

# Planar TR1300/1, TR5048 and TR7530

Vector Network Analyzer

Programming Manual COM/DCOM



---

Revision: 2.3  
October 2014



## TABLE OF CONTENTS

Introduction .....	7
1 Registering COM Server .....	7
2 COM Technology Overview .....	7
3 Automation Server .....	8
4 Automation Controllers .....	8
5 Local and Remote Server .....	8
6 Structure of COM Objects .....	10
7 Accessing the Application Object .....	11
8 Object Methods .....	13
9 Object Properties .....	13
10 Error Handling .....	14
11 COM Automation Data Types .....	15
12 Measurement Data Arrays .....	16
13 COM Server Commands .....	17
NAME .....	17
Ready .....	18
SCPI.ABORT .....	19
SCPI.CALCulate( <i>Ch</i> ).FSIMulator.SENDEd.DEEMbed.PORT( <i>Pt</i> ).USER.FILename .....	20
SCPI.CALCulate( <i>Ch</i> ).FSIMulator.SENDEd.DEEMbed.PORT( <i>Pt</i> ).STATe .....	22
SCPI.CALCulate( <i>Ch</i> ).FSIMulator.SENDEd.PMCircuit.PORT( <i>Pt</i> ).USER.FILename .....	23
SCPI.CALCulate( <i>Ch</i> ).FSIMulator.SENDEd.PMCircuit.PORT( <i>Pt</i> ).STATe .....	24
SCPI.CALCulate( <i>Ch</i> ).FSIMulator.SENDEd.ZCONversion.PORT( <i>Pt</i> ).Z0.R .....	25
SCPI.CALCulate( <i>Ch</i> ).FSIMulator.SENDEd.ZCONversion.STATe .....	26
SCPI.CALCulate( <i>Ch</i> ).PARAmeter.COUNT .....	27
SCPI.CALCulate( <i>Ch</i> ).PARAmeter( <i>Tr</i> ).DEFine .....	28
SCPI.CALCulate( <i>Ch</i> ).PARAmeter( <i>Tr</i> ).SELect .....	30
SCPI.CALCulate( <i>Ch</i> ).SELected.CONVersion.FUNCTion .....	31
SCPI.CALCulate( <i>Ch</i> ).SELected.CONVersion.STATe .....	32
SCPI.CALCulate( <i>Ch</i> ).SELected.CORRection.EDELay.TIME .....	33
SCPI.CALCulate( <i>Ch</i> ).SELected.CORRection.OFFSet.PHASE .....	34
SCPI.CALCulate( <i>Ch</i> ).SELected.DATA.FDATa .....	35
SCPI.CALCulate( <i>Ch</i> ).SELected.DATA.FMEMory .....	36
SCPI.CALCulate( <i>Ch</i> ).SELected.DATA.SDATa .....	37
SCPI.CALCulate( <i>Ch</i> ).SELected.DATA.SMEMory .....	38
SCPI.CALCulate( <i>Ch</i> ).SELected.FILTer.GATE.TIME.CENTer .....	39
SCPI.CALCulate( <i>Ch</i> ).SELected.FILTer.GATE.TIME.SHAPe .....	40
SCPI.CALCulate( <i>Ch</i> ).SELected.FILTer.GATE.TIME.SPAN .....	41
SCPI.CALCulate( <i>Ch</i> ).SELected.FILTer.GATE.TIME.START .....	42
SCPI.CALCulate( <i>Ch</i> ).SELected.FILTer.GATE.TIME.STATe .....	43
SCPI.CALCulate( <i>Ch</i> ).SELected.FILTer.GATE.TIME.STOP .....	44
SCPI.CALCulate( <i>Ch</i> ).SELected.FILTer.GATE.TIME.TYPE .....	45
SCPI.CALCulate( <i>Ch</i> ).SELected.FORMat .....	46
SCPI.CALCulate( <i>Ch</i> ).SELected.FUNCTion.DATA .....	47
SCPI.CALCulate( <i>Ch</i> ).SELected.FUNCTion.DOMain.COUPLE .....	48
SCPI.CALCulate( <i>Ch</i> ).SELected.FUNCTion.DOMain.START .....	49
SCPI.CALCulate( <i>Ch</i> ).SELected.FUNCTion.DOMain.STATe .....	50
SCPI.CALCulate( <i>Ch</i> ).SELected.FUNCTion.DOMain.STOP .....	51
SCPI.CALCulate( <i>Ch</i> ).SELected.FUNCTion.EXECute .....	52
SCPI.CALCulate( <i>Ch</i> ).SELected.FUNCTion.PEXCursion .....	53
SCPI.CALCulate( <i>Ch</i> ).SELected.FUNCTion.POINts .....	54
SCPI.CALCulate( <i>Ch</i> ).SELected.FUNCTion.PPOLarity .....	55
SCPI.CALCulate( <i>Ch</i> ).SELected.FUNCTion.TARGet .....	56
SCPI.CALCulate( <i>Ch</i> ).SELected.FUNCTion.TTRansition .....	57
SCPI.CALCulate( <i>Ch</i> ).SELected.FUNCTion.TYPE .....	58
SCPI.CALCulate( <i>Ch</i> ).SELected.LIMit.DATA .....	59
SCPI.CALCulate( <i>Ch</i> ).SELected.LIMit.DISPLAy.STATe .....	60
SCPI.CALCulate( <i>Ch</i> ).SELected.LIMit.FAIL .....	61
SCPI.CALCulate( <i>Ch</i> ).SELected.LIMit.OFFSet.AMPLitude .....	62
SCPI.CALCulate( <i>Ch</i> ).SELected.LIMit.OFFSet.STIMulus .....	63
SCPI.CALCulate( <i>Ch</i> ).SELected.LIMit.REPort.ALL .....	64

SCPI.CALCulate( <i>Ch</i> ).SELEcted.LIMit.REPort.DATA .....	65
SCPI.CALCulate( <i>Ch</i> ).SELEcted.LIMit.REPort.POINts .....	66
SCPI.CALCulate( <i>Ch</i> ).SELEcted.LIMit.STATe .....	67
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).ACTivate .....	68
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).BWIDth.DATA .....	69
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).BWIDth.REFERence .....	70
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).BWIDth.STATe .....	71
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).BWIDth.THReshold .....	72
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).BWIDth.TYPE .....	73
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).COUple .....	74
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNCTion.DOMain.START .....	75
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNCTion.DOMain.STATe .....	76
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNCTion.DOMain.STOP .....	77
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNCTion.EXECute .....	78
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNCTion.PEXCursion .....	79
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNCTion.PPOLarity .....	80
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNCTion.TARGET .....	81
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNCTion.TRACKing .....	82
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNCTion.TTRansition .....	83
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNCTion.TYPE .....	84
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).REFERence.STATe .....	85
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).SET_ .....	86
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).STATe .....	87
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).X .....	88
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).Y .....	89
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).COUNT .....	90
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MATH.FUNCTion .....	91
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MATH.MEMorize .....	92
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MAXhold.STATe .....	93
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MSTATistics.DATA .....	94
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MSTATistics.DOMain.MARKer.START .....	95
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MSTATistics.DOMain.MARKer.STOP .....	96
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MSTATistics.DOMain.STATe .....	97
SCPI.CALCulate( <i>Ch</i> ).SELEcted.MSTATistics.STATe .....	98
SCPI.CALCulate( <i>Ch</i> ).SELEcted.RLIMit.DATA .....	99
SCPI.CALCulate( <i>Ch</i> ).SELEcted.RLIMit.DISPlay.LINE .....	100
SCPI.CALCulate( <i>Ch</i> ).SELEcted.RLIMit.FAIL .....	101
SCPI.CALCulate( <i>Ch</i> ).SELEcted.RLIMit.REPort.DATA .....	102
SCPI.CALCulate( <i>Ch</i> ).SELEcted.RLIMit.STATe .....	103
SCPI.CALCulate( <i>Ch</i> ).SELEcted.SMOothing.APERTure .....	104
SCPI.CALCulate( <i>Ch</i> ).SELEcted.SMOothing.STATe .....	105
SCPI.CALCulate( <i>Ch</i> ).SELEcted.TRANSform.TIME.CENTer .....	106
SCPI.CALCulate( <i>Ch</i> ).SELEcted.TRANSform.TIME.IMPulse.WIDTh .....	107
SCPI.CALCulate( <i>Ch</i> ).SELEcted.TRANSform.TIME.KBESsel .....	108
SCPI.CALCulate( <i>Ch</i> ).SELEcted.TRANSform.TIME.LPFRequency .....	109
SCPI.CALCulate( <i>Ch</i> ).SELEcted.TRANSform.TIME.REFlection .....	110
SCPI.CALCulate( <i>Ch</i> ).SELEcted.TRANSform.TIME.SPAN .....	111
SCPI.CALCulate( <i>Ch</i> ).SELEcted.TRANSform.TIME.START .....	112
SCPI.CALCulate( <i>Ch</i> ).SELEcted.TRANSform.TIME.STATe .....	113
SCPI.CALCulate( <i>Ch</i> ).SELEcted.TRANSform.TIME.STEP.RTIME .....	114
SCPI.CALCulate( <i>Ch</i> ).SELEcted.TRANSform.TIME.STIMulus .....	115
SCPI.CALCulate( <i>Ch</i> ).SELEcted.TRANSform.TIME.STOP .....	116
SCPI.CALCulate( <i>Ch</i> ).SELEcted.TRANSform.TIME.TYPE .....	117
SCPI.CALCulate( <i>Ch</i> ).SELEcted.TRANSform.TIME.UNIT .....	118
SCPI.CALCulate( <i>Ch</i> ).TRACe( <i>Tr</i> ).DATA.FDATa .....	119
SCPI.CALCulate( <i>Ch</i> ).TRACe( <i>Tr</i> ).DATA.FMEMory .....	120
SCPI.CALCulate( <i>Ch</i> ).TRACe( <i>Tr</i> ).DATA.SDATa .....	121
SCPI.CALCulate( <i>Ch</i> ).TRACe( <i>Tr</i> ).DATA.SMEMory .....	122
SCPI.DISPlay.ANNotation.FREQuency.STATe .....	123
SCPI.DISPlay.COLor.BACK .....	124
SCPI.DISPlay.COLor.GRATICule .....	125
SCPI.DISPlay.COLor.RESet .....	126
SCPI.DISPlay.COLor.TRACe( <i>Tr</i> ).DATA .....	127

