** SYST:CAP?  
DCPSUPPLY WITH (MEASURE|MULTIPLE|TRIGGER)

### 5.17.5. SYSTem:CHANnel[:COUNT]?

**Syntax** SYSTem:CHANnel[:COUNT]?

**Description** This query returns the number of output channels in a mainframe.

**Usage example** SYST:CHAN?  
2

**Related** INSTRUMENT[:SELECT]  
**Commands** INSTRUMENT:NSELECT

#### 5.17.6. SYSTem:CHANnel:INFOrmation:CURRent?

**Syntax** SYSTem:CHANnel:INFOrmation:CURRent? [<channel>]

**Description** Use this query to get currently selected channel output current capability.

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4  CH5 CH6	–

**Usage example**  
 SYST:CHAN:INFO:CURR? CH2  
 5.00

#### 5.17.7. SYSTem:CHANnel:INFOrmation:ONTime:LAST?

**Syntax** SYSTem:CHANnel:INFOrmation:ONTime:LAST? [<channel>]

**Description** This query returns time passed after last activation of the currently selected channel. Resolution is 1 minute and this information is stored every 10 minutes in non-volatile memory. Therefore it's possible that up to 10 minutes is lost after restart caused with power outage or system reset.

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4  CH5 CH6	–

**Usage example**  
 SYST:CHAN:INFO:ONT:LAST?  
 15m

#### 5.17.8. SYSTem:CHANnel:INFOrmation:ONTime:TOTal?

**Syntax** SYSTem:CHANnel:INFOrmation:ONTime:TOTal? [<channel>]

**Description** This query returns total active time of the currently selected channel. Resolution is 1 minute and this information is stored every 10 minutes in non-volatile memory. Therefore it's possible that up to 10 minutes is lost after restart caused with power outage or system reset.

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4  CH5 CH6	–

**Usage example**  
 SYST:CHAN:INFO:ONT:TOT?  
 1h 45m

#### 5.17.9. SYSTem:CHANnel:INFOrmation:POWer?

**Syntax** SYSTem:CHANnel:INFOrmation:POWer? [<channel>]

**Description** Use this query to get currently selected channel output power capability.

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4  CH5 CH6	–

**Usage example**  
 SYST:CHAN:INFO:POW?  
 160.00

**5.17.10. SYSTem:CHANnel:INFOrmation:VOLTage?****Syntax** SYSTem:CHANnel:INFOrmation:VOLTage? [<channel>]**Description** Use this query to get currently selected channel output voltage capability.

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4 CH5 CH6	–

**Usage example**

```
SYST:CHAN:INFO:VOLT?
40.00
```

**5.17.11. SYSTem:CHANnel:MODEl?****Syntax** SYSTem:CHANnel:MODEl? [<channel>]**Description** This query returns the model name and version of the specified channel.

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4 CH5 CH6	–

**Usage example**

```
SYST:CHAN:MOD?
"DCP405"
```

**Related Commands**

```
SYSTem:CHANnel:VERSion?
SYSTem:CPU:MODEl?
```

**5.17.12. SYSTem:CHANnel:OPTion?****Syntax** SYSTem:CHANnel:OPTion? [<channel>]**Description** This query returns names of all selected channel resources that can be controlled by firmware. Depending of the board model (see the SYSTem:CHANnel:MODEl? query) various combination of the following features can be returned:

- Volt – program the output voltage while channel is in the CV mode of operation (see [SOURce[<n>]]:VOLTage and APPLy commands)
- Current – program the output current while channel is in the CC mode of operation (see [SOURce[<n>]]:CURRent and APPLy commands)
- Power – set max. allowed output power regardless of the channel mode of operation (see [SOURce[<n>]]:POWer:LIMit)
- OE – set channel power output (see OUTPut[::STATe])
- Dprog – control down-programmer circuit (see OUTPut:DPRog)
- Rprog – control output voltage programming source (see [SOURce[<n>]]:VOLTage:PROGram[:SOURce])
- Coupled – power outputs supports advanced coupling, i.e. in series and parallel (see

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4 CH5 CH6	–

**Usage example**

Reported resources for channel 1 on the DCP405 module:

```
SYST:CHAN:OPT? CH1
"Volt", "Current", "Power", "OE", "DProg", "Rprog", "Coupled"
```

Reported resources for channel 3 on the DCM220 module:

```
SYST:CHAN:OPT? CH1
```

"Volt", "Current", "Power", "OE"

**Related Commands** APPLY  
 OUTPut:DPRog  
 OUTPut:MODE?  
 OUTPut[:STATe]  
 OUTPut:TRACk[:STATe] {<chanlist>}  
 [SOURce[<n>]]:CURRent  
 [SOURce[<n>]]:POWer:LIMit  
 [SOURce[<n>]]:VOLTage  
 [SOURce[<n>]]:VOLTage:PROGram[:SOURce])  
 SYSTem:CHANnel:MODEl?

#### 5.17.13. SYSTem:CHANnel:SLOT?

**Syntax** SYSTem:CHANnel:SLOT? [<channel>]

**Description** This query returns the DIB backplane slot number of the specified channel.

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4  CH5 CH6	—

**Usage example** SYST:CHAN:SLOT? CH4  
 3

**Related Commands** SYSTem:CHANnel:SNO?  
 SYSTem:CHANnel:VERSion?

#### 5.17.14. SYSTem:CHANnel:SNO?

**Syntax** SYSTem:CHANnel:SNO? [<channel>]

**Description** This query returns the serial number of the specified channel.

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4  CH5 CH6	—

**Usage example** SYST:CHAN:SNO? CH4  
 "002F0036434E510A20373437"

**Related Commands** SYSTem:CHANnel:SLOT?  
 SYSTem:CHANnel:VERSion?

#### 5.17.15. SYSTem:CHANnel:VERSion?

**Syntax** SYSTem:CHANnel:VERSion? [<channel>]

**Description** This query returns the version number of the specified channel.

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4  CH5 CH6	—

**Usage example** SYST:CHAN:VERS? CH4  
 "R2B6"

**Related Commands** SYSTem:CHANnel:SLOT?  
 SYSTem:CHANnel:VERSion?

**5.17.16. SYSTem:COMMunicate:ENABLE**

**Syntax** SYSTem:COMMunicate:ENABLE {<bool>}, {<interface>}  
 SYSTem:COMMunicate:ENABLE? {<interface>}

**Description** Enables or disables USB, Ethernet, MQTT, NTP service, or the remote service Sockets. The setting is effective after rebooting the BB3. This command setting is not changed by power off or the [\\*RST](#) command.

*Enabling the Ethernet interface (SYSTem:COMMunicate:ENABLE 1, ETH) will have no effect until a restart is performed using the SYSTem:REStart command.*

*Remote service Sockets is not supported yet*

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	–
	<interface>	Discrete	USB ETHernet MQTT NTP SOCKets	–

**Return** This query returns the status of the selected interface that could be 0 (OFF) or 1 (ON).

**Usage example** SYST:COMM:ENAB 0, SER

**Related Commands** SYSTem:REStart

**5.17.17. SYSTem:COMMunicate:ETHernet:ADDRess**

**Syntax** SYSTem:COMMunicate:ETHernet:ADDRess {<ip\_address>}  
 SYSTem:COMMunicate:ETHernet:ADDRess?

**Description** Set the IP address of the BB3 manually if the DHCP mode is not enabled. If the DHCP mode is enabled setting the IP address using this command will be ignored.

Parameters	Name	Type	Range	Default
	<ip_address>	Quoted string	15 characters max. formatted as four groups of up to 3 digits (range 0–255), separated by “.”	–

**Return** The query returns the current IP address sets manually or assigned by the DHCP server.

**Usage example** SYST:COMM:ETH:ADDR?  
 "192.168.10.100"

**Related Commands** SYSTem:COMMunicate:ETHernet:DHCP

**5.17.18. SYSTem:COMMunicate:ETHernet:DHCP**

**Syntax** SYSTem:COMMunicate:ETHernet:DHCP {<bool>}  
 SYSTem:COMMunicate:ETHernet:DHCP?

**Description** Enable or disable the DHCP mode. In DHCP mode, the DHCP server in the current network assigns network parameters (IP Address, DNS address, GATEway address and the Subnet MASK) for the BB3.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	ON

**Return** Query the status of the DHCP mode.

**Usage example** SYST:COMM:ETH:DHCP?  
 1

**5.17.19. SYSTem:COMMunicate:ETHerneT:DNS**

**Syntax** SYSTem:COMMunicate:ETHerneT:DNS {<ip\_address>}  
SYSTem:COMMunicate:ETHerneT:DNS?

**Description** Set the DNS (Domain Name Service) address if the DHCP mode is not enabled. If the DHCP mode is enabled setting the DNS address using this command will be ignored.

Parameters	Name	Type	Range	Default
	<ip_address>	Quoted string	15 characters max. formatted as four groups of up to 3 digits (range 0–255), separated by “.”	–

**Return** The query returns the DNS address sets manually or assigned by the DHCP server. If DHCP is used query will return “unknown”.

**Usage example** SYST:COMM:ETH:DNS "192.168.1.200"

**Related Commands** SYSTem:COMMunicate:ETHerneT:DHCP

**5.17.20. SYSTem:COMMunicate:ETHerneT:GATEway**

**Syntax** SYSTem:COMMunicate:ETHerneT:GATEway {<ip\_address>}  
SYSTem:COMMunicate:ETHerneT:GATEway?

**Description** Set the network gateway address if the DHCP mode is not enabled. If the DHCP mode is enabled setting the network gateway address using this command will be ignored.

Parameters	Name	Type	Range	Default
	<ip_address>	Quoted string	15 characters max. formatted as four groups of up to 3 digits (range 0–255), separated by “.”	–

**Return** The query returns the network gateway address sets manually or assigned by the DHCP server. If DHCP is used query will return “unknown”.

**Usage example** SYST:COMM:ETH:GATE?  
"192.168.10.1"

**Related Commands** SYSTem:COMMunicate:ETHerneT:DHCP

**5.17.21. SYSTem:COMMunicate:ETHerneT:HOSTname**

**Syntax** SYSTem:COMMunicate:ETHerneT:HOSTname {<hostname>}  
SYSTem:COMMunicate:ETHerneT:HOSTname?

**Description** This command sets the BB3 local area network (LAN) connection hostname.

Parameters	Name	Type	Range	Default
	<hostname>	Quoted string	1 – 63 characters	EEZ-BB3

**Return** The query returns the BB3 local area network (LAN) connection hostname as a quoted string.

**Usage example** SYST:COMM:ETH:HOST?  
"bb3-test"

**Related Commands** SYSTem:COMMunicate:ETHerneT:ADDReSS  
SYSTem:COMMunicate:ETHerneT:DHCP



**5.17.22. SYSTem:COMMunicate:ETHernet:MAC**

**Syntax** SYSTem:COMMunicate:ETHernet:MAC {<mac\_address>}  
SYSTem:COMMunicate:ETHernet:MAC?

**Description** Use this command to set Ethernet communication port MAC address. You can set any combination of six hexadecimal values separated by "-". The Ethernet connection will work as long as two different machine in the LAN don't have the same MAC address.

Parameters	Name	Type	Range	Default
	<mac_address>	Quoted string	17 characters formatted as six groups of 2 digits separated by "-"	74-69-69-2D-30-00

**Return** The query returns the MAC address as a quoted string (six hexadecimal values separated by "-").

**Usage example** SYST:COMM:ETH:MAC?  
"70-60-50-40-30-20"

**5.17.23. SYSTem:COMMunicate:ETHernet:PORT**

**Syntax** SYSTem:COMMunicate:ETHernet:PORT {<number>}  
SYSTem:COMMunicate:ETHernet:PORT?

**Description** Use this command to change default (5025) Ethernet communication port for SCPI.

Parameters	Name	Type	Range	Default
	<number>	NR1	1-65535	5025

**Return** The query returns the Ethernet communication port number.

**Usage example** SYST:COMM:ETH:PORT?  
5025

**Related Commands** SYSTem:COMMunicate:ENABLE  
SYSTem:COMMunicate:ETHernet:CONTRol?

**5.17.24. SYSTem:COMMunicate:ETHernet:SMASk**

**Syntax** SYSTem:COMMunicate:ETHernet:SMASk {<mask>}  
SYSTem:COMMunicate:ETHernet:SMASk?

**Description** Set the subnet mask if the DHCP mode is not enabled. If the DHCP mode is enabled setting the network gateway address using this command will be ignored.

Parameters	Name	Type	Range	Default
	<mask>	Quoted string	15 characters max. formatted as four groups of up to 3 digits (range 0-255), separated by "."	-

**Return** The query returns the subnet mask sets manually or assigned by the DHCP server. If DHCP is used query will return "unknown".

**Usage example** SYST:COMM:ETH:SMAS "255.255.255.0"

**Related Commands** SYSTem:COMMunicate:ETHernet:DHCP

**5.17.25. SYSTem:COMMunicate:MQTT:SETTings**

**Syntax** SYSTem:COMMunicate:MQTT:SETTings {<address>}, {<port>}, {<user>}, {<password>}, {<period>}

**Description** [MQTT](#) (MQ Telemetry Transport) is an open OASIS and ISO standard (ISO/IEC PRF 20922) lightweight, publish-subscribe network protocol that transports messages between devices (i.e. clients).

The EEZ BB3 becomes a MQTT client (or node) by establishing connection with this command to the MQTT broker. A broker is a server that receives all messages from the clients and then routes the messages to the appropriate destination clients. It acts as a post office, MQTT doesn't use the address of the intended recipient but uses the subject line called "Topic", and anyone who wants a copy of that message will subscribe to that topic. EEZ The BB3 can produce and receive data by both publishing and subscribing. MQTT connection is initiated by sending a sequence of five comma separated connection parameters.

Parameters	Name	Type	Range	Default
	<address>	Quoted string	Max. 64 characters	–
	<port>	NR1	1 – 99999	–
	<user>	Quoted string	Max. 32 characters	–
	<password>	Quoted string	Max. 32 characters	–
	<period>	NR2	0.1 – 120	1
<b>Usage example</b>	SYST:COMM:MQTT:CONN "farmer.cloudmqtt.com", 1883, "user", "password", 10			
<b>Errors</b>	-104,"Data type error"			
<b>Related Commands</b>	SYSTem:COMMunicate:ENABLE SYSTem:COMMunicate:MQTT:STATe			

#### 5.17.26. [SYSTem:COMMunicate:MQTT:STATe](#)

**Syntax** [SYSTem:COMMunicate:MQTT:STATe?](#)

**Description** This query returns the current status of the MQTT connection.

**Return** Returns -1 if an error occurred, 0 if connection is not established, 1 if connection is established. Return codes greater than 1 indicates various transient states.

**Usage example**  
SYST:COMM:MQTT:STAT?  
1

**Related Commands** SYSTem:COMMunicate:ENABLE  
SYSTem:COMMunicate:MQTT:SETTings

#### 5.17.27. [SYSTem:COMMunicate:NTP](#)

**Syntax** [SYSTem:COMMunicate:NTP {<server>}](#)  
[SYSTem:COMMunicate:NTP?](#)

**Description** Use this command to set the NTP service server network address. The BB3 will try to establish connection with selected NTP service on each power up (hard reset), when \*RST is issued or once per day (24h as defined with [CONF\\_NTP\\_PERIOD\\_SEC](#) parameter in firmware)

Parameters	Name	Type	Range	Default
	<server>	Quoted string	IP address or NTP server host name up to 32 characters	europe.pool.ntp.org
<b>Return</b>	The query command returns the NTP service server network address.			
<b>Usage example</b>	SYST:COMM:ENAB NTP SYST:COMM:NTP "ntp.ubuntu.com"			

**Related Commands** \*RST  
 SYSTem:COMMunicate:ENABLE  
 SYSTem:DATE  
 SYSTem:TIME

#### 5.17.28. SYSTem:COMMunicate:USB:CLAss

**Syntax** SYSTem:COMMunicate:USB:CLAss {<usb\_class>}  
 SYSTem:COMMunicate:USB:CLAss?

**Description** Use this command to set the USB interface class.

Parameters	Name	Type	Range	Default
	<usb_class>	Discrete	VCOM MSTorage HID	–

**Return** The query command returns the USB interface class.

**Usage example** SYST:COMM:USB:MODE "DEV"  
 SYST:COMM:USB:CLA "VCOM"

**Related Commands** \*RST  
 SYSTem:COMMunicate:ENABLE  
 SYSTem:COMMunicate:USB:MODE

#### 5.17.29. SYSTem:COMMunicate:USB:MODE

**Syntax** SYSTem:COMMunicate:USB:MODE {<usb\_mode>}  
 SYSTem:COMMunicate:USB:MODE?

**Description** Use this command to set the USB interface mode that could be one of the following:

- DEVIce – the BB3 acts as an USB device when communicate with connected computer.
- HOSt – in this mode the BB3 provides VBUS power for connected USB device (e.g. mouse, keypad, foot pedal, etc.). The USB class will be automatically set to HID.
- OTG – “on-the-go” mode allows detection of USB ID pin (set by USB cable or adapter) and adopt its USB mode and class in accordance with detected device.