SCPI.DISPlay.COLor.TRACe( <i>Tr</i> ).MEMory .....	128
SCPI.DISPlay.FSIGn .....	129
SCPI.DISPlay.IMAGe.....	130
SCPI.DISPlay.MAXimize.....	131
SCPI.DISPlay.SPLit.....	132
SCPI.DISPlay.UPDate_ IMMEDIATE.....	133
SCPI.DISPlay.WINDow( <i>Ch</i> ).ACTivate .....	133
SCPI.DISPlay.WINDow( <i>Ch</i> ).ANNotation.MARKer.ALIGN.TYPE .....	134
SCPI.DISPlay.WINDow( <i>Ch</i> ).ANNotation.MARKer.SINGLE.STATE.....	135
SCPI.DISPlay.WINDows( <i>Ch</i> ).MAXimize .....	136
SCPI.DISPlay.WINDows( <i>Ch</i> ).SPLit .....	137
SCPI.DISPlay.WINDow( <i>Ch</i> ).TITLE.DATA.....	138
SCPI.DISPlay.WINDow( <i>Ch</i> ).TITLE.STATE .....	139
SCPI.DISPlay.WINDow( <i>Ch</i> ).TRACe( <i>Tr</i> ).ANNotation.MARKer.POSition.X .....	140
SCPI.DISPlay.WINDow( <i>Ch</i> ).TRACe( <i>Tr</i> ).ANNotation.MARKer.POSition.Y .....	141
SCPI.DISPlay.WINDow( <i>Ch</i> ).TRACe( <i>Tr</i> ).Y.SCALE.AUTO.....	142
SCPI.DISPlay.WINDow( <i>Ch</i> ).TRACe( <i>Tr</i> ).Y.SCALE.PDIVision.....	143
SCPI.DISPlay.WINDow( <i>Ch</i> ).TRACe( <i>Tr</i> ).Y.SCALE.RLEVel .....	144
SCPI.DISPlay.WINDow( <i>Ch</i> ).TRACe( <i>Tr</i> ).Y.SCALE.RPOSITION .....	145
SCPI.DISPlay.WINDow( <i>Ch</i> ).Y.SCALE.DIVisions .....	146
SCPI.HCOPy.DATE.STAMP .....	147
SCPI.HCOPy.IMAGe .....	148
SCPI.HCOPy.IMMEDIATE .....	149
SCPI.HCOPy.PAINt .....	150
SCPI.IEEE4882.IDN.....	151
SCPI.IEEE4882.RST.....	152
SCPI.IEEE4882.TRG.....	152
SCPI.IEEE4882.WAI.....	153
SCPI.INITiate( <i>Ch</i> ).CONTinuous .....	154
SCPI.INITiate( <i>Ch</i> ).IMMEDIATE.....	155
SCPI.MMEMory.COPY( <i>Src</i> , <i>Dst</i> ) .....	156
SCPI.MMEMory.DELeTe( <i>File</i> ).....	156
SCPI.MMEMory.LOAD.CKIT( <i>Ck</i> ).....	157
SCPI.MMEMory.LOAD.CALibration.....	158
SCPI.MMEMory.LOAD.LIMit.....	159
SCPI.MMEMory.LOAD.RLIMit.....	160
SCPI.MMEMory.LOAD.SEGMent .....	161
SCPI.MMEMory.LOAD.STATE .....	162
SCPI.MMEMory.MDIRectory .....	163
SCPI.MMEMory.STORe.CKIT( <i>Ck</i> ).....	164
SCPI.MMEMory.STORe.CALibration .....	165
SCPI.MMEMory.STORe.FDATA .....	166
SCPI.MMEMory.STORe.IMAGe.....	167
SCPI.MMEMory.STORe.LIMit.....	168
SCPI.MMEMory.STORe.RLIMit .....	169
SCPI.MMEMory.STORe.SEGMent .....	170
SCPI.MMEMory.STORe.SNP.DATA .....	171
SCPI.MMEMory.STORe.SNP.FORMat .....	172
SCPI.MMEMory.STORe.SNP.TYPE.S1P.....	173
SCPI.MMEMory.STORe.SNP.TYPE.S2P.....	174
SCPI.MMEMory.STORe.STATE.....	175
SCPI.MMEMory.STORe.STYPE.....	176
SCPI.OUTPUT.STATE .....	177
SCPI.SENSE( <i>Ch</i> ).AVERAge.CLEAR .....	178
SCPI.SENSE( <i>Ch</i> ).AVERAge.COUNT.....	179
SCPI.SENSE( <i>Ch</i> ).AVERAge.CURRENT.....	180
SCPI.SENSE( <i>Ch</i> ).AVERAge.STATE.....	181
SCPI.SENSE( <i>Ch</i> ).BANDwidth.RESolution.....	182
SCPI.SENSE( <i>Ch</i> ).CORRection.CLEAR .....	183
SCPI.SENSE( <i>Ch</i> ).CORRection.COEFFicient.DATA( <i>Str</i> , <i>Pt_r</i> , <i>Pt_s</i> ).....	184
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.ACQUIRE.LOAD.....	185
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.ACQUIRE.OPEN .....	186
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.ACQUIRE.SHORT.....	187

SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.ACQUIRE.THRU	188
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.CKIT.LABEL	189
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.CKIT.RESet	190
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.CKIT.SELect	190
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.CKIT.STAN( <i>Std</i> ).C0	192
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.CKIT.STAN( <i>Std</i> ).C1	193
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.CKIT.STAN( <i>Std</i> ).C2	194
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.CKIT.STAN( <i>Std</i> ).C3	195
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.CKIT.STAN( <i>Std</i> ).DELay	196
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.CKIT.STAN( <i>Std</i> ).L0	197
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.CKIT.STAN( <i>Std</i> ).L1	198
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.CKIT.STAN( <i>Std</i> ).L2	199
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.CKIT.STAN( <i>Std</i> ).L3	200
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.CKIT.STAN( <i>Std</i> ).LABel	201
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.CKIT.STAN( <i>Std</i> ).LOSS	202
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.CKIT.STAN( <i>Std</i> ).TYPE	203
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.CKIT.STAN( <i>Std</i> ).Z0	204
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.CLEar	205
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.ECAL.INFOrmation	206
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.ECAL.ORIenation.STATe	207
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.ECAL.PATH( <i>Pt</i> )	208
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.ECAL.SOLT1	209
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.ECAL.UCHar	210
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.METHod.ERESponse	211
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.METHod.RESPonse.OPEN	212
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.METHod.RESPonse.SHORt	213
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.METHod.RESPonse.THRU	214
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.METHod.SOLT1	215
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.METHod.TYPE	216
SCPI.SENSE( <i>Ch</i> ).CORRection.COLLECT.SAVE	217
SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.PORT( <i>Pt</i> ).FREQuency( <i>LS</i> )	218
SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.PORT( <i>Pt</i> ).INCLude( <i>LS</i> ).STATe	219
SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.PORT( <i>Pt</i> ).LDC	220
SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.PORT( <i>Pt</i> ).LOSS( <i>LS</i> )	221
SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.PORT( <i>Pt</i> ).TIME	222
SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.STATe	223
SCPI.SENSE( <i>Ch</i> ).CORRection.IMPedance.INPut.MAGNitude	224
SCPI.SENSE( <i>Ch</i> ).CORRection.STATe	225
SCPI.SENSE( <i>Ch</i> ).CORRection.TRANSform.TIME.FREQuency	226
SCPI.SENSE( <i>Ch</i> ).CORRection.TRANSform.TIME.LOSS	227
SCPI.SENSE( <i>Ch</i> ).CORRection.TRANSform.TIME.RVELocity	228
SCPI.SENSE( <i>Ch</i> ).CORRection.TRANSform.TIME.STATe	229
SCPI.SENSE( <i>Ch</i> ).CORRection.TYPE( <i>Tr</i> )	230
SCPI.SENSE( <i>Ch</i> ).FREQuency.CENTer	231
SCPI.SENSE( <i>Ch</i> ).FREQuency.CW	232
SCPI.SENSE( <i>Ch</i> ).FREQuency.DATA	233
SCPI.SENSE( <i>Ch</i> ).FREQuency.SPAN	234
SCPI.SENSE( <i>Ch</i> ).FREQuency.STARt	235
SCPI.SENSE( <i>Ch</i> ).FREQuency.STOP	236
SCPI.SENSE( <i>Ch</i> ).ROSCillator.SOURce	237
SCPI.SENSE( <i>Ch</i> ).SEGMENT.DATA	238
SCPI.SENSE( <i>Ch</i> ).SWEep.POINt.TIME	239
SCPI.SENSE( <i>Ch</i> ).SWEep.POINts	240
SCPI.SENSE( <i>Ch</i> ).SWEep.REVERSE.STATe	241
SCPI.SENSE( <i>Ch</i> ).SWEep.TYPE	242
SCPI.SERVice.CHANnel( <i>Ch</i> ).ACTive	243
SCPI.SERVice.CHANnel( <i>Ch</i> ).COUNt	243
SCPI.SERVice.CHANnel( <i>Ch</i> ).TRACe.ACTive	244
SCPI.SERVice.CHANnel( <i>Ch</i> ).TRACe.COUNt	244
SCPI.SERVice.PORT.COUNt	245
SCPI.SERVice.SWEep.FREQuency.MAXimum	245
SCPI.SERVice.SWEep.FREQuency.MINimum	246
SCPI.SERVice.SWEep.POINts	246

SCPI.SOURce( <i>Ch</i> ).POWER.CENTer.....	247
SCPI.SOURce( <i>Ch</i> ).POWER.LEVel.IMMediate.AMPLitude.....	248
SCPI.SOURce( <i>Ch</i> ).POWER.LEVel.SLOPe.DATA.....	249
SCPI.SOURce( <i>Ch</i> ).POWER.PORT( <i>Pt</i> ).CORRection.DATA.....	250
SCPI.SOURce( <i>Ch</i> ).POWER.PORT( <i>Pt</i> ).CORRection.STATe.....	251
SCPI.SOURce( <i>Ch</i> ).POWER.SPAN.....	252
SCPI.SOURce( <i>Ch</i> ).POWER.START.....	253
SCPI.SOURce( <i>Ch</i> ).POWER.STOP.....	254
SCPI.SYSTem.COMMunicate.ECAL.TEMPerature.SENSor.....	255
SCPI.SYSTem.CORRection.STATe.....	256
SCPI.SYSTem.DATE.....	257
SCPI.SYSTem.HIDe.....	258
SCPI.SYSTem.LOCal.....	259
SCPI.SYSTem.PRESet.....	260
SCPI.SYSTem.REMote.....	261
SCPI.SYSTem.RWLock.....	262
SCPI.SYSTem.SHOW.....	263
SCPI.SYSTem.TERMinate.....	264
SCPI.SYSTem.TIME.....	265
SCPI.TRIGger.INPut.POLarity.....	266
SCPI.TRIGger.SEQuence.IMMediate.....	267
SCPI.TRIGger.SEQuence.POINt.....	268
SCPI.TRIGger.SEQuence.SINGle.....	269
SCPI.TRIGger.SEQuence.SOURce.....	270
SCPI.TRIGger.SEQuence.STATe.....	271
<b>Appendix 1. Error Codes</b> .....	272
<b>Appendix 2. Programming Examples</b> .....	273

## Introduction

This Programming Manual contains information on remote control over Planar TR1300/1, TR5048 or TR7530 Vector Network Analyzer (Analyzer) and its data communication by means of user programs written with COM/DCOM technology.

COM technology is used when a user program runs together with an external measurement instrument program on one PC. DCOM technology is used when a user program runs on a PC connected with the measurement instrument by LAN.

Methods and techniques for writing of user programs are same for the both technologies. The only difference between the technologies is that the DCOM technology requires additional LAN setting performed by the LAN administrator.

Before reading this Manual, familiarize yourself with Analyzer Operating Manual.

## 1 Registering COM Server

To register COM server of the analyzer run the executable module from command prompt with the `/regserver` keyword. To unregister COM server of the analyzer run the executable module from command prompt with the `/unregserver` keyword. Administrative rights is required to register/unregister COM server. Also user has ability to register COM server during the software installation procedure.

Example of the COM server registration command:

```
PlanarTR1300.exe /regserver
TR5048.exe /regserver
TR7530.exe /regserver
```

## 2 COM Technology Overview

COM stands for *Component Object Model*. This programming technology was developed by Microsoft for two purposes:

- the model provides the specification for interaction of binary modules created in different programming languages;
- the model defines the interfacing between a client application and a server application running either on the same PC or on two different PCs. In the latter case, the technology has DCOM abbreviation – Distributed COM.

### 3 Automation Server

The network analyzer executable module contains a built-in COM server that enables other programs to access its functionality. The COM server was developed in conformity with the *COM automation* specification. COM automation is a technology allowing control over the COM server by the programs written in both traditional compiling programming languages and interpreting programming languages, such as VBScript. This enables the server applications to make their functionality accessible to many more clients.

### 4 Automation Controllers

*Automation controllers* are client programs, which use internal functionality of COM servers. Automation controller programs are developed by users for writing their own add-ons for the system.

User programs can be written in different languages:

- programming languages with built-in COM support, such as Visual Basic®, Delphi, Java;
- universal programming languages, such as C, C++;
- Microsoft Excel and Word office applications as they include built-in programming language Visual Basic for Applications®;
- program generators, such as National Instruments LabView®, or HP-VEE.

Examples represented in this Manual are written in Visual Basic (VB). Appendix 3 contains examples written in VB, and C++ languages.

Examples\COM\VBA folder contains source codes for examples written in Visual Basic for Applications® (Microsoft Excel files).

Examples\COM\CPP folder contains source codes for the C++ language examples.

### 5 Local and Remote Server

The network analyzer executable module can function either as a *local* server or as a *remote* server of COM automation.

*Local server* runs on the same PC with the automation controller and each of the programs is executed as an individual application in a separate window. COM technology is used in this case (Figure 1).

*Remote server* and the automation controller run on different PCs connected by LAN. DCOM (Distributed COM) technology is used in this case (Figure 2). When using



DCOM it is necessary to configure the local network by means of DCOM Windows tools.

The same automation controller is used for the both COM and DCOM technology. Some changes to the user program may be required in operators, which establish connection with the server. Moreover, DCOM technology requires additional settings of the LAN performed by the LAN administrator.

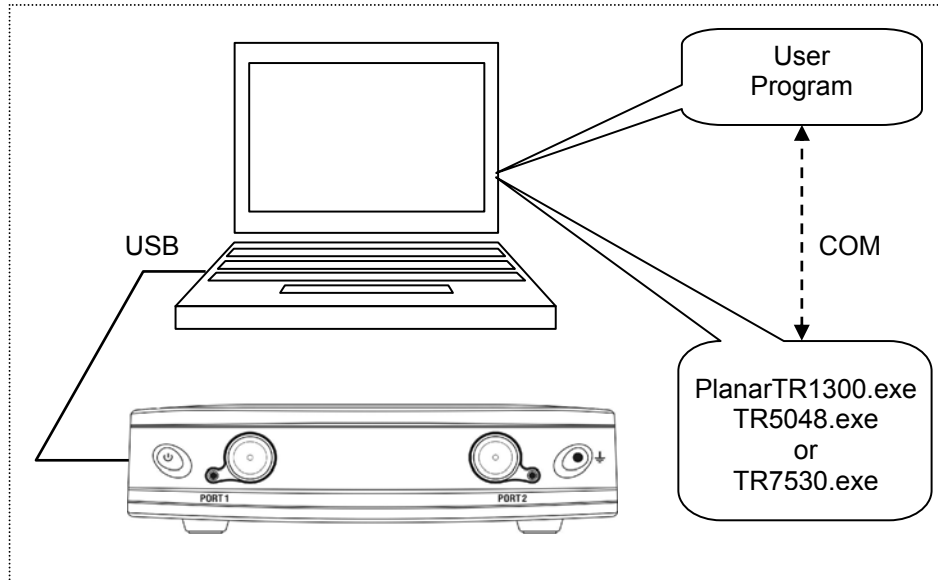


Figure 1. COM technology

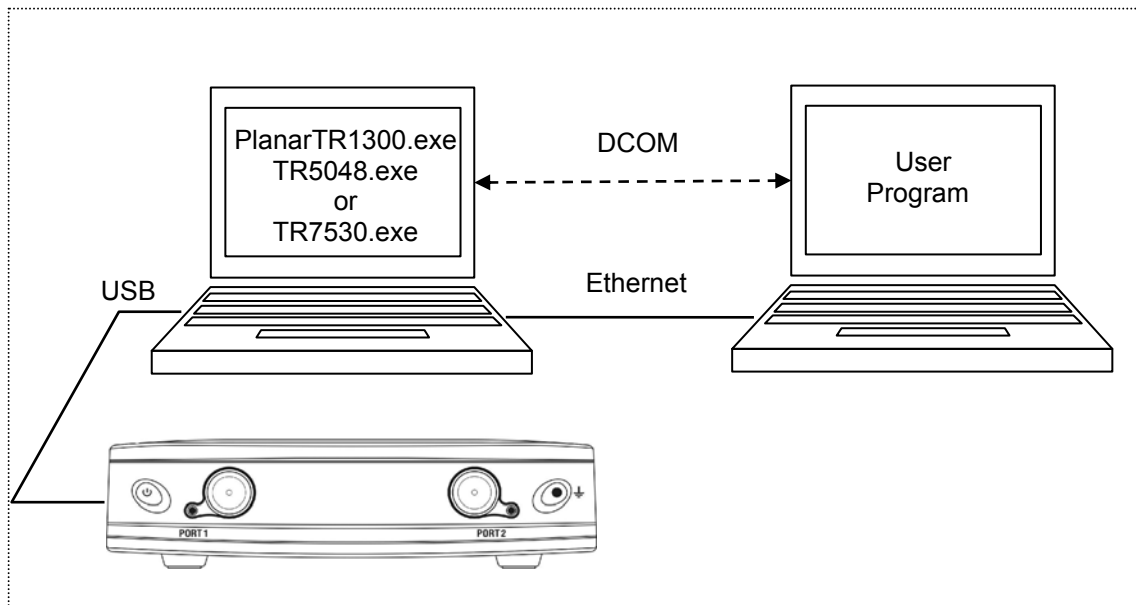


Figure 2. DCOM technology

## 6 Structure of COM Objects

COM server contains several *objects*, which provide different functionality of the server. The COM objects of the network analyzer executable module are organized in a hierarchical structure. Figure 3 shows the main COM objects, which comprise the first three levels of the hierarchical structure of the COM server. COM objects provide various *methods* and *properties*, which allow access to the server functions; besides, they allow access to the objects of the lower levels, which are not shown in Figure 3.

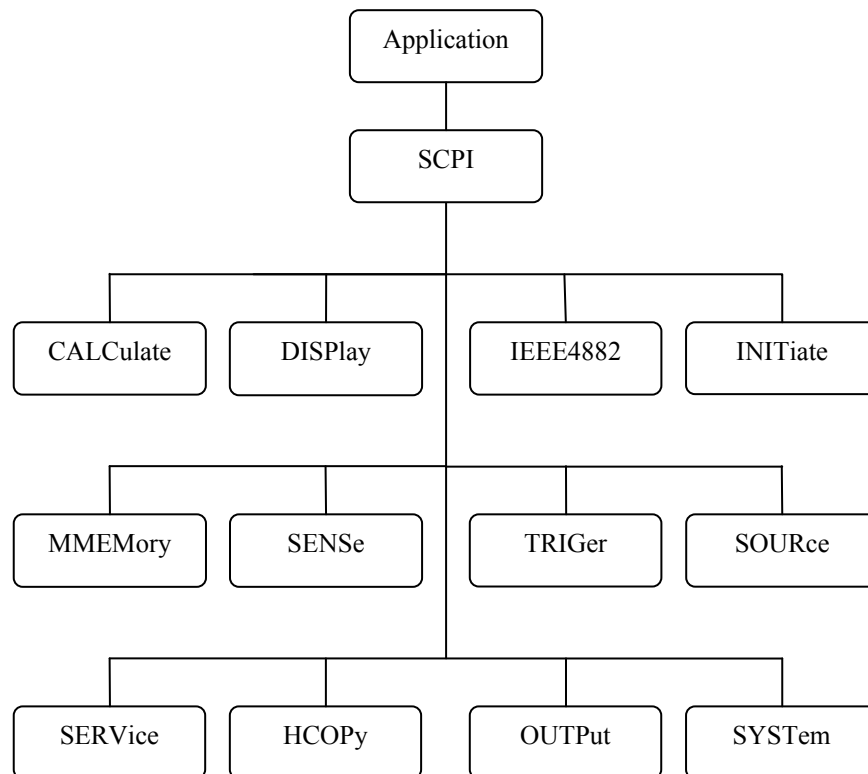


Figure 3. The structure of COM objects

The Object Application is in the top of the hierarchy COM server. Access to the lower level objects is implemented via higher level objects.

---

**Note** The hierarchy of COM objects is organized in accordance with the standard and syntax of the SCPI programming language. Operators in SCPI have hierarchical chain structure, for example:

```
CALCulate:PARAmeter:DEFine S11
```

The same COM command written in VB programming language is as follows:

```
app.SCPI.CALCulate.PARAmeter.DEFIne = "S11"
```

---

## 7 Accessing the Application Object

To establish connection with the COM server application, create an object reference in the client program. In COM programming the object reference needs to be acquired preliminarily, to be used later to access the object functionality. To define an object in Visual Basic perform the following:

- 1) Declare a variable as an object.
- 2) Assign the object to this variable.

To declare a variable, use *Dim* operator or other declaration statement (*Public*, *Private* or *Static*). The variables used for references should of the types *Variant*, *Object*, or a type of a specific object. For example, the following three operators declare *app* variable:

```
Dim app
Dim app as Object
Dim app as TR1300.Application
Dim app as TR5048.Application
Dim app as TR7530.Application
```

Use *Set* operator and *CreateObject* (*ObjectName*, *HostName*) function to assign a specific object to a variable.

<i>ObjectName</i>	Automation object name is always equal to “ <i>TR1300.Application</i> ”, “ <i>TR5048.Application</i> ” or “ <i>TR7530.Application</i> ”
<i>HostName</i>	Network name of the PC hosting the COM server. This parameter is not specified in case of a local server.

For example, the following operators create *Application* object and assign it to *app* variable:

```
Set app = CreateObject("TR5048.Application")
Set app = CreateObject("TR5048.Application", "Network_Name")
Set app = CreateObject("TR5048.Application", "192.168.10.10")
```

---

Note	The first form of the operator is used to create the reference to the local COM server, the second one is used to create the reference to the remote DCOM server.
------	---

---

To allow access to the objects of a lower level of the hierarchy, these objects are specified after the reference to the higher level object and separated from it by a dot. For example:

```
Dim SystObj
Set SystObj = app.SCPI.SYSTEM
```

COM objects can have indices. For example, *CALCulate*, *INITiate*, *SENSE*, *SOURCE* objects represent various aspects of the 4 measurement channels of the Analyzer. Therefore, it is necessary to write the channel index from 1 to 4 to acquire the data of these objects. For example:

```
Set SensObj1 = app.SCPI.SENSE(1)
Set SensObj2 = app.SCPI.SENSE(2)
```

Visual Basic allows omitting of such indices; in this case the indices are considered as equal to 1. For example, the following VB operators are equivalent:

```
Set SensObj = app.SCPI.SENSE(1)
Set SensObj = app.SCPI.SENSE
```

## 8 Object Methods

Objects have methods. Methods are actions that can be applied to objects. The object methods are specified after the object name and separated from it by a dot.