Parameters	Name	Type	Range	Default
	<usb_mode>	Discrete	DEVIce HOSt OTG	–

**Return** The query command returns the USB interface mode.

**Usage example** SYST:COMM:USB:MODE "HOST"

**Related Commands** \*RST  
 SYSTem:COMMunicate:ENABLE  
 SYSTem:COMMunicate:USB:CLAss

#### 5.17.30. SYSTem:COMMunicate:RLState

**Syntax** SYSTem:COMMunicate:RLState {<state>}  
 SYSTem:COMMunicate:RLState?

**Description** This command configures the remote/local state of the BB3 according to the following settings:

- LOCAl – The BB3 is set to front panel and remote interface control.
- REMote – The BB3 is set to front panel and remote interface control.
- RWLock – The front panel keys are disabled. The BB3 can only be controlled via the remote interface. This programmable setting is completely independent from

the front panel lock/unlock function that is available from the front panel menu. If you use this command to lock the front panel, the front panel will be unlocked when AC power is cycled.

The LOCal parameter is the same as SYSTem:LOCaL, the REMote parameter is the same as SYSTem:REMote, and the RWLock parameter is the same as SYSTem:RWLock.

The remote/local state is unaffected by \*RST or any SCPI commands other than SYSTem:COMMunicate:RLState.

Parameters	Name	Type	Range	Default
	<state>	Discrete	LOCaL REMote RWLock	LOCaL
<b>Return</b>	The query command returns control state of the BB3 that could be LOC, REM, or RWL.			
<b>Usage example</b>	SYST:COMM:RLST? "LOC"			
<b>Related Commands</b>	*RST SYSTem:LOCaL SYSTem:REMote SYSTem:RWLock			

#### 5.17.31. SYSTem:CPU:FIRMware?

**Syntax** SYSTem:CPU:FIRMware?

**Description** This query returns BB3 (master) firmware version.

**Usage example** SYST:CPU:FIRM?  
"1.0"

#### 5.17.32. SYSTem:CPU:INFOrmation:ONTime:LAST?

**Syntax** SYSTem:CPU:INFOrmation:ONTime:LAST?

**Description** This query returns time passed after last activation of the BB3. Resolution is 1 minute and this information is stored every 10 minutes in non-volatile memory. Therefore it's possible that up to 10 minutes is lost after restart caused with power outage or system reset.

**Usage example** SYST:CPU:INFO:ONT:LAST?  
15m

#### 5.17.33. SYSTem:CPU:INFOrmation:ONTime:TOTal?

**Syntax** SYSTem:CPU:INFOrmation:ONTime:TOTal?

**Description** This query returns total active time of the BB3. Resolution is 1 minute and this information is stored every 10 minutes in non-volatile memory. Therefore it's possible that up to 10 minutes is lost after restart caused with power outage or system reset.

**Usage example** SYST:CPU:INFO:ONT:TOT?  
"1h 45m"

#### 5.17.34. SYSTem:CPU:MODEl?

**Syntax** SYSTem:CPU:MODEl?

**Description** This query returns the model name and version of the MCU board. If simulator is used it returns "Simulator".

**Usage** SYST:CPU:MOD?

**example** "STM32F7, M1 0.3"

**Related Commands** SYSTem:CPU:INfOrmation:ONTime:LAST?  
SYSTem:CPU:INfOrmation:ONTime:TOTal?

#### 5.17.35. SYSTem:CPU:SNO?

**Syntax** SYSTem:CPU:SNO?

**Description** This query returns the BB3 serial number.

**Usage example** SYST:CPU:SNO?  
"002A003D3338510738323535"

**Related Commands** \*IDN?

#### 5.17.36. SYSTem:CPU:VERSion?

**Syntax** SYSTem:CPU:VERSion?

**Description** This query returns the BB3 version number.

**Usage example** SYST:CPU:VERS?  
"R2B4"

**Related Commands** \*IDN?

#### 5.17.37. SYSTem:DATE

**Syntax** SYSTem:DATE {<year>}, {<month>}, {<day>}  
SYSTem:DATE?

**Description** Sets the date of the system clock (RTC). Specify the year, month, and day. The self-test procedure compare date and time stored in RTC registers with values stored in the non-volatile memory (EEPROM). When the later is greater then former or any of them lost integrity (i.e. any of value is outside allowed range: for example seconds are higher then 60 or months are higher then 12, etc.) self-test will failed. The \*TST? will return 1 and detailed report could be queried using the DIAGnostic:TEST? command.

The bit 3 (TIME) of the Questionable Status register will be set (see [Section 3.4](#)) if date-time self-test failed or datetime was never set.

Parameters	Name	Type	Range	Default
	<year>	NR1	2000 – 2099	–
	<month>	NR1	1 – 12	–
	<day>	NR1	1 – 31	–
<b>Return</b>	Query the current date of the system clock in YYYY, MM, DD format.			
<b>Usage example</b>	SYST:DATE? 2015, 10, 24			
<b>Related Commands</b>	*TST? DIAGnostic[:INfOrmation]:TEST? SYSTem:FORMat:TIME SYSTem:TIME			

#### 5.17.38. SYSTem:DELay

**Syntax** SYSTem:DELay {<time>}

**Description** This command inserts pause in parsing and executing SCPI commands. Execution of the

next SCPI command will start only when the set delay expires. The programmed value can be up to 10 seconds (set in milliseconds).

Parameters	Name	Type	Range	Default
	<time>	NR1	1 – 10000	–
<b>Usage example</b>	SYST:DEL 1500			

#### 5.17.39. SYSTem:DIgital:INPut:DATA

**Syntax** SYSTem:DIgital:INPut:DATA? {<pin>}

**Description** This query reads the state of the digital control port.  
*Applies only to pin 1.*

Parameters	Name	Type	Range	Default
	<pin>	NR1	1	1

**Return** The query returns the value of the state of input pin.

**Usage example**  
SYST:DIG:INP:DATA? 1  
1

**Related Commands** SYSTem:DIgital:PIN<n>:FUNctIon  
SYSTem:DIgital:PIN<n>:POLarity

#### 5.17.40. SYSTem:DIgital:OUTPut:DATA

**Syntax** SYSTem:DIgital:OUTPut:DATA {<pin>}, {<state>}  
SYSTem:DIgital:OUTPut:DATA? {<pin>}

**Description** This command sets the output data on the digital output pin. Applies only to pin 3 and 4.

Parameters	Name	Type	Range	Default
	<pin>	NR1	3 – 4	–
	<state>	Discrete	ON OFF 0 1	–

**Return** The query returns the last programmed value on the selected pin.

**Usage example**  
SYST:DIG:OUTP:DATA 3, 0

**Errors** -114, "Header suffix out of range"

**Related Commands** SYSTem:DIgital:PIN<n>:FUNctIon  
SYSTem:DIgital:PIN<n>:POLarity

#### 5.17.41. SYSTem:DIgital:OUTPut:PWM:DUTY

**Syntax** SYSTem:DIgital:OUTPut:PWM:DUTY {<pin>}, {<duty>}  
SYSTem:DIgital:OUTPut:PWM:DUTY? {<pin>}

**Description** Use this command to set duty cycle of the output pin defined as PWM (see the SYSTem:DIgital:PIN<n>:FUNctIon command). Applies only to 4.

Parameters	Name	Type	Range	Default
	<pin>	NR1	4	–
	<duty>	NR3	0 – 100	50

**Return** The query returns the square wave generator duty cycle for the specified output pin.

**Usage example**  
SYST:DIG:OUTP:PWM:DUTY 4, 10

**Errors** -224, "Illegal parameter value"

-230,"Digital pin function mismatch"

**Related Commands** SYSTem:DIGital:OUTPut:PWM:FREQuency  
SYSTem:DIGital:PIN<n>:FUNCTion

#### 5.17.42. SYSTem:DIGital:OUTPut:PWM:FREQuency

**Syntax** SYSTem:DIGital:OUTPut:PWM:FREQuency {<pin>}, {<frequency>}  
SYSTem:DIGital:OUTPut:PWM:FREQuency? {<pin>}

**Description** Use this command to set frequency of the output pin defined as PWM (see the SYSTem:DIGital:PIN<n>:FUNCTion command). Applies only to 4.

Parameters	Name	Type	Range	Default
	<pin>	NR1	4	–
	<frequency>	NR3	0.03 – 5000000	0

**Return** The query returns the square wave generator frequency for the specified output pin.

**Usage example** SYST:DIG:OUTP:PWM:FREQ 4, 1000

**Errors** -224,"Illegal parameter value"  
-230,"Digital pin function mismatch"

**Related Commands** SYSTem:DIGital:OUTPut:PWM:DUTY  
SYSTem:DIGital:PIN<n>:FUNCTion

#### 5.17.43. SYSTem:DIGital:PIN<n>:FUNCTion

**Syntax** SYSTem:DIGital:PIN<n>:FUNCTion {<function>}  
SYSTem:DIGital:PIN<n>:FUNCTion?

**Description** Use this command to set function of the selected digital port pin. The pin function is saved in non-volatile memory.

*All input functions applies only to pin 1 and 2 and all output functions applies only to pin 3 and pin 4.*

- DINPut – The pin is in digital input mode.
- DLOGtrig – The pin is configured as a DLOG trigger input.
- DOUTput – The pin is in digital output mode.
- FAULT – Setting FAULT means that pin functions as an isolated fault output. The fault signal is true when any output is in a protected state (from OCP, OVP, OTP, OPP) or Fan fault is detected.
- INHibit – When pin is configured as an inhibit input, a true signal at the pin will disable all output channels.
- ONCouple – output pin synchronize channel output state.
- PWM – (*pin 4 only*) square wave generator output.
- SYSTrig – The pin is configured as a system trigger input. When configured as a trigger input, the pin can be selected as the source for trigger signals. See TRIGger[:SEQuence]:SOURce.
- TOUTput – This allows a BUS trigger to be sent to any digital port pin that has been configured as a trigger output. A trigger out pulse is generated when the state is on and a bus trigger is received. A BUS trigger is generated using the \*TRG command.

Parameters	Name	Type	Range	Default
	<function>	Discrete	DINP DLOG DOUT  FAULT INH ONC  PWM SYST TOUT	–

**Return** The query command returns DINP, DOUT, FAULT, INH or TINP.

<b>Usage example</b>	SYST:DIG:PIN1:FUNC INH SYST:DIG:PIN2:FUNC Tinp SYST:DIG:PIN3:FUNC FAUL SYST:DIG:PIN4:FUNC ONC
<b>Errors</b>	-114, "Header suffix out of range"
<b>Related Commands</b>	TRIGger[:SEquence]:SOURce SYSTem:DIGital:INPut:DATA SYSTem:DIGital:OUTPut:DATA SYSTem:DIGital:OUTPut:PWM:DUTY SYSTem:DIGital:OUTPut:PWM:FREQuency

#### 5.17.44. SYSTem:DiGital:PIN<n>:POLarity

<b>Syntax</b>	SYSTem:DiGital:PIN<n>:POLarity {<polarity>} SYSTem:DiGital:PIN<n>:POLarity?
<b>Description</b>	<p>This command sets the polarity of the selected digital port pin. The pin polarity is saved in non-volatile memory.</p> <ul style="list-style-type: none"> <li>POSitive – a logical true signal is a voltage high at the pin. For trigger inputs and outputs, POSitive means a rising edge.</li> <li>NEGative – a logical true signal is a voltage low at the pin. For trigger inputs and outputs, NEGative means a falling edge.</li> </ul>

Parameters	Name	Type	Range	Default
	<polarity>	Discrete	POSitive NEGative	–
<b>Return</b>	The query command returns POS or NEG.			
<b>Usage example</b>	SYST:DIG:PIN1:POL POS SYST:DIG:PIN2:POL NEG			
<b>Errors</b>	-114, "Header suffix out of range"			
<b>Related Commands</b>	TRIGger[:SEquence]:SOURce SYSTem:DIGital:INPut:DATA SYSTem:DIGital:OUTPut:DATA			

#### 5.17.45. SYSTem:ERRor

<b>Syntax</b>	SYSTem:ERRor[:NEXT]?
<b>Description</b>	<p>This query command reads and clear errors from the error queue. A record of up to 20 errors can be stored in the BB3's error queue. See also "Error Messages" in <a href="#">Section 7</a>. Errors are retrieved in first-in-first-out (FIFO) order. The first error returned is the first error that was stored. The BB3 beeps once each time an error is generated. The error queue is cleared when power has been off or after a *CLS command.</p>
<b>Return</b>	SYSTem:ERRor[:NEXT]? queries and clears the error messages in the error queue. The query returns the number and content of the error message.
<b>Usage example</b>	SYST:ERR? -113, "Undefined header"
<b>Errors</b>	<p>If more than 20 errors have occurred, the last error stored in the queue (the most recent error) is replaced with:</p> <p>-350, "Queue overflow"</p> <p>No additional errors are stored until you remove errors from the queue.</p>
<b>Related Commands</b>	*CLS *RST SYSTem:ERRor:COUNT



**5.17.46. SYSTem:ERRor:COUNT?****Syntax** SYSTem:ERRor:COUNT?**Description** This query command queries the error/event queue for the number of unread items. As errors and events may occur at any time, more items may be present in the queue at the time it is actually read.**Usage example**  
SYST:ERR:COUN?  
10**Related Commands**  
\*CLS  
\*RST  
SYSTem:ERRor[:NEXT]**5.17.47. SYSTem:FAN:SPEed?****Syntax** SYSTem:FAN:SPEed?**Description** Use this query to obtain cooling fan speed in rpm.**Usage example**  
SYST:FAN:SPE?  
1450**Related Commands**  
SYSTem:FAN:STATus?**5.17.48. SYSTem:FAN:STATus?****Syntax** SYSTem:FAN:STATus?**Description** Use this query to obtain information about cooling fan state. Cooling fan is periodically tested while it's working (that depends of measured channel's temperature sensor value). When it does not passed the test, programmed output current is automatically limited to 2 A.**Return** Returns numerical status of the cooling fan:

- 0 – fault is detected
- 1 – fan is up and running
- 2 – fan testing is in progress
- 3 – fan is not installed

**Usage example**  
SYST:FAN:STAT?  
1**Related Commands**  
SYSTem:FAN:SPEed?**5.17.49. SYSTem:FORMat:DATE****Syntax** SYSTem:FORMat:DATE {<format>}  
SYSTem:FORMat:DATE?**Description** Set format that will be used for displaying date.

Parameters	Name	Type	Range	Default
	<format>	Discrete	DMY MDY	–

**Return** The query command returns “DMY” or “MDY”.**Usage example**  
SYST:FORM:DATE MDY**Related Commands**  
SYSTem:DATE  
SYSTem:FORMat:DATE

SYSTem:FORMat:TIME

**5.17.50. SYSTem:FORMat:TIME**

**Syntax** SYSTem:FORMat:TIME {<format>}  
SYSTem:FORMat:TIME?

**Description** Set clock format that will be used for displaying time.

Parameters	Name	Type	Range	Default
	<format>	NR1	12 24	—

**Return** The query command returns 12 or 24 clock format.

**Usage example** SYST:FORM:TIME 12

**Related Commands** SYSTem:DATE  
SYSTem:FORMat:DATE  
SYSTem:FORMat:TIME

**5.17.51. SYSTem:INHibit?**

**Syntax** SYSTem:INHibit?

**Description** Use this command to query BB3's inhibit state that is controlled with digital input (see [SOURce]:DIGital:PIN<n>:FUNctIon command). When BB3 is in inhibited state, all channel's output will be disabled (OFF) and triggered action (e.g. LIST) will be paused.

Channel output state command (OUTPut[:STATe]) can be used when BB3 is in inhibited mode but it will not affect output (i.e. change it to enable state).

**Return** Returns 0 if inhibit is not active or 1 if inhibit is active.

**Usage example** SYST:INH?  
0

**Related Commands** OUTPut[:STATe] {<bool>} [, <channel>]  
[SOURce]:DIGital:PIN<n>:FUNctIon

**5.17.52. SYSTem:KLOCK**

**Syntax** SYSTem:KLOCK

**Description** This command similar to the SYSTem:REMOte command disables all front-panel options except for [Lock/Unlock] icon. You can push and hold the [Lock/Unlock] for a few seconds to unlock the front panel. The system password may be needed if it is set. Lock state is saved in non-volatile memory. Therefore, the front panel remains locked even after AC power is cycled.

**Usage example** SYST:KLOC

**Related Commands** SYSTem:COMMunicate:RLState  
SYSTem:REMOte

**5.17.53. SYSTem:LOCal**

**Syntax** SYSTem:LOCal

**Description** This command places the BB3 in the local mode during remote operation. All options on the front panel are fully functional.

**Usage example** SYST:LOC

**Related Commands** SYSTem:COMMunicate:RLState  
SYSTem:REMOte

SYSTem:RWLock

MEASure[:SCALar]:TEMPerature[:THERmistor][:DC]

Syntax MEASure[:SCALar]:TEMPerature[:THERmistor][:DC]? {&lt;sensor&gt;}

Description Query the temperature measured using the specified temperature sensors as follows:  
 AUX – auxiliary temperature sensor (built-in fan controller) used to monitor temperature inside BB3 chassis  
 CH1, CH2, CH3, CH4, CH5, CH6 – temperature sensor connected to channel power module used to regulate fan speed  
 All installed sensors are periodically tested. When sensor does not passed the test, programmed output current is automatically limited to 2 A. If load that draws more current then 2 A output current will be set to zero.

Parameters	Name	Type	Range	Default
	<sensor>	Discrete	AUX CH1 CH2 CH3 CH4 CH5 CH6	AUX
Return	Returns the average temperature value in degrees Celsius (°C) as decimal number (NR2). If self-test detect selected temperature sensor failure or sensor is not installed a -240 or -241 error will be generated.			
Usage example	ex- MEAS:TEMP? CH2 49			
Errors	-240,"Hardware error" -241,"Hardware missing"			
Related Commands	DIAGnostic[:INFormation]:FAN? SYSTem:CPU:OPTion? SYSTem:TEMPerature:PROTection[:HIGH][:LEVel] SYSTem:TEMPerature:PROTection[:HIGH]:DELay[:TIME] SYSTem:TEMPerature:PROTection[:HIGH]:STATE SYSTem:TEMPerature:PROTection[:HIGH]:TRIPped?			

#### 5.17.54. SYSTem:MEASure[:SCALar]:TEMPerature[:THERmistor][:DC]

Syntax SYSTem:MEASure[:SCALar]:TEMPerature[:THERmistor][:DC]? {&lt;sensor&gt;}

Description Query the temperature measured using the specified temperature sensors as follows:

- AUX – auxiliary temperature sensor (built-in fan controller) used to monitor temperature inside BB3 chassis
- CH1, CH2, CH3, CH4, CH5, CH6 – temperature sensor connected to channel power module used to regulate fan speed

All installed sensors are periodically tested. When sensor does not passed the test, programmed output current is automatically limited to 2 A. If load that draws more current then 2 A output current will be set to zero.

Parameters	Name	Type	Range	Default
	<sensor>	Discrete	AUX CH1 CH2 CH3 CH4 CH5 CH6	AUX
Return	Returns the average temperature value in degrees Celsius (°C) as decimal number (NR2). If self-test detect selected temperature sensor failure or sensor is not installed a -240 or -241 error will be generated.			
Usage example	SYST:MEAS:TEMP? CH2 49			
Errors	-240,"Hardware error" -241,"Hardware missing"			
Related Commands	DIAGnostic[:INFormation]:FAN? SYSTem:CPU:OPTion?			

SYSTem:TEMPerature:PROTection[:HIGH][:LEVel]  
 SYSTem:TEMPerature:PROTection[:HIGH]:DELay[:TIME]  
 SYSTem:TEMPerature:PROTection[:HIGH]:STATE  
 SYSTem:TEMPerature:PROTection[:HIGH]:TRIPped?

#### 5.17.55. SYSTem:MEASure[:SCALar][:VOLTage][:DC]?

**Syntax** SYSTem:MEASure[:SCALar][:VOLTage][:DC]? {<device>}

**Description** Returns voltage of the RTC (Real-time-clock) backup battery.

Parameters	Name	Type	Range	Default
	<device>	Discrete	RTC	RTC
<b>Usage example</b>	SYST:MEAS? RTC 3.07			

#### 5.17.56. SYSTem:PASSword:CALibration:RESet

**Syntax** SYSTem:PASSword:CALibration:RESet

**Description** This command resets the calibration password to the firmware default setting, which is "eezbb3". This command does not reset the system password.

**Usage example** SYST:PASS:CAL:RES

**Related Commands** CALibration[:MODE]  
CALibration:PASSword:NEW

#### 5.17.57. SYSTem:PASSword:FPANel:RESet

**Syntax** SYSTem:PASSword:FPANel:RESet

**Description** This command resets the front panel lockout password to the firmware default setting, which is empty space (""). This command does not reset the calibration password.

**Usage example** SYST:PASS:FPAN:RES

**Related Commands** SYSTem:KLOCK  
SYSTem:PASSword:NEW

#### 5.17.58. SYSTem:PASSword:NEW

**Syntax** SYSTem:PASSword:NEW {<old>}, {<new>}

**Description** Enter a new system password. To change the password, first unsecure the BB3 using the old password. Then, the new code has to be entered. The calibration code may contain up to 16 characters over the remote interface. Minimum length is 4 characters. The new password is automatically stored in non-volatile memory

Parameters	Name	Type	Range	Default
	<old>	Quoted string	0 to 16 characters	—
	<new>	Quoted string	4 to 16 characters	—

**Usage example** SYST:PASS:NEW "", "mypass2016"

**Errors** 122, "Invalid sys password"  
125, "Sys password too long"  
126, "Sys password too short"

**Related Commands** SYSTem:KLOCK

**5.17.59. SYSTem:PON:OUTPut:DISable**

**Syntax** SYSTem:PON:OUTPut:DISable {<bool>}  
SYSTem:PON:OUTPut:DISable?

**Description** This command controls status off all channel outputs on power up. If enabled (ON), all outputs will be disabled regardless of what is stored in user profile selected for auto recall.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	OFF

**Return** Query returns status of forced output disabling on power up.

**Usage example**  
 OUP?  
 1  
 SYST:PON:OUTP:DIS 1  
 (Restart)  
 OUP?  
 0

**Related Commands** MEMory:STATe:RECall:AUTO  
SYSTem:POWer

**5.17.60. SYSTem:POWer**

**Syntax** SYSTem:POWer {<bool>}  
SYSTem:POWer?

**Description** This command controls powering down and powering up sequence of the AC power inputs. The “Standby” indicator will be switched on when the BB3 enters the standby mode.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	OFF

**Return** Query returns BB3 power standby status.

**Usage example**  
 SYST:POW ON  
 SYST:POW?  
 1

**Related Commands** \*RST  
\*TST?  
SYSTem:BEEP:STATe

**5.17.61. SYSTem:POWer:PROTection:TRIP**

**Syntax** SYSTem:POWer:PROTection:TRIP {<bool>}  
SYSTem:POWer:PROTection:TRIP?

**Description** Use this command to shut down (set to standby mode) the BB3 when any of channel's protection tripped. The SYSTem:POWer command has to be used to power the BB3 on again.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	OFF

**Return** Query returns status of shutdown when tripped mode.

**Usage example**  
 SYST:POW:PROT:TRIP?  
 1

**Related** SYSTem:POWer

## Commands

**5.17.62. SYSTem:RELAy:CYCLes?****Syntax** SYSTem:RELAy:CYCLes? {<chanlist>}**Modules** SMX**Description** This query returns the cycle count on the specified channels.  
The SMX46 module channels are mapped in the following way (n is slot number):

- (@n03) – Power relay (max. 250 Vac, 8 A per contact)
- (@nrc) – Signal relay (1 A max., 30 Vdc / 0.3 A, 125 Vac) in switch matrix consists of 4 rows (r value) and 6 columns (c value). For example, use (@n23) to address relay for switch in row 2 and column 3.

*When list of channels separated with comma is specified no space is allowed.*

Parameters	Name	Type	Range	Default
	<chanlist>	ChannelList	See channels mapping described above	–

**Usage example** Query cycle count of power relay on the SMX46 installed in slot 2:

SYST:REL:CYCL? (@203)

44

Query cycle count of multiple signal relays on the SMX46 installed in slot 3:

SYST:REL:CYCL? (@311,312,313)

13,44,1023

**Related Commands** SYSTem:SLOT[:COUNT]?  
SYSTem:SLOT:STATE  
SYSTem:SLOT:VERSion?

**5.17.63. SYSTem:REMOte****Syntax** SYSTem:REMOte**Description** Use this command to place the BB3 into remote mode for USB (Virtual COM) or Ethernet remote control. All front-panel options are disabled except for [Lock/Unlock] icon. You can push and hold the [Lock/Unlock] for a few seconds to unlock the front panel. The system password may be needed if it is set.**Usage example** SYST:REM

**Related Commands** SYSTem:COMMunicate:RLState  
SYSTem:LOCal  
SYSTem:RWLock

**5.17.64. SYSTem:REStart****Syntax** SYSTem:REStart**Description** Use this command to initiate low-level (hardware) reset. When executed the power up procedure will start and currently active SCPI session will be lost.**Usage example** SYST:RES

**Related Commands** \*RST

**5.17.65. SYSTem:RWLock****Syntax** SYSTem:RWLock**Description** Places the BB3 in the remote mode for USB (Virtual COM) or Ethernet remote control. This command is the same as SYSTem:REMOte, except that all front panel options are disabled, including the [Lock/Unlock] icon.**Usage example** SYST:RWL**Related Commands** SYSTem:COMMunicate:RLState  
SYSTem:LOCal  
SYSTem:REMOte**5.17.66. SYSTem:SLOT[:COUNT]?****Syntax** SYSTem:SLOT[:COUNT]?**Description** This query returns the number of available slots in BB3 enclosure. That number is always 3.**Usage example** SYST:SLOT?  
3**Related Commands** SYSTem:SLOT:MODEl?  
SYSTem:SLOT:STATE  
SYSTem:SLOT:VERSion?**5.17.67. SYSTem:SLOT:MODEl?****Syntax** SYSTem:SLOT:MODEl {<slot>}?**Description** This query returns the model name of the module installed into specified slot.

Parameters	Name	Type	Range	Default
	<slot>	NR2	1 – 3	–

**Usage example** SYST:SLOT:MOD? 2  
"DCP405"**Related Commands** SYSTem:SLOT[:COUNT]?  
SYSTem:SLOT:STATE  
SYSTem:SLOT:VERSion?**5.17.68. SYSTem:SLOT:STATE****Syntax** SYSTem:SLOT:STATE {<slot>}, {<bool>}  
SYSTem:SLOT:STATE? [<slot>]**Description** Use this command to enable or disable module installed into specified slot. When disabled an attempt to communicate with module will generate an execution error.

Parameters	Name	Type	Range	Default
	<slot>	NR2	1 – 3	–
	<bool>	Boolean	ON OFF 0 1	–

**Return** Query the status of module on the selected BB3 slot.**Errors** -200, "Execution error"**Usage example** SYST:SLOT:STAT 1,0**Related Commands** \*RST  
DIAGnostic[:INFORMATION]:TEST?  
SYSTem:SLOT[:COUNT]?

SYSTem:SLOT:MODEL?  
SYSTem:SLOT:VERSion?

#### 5.17.69. SYSTem:SLOT:VERSion?

**Syntax** SYSTem:SLOT:VERSion {<slot>}?

**Description** This query returns the version number of the module installed into specified slot.

Parameters	Name	Type	Range	Default
	<slot>	NR2	1 – 3	–

**Usage example**  
SYST:SLOT:VERS? 2  
"R2B11"

**Related Commands**  
\*RST  
DIAGnostic[:INFormation]:TEST?  
SYSTem:CHANnel[:COUNT]?

#### 5.17.70. SYSTem:TEMPerature:PROTection[:HIGH][:LEVel]

**Syntax** SYSTem:TEMPerature:PROTection[:HIGH][:LEVel] {<temperature>} [, <sensor>]  
SYSTem:TEMPerature:PROTection[:HIGH][:LEVel]? [<sensor>]

**Description** Set the over-temperature protection (OTP) value in degrees Celsius (°C) of the selected temperature sensor. When the over-temperature protection function of the specified temperature sensor is enabled (SYSTem:TEMPerature:PROTection[:HIGH]:STATe), one of the following action will be performed when the temperature exceeds the over-temperature protection value currently set:

- AUX – Switch off power of the main transformer and set bit 4 of the Questionable Status register
- CH1, CH2, CH3, CH4, CH5, CH6 – Disable channel output (OUTPut OFF) and set bit 4 of the Questionable Instrument lsummary register.

If any of above mentioned temperature sensors cause over-temperature condition an error tone will also follow if beeper is enabled (see SYSTem:BEEPer:STATe).

SYSTem:TEMPerature:PROTection[:HIGH]:TRIPped? command can be used to query whether over-temperature protection occurred on the selected temperature sensor.

Parameters	Name	Type	Range	Default
	<temperature>	NR2	10 – 100	70 for CH1 to CH2, and 50 for AUX
	<sensor>	Discrete	AUX CH1 CH2  CH3 CH4 CH5 CH6	AUX

**Return** Query the over-temperature protection (OTP) value of the selected temperature sensor.

**Usage example**  
SYST:TEMP:PROT 50, AUX  
SYST:TEMP:PROT?  
50

**Related Commands**  
\*RST  
SYSTem:TEMPerature:PROTection[:HIGH]:STATe  
SYSTem:TEMPerature:PROTection[:HIGH]:TRIPped?

#### 5.17.71. SYSTem:TEMPerature:PROTection[:HIGH]:CLEar

**Syntax** SYSTem:TEMPerature:PROTection[:HIGH]:CLEar [<sensor>]

**Description** This command clears the latched protection status when an over-temperature is detected.  
All conditions that generate the fault must be removed before the latched status can be



cleared. The output is restored to the state it was in before the fault condition occurred.

Name	Type	Range	Default
<sensor>	Discrete	AUX CH1 CH2 CH3 CH4  CH5 CH6	AUX

**Usage example** `SYST:TEMP:PROT:CLE`

**Related Commands** `SYSTem:TEMPerature:PROTection[:HIGH]:TRIPped`

#### 5.17.72. **SYSTem:TEMPerature:PROTection[:HIGH]:DELay[:TIME]**

**Syntax** `SYSTem:TEMPerature:PROTection[:HIGH]:DELay[:TIME] {<delay>} [, <sensor>]`  
`SYSTem:TEMPerature:PROTection[:HIGH]:DELay[:TIME]? [<sensor>]`

**Description** This command sets the over-temperature protection delay. The over-temperature protection function will not be triggered during the delay time. After the delay time has expired, the over-temperature protection function will be active. Programmed values can range from 0 to 300 seconds. See also [Section 8.1](#)

Parameters	Name	Type	Range	Default
	<delay>	NR1	0 – 300 seconds	10
	<sensor>	Discrete	AUX CH1 CH2 CH3 CH4  CH5 CH6	AUX

**Return** The query returns programmed over-temperature protection delay.

**Usage example** `SYST:TEMP:PROT:DEL 30, CH2`

**Related Commands** `*RST`  
`SYSTem:TEMPerature:PROTection[:HIGH][:LEVel]`

#### 5.17.73. **SYSTem:TEMPerature:PROTection[:HIGH]:STATe**

**Syntax** `SYSTem:TEMPerature:PROTection[:HIGH]:STATe {<bool>} [, <sensor>]`  
`SYSTem:TEMPerature:PROTection[:HIGH]:STATe? [<sensor>]`

**Description** This command enables or disables the over-temperature protection (OTP) function. The enabled state is ON (1); the disabled state is OFF (0). If the over-temperature protection function is enabled and the measured output power reach value set by [SOURce[<n>]]:POWer:PROTection[:LEVel] the output is disabled and the Questionable Condition status register OPP bit 10 is set.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	OFF
	<sensor>	Discrete	AUX CH1 CH2 CH3 CH4  CH5 CH6	AUX

**Return** The query command returns 0 if the current protection state is OFF, and 1 if the current protection state is ON.

**Usage example** `SYST:TEMP:PROT:STAT? CH1`  
 0

**Related Commands** `*RST`  
`SYSTem:TEMPerature:PROTection[:HIGH]:CLEAr`

#### 5.17.74. **SYSTem:TEMPerature:PROTection[:HIGH]:TRIPped?**

**Syntax** `SYSTem:TEMPerature:PROTection[:HIGH]:TRIPped? [<sensor>]`

**Description** Query whether OTP occurred on the selected temperature sensor. When protection is

tripped bit 4 (TEMPerature) of the Questionable Status register will be set (see [Section 3.4](#)).

The SYSTem:TEMPerature:PROTection[:HIGH]:CLEar command can be send to clear OTP condition caused by the selected temperature sensor.

Parameters	Name	Type	Range	Default
	<sensor>	Discrete	AUX CH1 CH2 CH3 CH4 CH5 CH6	AUX
<b>Return</b>	This query returns a 1 if the protection circuit is tripped and a 0 if it is untripped.			
<b>Usage example</b>	SYST:TEMP:PROT:TRIP? 0			
<b>Related Commands</b>	SYSTem:TEMPerature:PROTection[:HIGH]:CLEar SYSTem:TEMPerature:PROTection[:HIGH]:STATE			

#### 5.17.75. SYSTem:TIME

**Syntax** SYSTem:TIME {<hours>}, {<minutes>}, {<seconds>}  
SYSTem:TIME?

**Description** Sets the time of the system clock (RTC). Specify the hours, minutes, and seconds. The self-test procedure compare date and time stored in RTC registers with values stored in the non-volatile memory (EEPROM). When the later is greater then former or any of them lost integrity (i.e. any of value is outside allowed range: for example seconds are higher then 60 or months are higher then 12, etc.) self-test will failed. The \*TST? will return 1 and detailed report could be queried using the DIAGnostic:TEST? command.

*The bit 3 (TIME) of the Questionable Status register will be set (see [Section 3.4](#)) if date-time self-test failed or datetime was never set.*

Parameters	Name	Type	Range	Default
	<hours>	NR1	0 – 23	–
	<minutes>	NR1	0 – 59	–
	<seconds>	NR1	0 – 59	–
<b>Return</b>	Query the current time of the system clock in HH, MM, SS format.			
<b>Usage example</b>	SYST:TIME? 15, 10, 33			
<b>Related Commands</b>	*TST? DIAGnostic[:INFOrmation]:TEST? SYSTem:DATE SYSTem:FORMat:TIME			

#### 5.17.76. SYSTem:TIME:DST

**Syntax** SYSTem:TIME:DST {<rules>}  
SYSTem:TIME:DST?

**Description** Use this command to define daylight saving time (DST) rules used in your region.

*Firmware v1.0 support limited number of region: Europe, US/Canada and Australia/New Zealand.*

Parameters	Name	Type	Range	Default
	<rules>	Discrete	OFF EU USA AUS	OFF
<b>Return</b>	Query the DTS rule name used for adjust time.			
<b>Usage example</b>	SYST:TIME:DST EU			

**Related Commands** SYSTem:COMMunicate:ENABLE  
 SYSTem:COMMunicate:NTP  
 SYSTem:TIME

#### 5.17.77. SYSTem:TIME:ZONE

**Syntax** SYSTem:TIME:ZONE {<zone>}  
 SYSTem:TIME:ZONE?

**Description** Use this command to define time zone as offset from GMT.

Parameters	Name	Type	Range	Default
	<zone>	Quoted string	-12:00 to 14:00	—

**Return** Query the time zone as offset from GMT in the following format: “[sign]hh:mm GMT”

**Usage example** SYST:TIME:ZONE 1  
 SYST:TIME:ZONE?  
 "+01:00 GMT"

**Related Commands** SYSTem:COMMunicate:ENABLE  
 SYSTem:COMMunicate:NTP  
 SYSTem:TIME

#### 5.17.78. SYSTem:VERSion?

**Syntax** SYSTem:VERSion?

**Description** This command returns the version of the SCPI (Standard Commands for Programmable Instruments) standard with which the instrument is in compliance

**Return** The command returns a string in the form “YYYY.V”, where YYYY represents the year of the version and V represents a version for that year.