The following example shows the *PRESet* method of *SYSTEM* object. This method performs setting of the Analyzer to the preset condition:

```
app.SCPi.SYSem.PRESet
```

## 9 Object Properties

Along with methods, objects have properties. Properties are object characteristics that can be set or read out. The object properties are specified after the object name and separated from it by a dot.

To modify an object characteristic, write the value of the corresponding property. To define an object characteristic, read out the value of its property. The following example show the setting of the *POINts* property of *SWEep* object, i.e. the number of sweep points:

```
app.SCPi.SENSE.SWEp.POINts = 201
```

---

Note	Some object properties cannot be written, and some object properties cannot be read. In such cases, the properties are indicated as “read only” or “write only”.
------	--

---

## 10 Error Handling

You can use different approaches to error handling in VB program:

- check the value of `Err.Number` variable after execution of VB operator, which contains the call to COM server object;
- use `On Error goto` VB operator.

These approaches are represented in the examples below. The following operator causes an error in VB program as "S13" value of the *DEFine* property is incorrect.

```
app.SCPI.PARAmeter.DEFine = "S13"
```

In the first example, the value of the *Err.Number* variable is checked after execution of the VB operator, which contains the call to COM server object. *On Error Resume Next* directive instructs VB not to interrupt the program execution when the error is detected but to pass control to the next operator in natural order.

```
Dim app
Public Sub HandleError1()
Set app = CreateObject("TR5048.Application")
On Error Resume Next
app.SCPI.PARAmeter.DEFine = "S13"
If Err.Number <> 0 Then
Msg = "Error # " & Str(Err.Number) & " was generated by " &_
Err.Source & Chr(13) & Err.Description
MsgBox Msg, , "Error"
End If
...
End Sub
```

In the second example, *On Error GoTo ErrHandler* directive instructs VB to interrupt the program execution when the error is detected and to pass control to *ErrHandler* label.

```
Dim app
Public Sub HandleError2()
Set app = CreateObject("TR5048.Applcation")
On Error GoTo ErrHandler
app.SCPI.PARAmeter.DEFine = "S13"
...
Exit Sub
ErrHandler:
Msg = "Error # " & Str(Err.Number) & " was generated by " & _
Err.Source & Chr(13) & Err.Description
MsgBox Msg, , "Error"
End Sub
```

## 11 COM Automation Data Types

In COM automation, there are the following data types, which can be used for client-to-server communication:

<b>Long</b>	32-bit signed integer, value range from -2147483648 to 2147483647
<b>Double</b>	64-bit double-precision floating point, value range from -1.79769313486232E308 to -4.94065645841247E-324 for negative values, and from 4.94065645841247E-324 to 1.79769313486232E308 for positive values
<b>Boolean</b>	16-bit integer, two values 0 – <i>False</i> , 1 – <i>True</i>
<b>String</b>	Variable-length string
<b>VARIANT</b>	Can be either a value of arbitrary type or an array of values of arbitrary type. In this case, the term “arbitrary type” means any allowed type of COM automation. A variable contains information about its type and array size (if it is an array). It is used for communication of data arrays between a client and a server.

## 12 Measurement Data Arrays

Measurement data can be either complex values or real values. This depends on the format selected by the user. For example, the data is real in logarithmic magnitude format and the data is complex in polar format.

The measurement data is transferred in a *Variant* type variable, which represents an array of *Double* type. To transfer one complex measurement, two adjacent array cells are used. To transfer one real measurement two adjacent array cells are used as well but the second cell is always equal to 0. Thus, measurement data array size is a double number of the measurement points.

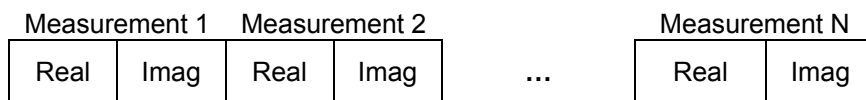


Figure 4. Array of complex measurements

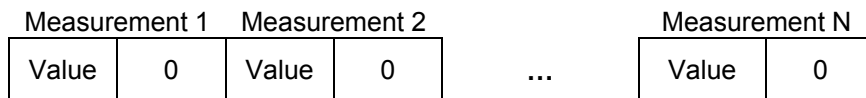


Figure 5. Array of real measurements



## 13 COM Server Commands

### NAME

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	String
<i>Description</i>	Instrument information string. String format: manufacturer, model, serial number, number of firmware version and number of software version.
<i>Range</i>	up to 40 characters
<i>Syntax</i>	Dim <i>ID</i> As String <i>ID</i> = <i>app.NAME</i>
<i>Equivalent Softkeys</i>	<b>None</b>

## Ready

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Boolean
<i>Description</i>	Ready state of the instrument. Reads out the <i>True</i> value after successful completion of the boot process (about 10 sec). The vector analyzer must be connected to PC by a USB cable.
<i>Syntax</i>	Dim <i>State</i> as Boolean <i>State</i> = <i>app</i> .Ready
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.ABORT**

<i>Object Type</i>	Method
<i>Description</i>	Aborts the sweep. Switches trigger mode from <i>Single</i> to <i>Hold</i> , or from <i>Continuous</i> to waiting for a trigger. If the trigger source is set to <i>Internal</i> , starts a new sweep.
<i>Syntax</i>	<code>app.SCPI.ABORT</code>
<i>Equivalent Softkeys</i>	<b>None</b>

## SCPI.CALCulate(*Ch*).FSIMulator.SENDEd.DEEMbed.PORT(*Pt*).USER. FILename

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Port <i>Pt</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Pt</i> : port number 1–2 (see Table 2 on page 21)
<i>Description</i>	De-embedding function file name (*.s2p). The file contains the circuit S-parameters in Touchstone format.
<i>Range</i>	up to 256 characters
<i>Preset Value</i>	""
<i>Syntax</i>	Dim <i>File</i> As String <i>File</i> = <code>app.SCPI.CALCulate(<i>Ch</i>).FSIMulator.SENDEd.DEEMbed.PORT(<i>Pt</i>).USER.FILename</code>  <code>app.SCPI.CALCulate(<i>Ch</i>).FSIMulator.SENDEd.DEEMbed.PORT(<i>Pt</i>).USER.FILename</code> = "network.s2p"
<i>Notes</i>	If the full path to the file is not specified, the <code>\FixtureSim</code> subdirectory of the main directory will be searched for the file.
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Fixture Simulator &gt; De-Embedding &gt; S-parameters File</b>

**Table 1. Ch: Channel Number**

<i>Data Type</i>	Long
<i>Description</i>	Channel number.
<i>Range</i>	from 1 to 4
<i>Out of Range</i>	An error occurs. Error code: 201.

**Table 2. Pt: Port Number**

<i>Data Type</i>	Long
<i>Description</i>	Port number.
<i>Range</i>	from 1 to 2
<i>Out of Range</i>	An error occurs. Error code: 114.

**SCPI.CALCulate(*Ch*).FSIMulator.SENDEd.DEEMbed.PORT(*Pt*).STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
	Port <i>Pt</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Pt</i> : port number 1–2 (see Table 2 on page 21)
<i>Description</i>	The ON/OFF state of the e-embedding function.
<i>Allowable Values</i>	True: De-embedding function ON False: De-embedding function OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = app.SCPi.CALCulate( <i>Ch</i> ).FSIMulator.SENDEd.DEEMbed.PORT( <i>Pt</i> ).STATe app.SCPi.CALCulate( <i>Ch</i> ).FSIMulator.SENDEd.DEEMbed.PORT( <i>Pt</i> ).STATe = True
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Fixture Simulator &gt; De-Embedding &gt; Port n</b>

## SCPI.CALCulate(*Ch*).FSIMulator.SENDEd.PMCircuit.PORT(*Pt*).USER. FILEname

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Port <i>Pt</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Pt</i> : port number 1–2 (see Table 2 on page 21)
<i>Description</i>	Embedding function file name (*.s2p). The file contains the circuit S-parameters in Touchstone format.
<i>Range</i>	up to 256 characters
<i>Preset Value</i>	""
<i>Syntax</i>	Dim <i>File</i> As String <i>File</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).FSIMulator.SENDEd.PMCircuit.PORT( <i>Pt</i> ).USER.FILEname  <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).FSIMulator.SENDEd.PMCircuit.PORT( <i>Pt</i> ).USER.FILEname = "network.s2p"
<i>Notes</i>	If the full path to the file is not specified, the \FixtureSim subdirectory of the main directory will be searched for the file.
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Fixture Simulator &gt; Embedding &gt; User File</b>