**Usage example** SYST:VERS?  
 1999.0

## 5.18. TRIGger

The BB3's triggering system allows a change in output voltage, current or start internal data logging when receiving a trigger from selected source. Triggering the BB3 is a multi-step process and consists of the following steps:

- An output has to be selected (the INSTRument:SElect command) following by configuring the BB3 for the triggered output level by using CURRent:TRIGgered and VOLTage:TRIGgered commands.
- The source from which the BB3 will accept the trigger must be specified. The BB3 could accept e.g. a BUS (software) trigger or an IMMEDIATE trigger from the remote interface.
- The time delay between the detection of the trigger on the specified trigger source and the start of any corresponding output change can be programmed if needed. Such time delay is valid only for the BUS trigger source.
- Trigger programming is completed by providing an INITiate[:IMMEDIATE] command. If the IMMEDIATE source is selected, the selected output is set to the triggered level immediately. But if the trigger source is the BUS, the BB3 is set to the triggered level after receiving the \*TRG command.

SCPI command	Description
TRIGger	
[:SEquence]	
[:IMMEDIATE]	Starts the trigger immediately
:DELay {<delay>}	Sets the time delay between the detection of a trigger event and the start of any corresponding trigger action
:EXIT	
:CONDition {<condition>}	Sets channels condition when LIST execution stopped
:SOURce {<source>}	Sets the trigger source
:DLOG	
[:IMMEDIATE]	
:SOURce {<source>}	Sets the internal data logger trigger source

### 5.18.1. TRIGger[:SEquence][:IMMEDIATE]

**Syntax** TRIGger[:SEquence][:IMMEDIATE]

**Description** This event command causes a defined LIST to immediately start without the selected trigger occurring.

**Usage example** TRIG

**Related Commands** \*TRG

### 5.18.2. TRIGger[:SEquence]:DELay

**Syntax** TRIGger[:SEquence]:DELay {<delay>}  
TRIGger[:SEquence]:DELay?

**Description** This command sets the time delay between the detection of an event on the specified trigger source and the start of any corresponding trigger action on the peripheral module output.

Parameters	Name	Type	Range	Default
	<delay>	NR1 Discrete	0 – 3600 MIN MAX	MIN
<b>Return</b>	The query command returns the programmed delay in seconds.			
<b>Usage</b>	TRIG:DEL 10			

## example

## 5.18.3. TRIGger[:SEQuence]:EXIT:CONDition

**Syntax** TRIGger[:SEQuence]:EXIT:CONDition {<condition>}  
 TRIGger[:SEQuence]:EXIT:CONDition?

**Description** This command sets channels condition when LIST execution is not prematurely stopped (e.g. with ABORt command or by user action).  
 Use [SOURce[<n>]]:LIST:COUNT to set finite number of LIST loops.

Parameters	Name	Type	Range	Default
	<condition>	Discrete	OFF FIRST LAST STANdby	OFF

**Return** The query command returns the programmed exit condition.

**Usage example** TRIG:EXIT:COND STAN

**Related Commands** ABORt  
 INITiate  
 [SOURce[<n>]]:LIST:COUNT  
 [SOURce[<n>]]:CURRent:MODE  
 [SOURce[<n>]]:VOLTage:MODE

## 5.18.4. TRIGger[:SEQuence]:SOURce

**Syntax** TRIGger[:SEQuence]:SOURce {<source>}  
 TRIGger[:SEQuence]:SOURce?

**Description** This command selects the source from which the BB3 will accept a trigger.

- BUS – enables LAN and serial (via USB) triggering using the \*TRG command.
- IMMEDIATE – the BB3 executes a complete trigger operation immediately after executing the INITiate command without delay.
- MANual – enables triggering by selecting the encoder knob switch.
- PIN<n> – selects a digital port pin configured as a trigger input. <n> specifies the pin number.

When the trigger source is set to BUS, the \*WAI command can ensure the synchronization. After executing the \*WAI command, the BB3 will only execute new command when all the pending operations are completed.

Also when the trigger source is set to BUS, you can use the \*OPC command to report that the operation is completed. The \*OPC? command will return “1” to the output buffer and the \*OPC command will set the bit 0 (OPC bit, operation complete) in the standard event register when the operation is finished.

The wait for the BUS, EXTERNAL, or KEY trigger can be bypassed by sending the TRIGger[:SEQuence][:IMMEDIATE] command.

*The APPLY command automatically sets the source to IMMEDIATE.*

Parameters	Name	Type	Range	Default
	<source>	Discrete	BUS IMM MAN PIN1 PIN2	–

**Return** The query command returns the programmed trigger subsystem source.

**Usage example** TRIG:SOUR BUS  
 TRIG:SOUR?

BUS

**Related Commands** \*OPC  
 \*RST  
 \*TRG

\*WAI  
 ABORt  
 APPLy  
 INITialize  
 TRIGger[:SEQuence][:IMMediate]

#### 5.18.5. TRIGger:DLOG[:IMMediate]

**Syntax** TRIGger:DLOG[:IMMediate]

**Description** The command sends an immediate trigger signal to the data logger. This will trigger the internal data log session regardless of the selected trigger source. You must initiate (see the INIT:DLOG command) the data logger before you trigger it.

**Usage example** TRIG:DLOG

**Related Commands** \*TRG  
 INITialize:DLOG

#### 5.18.6. TRIGger:DLOG:SOURce

**Syntax** TRIGger:DLOG:SOURce {<source>}  
 TRIGger:DLOG:SOURce?

**Description** The command selects the trigger source for the data logger in the same fashion as the TRIGger[:SEQuence]:SOURce command.

Parameters	Name	Type	Range	Default
	<source>	Discrete	BUS IMM MAN PIN1  PIN2	–

**Return** The query command returns the programmed trigger subsystem source.

**Usage example** TRIG:DLOG:SOUR PIN1  
 TRIG:SOUR?  
 PIN1

**Related Commands** INITialize  
 TRIGger[:SEQuence]:SOURce  
 TRIGger:DLOG[:IMMediate]

## 6. Device-specific (unclassified) commands

The commands in this section are specific to the BB3, and so are not included in the 1999.0 version of the SCPI standard. However, these commands are designed with the SCPI standard in mind, and they follow all of the command syntax rules defined by the standard.

### 6.1.1. APPLy

The APPLy command provides the most straightforward method to program the BB3 remotely.

**Syntax**      `APPLy {<channel>}, {<voltage>} [, <current>]`  
`APPLy? {<channel>} [, <query param>]`

**Description** This command is a combination of the [INSTrument:SElect](#) (or [INSTrument:NSElect](#)), [\[SOURce\[<n>\]\]:VOLTage](#) and [\[SOURce\[<n>\]\]:CURRent](#) commands.

The APPLy changes the power module's output to the newly programmed values only if the programmed values are valid within the presently selected power module operating range. An execution error will occur if the programmed values are not valid within the selected range. You can substitute MINimum, MAXimum, or DEFault in place of a specific value for the voltage and current parameters (see table below).

Parameters	Name	Type	Range	Default
	<channel>	Discrete	CH1 CH2 CH3 CH4 CH5 CH6	–
	<voltage>	NR2 Discrete	0 to MAXimum, MIN DEF MAX UP DOWN The MAXimum value is dependent on the power module voltage rating. See <a href="#">Section 8.1</a>	–
	<current>	NR2 Discrete	0 to MAXimum, MIN DEF MAX UP DOWN The MAXimum value is dependent on the power module current rating. See <a href="#">Section 8.1</a>	–
	<query param>	Discrete	CURR VOLT	–

**Return**      APPLy? query the voltage/current of the specified channel.

**Usage example**      Set the voltage and current of CH1 to 35.5V and 0.5A respectively:

```
APPL CH1, 35.5, 0.5
```

Query the voltage and current settings of the first channel:

```
APPL? CH1
```

```
CH1:50V/3A, 35.500, 0.500
```

Query only current setting of the second channel:

```
APPL? CH2, CURR
```

```
0.25
```

**Errors**      -221, "Power limit exceeded"  
              -222, "Data out of range"

**Related Commands**      [INSTrument:NSElect](#)  
                          [INSTrument\[:SElect\]](#)  
                          [\[SOURce\[<n>\]\]:VOLTage\[:LEVel\]\[:IMMediate\]\[:AMPLitude\]](#)  
                          [\[SOURce\[<n>\]\]:CURRent\[:LEVel\]\[:IMMediate\]\[:AMPLitude\]](#)  
                          [TRIGger\[:SEQUence\]:SOURce](#)

### 6.1.2. DEBUg

**Syntax**      `DEBUg?`

**Description** Use this command to collect various runtime information that can be used in debugging

	process.
<b>Usage example</b>	DEBU? CH1 U_DAC = 0 CH2 U_DAC = 32768 CH1 U_MON = 0 CH2 U_MON = 4095 CH1 U_MON_DAC = 0 CH2 U_MON_DAC = 16383 CH1 I_DAC = 39321 CH2 I_DAC = 6554 CH1 I_MON = 0 CH2 I_MON = 3276 CH1 I_MON_DAC = 19660 CH2 I_MON_DAC = 3276 AUX TEMP = 373 CH1 TEMP = 561 CH2 TEMP = 373 MAIN_LOOP_DURATION = 15998 16696 17409 / 0 0 0 / 1844 346805388 ADC_COUNTER = 120 / 0 / 296
<b>Related Commands</b>	DIAGnostic[:INFOrmation]:ADC?



## 7. Error messages

The system-defined error/event numbers are chosen on an enumerated ("1 of N") basis. The SCPI-defined error/event numbers and the <error/event\_description> portions of the full queue item are listed here. The first error/event described in each class (for example, -100, -200, -300, -400) is a "generic" error.

### 7.1. Command Error [-199, -100]

An <error/event number> in the range [-199, -100] indicates that a syntax error has been detected by the BB3's SCPI command parser. The occurrence of an error in this class cause the command error bit (CME, bit 5) in the Standard Event Status Register (see [Section 3.1](#)) to be set.

Return string	Description
0, "No error"	The queue is completely empty. Every error/event in the queue has been read, or the queue was purposely cleared by a power-on event, *CLS, etc.
-100, "Command error"	Generic syntax error.
-101, "Invalid character"	An invalid character was found in the command string. There may be a character such as #, \$, or % in the command keyword or within a parameter.  Example: OUTP:STAT #ON
-103, "Invalid separator"	An invalid separator was found in the command string. There may be a comma instead of a colon, semicolon, or blank space, or a blank space instead of a comma.  Example: TRIG:SOUR,BUS
-104, "Data type error"	The wrong parameter type was found in the command string. A string might have been sent when a string was expected, or vice versa.  Example (the password must be a quoted string): CAL ON, 123
-108, "Parameter not allowed"	More parameters were received than expected for the command. There may be an extra parameter, or parameter added to a command that does not accept a parameter.  Example: INST CH1, CH2
-109, "Missing parameter"	Fewer parameters were received than expected for the command. One or more parameters that are required for this command was not received.  Example: APPL
-113, "Undefined header"	A command was received that is not valid for the BB3. The command may have been misspelled, or it may not be a valid command. Short forms of commands, may contain up to four letters.


	<p>Example:</p> <pre>MEASU:CURR?</pre>
-114, "Header suffix out of range"	<p>The numeric suffix attached to a command header is not one of the allowable values.</p>
	<p>Example:</p> <pre>STAT:QUES:INST:ISUM3?</pre>
-131, "Invalid suffix"	<p>A suffix was incorrectly specified for a numeric parameter. It may have been misspelled.</p> <p>Example (use A instead of V):</p> <pre>VOLT 3A</pre>
-138, "Suffix not allowed"	<p>A suffix was received following a numeric parameter which does not accept a suffix.</p> <p>Example (SEC is not a valid suffix):</p> <pre>STAT:QUES:ENAB 10 SEC</pre>
-151, "Invalid string data"	<p>An invalid character string was received. Check to see if you have enclosed the character string in single or double quotes.</p> <p>Example:</p> <pre>DISP:TEXT 'ON</pre>

## 7.2. Execution Error [-299, -200]

An <error/event number> in the range [-299, -200] indicates an error has been detected by the BB3's execution control block. The occurrence of any error in this class sets the execution error bit (EXE, bit 4) in the Standard Event Status Register (see [Section 3.1](#)). One of the following events has occurred:

- A <PROGRAM DATA> element following a header was evaluated by the BB3 as outside of its legal input range, or as otherwise inconsistent with the BB3's capabilities.
- A valid program message could not be properly executed. Probably due to some BB3 condition.

Execution errors will be reported by the BB3 after rounding and expression evaluation has taken place. Rounding a numeric data element, for example, will not be reported as an execution error. Events that generate execution errors will not generate Command Errors, device-specific errors, or Query Errors; see the other error definitions in this section.

Return string	Description
-200, "Execution error"	This is the generic execution error when more specific error is not assigned in the case that command execution failed.
 -211, "Settings conflict"	Indicates that a legal program data element was parsed but could not be executed due to the current device state.
-222, "Data out of range"	<p>A numeric parameter value is outside the valid range for the command.</p> <p>Example:</p> <pre>VOLT 166</pre>
-223, "Too much data"	A character string was received but could not be executed because the string length was more than 32 characters. This error can be generated by the CALibration:REMark command.

-224, "Illegal parameter value"

A discrete parameter was received which was not a valid choice for the command. You may have used an invalid parameter choice.

Example:

VOLT ON

-230, "Digital pin function mismatch"

Operation with digital pin is not possible due to incorrect function definition.

Example:

DIG:PIN1:FUNC INH  
DIG:INP:DATA? 1

-240, "Hardware error"

Command or query could not be executed because failure is detected during power-up self-test. Use the [\\*TST?](#) command to query self-test results. See also [Section 7.3.1](#)

-241, "Hardware missing"

Command or query could not be executed because of missing BB3 hardware.

Example (remote sense cannot be activated on the DCM220 power module):

OUTP:SENS ON

-250, "Mass storage error"

Indicates that a SD card error occurred. This error message is generated when the firmware cannot detect the more specific errors described for errors -252 through -258.

-252, "Missing media"

Command or query could not be executed because of a missing SD card.

-256, "File name not found"

Command or query could not be executed because the file name on the device media was not found; for example, an attempt was made to read or copy a nonexistent file.

-257, "File name error"

Command or query could not be executed because the file name on the device media was in error; for example, an attempt was made to copy to a duplicate file name.

-258, "Media protected"

Command or query could not be executed because the SD card access is locked (MMEMory:LOCK command).

-259, "File transfer aborted"

Command or query for file transfer is aborted before transfer is completed.

-260, "CH1 fault detected"

A POWERGOOD signal failure has been detected on Channel 1. If such a condition has happened, BB3 firmware will be immediately put into the standby mode.

-261, "CH2 fault detected"

A POWERGOOD signal failure has been detected on Channel 2. If such a condition has happened, BB3 firmware will be immediately put into the standby mode.

-262, "CH3 fault detected"

A POWERGOOD signal failure has been detected on Channel 3. If such a condition has happened, BB3 firmware will be immediately put into the standby mode.

-263, "CH4 fault detected"

A POWERGOOD signal failure has been detected on Channel 4. If such a condition has happened, BB3 firmware will be immediately put into the standby mode.

-264, "CH5 fault detected"

A POWERGOOD signal failure has been detected on Channel 5. If such a condition has happened, BB3 firmware will be immediately put into the standby mode.

-265, "CH6 fault detected"	A POWERGOOD signal failure has been detected on Channel 6. If such a condition has happened, BB3 firmware will be immediately put into the standby mode.
-270, "CH1 output fault detected"	A prohibited state has been detected on the Channel 1 output (e.g., negative power is measured for extended period of time). The Channel 1 output will be turned off.
-271, "CH2 output fault detected"	A prohibited state has been detected on the Channel 2 output (e.g., negative power is measured for extended period of time). The Channel 1 output will be turned off.
-272, "CH3 output fault detected"	A prohibited state has been detected on the Channel 3 output (e.g., negative power is measured for extended period of time). The Channel 1 output will be turned off.
-273, "CH4 output fault detected"	A prohibited state has been detected on the Channel 4 output (e.g., negative power is measured for extended period of time). The Channel 1 output will be turned off.
-274, "CH5 output fault detected"	A prohibited state has been detected on the Channel 5 output (e.g., negative power is measured for extended period of time). The Channel 1 output will be turned off.
-275, "CH6 output fault detected"	A prohibited state has been detected on the Channel 6 output (e.g., negative power is measured for extended period of time). The Channel 1 output will be turned off.

### 7.3. Device-Specific Error [-399, -300], [1, 32767]

An <error/event number> in the range [-399, -300] or [1, 32767] indicates that the BB3 has detected an error which is not a command error, a query error, or an execution error. Most likely, some BB3 operations did not properly complete, possibly due to an abnormal hardware or firmware condition. These codes are also used for self-test response errors. The occurrence of any error in this class sets the device-specific error bit (DDE, bit 3) in the Standard Event Status Register (see [Section 3.1](#)).

Events that generate device-specific errors do not generate command errors, execution errors, or query errors; see the other error definitions in this section.

Return string	Description
-211, "Trigger ignored"	A *TRG was received but the trigger was ignored. The trigger source should be set to the BUS and the trigger subsystem should be initiated by the INITiate[:IMMdate] command.
-213, "Init ignored"	An INITiate command was received but could not be executed because a measurement was already in progress.
-310, "System error"	An internal firmware error has been detected.
-330, "Self-test failed"	The BB3's complete self-test failed from the remote interface (*TST? command). In addition to this error, more specific self-test errors are also reported. See also <a href="#">Section 7.3.1</a>
-350, "Queue overflow"	The error queue is full because more than 16 errors have occurred. No additional errors will be stored until errors have been removed from the queue. The error queue is cleared when power has been turned off, or after a *CLS (clear status) command has been executed.
-363, "Input buffer overrun"	Input buffer overrun. The Serial or Ethernet port input buffer has lost data due to an overflow.
100, "Channel not found"	A non existing channel number has been specified.

Example:

```
SOUR3:VOLT?
```

101, "Calibration mode is off"	Calibration is not enabled. The BB3 will not accept calibration commands.
102, "Invalid password"	The calibration password is incorrect.
104, "Bad sequence of cal commands"	Calibration commands have not been entered in the proper sequence.
105, "Password too long"	A calibration password was received which contained more than 16 characters.
106, "Password too short"	A calibration password was received which contained less than 4 characters.
107, "Cal value out of range"	The specified calibration value (CALibration:CURRent:DATA or CALibration:VOLTagE:DATA) is not valid for the BB3's present measurement function and range.
108, "Cal output disabled"	The Calibration operation has been aborted by sending the OUTPut OFF command during an output calibration.
109, "Invalid cal data"	One or more calibration data values are impossible or out of range, and will prohibit the calibration parameters calculation.  For example the MID value is lower then MIN, or the MIN value is higher then MAX, or the MID value is so out of accepted tolerance that it cannot be predicted with newly calculated calibration parameters.
110, "Cal params missing or corrupted"	Calibration parameters activation started by the CALibration:STATe ON, "<password>" command failed because calibration has never been conducted or existing parameters are corrupted.
111, "No new cal data exists"	There was an attempt to save calibration data with the CALibration:SAVE command without calibration data.
150, "Power limit exceeded"	The product of voltage and current exceeds channel power limitation.  For example if channel power limit is 155 W and the following sequence is executed: VOLT 38 CURR 4.4
151, "Voltage limit exceeded"	Set voltage exceeds channel capability.  Example: VOLT 60
152, "Current limit exceeded"	Set current exceeds channel capability.  Example: CURR 5.5
159, "Module total power limit exceeded"	Set power on module with multiple channels exceeds total power limitation.  Example, an attempt to set 80 W (160 W in total) on both channels while module total power limit is 155 W:  APPL CH1, 20, 4 APPL CH2, 20, 4
201, "Cannot execute before clearing protection"	A command such as OUTP ON cannot be executed on a channel where one or more protections tripped.

202, "Cannot init trigger while Rprog is enabled"	Trigger initialization failed because remote programming option is active on one or more channels (see [SOURce[<n>]]:VOLTage:PROGram[:SOURce])
304, "Incompatible transient modes"	For example, VOLTage and CURRent cannot be in both STEP and LIST modes at the same time (see [SOURce[<n>]]:CURRent:MODE and [SOURce[<n>]]:VOLTage:MODE)
306, "Too many list points"	Too many list points have been specified.
307, "List lengths are not equivalent"	One or more lists are not the same length.  For example, number of CURRent steps is not equal to one or number of VOLTage steps  LIST:VOLT 0, 5, 10, 15, 20 LIST:CURR 1, 2
308, "Cannot be changed while transient trigger is initiated"	An attempt to change a setting which cannot be changed while the instrument is waiting for or executing a trigger sequence.
309, "Cannot initiate while in fixed mode"	Cannot initiate transient generator. Either the VOLTage or CURRent function is set to FIXed mode.
310, "File not found"	A file cannot be found on the SD card.
311, "List is empty"	A list cannot be started because it's empty.
312, "Cannot execute when the channels are coupled"	Cannot execute a command when the channel outputs are coupled. Example (an attempt to set remote sensing on the channel that is coupled in series):  INST:COUP:TRAC SER VOLT:SENS EXT
313, "Cannot execute in tracking mode"	Cannot execute a command when the tracking mode is active.
314, "Cannot set list value"	List include one or more values that cannot be set on target instrument. For example, DCM220 dual output DC source output current can be set in 20 mA increments. Setting current to 22 mA will generate this error.
320, "Firmware update error"	An attempt to upload firmware via LAN has failed due to communication error, checksum error, etc. ( <i>Not implemented yet</i> ).
400, "Cannot load empty profile"	An attempt has been made to recall an empty or erased user profile with *RCL command.
401, "Module mismatch in profile"	An attempt has been made to recall user profile that contains different peripheral modules configuration.
410, "No FAT file system on mass media"	Command or query could not be executed because the SD card was not formatted or has filesystem that is not supported.
500, "Down-programmer on CH1 switched off"	The down programmer has been switched off because a negative output power (DP_NEG_LEV) has been measured on Channel 1 that has lasted more then DP_NEG_DELAY seconds.
501, "Down-programmer on CH2 switched off"	The down programmer has been switched off because a negative output power (DP_NEG_LEV) has been measured on Channel 2 that has lasted more then DP_NEG_DELAY seconds.
502, "Down-programmer on CH3 switched off"	The down programmer has been switched off because a negative output power (DP_NEG_LEV) has been measured on Channel 3 that has lasted more then DP_NEG_DELAY seconds.
503, "Down-programmer on CH4"	The down programmer has been switched off because a nega-




switched off"	tive output power ( <code>DP_NEG_LEV</code> ) has been measured on Channel 4 that has lasted more then <code>DP_NEG_DELAY</code> seconds.
504, "Down-programmer on CH5 switched off"	The down programmer has been switched off because a negative output power ( <code>DP_NEG_LEV</code> ) has been measured on Channel 5 that has lasted more then <code>DP_NEG_DELAY</code> seconds.
510, "External voltage on CH1 detected"	An external voltage has been measured (e.g., from a battery or other power source intentionally or mistakenly left connected) on Channel 1 while output is switched from off to on. ( <i>Not implemented yet</i> ).
511, "External voltage on CH2 detected"	An external voltage has been measured (e.g., from a battery or other power source intentionally or mistakenly left connected) on Channel 2 while output is switched from off to on. ( <i>Not implemented yet</i> ).
512, "External voltage on CH3 detected"	An external voltage has been measured (e.g., from a battery or other power source intentionally or mistakenly left connected) on Channel 3 while output is switched from off to on. ( <i>Not implemented yet</i> ).
513, "External voltage on CH4 detected"	An external voltage has been measured (e.g., from a battery or other power source intentionally or mistakenly left connected) on Channel 4 while output is switched from off to on. ( <i>Not implemented yet</i> ).
514, "External voltage on CH5 detected"	An external voltage has been measured (e.g., from a battery or other power source intentionally or mistakenly left connected) on Channel 5 while output is switched from off to on. ( <i>Not implemented yet</i> ).
615, "MCU EEPROM save failed"	The STM32F7 MCU board non-volatile memory (an EEPROM) is damaged or not responsive.
616, "SLOT1 EEPROM save failed"	The non-volatile memory (an EEPROM) on the periperal module inserted into slot #1 is damaged or not responsive.
617, "SLOT2 EEPROM save failed"	The non-volatile memory (an EEPROM) on the periperal module inserted into slot #2 is damaged or not responsive.
618, "SLOT3 EEPROM save failed"	The non-volatile memory (an EEPROM) on the periperal module inserted into slot #3 is damaged or not responsive.

### 7.3.1. Self-Test Error Messages

During power-up, the BB3 will start self-test sequences when communication with all SPI devices that is marked as installed will be established. The scope of the self-test depends on device capability and it could vary from simply reading device registers, and waiting for expected responses to more complex operations such as setting DAC registers and reading back those values using the ADC (if ADC test passed). Every test failure will be announced by an error beep, and one error message per failed test will be inserted into the error queue. The device-specific error bit (DDE, bit 3) in the Standard Event Status Register (see [Section 3.1](#)) will also be set.

Return string	Description
210, "CH1 IOEXP test failed"	Communication with the Channel 1 I/O expander is not possible. Probably because the I/O expander is not functional.
211, "CH2 IOEXP test failed"	Communication with the Channel 2 I/O expander is not possible. Probably because the I/O expander is not functional.
212, "CH3 IOEXP test failed"	Communication with the Channel 3 I/O expander is not possible. Probably because the I/O expander is not functional.
213, "CH4 IOEXP test failed"	Communication with the Channel 4 I/O expander is not possible. Probably because the I/O expander is not functional.

214,"CH5 IOEXP test failed"	Communication with the Channel 5 I/O expander is not possible. Probably because the I/O expander is not functional.
220,"CH1 ADC test failed"	Communication with the Channel 1 ADC is not possible. Probably because values written into various registers are not equal to the returned values.
221,"CH2 ADC test failed"	Communication with the Channel 2 ADC is not possible. Probably because values written into various registers are not equal to the returned values.
222,"CH3 ADC test failed"	Communication with the Channel 3 ADC is not possible. Probably because values written into various registers are not equal to the returned values.
223,"CH4 ADC test failed"	Communication with the Channel 4 ADC is not possible. Probably because values written into various registers are not equal to the returned values.
224,"CH5 ADC test failed"	Communication with the Channel 5 ADC is not possible. Probably because values written into various registers are not equal to the returned values.
230,"CH1 DAC test failed"	The Channel 1 DAC is not functional. Possibly because communication has failed, or because there's a difference between the test voltage data sent to the Channel 1 DAC and that read back.
231,"CH2 DAC test failed"	The Channel 2 DAC is not functional. Possibly because communication has failed, or because there's a difference between the test voltage data sent to the Channel 2 DAC and that read back.
232,"CH3 DAC test failed"	The Channel 3 DAC is not functional. Possibly because communication has failed, or because there's a difference between the test voltage data sent to the Channel 1 DAC and that read back.
233,"CH4 DAC test failed"	The Channel 4 DAC is not functional. Possibly because communication has failed, or because there's a difference between the test voltage data sent to the Channel 1 DAC and that read back.
234,"CH5 DAC test failed"	The Channel 5 DAC is not functional. Possibly because communication has failed, or because there's a difference between the test voltage data sent to the Channel 1 DAC and that read back.
235,"CH6 DAC test failed"	The Channel 6 DAC is not functional. Possibly because communication has failed, or because there's a difference between the test voltage data sent to the Channel 1 DAC and that read back.
 , "MCU EEPROM test failed"	The non-volatile memory on the STM32F7 MCU board has failed the checksum integrity test.
250,"RTC test failed"	The RTC (real time clock) on the STM32F7 MCU board is not present, or the date or time values returned are not valid, or the last datetime value stored into non-volatile memory is greater than RTC datetime.
260,"Ethernet test failed"	The Ethernet controller on the STM32F7 MCU board test failed.
630,"Fan test failed"	The measured fan speed during the self-test procedure is not within expected range.

For example, if SPI-bus cable is accidentally not connected, or wrongly wired, on Channel 1 two errors will be generated and placed into the error queue which can be checked using the following command



sequence:

SYST:ERR:COUN?

2

SYST:ERR?

210,"CH1 IOEXP test failed"

SYST:ERR?

220,"CH1 ADC test failed"

## 8. Parameters and settings

### 8.1. Programming parameters

The BB3 firmware can be build to support any combination of various type of peripheral modules that that are inserted into BB3 chassis.

Use the [\\*IDN?](#) command to find out what channels are defined in the BB3's firmware.

#### 8.1.1. Voltage

Power module model	DCP405	DCM220
MAXimum [V]	40	20
MINimum [V]	0	1
DEFault [V]	0	0
Value after *RST or *TST?	0	0
STEP MINimum [V]	10m	10m
STEP MAXimum [V]	5	2
STEP DEFault [V]	100m	100m
PROtection DELay MINimum [s]	0	0
PROtection DELay MAXimum [s]	10	10
PROtection DELay DEFault [s]	5m	5m
CALibration VALue MINimum[V]	0.15	2
CALibration VALue MIDdle[V]	20	10
CALibration VALue MAXimum[V]	38	18

#### 8.1.2. Current

Programming range or model	DCP405	DCM220
MAXimum [A]	5	4
MINimum [A]	0	0
DEFault [A]	0	0
Value after *RST or *TST?	0	0
STEP and STEP MINimum [A]	10m	10m
STEP MAXimum [A]	1	1
STEP DEFault [A]	50m	50m
PROtection DELay MINimum [s]	0	0
PROtection DELay MAXimum [s]	10	10
PROtection DELay DEFault [s]	20m	20m
CALibration VALue MINimum[A]	50m	500m
CALibration VALue MIDdle[A]	2.425	2
CALibration VALue MAXimum[A]	4.8	3.5

#### 8.1.3. Power

The total number of possible combinations for the BB3's power parameters is 6, of which two typical examples are given below:

Programming range or model	DCP405	DCM220
----------------------------	--------	--------

MINimum [W]	0	0
DEFault [W]	155	80
MAXimum [W]	155	80
PROtection LEVel DEFault [W]	155	80
PROtection DELay MINimum [s]	1	1
PROtection DELay MAXimum [s]	300	300
PROtection DELay DEFault [s]	10	10

## 8.2. Reset Settings (\*RST)

At power-on or after execution of the [\\*RST](#) common command, device settings will be set to the states listed in the following table. See also the [MEMory:STATE:RECall:AUTO](#) command.

Command	Power on	*RST
*ESE	0	Not affected
*ESR	0	Not affected
*SRE	0	Not affected
*STB?	0	???
CAL[:MODE]	OFF	
CAL:STAT	ON if valid calibrating data for both voltage and current exists in the non-volatile memory, otherwise OFF.	
DISP:BRIG	Power down state	20
DISP:VIEW	Power down state	1
DISP[:STAT]	Power down state	Not affected
INST:NSEL	1	
INST:SEL	CH1	
MEMM:CDIR	/	
MEMM:DOWN:FNAME	""	
MMEM:LOCK	Power down state	Not affected
OUTP[:STAT]	Power down state	OFF
OUTP:DPR	Power down state	Not affected
OUTP:PROT:COUP	Power down state	OFF
OUTP:TRAC	Power down state	OFF
SENS:CURRE:RANG	Power down state	BEST
SENS:DLOG:FUNC:CURRE	Power down state	OFF
SENS:DLOG:FUNC:POW	Power down state	OFF
SENS:DLOG:FUNC:VOLT	Power down state	OFF
SENS:DLOG:PER	Power down state	0.02
SENS:DLOG:TIME	Power down state	60
[SOUR[n]]:CURRE	Power down state	DEF (see <a href="#">Section 8.1</a> )
[SOUR[n]]:CURRE:PROT:DEL	Power down state	DEF (see <a href="#">Section 8.1</a> )
[SOUR[n]]:CURRE:PROT:STAT	Power down state	OFF
[SOUR[n]]:CURRE:PROT:TRIP?	0	
[SOUR[n]]:CURRE:STEP	Power down state	DEF (see <a href="#">Section 8.1</a> )

[SOUR[n]]:POW:PROT[:LEV]	Power down state	DEF (see <a href="#">Section 8.1</a> )
[SOUR[n]]:POW:PROT:DEL	Power down state	DEF (see <a href="#">Section 8.1</a> )
[SOUR[n]]:POW:PROT:STAT	Power down state	ON
[SOUR[n]]:POW:PROT:TRIP?	0	
[SOUR[n]]:VOLT	Power down state	DEF (see <a href="#">Section 8.1</a> )
[SOUR[<n>]]:VOLT:PROG[:SOUR]	INT	
[SOUR[n]]:VOLT:PROT:DEL	Power down state	DEF (see <a href="#">Section 8.1</a> )
[SOUR[n]]:VOLT:PROT:STAT	Power down state	OFF
[SOUR[n]]:VOLT:PROT:TRIP?	0	
[SOUR[n]]:VOLT:STEP	Power down state	DEF (see <a href="#">Section 8.1</a> )
[SOUR[<n>]]:VOLT:SENS[:SOUR]	Power down state	OFF
STAT:OPER[:EVEN]	0	Not affected
STAT:OPER:COND	0	Not affected
STAT:OPER:ENAB	0	Not affected
STAT:OPER:INST[:EVEN]	0	Not affected
STAT:OPER:INST:COND	0	Not affected
STAT:OPER:INST:ENAB	0	Not affected
STAT:OPER:INST:ISUM[:EVEN]	0	Not affected
STAT:OPER:INST:ISUM:COND	0	Not affected
STAT:OPER:INST:ISUM:ENAB	0	Not affected
STAT:QUES[:EVEN]	0	Not affected
STAT:QUES:COND	0	Not affected
STAT:QUES:ENAB	0	Not affected
STAT:QUES:INST[:EVEN]	0	Not affected
STAT:QUES:INST:COND	0	Not affected
STAT:QUES:INST:ENAB	0	Not affected
STAT:QUES:INST:ISUM[:EVEN]	0	Not affected
STAT:QUES:INST:ISUM:COND	0	Not affected
STAT:OPER:INST:ISUM:ENAB	0	Not affected
SYST:CHAN:INFO:ONT:LAST?	0	
SYST:CHAN:INFO:ONT:TOT?	Power down state	Not affected
SYST:CPU:INFO:ONT:LAST?	0	
SYST:CPU:INFO:ONT:TOT?	Power down state	Not affected
SYST:ERR:COUN?	0	
SYST:POW	Power down state	ON
SYST:TEMP:PROT [AUX]	Power down state	55
SYST:TEMP:PROT:DEL [AUX]	Power down state	30
SYST:TEMP:PROT:STAT [AUX]	Power down state	ON
SYST:TEMP:PROT CH1 CH2	Power down state	75
SYST:TEMP:PROT:DEL CH1 CH2	Power down state	30
SYST:TEMP:PROT:STAT CH1 CH2	Power down state	ON
TRIG:DEL	Power down state	0