**SCPI.CALCulate(*Ch*).FSIMulator.SENDEd.PMCircuit.PORT(*Pt*).STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Port <i>Pt</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Pt</i> : port number 1–2 (see Table 2 on page 21)
<i>Description</i>	The ON/OFF state of the embedding function.
<i>Allowable Values</i>	True: Embedding function ON False: Embedding function OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).FSIMulator.SENDEd.DEEMbed.PORT( <i>Pt</i> ).STATe <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).FSIMulator.SENDEd.DEEMbed.PORT( <i>Pt</i> ).STATe = True
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Fixture Simulator &gt; Embedding &gt; Port n</b>



**SCPI.CALCulate(*Ch*).FSIMulator.SENDEd.ZCONversion.PORT(*Pt*).Z0.R**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Port <i>Pt</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Pt</i> : port number 1–2 (see Table 2 on page 21)
<i>Description</i>	The value of the impedance for port impedance conversion function.
<i>Range</i>	from 1e–6 to 1e6
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	50
<i>Unit</i>	Ω (Ohm)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).FSIMulator.SENDEd.ZCONversion.PORT( <i>Pt</i> ).Z0.R <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).FSIMulator.SENDEd.ZCONversion.PORT( <i>Pt</i> ).Z0.R = 75
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Fixture Simulator &gt; Port Z Conversion &gt; Port n Z0</b>

**SCPI.CALCulate(*Ch*).FSIMulator.SENDEd.ZCONversion.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the port impedance conversion function.
<i>Allowable Values</i>	True: Port Z conversion function ON False: Port Z conversion function OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <code>app.SCPI.CALCulate(<i>Ch</i>).FSIMulator.SENDEd.ZCONversion.STATe</code> <code>app.SCPI.CALCulate(<i>Ch</i>).FSIMulator.SENDEd.ZCONversion.STATe = True</code>
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Fixture Simulator &gt; Port Z Conversion &gt; Port Z Conversion</b>

**SCPI.CALCulate(*Ch*).PARAmeter.COUNT**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Long
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The number of traces in the channel.
<i>Range</i>	from 1 to 8
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	1
<i>Syntax</i>	Dim <i>TraceNum</i> As Long <i>TraceNum</i> = app.SCPI.CALCulate( <i>Ch</i> ).PARAmeter.COUNT app.SCPI.CALCulate( <i>Ch</i> ).PARAmeter.COUNT = 2
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).PARAmeter(*Tr*).DEFine**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Trace <i>Tr</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Tr</i> : trace number 1–8 (see Table 3 on page 29)
<i>Description</i>	The measurement parameter of the trace.
<i>Allowable Values</i>	"S11" : S11 parameter "S21" : S21 parameter
<i>Out of Range</i>	An error occurs. Error code: 208.
<i>Preset Value</i>	Depends on the trace number. Tr 1, Tr 3, Tr 5, Tr 7: "S11" Tr 2, Tr 4, Tr 6, Tr 8: "S21"
<i>Syntax</i>	Dim <i>Meas</i> As String <i>Meas</i> = app.SCPI.CALCulate( <i>Ch</i> ).PARAmeter( <i>Tr</i> ).DEFine app.SCPI.CALCulate( <i>Ch</i> ).PARAmeter( <i>Tr</i> ).DEFine = "S11"
<i>Equivalent Softkeys</i>	<b>Response &gt; Measurement &gt; S11   S21</b>

**Table 3. *Tr*: Trace Number**

<i>Data Type</i>	Long
<i>Description</i>	Trace number
<i>Range</i>	from 1 to 8
<i>Out of Range</i>	An error occurs. Error code: 202.
<i>Notes</i>	If the trace number is not specified, it is taken as equal to 1.

**SCPI.CALCulate(*Ch*).PARAmeter(*Tr*).SELEct**

<i>Object Type</i>	Method
<i>Target</i>	Trace <i>Tr</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Tr</i> : trace number 1–8 (see Table 3 on page 29)
<i>Description</i>	Sets the active channel and trace.
<i>Syntax</i>	<code>app.SCPI.CALCulate(<i>Ch</i>).PARAmeter(<i>Tr</i>).SELEct</code>
<i>Notes</i>	If the channel number is greater than the number of the channels displayed, an error occurs and the command is ignored. If the trace number is greater than the number of the traces displayed in the channel, an error occurs and the command is ignored.
<i>Equivalent Softkeys</i>	<b>Channel &gt; Active Channel</b> <b>Trace &gt; Active Trace</b>

**SCPI.CALCulate(*Ch*).SElected.CONVersion.FUNcTion**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The S–parameter conversion function type.
<i>Range</i>	"IMPedance" : Reflection or Transmission equivalent impedance according to the trace measurement parameter (S11 or S21) "ADMittance" : Reflection or Transmission equivalent admittance according to the trace measurement parameter (S11 or S21) "INVersion" : Inverse S–parameter "CONJugation" : S–parameter conjugate
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	An error occurs. Error code: 217.
<i>Preset Value</i>	"IMP"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.CONVersion.FUNcTion app.SCPI.CALCulate( <i>Ch</i> ).SElected.CONVersion.FUNcTion = "INV"
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Conversion &gt; Function &gt; Impedance Z   AdmittanceY   Inverse 1/S   Conjugation</b>

**SCPI.CALCulate(*Ch*).SElected.CONVersion.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the S-parameter conversion function.
<i>Allowable Values</i>	True: S-parameter conversion function ON False: S-parameter conversion function OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.CONVersion.STATe <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.CONVersion.STATe = True
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Conversion &gt; Conversion</b>



**SCPI.CALCulate(*Ch*).SElected.CORRection.EDElay.TIME**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The value of the electrical delay.
<i>Range</i>	from –10 to 10
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	0
<i>Unit</i>	s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.CORRection.EDElay.TIME <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.CORRection.EDElay.TIME = 1e–9
<i>Equivalent Softkeys</i>	<b>Scale &gt; Electrical Delay</b>

**SCPI.CALCulate(*Ch*).SElected.CORRection.OFFSet.PHASE**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The value of the phase offset.
<i>Range</i>	from –360 to 360
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	0
<i>Unit</i>	° (degree)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.CORRection.OFFSet.PHASE app.SCPI.CALCulate( <i>Ch</i> ).SElected.CORRection.OFFSet.PHASE = 360
<i>Equivalent Softkeys</i>	<b>Scale &gt; Phase Offset</b>

**SCPI.CALCulate(*Ch*).SElected.DATA.FDATA**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	<p>The formatted data array. The array elements contain measurements in the current format, for example, in logarithmic magnitude format (Log Mag). Also, see section “Measurement Data Arrays” on page 19.</p> <p>The array size is 2N, where N is the number of measurement points.</p> <p>For the <i>n</i>–th point, where <i>n</i> from 1 to N:</p> <p style="padding-left: 40px;"><i>Data</i>(2<i>n</i>–2) real number in rectangular format, real part in polar and Smith chart formats;</p> <p style="padding-left: 40px;"><i>Data</i>(2<i>n</i>–1) 0 in rectangular format, imaginary part in polar and Smith chart formats.</p>
<i>Syntax</i>	<p>Dim <i>Data</i> As Variant</p> <p><i>Data</i> = <i>app</i>.SCPI.CALCulate(<i>Ch</i>).SElected.DATA.FDATA</p>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.DATA.FMEMory**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	<p>The formatted memory array. The array elements contain saved measurements in the current format, for example, in logarithmic magnitude format (Log Mag). Also, see section “Measurement Data Arrays”.</p> <p>The array size is 2N, where N is the number of measurement points.</p> <p>For the n–th point, where n from 1 to N:</p> <p style="padding-left: 40px;"><i>Data</i>(2<i>n</i>–2) real number in rectangular format, real part in polar and Smith chart formats;</p> <p style="padding-left: 40px;"><i>Data</i>(2<i>n</i>–1) 0 in rectangular format, imaginary part in polar and Smith chart formats.</p>
<i>Syntax</i>	<p>Dim <i>Data</i> As Variant</p> <p><i>Data</i> = <i>app</i>.SCPI.CALCulate(<i>Ch</i>).SElected.DATA.FMEMory</p>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.DATA.SDATA**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The corrected data array. The corrected measurements are complex numbers. Also, see section “Measurement Data Arrays” on page 19. The array size is 2N, where N is the number of measurement points. For the n–th point, where n from 1 to N: <i>Data</i> (2n–2) the real part of corrected measurement; <i>Data</i> (2n–1) the imaginary part of corrected measurement.
<i>Syntax</i>	Dim <i>Data</i> As Variant <i>Data</i> = app.SCPICALCulate( <i>Ch</i> ).SElected.DATA.SDATA
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.DATA.SMEMory**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	<p>The corrected memory array. The corrected measurements are complex numbers. Also, see section “Measurement Data Arrays” on page 19.</p> <p>The array size is 2N, where N is the number of measurement points.</p> <p>For the n–th point, where n from 1 to N:</p> <p style="padding-left: 40px;"><i>Data</i>(2<i>n</i>–2) the real part of corrected measurement memory;</p> <p style="padding-left: 40px;"><i>Data</i>(2<i>n</i>–1) the imaginary part of corrected measurement memory.</p>
<i>Syntax</i>	<p>Dim <i>Data</i> As Variant</p> <p><i>Data</i> = app.SCPi.CALCulate(<i>Ch</i>).SElected.DATA.SMEMory</p>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.FILTer.GATE.TIME.CENTer**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The gate center value of the gating function.
<i>Range</i>	Varies depending on the frequency span and the number of points.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	0
<i>Unit</i>	s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.FILTer.GATE.TIME.CENTer <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.FILTer.GATE.TIME.CENTer = 1e–8
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Gating &gt; Center</b>