TRIG:SLOP	Power down state	POS
TRIG:SOUR	Power down state	IMM
TRIG:SOUR:DLOG	Power down state	IMM

### 8.3. Special modes of operation

Operations will differ when tracking or any of coupling modes between channels have been selected. The following table shows how special modes of operation affects some features:

	Calibration (CAL)	Remote sens- ing (VOLT:SENS)	Voltage programming (VOLT:PROG)	Tracking control (OUTP:TRAC)	Coupling (INST:COUP: TRAC)
TRACking	Disabled		Disabled		Disabled
Common ground	Disabled				
Split rails coupling	Disabled				
Coupled in PARallel	Disabled		Disabled	Disabled	
Coupled in SERies	Disabled	Disabled	Disabled	Disabled	

### 8.4. Default settings

Function	State / Value	SCPI command reference
Over-current protection (OCP)	OFF	CURR:PROT:STAT
Over-voltage protection (OVP)	OFF	VOLT:PROT:STAT
Over-power protection (OPP)	ON	POW:PROT:STAT
OPP trip level	155.00 W	POW:PROT:TRIP
OPP delay	10 s	POW:PROT:DEL
Channel Over-temperature protection (OTP)	ON	SYST:TEMP:PROT:STAT
Channel OTP trip level	60.00 °C	SYST:TEMP:PROT:TRIP
Channel OTP delay	30 s	SYST:TEMP:PROT:DEL
AUX sensor OTP	ON	SYST:TEMP:PROT:STAT AUX
AUX sensor OTP trip level	50 °C	SYST:TEMP:PROT:TRIP AUX
AUX sensor OTP delay	10 s	SYST:TEMP:PROT:DEL AUX
Shutdown when protection tripped	OFF	SYST:POW:PROT:TRIP
Switch off all outputs when protection tripped	OFF	OUTP:PROT:COUP
Remote voltage programming	OFF	VOLT:PROG
Remote voltage sense	OFF	VOLT:SENS
Coupling mode	NONE	INST:COUP:TRAC
Tracking mode	OFF	OUTP:TRAC
Calibration password	eezbb3	CAL:PASS:NEW
System password	Not defined	SYST:PASS:NEW
Communication mode	Local	SYST:LOC
Front panel lock	OFF	SYST:KLOC
Front panel display	ON	DISP:STAT
Ethernet communication	OFF	SYST:COMM:ENAB 0, ETH
Force disabling of all outputs on power up	OFF	SYST:PON:OUTP:DIS

Beeper	ON	SYST:BEEP:STAT
“Key pressed” click tone	OFF	SYST:BEEP:KEY:STAT
Ethernet port communication	5025	SYST:COMM:ETH:PORT
Ethernet MAC	74-69-69-2D-30-00	SYST:COMM:ETH:MAC
Serial data bits	8	–
Serial stop bits	1	–
Serial port speed	115200	–
Serial port parity bit	NONE	–
NTP service	OFF	SYST:COMM:ENAB 0, NTP
NTP server	europe.pool.ntp.org	SYST:COMM:NTP

## 9. Software simulator

The BB3 firmware can be compiled and executed on a Windows, Linux or OS X system. The software simulator is a terminal application that can respond to almost all of currently supported SCPI command described in this document. SCPI commands could be entered directly in the simulator's terminal window (Fig. 4).

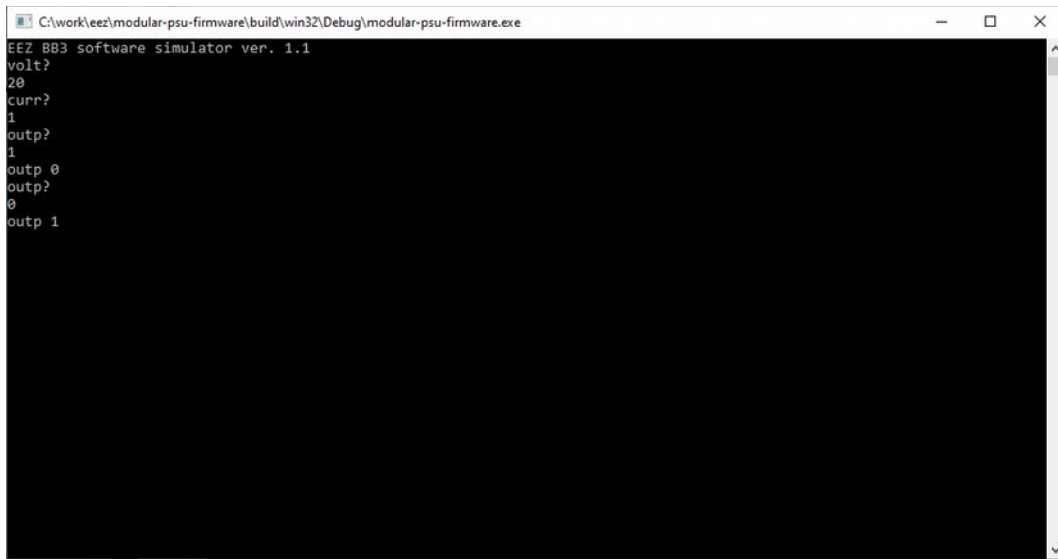


Fig. 4: Simulator terminal session

The simulator also has built-in GUI. When started, a separate window is opened with the picture of the BB3 front panel.

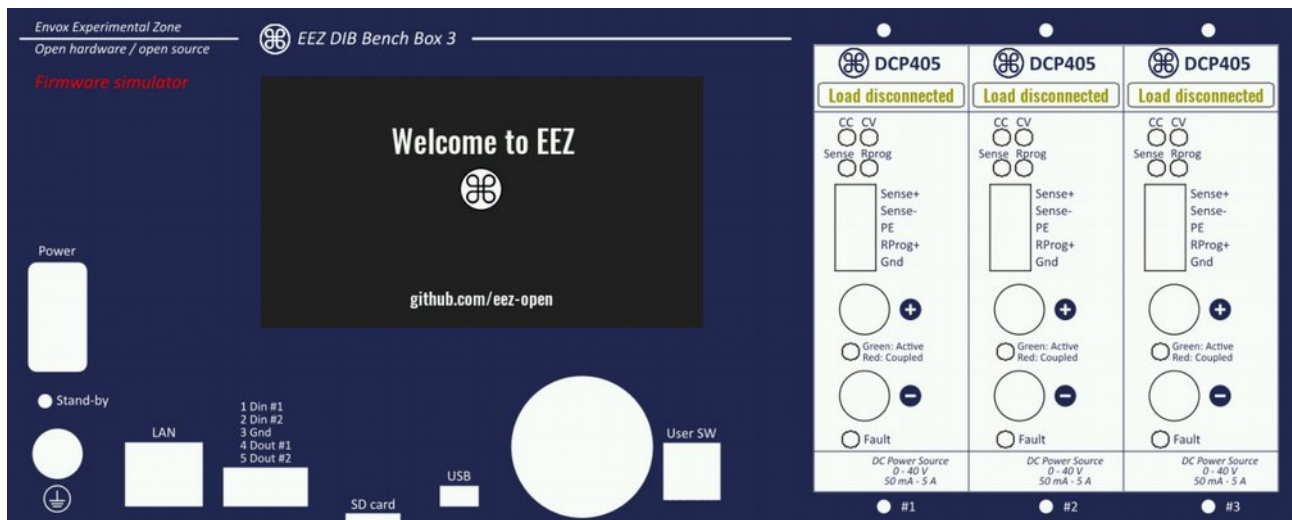


Fig. 5: Firmware simulator GUI

Additionally, firmware features could be evaluated using the Web firmware simulator that can be found on the following address: <https://www.envox.hr/eez-modular-firmware-simulator/>

## 9.1. SIMUlator

The SIMUlator software subsystem implements a set of unclassified SCPI commands for managing external parameters and events such as load impedance, connection and disconnection of the load, sensor temperature or the BB3 control circuit power supply state. Thus it is possible to exercise the measuring and protection commands that depend on external events.

For example, a MEASure:CURRent? command without a connected load will always return zero, or activation of the VOLTage:PROTection:STATe will automatically trip the OVP signal since a channel in CC operation cannot start when output is switched on, etc.

SCPI command	Description
SIMUlator	
<a href="#">:EXIT</a>	Closes simulator
<a href="#">:GUI</a>	Starts simulator's GUI
<a href="#">:LOAD {&lt;resistance&gt;}</a>	Sets the value of the simulated load
<a href="#">:STATe {&lt;bool&gt;}</a>	"Connects" a simulated load to the channel output
<a href="#">:PIN1 {&lt;bool&gt;}</a>	Sets the value of the PIN1 input
<a href="#">:PIN2 {&lt;bool&gt;}</a>	Sets the value of the PIN2 input
<a href="#">:PWRGood {&lt;bool&gt;}</a>	Sets the PWRGOOD signal state
<a href="#">:QUIT</a>	Closes simulator
<a href="#">:RPOL {&lt;bool&gt;}</a>	Sets the RPOL signal
<a href="#">:TEMP {&lt;temperature&gt;}</a>	Sets the temperature sensor value
<a href="#">:VOLT:PROG:EXT {&lt;voltage&gt;}</a>	Sets the output voltage when a channel is in external programming mode

### 9.1.1. SIMUlator:EXIT

**Syntax** [SIMUlator:EXIT](#)

**Description** This command close all Simulator windows (terminal and GUI if started).

**Usage example** `SIMU:EXIT`

### 9.1.2. SIMUlator:GUI

**Syntax** [SIMUlator:GUI](#)

**Description** This command starts the GUI simulation in a new window (see Fig. 5).

**Usage example** `SIMU:GUI`

**Related Commands** [SIMUlator:EXIT](#)

### 9.1.3. SIMUlator:LOAD

**Syntax** [SIMUlator:LOAD {<resistance>}](#)  
[SIMUlator:LOAD?](#)

**Description** This command define the impedance of a simulated load connected to a channel output. Units are in ohms. With a load connected it is possible to simulate several BB3 operations: e.g., CC mode of operation, current and power measurement, OCP and OPP functionality, etc.



The simulated load value can be also changed by clicking and dragging the load's image. Move to the left to decrease, or to the right to increase the simulated value in increments of 1  $\Omega$ .

*The simulator currently cannot emulate the "UR" mode of operation (see the [OUTPut:MODE?](#) command).*

Parameters	Name	Type	Range	Default
	<resistance>	NR2 Discrete	0 – 99999999 INFinite	–
<b>Return</b>	The query command returns the programmed load value.			
<b>Usage example</b>	SIMU:LOAD 8.2			
<b>Related Commands</b>	OUTPut:MODE?			

#### 9.1.4. SIMUlator:LOAD:STATe

**Syntax** [SIMUlator:LOAD:STATe {<bool>}](#)  
[SIMUlator:LOAD:STATe?](#)

**Description** This command "connects" or "disconnects" the simulated load on the channel output. If the GUI simulator has been started (via the SIMUlator:GUI command) a load symbol with the currently selected value in Ohms will be displayed.

Another possibility to connect (or disconnect) a load is by click on its image.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	–
<b>Return</b>	The query command returns the load state.			
<b>Usage example</b>	SIMU:LOAD:STAT?			
	0			
	MEAS:CURRE?			
	0.00			
	SIMU:LOAD:STAT ON			
	MEAS:CURRE?			
	1.50			
<b>Related Commands</b>	OUTPut:MODE? SIMUlator:GUI SIMUlator:LOAD			

#### 9.1.5. SIMUlator:PIN1

**Syntax** [SIMUlator:PIN1 {<bool>}](#)  
[SIMUlator:PIN1?](#)

**Description** This command can be used to simulate the state of the PIN1 input on the BB3 front panel push-in connector that can be used for initiate trigger.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	–
<b>Return</b>	The query command returns the state of the simulated PIN1 input.			
<b>Usage example</b>	VOLT:TRIG 12.00			
	CURRE:TRIG 2.50			
	TRIG:SOUR PIN1			
	OUTP 1			

```
INIT
SIMU:PIN1
```

**Related Commands**

```
ABORT
INITiate[:IMMediate]
TRIGger[:SEquence]:DELay
TRIGger[:SEquence]:SLOPe
TRIGger[:SEquence]:SOURce
```

### 9.1.6. SIMUlator:PIN2

**Syntax** `SIMUlator:PIN2 {<bool>}`  
`SIMUlator:PIN2?`

**Description** This command can be used to simulate the state of the PIN2 input on the BB3 front panel push-in connector that can be used for initiate trigger.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	–

**Return** The query command returns the state of the simulated PIN2 input.

**Usage example**

```
VOLT:TRIG 12.00
CURR:TRIG 2.50
TRIG:SOUR PIN1
OUTP 1
INIT
SIMU:PIN1
```

**Related Commands**

```
ABORT
INITiate[:IMMediate]
TRIGger[:SEquence]:DELay
TRIGger[:SEquence]:SLOPe
TRIGger[:SEquence]:SOURce
```

### 9.1.7. SIMUlator:PWRGood

**Syntax** `SIMUlator:PWRGood {<bool>}`  
`SIMUlator:PWRGood?`

**Description** This command can be used to simulate an internal power supply failure. When the PWR-GOOD signal is changed from 1 to 0 the BB3 will go into the standby mode (equal to the command SYSTem:POW OFF).

*The simulated BB3 mode cannot be changed until PWRGOOD is not changed to 1.*

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	ON

**Return** The query command returns the PWRGOOD signal state.

**Usage example**

```
SYST:POW?
1

SIMU:PWRG 0
SYST:POW?
0
```

**Related Commands**

```
SYSTem:POWer
```

### 9.1.8. SIMUlator:QUIT

**Syntax** `SIMUlator:QUIT`

**Description** Same as SIMUlator:EXIT

**Usage example** SIMU:QUIT

### 9.1.9. SIMUlator:RPOL

**Syntax** SIMUlator:RPOL {<bool>}  
SIMUlator:RPOL?

**Description** This command can be used to simulate detection of a remote sensing reverse polarity condition.

Parameters	Name	Type	Range	Default
	<bool>	Boolean	ON OFF 0 1	ON

**Return** The query command returns the RPOL signal state.

**Usage example**  
STAT:QUES:INST:ISUM2?  
0  
  
SIMU:RPOL 1  
STAT:QUES:INST:ISUM2?  
128

**Errors** 312, "Cannot execute when the channels are coupled"

**Related Commands** INSTRument:COUPle:TRACking  
[SOURce[<n>]]:VOLTage:SENSe[:SOURce]  
STATus:QUESTionable:INSTRument[:EVENT]?

### 9.1.10. SIMUlator:TEMPerature

**Syntax** SIMUlator:TEMPerature {<temperature>} [, <sensor>]  
SIMUlator:TEMPerature? [<sensor>]

**Description** This command sets the simulated temperature in degrees Celsius (°C), and then reads it from the simulated temperature sensor.

Parameters	Name	Type	Range	Default
	<temperature>	NR2	0 – 100	–
	<sensor>	Discrete	AUX CH1 CH2 CH3  CH4 CH5 CH6	AUX

**Return** The query command returns the set temperature value.

**Usage example**  
SIMU:TEMP 45, CH2  
MEAS:TEMP? CH2  
45.00

**Related Commands** MEASure[:SCALar]:TEMPerature[:THERmistor][:DC]

### 9.1.11. SIMUlator:VOLTage:PROGram:EXTernal

**Syntax** SIMUlator:VOLTage:PROGram:EXTernal {<voltage>}  
SIMUlator:VOLTage:PROGram:EXTernal?

**Description** This command sets the simulated voltage that will be used for output voltage programming when the simulated channel is set in external / remote programming mode (see [SOURce[<n>]]:VOLTage:PROGram[:SOURce]). For full range, apply 2.5 V; if a higher value is entered channel's OVP will trip.

Parameters	Name	Type	Range	Default
------------	------	------	-------	---------

	<voltage>	NR2	Positive value	0.00
<b>Return</b>	The query command returns the simulated external output voltage programming value.			
<b>Usage example</b>	<pre>VOLT:PROG EXT SIMU:VOLT:PROG:EXT 1.25 MEAS?  20.00</pre>			
<b>Errors</b>	312, "Cannot execute when the channels are coupled"			
<b>Related Commands</b>	<pre>INSTrument:COUPle:TRACking [SOURce[&lt;n&gt;]]:VOLTage:PROGram[:SOURce] [SOURce[&lt;n&gt;]]:VOLTage:PROTection:TRIPped?</pre>			

## 10. Programming examples

### 10.1. Set channel output values and working with the OCP

This is a SCPI commands sequence that sets a voltage, current, and the over-current protection (OCP) on the channel two:

INST?	<i>Check currently selected output</i>
1	
INST CH2	<i>Select channel two as current channel</i>
VOLT 10	<i>Set output voltage</i>
CURR	<i>Set output current</i>
CURR:PROT:STAT?	<i>Check OCP status</i>
0	
CURR:PROT:STAT 1	<i>Enable OCP</i>
CURR:PROT:DEL 100ms	<i>Set OCP delay</i>
OUTP 1	<i>Enable output</i>
MEAS?	<i>Measure output voltage</i>
10.00	
MEAS:CURR?	<i>Measure output current</i>
0.00	<i>Current is zero since no load is connected</i>

If [software simulator](#) is used, connection of the load can be also simulated:

SIMU:LOAD 20	<i>Define connected load impedance</i>
MEAS?	<i>Measure voltage once again</i>
10.00	
MEAS:CURR?	<i>Measure current once again</i>
0.50	<i>Measured current</i>

The following command sequence could be used to test channel mode with load previously defined and after the load impedance is lowered enough that output current reach programmed value. The OCP has to be disabled because previously defined 100 ms delay does not give us enough time to execute the whole sequence for testing channel mode and output voltage and current values:

OUTP:MODE?	<i>Check mode of operation</i>
"CV"	<i>The channel is in constant-voltage mode since output current is below previously programmed level</i>
SIMU:LOAD?	<i>Check load value</i>
10	
CURR:PROT:STAT?	<i>Check OCP status</i>
1	
CURR:PROT:STAT OFF	<i>Disable OCP</i>
SIMU:LOAD 4	<i>Decrease load impedance</i>
OUTP:MODE?	<i>Check once again mode of operation</i>
"CC"	<i>Channel enters constant-current mode since <math>I_{max} = U / R = 10 / 4 = 2.5 \text{ A}</math> and current is limited to the 1 A</i>

## EEZ BB3 SCPI reference

```
MEAS:CURRE?           Measure output current
1.00
MEAS?                 Measure output voltage
4.00                 Output voltage is decreased since  $U = I * R = 1 * 4 = 4\text{ V}$ 
```

The OCP will “trip” when output current reach programmed value and channel stay in the CC mode for more then programmed OCP delay time. To test that with e.g. the [software simulator](#) we'll disable channel output first, enable OCP and when change channel output back to enabled state:

```
OUTP OFF             Disable channel output
CURR:PROT:TRIP?      Check OCP status
0                   OCP is not activated
CURR:PROT:STAT ON    Enable OCP
VOLT?               Check programmed output voltage
10.00
CURR?               Check programmed output current
1.00
SIMU:LOAD?          Check simulated load value
4
OUTP ON             Enable channel output
CURR:PROT:TRIP?      Check OCP status once again
1                   OCP has been tripped
OUTP?               Check channel output state
0                   Channel output is changed to OFF stated by the OCP
```

The channel output state cannot be changed to enabled until any of protection is active. We have to clear protection first. If the same load that caused the first protection trip is still connected the channel output will be disabled immediately after the protection programmed delay time expired. Therefore we also have to disconnect load or disable protection. The later method will be used in the command sequence that follows:

```
OUTP ON             First attempt to enable channel output
OUTP?
0                   This attempt failed, the channel output remain disabled
OUTP:PROT:CLE       Channel protections reset
OUTP ON             Second to enable channel output
0                   Channel output was enabled for a short time (100ms) and returns back to OFF state
CURR:PROT:TRIP?      Check OCP status
1                   OCP has been tripped
OUTP:PROT:CLE       Reset channel protections once again
CURR:PROT:STAT OFF  Disable OCP
OUTP ON             Third attempt to enable channel output
OUTP?
1                   Output is finally enabled
OUTP:MODE?
```

"CC"

Channel enters CC mode of operation

## 10.2. Voltage and current calibration

For optimum calibration results the following condition are recommended:

- the calibration ambient temperature is stable and between 20 °C and 30 °C.
- ambient relative humidity is less than 80 %.
- Allow a one hour warm-up period before verification or calibration (use e.g. SYST:CHAN:INFO:ONT:LAST? or SYST:CPU:INFO:ONT:LAST? to get that info).
- Use short and thick cables to connect test setups.

Step	Commands	Description
1	*RST	
2	SYST:RWL	Make sure that BB3 is in remote mode and cannot be unlock from local console (TFT display)
3	INST {CH1 CH2}; OUTP ON	Select the channel to be calibrated and enable the channel output.
4	VOLT:PROT:STAT OFF CURR:PROT:STAT OFF POW:PROT:STAT OFF	Disable if required the voltage, current and power protections.
5	CAL ON, "<password>"	BB3 enters calibration mode on the channel selected in step 1. Both voltage and current on the selected channel are set to the MINimum value. The VOLT? and CURR? commands can be optionally used here to test channel output values.
6		For voltage calibration, connect a voltmeter across the power module's output terminals.
7	CAL:VOLT:LEV 1,0.15	Set the channel to the first calibration point and output voltage to 150 mV
8	CAL:VOLT 0.1455	Enter the reading you obtained from the external voltmeter.
9	CAL:VOLT:LEV 2,38	Set the channel to the second calibration point and output voltage 38 V
10	CAL:VOLT 39.292	Enter the reading you obtained from the voltmeter.
11		For current calibration, connect current monitoring resistor (shunt) across the output terminals and connect the ammeter across the shunt resistor. Its resistance has to be less then 5 $\Omega$ and rated for 25 W or more for measuring MAX current level.
12	CAL:CURR:RANG LOW	Set low current range (i.e. 0 – 50 mA).
13	CAL:CURR:LEV 1,0.0005	Set the channel to the first calibration point and output current to 0.5 mA.
14	CAL:CURR 0.000592	Enter the reading you obtained from the external ammeter.
15	CAL:CURR:LEV 2,0.048	Set the channel to the second calibration point and output current to 48 mA.
16	CAL:CURR 0.049863	Enter the reading you obtained from the ammeter.
17	CAL:CURR:RANG HIGH	Set high current range (i.e. 0 – 5 A).
18	CAL:CURR:LEV 1,0.005	Set the channel to the first calibration point and output current to 50 mA.
19	CAL:CURR 0.00603	Enter the reading you obtained from the external ammeter.
20	CAL:CURR:LEV 2,4.8	Set the channel to the second calibration point and output current to 4.8 A.
21	CAL:CURR 5.0731	Enter the reading you obtained from the ammeter.

22	CAL:REM "<string>"	Record calibration information such as next calibration due date for future reference. The calibration string may contain up to 32 characters. You don't need to enter current date and time since that information will be recorded automatically.
23	CAL:SAVE	Save to non-volatile memory new calibration data.
24	CAL OFF, "<password>"	BB3 exit calibration mode. Both voltage and current on the selected channel are again set to the MINimum value.
25	SYST:REM	Enable local console unlock. Alternatively SYST:LOC can be executed to make local console enabled again.

### 10.3. Working with profiles

The following command sequence could be used to store current set of parameters to the profile location 4 in the non-volatile memory:

MEM:STAT:VAL? 4	Check to see if profile selected location is empty
0	
MEM:STAT:NAME? 4	We can also check that by querying profile location name
--Not used--	
INST CH1	Examine currently programmed output values of the first channel
VOLT?;:CURR?;:OUTP?	
0.00;0.00;0	
INST CH2	Examine currently programmed output values of the second channel
VOLT?;:CURR?;:OUTP?	
0.00;0.00;0	
VOLT 12;:CURR 300mA	Reprogram both channel output values that will be stored as a new profile
INST CH1	
VOLT 12;:CURR 300mA	
OUTP 1;:OUTP 1, CH2	
*SAV 4	All profile parameters is now storing on the selected location
MEM:STAT:NAME? 4	Check profile name
" "	
MEM:STAT:NAME 4, "Dual 12V/300mA, Output ON"	Set the profile name (only ASCII characters are allowed!)
MEM:STAT:NAME? 4	Check the profile name once again
"Dual 12V/300ma, Output ON"	

We can now turn the BB3 off (when it enters the standby mode) turn it on again and check some of the programmed parameters:

SYST:POW 0	The BB3 enters the standby mode
SYST:POW 1	Returns back from the standby mode. Please note that this command can be executed with the minimum of 5 seconds delay otherwise a -200,"Execution error" will be generated (you can check that with the SYST:ERR? command)
VOLT?;:CURR?;:OUTP?	Query programmed voltage, current and output state of the currently selected channel
0.00;0.00;0	Returned data indicate that previously saved values in profile



<pre>*RCL 4 VOLT?;:CURR?;:OUTP? 12.00;0.30;1</pre>	<p><i>number 4 were not used</i></p> <p><i>Recall parameters from desired location and execute query once again</i></p> <p><i>The channel output values are now programmed using the selected profile</i></p>
--	---

We can automate above mentioned process that channel profile parameters stored in non-volatile memory are using on power up. First we'll check what is a current status of automatic recall and what profile will be used in the case of automatic recall:

<pre>MEM:STAT:REC:AUTO? 0 MEM:STAT:REC:AUTO ON MEM:STAT:REC:SEL? 0 MEM:STAT:REC:SEL 4 SYST:POW 0 SYST:POW 1 VOLT?;:CURR?;:OUTP? 12.00;0.30;1</pre>	<p><i>Query status of automatic profile recall during power on sequence</i></p> <p><i>Automatic recall is turned off</i></p> <p><i>Turn on automatic recall</i></p> <p><i>Query which profile will be used when automatic recall is turned on</i></p> <p><i>Selected profile was 0</i></p> <p><i>Change power on profile to 4</i></p> <p><i>Switch the BB3 to the standby mode once again</i></p> <p><i>Returns back from the standby mode. Again wait at least 5 seconds before enters this command</i></p> <p><i>Query programmed voltage, current and output state of the currently selected channel</i></p> <p><i>The channel output values are programmed using the selected profile</i></p>
--	---

#### 10.4. Get identification info and self-test results

The BB3's identification information could be beneficial when more then one instrument are controlled. Additionally in the following example information about self-test will be queried:

<pre>*IDN? Envox,EEZ BB3 (STM32),00000000,M1 0.2 *TST? 0 DIAG:TEST? "2, EEPROM, installed, passed", "2, SD card, installed, passed", "2, Ethernet, installed, passed", "2, RTC, installed, passed", "2, DateTime, installed, passed", "4, BP option, not installed, failed", "2, Fan, installed, passed", "2, AUX temp, installed, passed", "2, CH1 temp, installed, passed", "2, CH2 temp, installed, passed"</pre>	<p><i>Query identification string</i></p> <p><i>BB3 with two different channels is identified, the first channel is 0-50V/3A and the second is 0-40V/5A. Serial number is 00001, and firmware version M1.0.93</i></p> <p><i>Execute self-test and query result</i></p> <p><i>Self-test is passed</i></p> <p><i>Query additional information about self-test</i></p>
--	---

The self-test could be performed even when the BB3 is in the standby mode. We'll first switch the BB3 into the standby mode. At the end of this example we are using additional diagnostic command that allows us to query information about channel's ADC measurements.

SYST:POW 0

*The BB3 enters the standby mode*

DIAG:TEST?

*Query additional information about self-test*

"2, EEPROM, installed, passed",  
 "2, SD card, installed, passed",  
 "2, Ethernet, installed, passed",  
 "2, RTC, installed, passed",  
 "2, DateTime, installed, passed",  
 "4, BP option, not installed, failed",  
 "2, Fan, installed, passed",  
 "2, AUX temp, installed, passed",  
 "2, CH1 temp, installed, passed",  
 "2, CH2 temp, installed, passed"

SYST:POW 1

*Returns back from the standby mode*

DIAG:ADC?

*Additional information about currently selected channel ADC inputs*

"U\_SET=12.02 V", "U\_MON=12.00  
 V", "I\_SET=0.30 A", "I\_MON=0.00 A"

*U\_SET and I\_SET are measured values of the DAC outputs, U\_MON and I\_MON are actual output values. I\_MON is 0 because no load is connected. A small difference between set and actual output voltage exists because calibration data are currently in use.*

## 10.5. Programming output voltage using the list of values

The BB3 comes with simple “arbitrary waveform generator” functionality that can be accomplished using the LIST commands. The following example changes in the loop output voltage between five output values each half a second long while current is set to 3 A.

INST CH1

*Select the channel that has to be programmed*

VOLT:MODE LIST

*Set voltage programming mode to the list of values*

LIST:VOLT 5, 10, 20, 40, 0

*Define sequence of output voltage*

CURR:MODE LIST

*Set current programming mode to the list of values.*

LIST:CURR 3

*Only one output current value is defined. Single value or number of values equivalent to other parameters (LIST:VOLT and LIST:DWEL) is allowed.*

LIST:DWEL 0.5

*Only one value for the duration of each step is chosen. Single value or number of values equivalent to other parameters (LIST:VOLT and LIST:CURR) is allowed.*

LIST:COUN INF

*Repeat continuously LIST sequence*

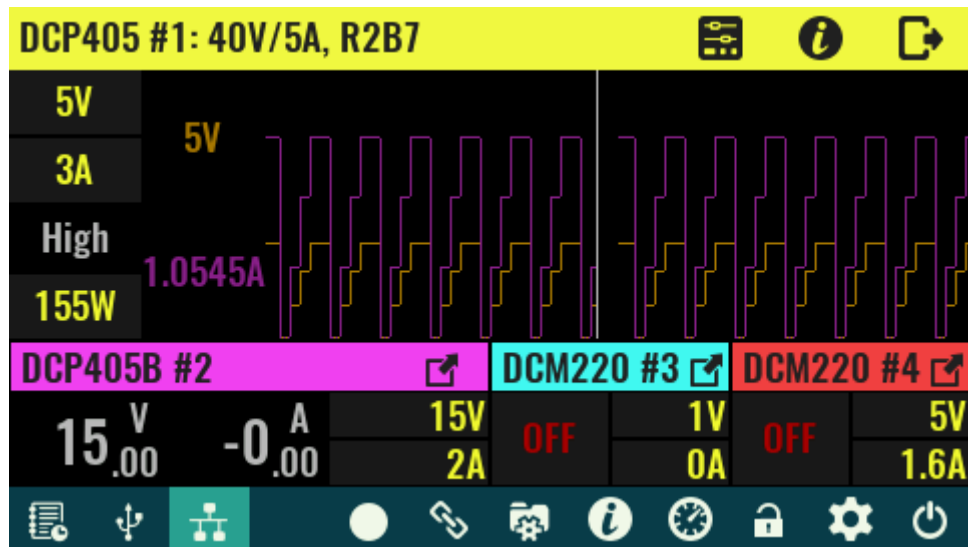
TRIG:SOUR IMM

*Define type of trigger. In this case the list execution will start immediately after INIT command is received.*

INIT

*Start the trigger.*

Resulting output voltage and current waveform with connected load of 4.7  $\Omega$  is shown on the picture below.



## 11. SCPI commands scheduled for upcoming releases

Please note that the following list is preliminary.

SCPI command	Description
DCL	Requires the client (controller) to send a "DCL\n" string
*SRE	Sets the value of the Service Request Enable register
*WAI	Waits until all pending commands are completed
CALibration	
:TEMPerature	
[:DATA] {<new value>}	Enters the calibration value
DISPlay	
:CMAP {<palette>}	Sets color palette (theme)
:CMAP:COLor:CATalog?	Returns names and RGB values of all available colors
:CMAP:COLor[:RGB] {<name>, <red>, <green>, <blue>}	Sets RGB value for the selected color
INSTrument	
:COUPle	
:TRIGger {<mode>}	Selects a coupling between channels trigger systems
MMEMory	
:CLOSe	Closes the file specified in NAME
:FEED	Sets data handle used to feed data into the file
:NAME {<filename>}	Sets the file name to be opened or closed
:OPEN	Opens the file specified in NAME
OUTPut	
:PROTection	
:MEASure {<bool>}	Enables measuring of output voltage before channel output is turned on
SENSe	
:AHOur	
:RESet {<channel>}	Resets the amp-hour (Ah) measurement to zero
:WHOur	
:RESet {<channel>}	Resets the watt-hour (Wh) measurement to zero
[SOURce[<n>]]	
:ARB	
:AMPlitude {<amplitude>}	Sets the arbitrary waveform amplitude
:COUNt {<count>}	Sets the number of arbitrary waveforms
:FREQuency {<frequency>}	Sets the arbitrary waveform frequency
:FUNCTion {<type>}	Sets the arbitrary waveform parameter type
:OFFset {<level>}	Sets the offset level of the arbitrary waveform
:PHASe[:ADJust] {<angle>}	Set the arbitrary waveform start phase
:SHAPE {<waveform>}	Selects the shape of the arbitrary waveform
SYSTEM	

:CHANnel	
:INFOrmation	
:AHOur	
TOTal?	Returns channel's total delivered energy in Ah
:WHOur	
TOTal?	Returns channel's total delivered energy in Wh
:COMMunicate	
:ETHernet	
:CONTRol?	Queries communication port for SRQ handling
:DIGital	
:TOUTput	
:BUS[:ENABLE] {<Bool>}	Enables/disables BUS-generated triggers on digital pins
:ERRor	
:ALL?	Queries the error/event queue for all the unread items
:CODE[:NEXT]?	Queries the error/event queue for the next item code
:KEY	
:DEFine {<key>, <block>}	Sets the definition of the key
:DELete {<key>}	Removes the definition of the key
:TIME	
:TIMer	
[:STATe]	Sets the internal timer state
:COUNt	Queries the current value of the internal timer