**SCPI.CALCulate(*Ch*).SElected.FILTer.GATE.TIME.SHAPe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The gate shape of the gating function.
<i>Range</i>	"MAXimum" : Maximum shape "WIDE" : Wide shape "NORMal" : Normal shape "MINimum" : Minimum shape
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	An error occurs. Error code: 218.
<i>Preset Value</i>	"NORM"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.FILTer.GATE.TIME.SHAPe app.SCPI.CALCulate( <i>Ch</i> ).SElected.FILTer.GATE.TIME.SHAPe = "MAX"
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Gating &gt; Shape &gt; Maximum   Wide   Normal   Minimum</b>



**SCPI.CALCulate(*Ch*).SElected.FILTer.GATE.TIME.SPAN**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The gate span value of the gating function.
<i>Range</i>	Varies depending on the frequency span and the number of points.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	2e–8
<i>Unit</i>	s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.FILTer.GATE.TIME.SPAN app.SCPI.CALCulate( <i>Ch</i> ).SElected.FILTer.GATE.TIME.SPAN = 1e–8
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Gating &gt; Span</b>

**SCPI.CALCulate(*Ch*).SElected.FILTer.GATE.TIME.START**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The gate start value of the gating function.
<i>Range</i>	Varies depending on the frequency span and the number of points.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	–1e–8
<i>Unit</i>	s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.FILTer.GATE.TIME.START app.SCPI.CALCulate( <i>Ch</i> ).SElected.FILTer.GATE.TIME.START = 1e–7
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Gating &gt; Start</b>

**SCPI.CALCulate(*Ch*).SElected.FILTer.GATE.TIME.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the gating function.
<i>Allowable Values</i>	True: Gating function ON False: Gating function OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.FILTer.GATE.TIME.STATe app.SCPI.CALCulate( <i>Ch</i> ).SElected.FILTer.GATE.TIME.STATe = <i>Status</i>
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Gating &gt; Gating</b>

**SCPI.CALCulate(*Ch*).SElected.FILTer.GATE.TIME.STOP**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The gate stop value of the gating function.
<i>Range</i>	Varies depending on the frequency span and the number of points.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	1e–8
<i>Unit</i>	s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.FILTer.GATE.TIME.STOP app.SCPI.CALCulate( <i>Ch</i> ).SElected.FILTer.GATE.TIME.STOP = 1e–7
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Gating &gt; Stop</b>

**SCPI.CALCulate(*Ch*).SELEcted.FILTer.GATE.TIME.TYPE**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The gate type of the gating function.
<i>Range</i>	"BPASs" : Bandpass type "NOTCh" : Notch type
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	An error occurs. Error code: 219.
<i>Preset Value</i>	"BPAS"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.CALCulate( <i>Ch</i> ).SELEcted.FILTer.GATE.TIME.TYPE app.SCPI.CALCulate( <i>Ch</i> ).SELEcted.FILTer.GATE.TIME.TYPE = "NOTC"
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Gating &gt; Type</b>

**SCPI.CALCulate(*Ch*).SElected.FORMAT**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Data format.
<i>Range</i>	<p>"MLOGarithmic" : Logarithmic magnitude  "PHASe" : Phase  "GDELay" : Group delay time  "SLINear" : Smith chart format (Lin)  "SLOGarithmic" : Smith chart format (Log)  "SCOMplex" : Smith chart format (Real/Imag)  "SMITH" : Smith chart format (R + jX)  "SADMittance" : Smith chart format (G + jB)  "PLINear" : Polar format (Lin)  "PLOGarithmic" : Polar format (Log)  "POLar" : Polar format (Real/Imag)  "MLINear" : Linear magnitude  "SWR" : Voltage standing wave ratio  "REAL" : Real part  "IMAGinary" : Imaginary part  "UPHase" : Expanded phase</p>
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	An error occurs. Error code: 209.
<i>Preset Value</i>	"MLOG"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.FORMAT app.SCPI.CALCulate( <i>Ch</i> ).SElected.FORMAT = "PHAS"
<i>Equivalent Softkeys</i>	<b>Response &gt; Format &gt; Log Mag   SWR   Phase   Expand Phase   Group Delay   Lin Mag   Real   Imag</b> <b>Response &gt; Format &gt; Smith &gt; Log / Phase   Lin / Phase   Real / Imag   R+jX   G+jB</b> <b>Response &gt; Format &gt; Polar &gt; Log / Phase   Ling / Phase   Real / Imag</b>

**SCPI.CALCulate(*Ch*).SELEcted.FUNcTion.DATA**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	<p>The data array of analysis executed by the SCPI.CALCulate(<i>Ch</i>).SELEcted.FUNcTion.EXECute method.</p> <p>The array size is 2N, where N is the number of points defined by the SCPI.CALCulate(<i>Ch</i>).SELEcted.FUNcTion.POINts property.</p> <p>For the n–th point, where n from 1 to N:</p> <p style="padding-left: 40px;"><i>Data</i>(2n–2) the response value in the n–th measurement point;</p> <p style="padding-left: 40px;"><i>Data</i>(2n–1) the stimulus value in the n–th measurement point. Always set to 0 for the analysis of mean value, standard deviation, and peak–to–peak value.</p>
<i>Syntax</i>	<p>Dim <i>Data</i> As Variant</p> <p><i>Data</i> = <i>app</i>.SCPI.CALCulate(<i>Ch</i>).SELEcted.FUNcTion.DATA</p>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SELEcted.FUNcTion.DOMain.COUPle**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	All traces of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF coupling state of the analysis range for the SCPI.CALCulate( <i>Ch</i> ).SELEcted.FUNcTion.EXECute method.
<i>Allowable Values</i>	True: Coupling state ON False: Coupling state OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SELEcted.FUNcTion.DOMain.COUPle <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SELEcted.FUNcTion.DOMain.COUPle = <i>Status</i>
<i>Equivalent Softkeys</i>	<b>None</b>



**SCPI.CALCulate(*Ch*).SElected.FUNcTion.DOMain.START**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	All traces of channel <i>Ch</i> (if the coupling is set to OFF by the SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.DOMain.COUPle property), the active trace of channel <i>Ch</i> (if otherwise), <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The start value of the analysis range set by the SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.EXECute method.
<i>Range</i>	From the stimulus current start value to the stimulus current stop value.
<i>Out of Range</i>	No limitation
<i>Preset Value</i>	0
<i>Unit</i>	Hz (Hertz)   s (second)   dBm (decibels above 1 milliwatt)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.DOMain.START app.SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.DOMain.START = 1e9
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.FUNcTion.DOMain.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	All traces of channel <i>Ch</i> (if the coupling is set to OFF by the SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.DOMain.COUPle property), the active trace of channel <i>Ch</i> (if otherwise), <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the arbitrary range when executing the analysis by the SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.EXECute method.
<i>Allowable Values</i>	True: Arbitrary range ON False: Arbitrary range OFF (entire sweep range)
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.DOMain.STATe <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.DOMain.STATe = <i>True</i>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.FUNcTion.DOMain.STOP**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	All traces of channel <i>Ch</i> (if the coupling is set to OFF by the SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.DOMain.COUPle property), the active trace of channel <i>Ch</i> (if otherwise), <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The stop value of the analysis range set by the SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.EXECute method.
<i>Range</i>	From the stimulus current start value to the stimulus current stop value.
<i>Out of Range</i>	No limitation
<i>Preset Value</i>	0
<i>Unit</i>	Hz (Hertz)   s (second)   dBm (decibels above 1 milliwatt)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.DOMain.STOP <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.DOMain.STOP = 2e9
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SELEcted.FUNcTion.EXECute**