## 12. SCPI commands summary

Common command	Description
*CLS	Clears all status data structures
*ESE {<value>}	Programs the Standard Event Status Enable register bits
*ESR?	Reads the Standard Event Status Register
*IDN?	Returns the UNIQUE identification of the BB3
*OPC	Operation Complete Command used for program synchronization
*RCL {<profile>}	Recalls the BB3 state stored in the specified storage location
*RST	Reset BB3 to the initial state
*SAV {<profile>}	Stores the current BB3 state in the specified storage location
*STB?	Reads the Status Byte register
*TRG	Generates a software trigger
*TST?	Returns Self-Test results
*WAI	Waits until all pending commands are completed

SCPI Command	Description
ABORT	Resets the trigger system to the Idle state
:DLOG	Stops the internal data logging session
CALibrate[:MODE] {<bool>, <password>}	Enables/disables calibration mode
:CLEar {<password>}	Clears all calibration parameters
:CURRent	
[:DATA] {<new value>}	Enters the calibration value
:LEVel {<level>}	Calibrates the output current programming
:RANGe {<range>}	Sets current range for multiple current range model
:PASSword	
:NEW {<old>, <new>}	Changes calibration password
:REMark {<string>}	Saves calibration information
:SAVE	Saves the new cal constants in non-volatile memory
:STATe {<bool>, <password>}	Enables calibration parameters
:VOLTage	
[:DATA] {<new value>}	Enters the calibration value
:LEVel {<level>}	Calibrates the output voltage programming
DIAGnostic	
[:INfOrmation]	
:ADC?	Returns the latest values acquired by ADC
:CALibration?	Returns a list of the calibration parameters
:FAN?	Returns status of the cooling fan.
:PROTection?	Returns the information about all protections.
:TEST?	Returns results of the most recent self-test
DISPlay	

:BRIGhtness {<value>}	Sets the intensity of the front panel TFT display
:CMAP {<palette>}	Sets color palette (theme)
:CMAP:COLor:CATalog?	Returns names and RGB values of all available colors
:CMAP:COLor[:RGB] {<name>, <red>, <green>, <blue>}	Sets RGB value for the selected color
:DATA?	Reads screen image data
:MODE {<mode>}	Sets the main page appearance
[:WINDow]	
[:STATe] {<bool>}	Sets the front panel TFT display state
:DLOG	Opens DLOG viewer
:INPUt? {<label>, {<type>} [, <min>, <max>, <value>]}	Displays entry form and wait for input on the front panel TFT display
:TEXT {<message>}	Displays a message on the front panel TFT display
:CLEar	Clear a message on the front panel TFT display
<hr/>	
INITiate	
[:IMMediate]	Completes one full trigger cycle
:DLOG {<filename>}	Initiates internal data logging
:TRACe {<filename>}	Initiates internal data logging
:CONTinuous {<bool>}	Enables/disables continuous transient triggers
<hr/>	
INSTrument	
[:SElect] {<channel>}	Selects the output to be programmed
:CATalog?	Returns a quoted string of the list of valid choices for the instrument channels
:FULL?	Returns a list of string – number pairs
:COUPle	
:TRACking {<type>}	Selects independent, parallel-tracking, or series-tracking mode
:DISPlay	
:TRACe[<n>] {<value>}	Selects output value on the specified display trace
:SWAP	Swaps positions of selected output values
:YT	
:RATE {<duration>}	Selects YT view sample duration
:NSElect {<channel>}	Selects the output to be programmed
<hr/>	
MEASure	
[:SCALar]	
:CURRent	
[:DC]? [<channel>]	Takes a measurement; returns the average current
:POWer	
[:DC]? [<channel>]	Takes a measurement; returns the average power
[:VOLTage]	
[:DC]? [<channel>]	Takes a measurement; returns the average voltage
<hr/>	
MEMory	
:NSTATes?	Returns total number of state storage memory loca-

	tions
:STATe	
:CATalog?	Lists the names associated with all ten state storage locations
:DELete {<profile>}	Deletes the contents of a state storage location
:ALL	Deletes the contents of all state storage locations
:NAME {<profile>, <name>}	Assigns a custom name to a state storage locations
:RECall	
:AUTO {<bool>}	Specifies whether the power-down state is recalled from location 0 on power-on
:SElect {<profile>}	Specifies which BB3 state will be used at power on
:VALid? {<profile>}	Determines whether a storage location contains a valid state
<hr/>	
MMEMemory	
:CATalog [<directory>]	Returns a list of items in the specified directory (folder)
:LENGth [<directory>]	Returns the number of items in the specified directory
:CDIRectory {<directory>}	Changes the current directory
:COPY {<source>, <destination>}	Copies <source> to <destination>
:DATE? {<filename>}	Returns date that the specified file was last saved
:DELete {<filename>}	Deletes an existing file
:DOWNload	
:ABORt	Aborts current download session
:DATA :DATA {#<length>, <encoding>, <block>}	Downloads data from the host computer
:FNAMe {<filename>}	Creates or opens the specified filename for download data
:SIZE {<filesize>}	Sets information about file size used for progress bar
:LOAD	
:LIST[<n>] {<filename>}	Loads stored LIST to the specified channel
:PROFile {<filename>}	Loads stored user profile
:STATe {<filename>}	Loads the instrument setup
:LOCK {<password>}	Sets write protection
:MDIRectory {<directory>}	Makes a new directory
:MOVE {<source>, <destination>}	Moves or renames <source> to <destination>
:RDIRectory {<directory>}	Removes the specified directory
:STORe	
:LIST[<n>] {<filename>}	Saves specified channel LIST
:PROFile {<filename>}	Saves specified user profile
:STATe {<filename>}	Saves the instrument setup
:TIME? {<filename>}	Returns time that the specified file was last saved
:UNLock {<password>}	Clears write protection
:UPLoad? {<filename>}	Uploads data to the host computer
<hr/>	
OUTPut	
[ :STATe ] {<bool>}	Enables the specified output channel(s)



TRIGgered {<bool>} [, <channel>]	Controls channel output state with trigger
:DELay	
:DURation {<duration>} [, <channel>]	Sets the output start delay duration
:MODE?	Returns the channel mode of operation
:PROTection	
:CLEar	Resets latched protection
:COUPle {<bool>}	Enables channel coupling for protection faults
:TRACK[:STATe] {<chanlist>}	Defines channels to operate in the tracking mode
<hr/>	
ROUTe	
:CHANnel	
:LABel {<label>},{<chanlist>}	Sets labels for relay matrix channels
:CLOSe {<chanlist>}	Closes (turns on) specified relay matrix channels
:OPEN {<chanlist>}	Opens (turns off) specified relay matrix channels
<hr/>	
SIMULator	
:EXIT	Closes simulator
:GUI	Starts simulator's GUI
:LOAD {<value>}	Sets value of the virtual load
:STATe {<bool>}	"Connects" virtual load to the channel output
:PIN1 {<bool>}	Sets value of the PIN1 input
:PWRGood {<bool>}	Sets the PWRGOOD signal state
:RPOL {<bool>}	Sets the RPOL signal state
:TEMP {<value>}	Sets the temperature sensor value
:VOLT:PROG:EXT {<voltage>}	Sets the output voltage when channel is in external programming mode
<hr/>	
SENSe	
:CURRent	
[:DC]	
RANGe[:UPPer] {<range>}	Selects a DC current measurement range
AUTO {<bool>}	Enables/disables seamless measurement auto ranging
:DLOG	
:FUNCTION	
:CURRent {<bool>}, {<channel>}	Enables/disables output current internal data logging
:POWer {<bool>}, {<channel>}	Enables/disables output power internal data logging
:VOLTage {<bool>}, {<channel>}	Enables/disables output voltage internal data logging
:PERiod {<time>}	Sets the sample period for internal data logging
:TIME {<time>}	Sets the sample duration for internal data logging
:TRACe	
:X	
:UNIT {<unit>}	Sets DLOG viewer X-axis units
:STEP {<step>}	Sets DLOG viewer X-axis step value
:LABel {<label>}	Sets DLOG viewer X-axis label

<code>[:RANGe]:MIN {&lt;min&gt;}</code>	Sets DLOG viewer X-axis min. value
<code>[:RANGe]:MAX {&lt;max&gt;}</code>	Sets DLOG viewer X-axis max. value
<code>:Y&lt;n&gt;</code>	
<code>:UNIT {&lt;unit&gt;}</code>	Sets DLOG viewer Y-axis units
<code>:LABel {&lt;label&gt;}</code>	Sets DLOG viewer Y-axis label
<code>[:RANGe]:MIN {&lt;min&gt;}</code>	Sets DLOG viewer Y-axis min. value
<code>[:RANGe]:MAX {&lt;max&gt;}</code>	Sets DLOG viewer Y-axis max. value

**[SOURce[<n>]]****:CURRent****[[:LEVel]**

`[[:IMMediate]][:AMPLitude]` Sets the output current  
`{<current>}`

`:STEP[:INCRement] {<step>}` Sets the step of the current change

`:TRIGgered [:AMPLitude]` Sets the triggered output current  
`{<current>}`

**:LIMit**

`[[:POSitive]][:IMMediate]` Sets the output current limit  
`[[:AMPLitude] {<current>}`

**:PROTection****:DELay**

`[[:TIME] {<time>}` Sets the over-current protection (OCP) programming delay

`:STATe {<bool>}` Enables/disables over-current protection on the selected channel

`:TRIPped?` Returns status of over-current protection activation

**:LIST**

`:COUNT` Sets the number of times that the list is executed

`:CURRent[:LEVel]` Specifies the current setting for each list step

`:DWELl` Specifies the dwell time for each list step

`:VOLTage[:LEVel]` Specifies the voltage setting for each list step

**:POWER**

`:LIMit {<power>}` Sets the output power limit

`:PROTection[:LEVel]` Sets the over-power protection (OPP) level

**:DELay**

`[[:TIME] {<time>}` Sets the over-power protection programming delay

`:STATe {<bool>}` Enables/disables over-power protection on the selected channel

`:TRIPped?` Returns status of over-power protection activation

**:VOLTage****[[:LEVel]**

`[[:IMMediate]][:AMPLitude]` Sets the output voltage  
`{<voltage>}`

`:STEP[:INCRement] {<step>}` Sets the step of the voltage change

`:TRIGgered [:AMPLitude]` Sets the triggered output voltage  
`{<voltage>}`

:LIMit	
[:POSitive][:IMMediate] [:AMPLitude] {<voltage>}	Sets the output voltage limit
:PROGram[:SOURce] {<source>}	Sets voltage programming source
:PROTection[:LEVel]	Sets the over-voltage protection (OVP) level
:DELay	
[:TIME] {<time>}	Sets the over-voltage protection (OVP) programming delay
:STATe {<bool>}	Enables/disables over-voltage protection on the selected channel
:TRIPped?	Returns status of over-voltage protection activation
:SENSe[:SOURce] {<source>}	Sets voltage sense inputs source

## STATus

:OPERation	
[:EVENT]?	Returns the value of the Operation Event register
:CONDition?	Returns the value of the Operation Instrument Condition register
:ENABLE {<value>}	Enables specific bits in the Operation Event register
:INSTrument[<n>]	
[:EVENT]?	Returns the value of the Operation Instrument Event register
:CONDition?	Returns the value of the Operation Instrument Condition register
:ENABLE {<value>}	Enables specific bits in the Operation Instrument Event register
:ISUMmary<n>	
[:EVENT]?	Returns the value of the Operation Instrument Isummary Event register
:CONDition?	Returns the value of the Operation Instrument Isummary Condition register
:ENABLE {<value>}	Enables specific bits in the Operation Instrument Isummary Event register
:PREset	Presets all enable registers to power-on state
:QUEStionable	
[:EVENT]?	Returns the value of the Questionable Event register
:CONDition?	Returns the value of the Questionable Condition register
:ENABLE {<value>}	Enables specific bits in the Questionable Event register
:INSTrument[<n>]	
[:EVENT]?	Returns the value of the Questionable Instrument Event register
:CONDition?	Returns the value of the Questionable Instrument Condition register
:ENABLE {<value>}	Enables specific bits in the Questionable Instrument Event register
:ISUMmary<n>	
[:EVENT]?	Returns the value of the Questionable Instrument Isummary Event register

:CONDition?	Returns the value of the Questionable Instrument Summary Condition register
:ENABle {<value>}	Enables specific bits in the Questionable Instrument Summary Event register

---

SYSTem	
:BEEPer[:IMMEDIATE]	Issues a single beep immediately
:STATe {<bool>}	Enables beeper function
:KEY	
:STATe {<bool>}	Enables click tone for local control
:CAPability?	Returns an <instrument_specifier>
:CHANnel	
:[:COUNT]?	Returns the number of output channels
:INFORMATION	
:AHOuR	
:TOTal?	Returns channel's total delivered energy in Ah
:CURRent?	Returns output current capability
:ONtime	
:LAST?	Returns time passed after last output enable
:TOTal?	Returns channel's total active time
:POWer?	Returns output power capability
:VOLTage?	Returns output voltage capability
:WHOuR	
:TOTal?	Returns channel's total delivered energy in Wh
:MODEl?	Returns the channel model and version name
:SLOT?	Returns the channel slot number
:SNO?	Returns the channel serial number
:VERSion?	Returns the channel version number
:COMMunicate	
:ENABle {<bool>, <interface>}	Enables the remote interface
:ETHernet	
:ADDReSS {<ip_address>}	Sets the static LAN (IP) address
:DHCP {<bool>}	Enables the use of the Dynamic Host Configuration Protocol (DHCP)
:DNS <ip_address>	Sets the IP address of the DNS server.
:GATEWay {<ip_address>}	Sets the IP address of the default gateway
:MAC?	Returns the MAC address
:PORT {<number>}	Sets the port number
:SMASk {<mask>}	Sets the static subnet mask
:NTP {<server>}	Set s NTP service server address
:RLSTate {<state>}	Places the instrument in remote or local mode
CPU	
:FIRMware?	Returns BB3 firmware version

:INFOrmation	
:ONtime	
LAST?	Returns time passed after last power on
TOTal?	Returns BB3's total active time
:TYPE?	Returns the type of CPU
:MODEl?	Returns the control board model name
:SNO?	Returns the BB3 serial number
:VERSion?	Returns the channel version number
:DATE {<yyyy>,<mm>,<dd>}	Sets the date of the system clock
:DIGital	
:INPut:DATA? [<pin>]	Reads the state of the digital port pins
:OUTPut	
:DATA {<pin>}, {<state>}	Sets the state of the digital port pins
:PWM	
:DUTY {<pin>}, {<duty>}	Sets square wave generator duty cycle
:FREQuency {<pin>}, {<duty>}	Sets square wave generator frequency
:PIN<n>	
:FUNCTion {<function>}	Sets the selected pin's function
:POLarity {<polarity>}	Sets the selected pin's polarity
:TOUTput	
:BUS[:ENABle] {<Bool>}	Enables/disables BUS-generated triggers on digital pins
:ERRor	
[:NEXT]?	Queries and clears errors from the error queue
:COUNT?	Queries the error/event queue for the number of unread items
:FAN	
:SPEEd	Returns speed of the cooling fan
:STATus	Returns status of the cooling fan
:FORMat	
:DATE	Sets format for displaying date
:TIME	Sets 12h or 24h clock format
:INHibit?	Queries system inhibit state
:KEY	
:DEFine {<key>, <block>}	Sets the definition of the key
:DELete {<key>}	Removes the definition of the key
:KLOCK	Disables front panel [lock/unlock] icon
:LOCAL	Places the BB3 in the local mode
:MEASure	
[:SCALar]	
:TEMPerature	
[:DC]? {<sensor>}	Takes a measurement; returns the average temperature

[:VOLTage]	
[:DC]? {<device>}	Takes a voltage measurement of the RTC battery
:PASSword	
:CALibration	
:RESet	Resets the calibration password to initial value
:FPANel	
:RESet	Resets the front panel lock password to initial value
:NEW {<old>, <new>}	Changes system password
:PON:OUTPut:DISable {<bool>}	Sets output state on power up
:POWEr {<bool>}	Enters the BB3 into the standby mode
:PROtection:TRIP {<bool>}	Enters the BB3 into the standby mode in case of protection trip
:RElay:CYCLes? {<chanlist>}	Returns the cycle count on the specified channels
:REMOte	Places the BB3 in the remote mode
:RWLock	Places the BB3 in the remote mode and disables front panel [lock/unlock] icon
:TEMPerature	
:PROtection	
[:HIGH]	
[:LEVel] {<temperature>[, <sensor>]}	Sets the OTP value
:CLEar [, {<sensor>}]	Clears the latched protection status of the over-temperature protection (OTP)
:DElay	Sets time-out period
[:TIME] {<delay>[, <sensor>]}	Sets the OTP programming delay
:STATe {<bool>[, <sensor>]}	Enables/disables OTP on the selected temperature sensor
:TRIPped? [ <sensor>]	Returns status of OTP activation
:TIME {<hh>,<mm>,<ss>}	Sets the time of the system clock
:DTS {rules}	Defines daylight saving time (DST) rules
:ZONE {zone}	Defines time zone
:VERSion?	Returns the SCPI version number

## TRIGger

[:SEquence]	
[:IMMediate]	
:DElay {<delay>}	Sets the time delay between the detection of a trigger event and the start of any corresponding trigger action
:EXIT	
:CONDition {<condition>}	Sets channel's condition when LIST execution stopped
:SOURce {<source>}	Sets the trigger source
:DLOG	
[:IMMediate]	
:SOURce {<source>}	Sets the internal data logger trigger source



---

For more info visit: [www.envox.hr/eez](http://www.envox.hr/eez)  
File repository: <https://github.com/eez-open>

Document version: 1.2  
Date: 2020-09-22