<i>Object Type</i>	Method
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Executes the analysis specified with the SCPI.CALCulate( <i>Ch</i> ).SELEcted.FUNcTion.TYPE property. The analysis result can then be read out with the SCPI.CALCulate( <i>Ch</i> ).SELEcted.FUNcTion.DATA property.
<i>Syntax</i>	<i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SELEcted.FUNcTion.EXECute
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.FUNcTion.PEXCursion**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The lower limit for the peak excursion value when executing the peak search by the SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.EXECute method.
<i>Range</i>	Varies depending on the trace format.
<i>Out of Range</i>	No limitation
<i>Preset Value</i>	3
<i>Unit</i>	dB (decibel)   ° (degree)   s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.PEXCursion <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.PEXCursion = 1.5
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.FUNcTion.POINts**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Long
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The number of points (data pairs) of the analysis result by the SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.EXECute method. Always equal to 1, when the search is executed for the maximum, minimum, mean, standard deviation, peak, and peak-to-peak values. The actual number of points is read out, when the search is executed for all peaks or all targets.
<i>Syntax</i>	Dim <i>Value</i> As Long <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.POINts
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.FUNcTion.PPOLarity**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The polarity selection when performing the peak search by the SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.EXECute method.
<i>Range</i>	"POSitive" : Positive peaks "NEGative" : Negative peaks "BOTH" : Both positive peaks and negative peaks
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	The value is ignored.
<i>Preset Value</i>	"POS"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.PPOLarity app.SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.PPOLarity = "NEG"
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.FUNcTion.TARGet**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The target level when performing the search for the trace and the target level crosspoints by the SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.EXECute method.
<i>Range</i>	Varies depending on the trace format.
<i>Out of Range</i>	No limitation
<i>Preset Value</i>	0
<i>Unit</i>	dB (decibel)   ° (degree)   s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.TARGet <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.TARGet = –10
<i>Equivalent Softkeys</i>	<b>None</b>



**SCPI.CALCulate(*Ch*).SElected.FUNcTion.TTRansition**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The transition type selection when performing the search for the trace and the target level crosspoints by the SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.EXECute method.
<i>Range</i>	"POSitive" : Positive peaks "NEGative" : Negative peaks "BOTH" : Both positive peaks and negative peaks
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	The value is ignored.
<i>Preset Value</i>	"POS"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.TTRansition app.SCPI.CALCulate( <i>Ch</i> ).SElected.FUNcTion.TTRansition = "BOTH"
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SELEcted.FUNcTion.TYPE**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The selection of the type of analysis executed by the SCPI.CALCulate( <i>Ch</i> ).SELEcted.FUNcTion.EXECute method.
<i>Range</i>	<p>"PTPeak" : Peak-to-peak (difference between the maximum value and the minimum value)</p> <p>"STDEV" : Standard deviation</p> <p>"MEAN" : Mean value</p> <p>"MAXimum" : Maximum value</p> <p>"MINimum" : Minimum value</p> <p>"PEAK" : Search for the peak</p> <p>"APEak" : Search for all the peaks</p> <p>"ATARget" : Search for all targets</p>
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	The value is ignored.
<i>Preset Value</i>	"PTP"
<i>Syntax</i>	<p>Dim <i>Param</i> As String</p> <p><i>Param</i> = app.SCPI.CALCulate(<i>Ch</i>).SELEcted.FUNcTion.TYPE</p> <p>app.SCPI.CALCulate(<i>Ch</i>).SELEcted.FUNcTion.TYPE = "STDEV"</p>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.LIMit.DATA**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	<p>The data array, which is the limit line in the limit test function. The array size is <math>1 + 5N</math>, where <math>N</math> is the number of measuring points.</p> <p>For the <math>n</math>-th point, where <math>n</math> from 1 to <math>N</math>:</p> <p><i>Data</i>(0) The number of limit line segments <math>N</math> is from 0 to 100. Setting 0 clears the limit line;</p> <p><i>Data</i>(<math>5n-4</math>) type of the <math>n</math>-th limit line segment; 0: OFF 1: Upper limit 2: Lower limit</p> <p><i>Data</i>(<math>5n-3</math>) the stimulus value in the start point of the <math>n</math>-th segment;</p> <p><i>Data</i>(<math>5n-2</math>) the stimulus value in the end point of the <math>n</math>-th segment;</p> <p><i>Data</i>(<math>5n-1</math>) the response value in the start point of the <math>n</math>-th segment;</p> <p><i>Data</i>(<math>5n-0</math>) the response value in the end point of the <math>n</math>-th segment.</p>
<i>Notes</i>	If the array size is not $1 + 5N$ , where $N$ is <i>Data</i> (0), an error occurs (error code 214). If <i>Data</i> ( $5n - 4$ ) is less than 0 or more than 2, an error occurs (error code 214). When <i>Data</i> ( $5n-3$ ), <i>Data</i> ( $5n-2$ ), <i>Data</i> ( $5n-1$ ) and <i>Data</i> ( $5n-0$ ) elements are out of allowable range, the value is set to the limit, which is closer to the specified value.
<i>Syntax</i>	<p>Dim <i>Data</i> As Variant</p> <p><i>Data</i> = app.SCPI.CALCulate(<i>Ch</i>).SElected.LIMit.DATA</p> <p>app.SCPI.CALCulate(<i>Ch</i>).SElected.LIMit.DATA = Array(1,2,800,900,-10,-10)</p>
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Limit Test &gt; Edit Limit Line</b>

**SCPI.CALCulate(*Ch*).SElected.LIMit.DISPlay.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the limit line display of the limit test function.
<i>Allowable Values</i>	True: Limit line display ON False: Limit line display OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.LIMit.DISPlay.STATe <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.LIMit.DISPlay.STATe = True
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Limit Test &gt; Limit Line</b>

**SCPI.CALCulate(*Ch*).SElected.LIMit.FAIL**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Boolean
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The limit test result.
<i>Allowable Values</i>	True: Fail False: Pass
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.LIMit.FAIL
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.LIMit.OFFSet.AMPLitude**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The value of the limit line offset along Y-axis.
<i>Range</i>	Varies depending on the trace format.
<i>Out of Range</i>	No limitation
<i>Preset Value</i>	0
<i>Unit</i>	dB (decibel)   ° (degree)   s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.LIMit.OFFSet.AMPLitude <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.LIMit.OFFSet.AMPLitude = –10
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Limit Test &gt; Limit Line Offsets &gt; Response Offset</b>

**SCPI.CALCulate(*Ch*).SElected.LIMit.OFFSet.STIMulus**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The value of the limit line offset along X-axis.
<i>Range</i>	From the stimulus current start value to the stimulus current stop value.
<i>Out of Range</i>	No limitation
<i>Preset Value</i>	0
<i>Unit</i>	Hz (Hertz)   s (second)   dBm (decibels above 1 milliwatt)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.LIMit.OFFSet.STIMulus <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.LIMit.OFFSet.STIMulus = 1e6
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Limit Test &gt; Limit Lines Offsets &gt; Stimulus Offset</b>

**SCPI.CALCulate(*Ch*).SElected.LIMit.REPort.ALL**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	<p>The data array, which is the limit test results. The array size is 4N, where N is the number of measurement points.</p> <p>For the n–th point, where n from 1 to N:</p> <p style="padding-left: 40px;"><i>Data</i>(4n–3) the stimulus value in the n–th point</p> <p style="padding-left: 40px;"><i>Data</i>(4n–2) the limit test result in the n–th point –1: No limit 0: Fail 1: Pass</p> <p style="padding-left: 40px;"><i>Data</i>(4n–1) the upper limit value in the n–th point (0 – if there is no limit)</p> <p style="padding-left: 40px;"><i>Data</i>(4n–0) the lower limit value in the n–th point (0 – if there is no limit)</p>
<i>Syntax</i>	<p>Dim <i>Data</i> As Variant</p> <p><i>Data</i> = app.SCPi.CALCulate(<i>Ch</i>).SElected.LIMit.REPort.ALL</p>
<i>Equivalent Softkeys</i>	<b>None</b>



**SCPI.CALCulate(*Ch*).SElected.LIMit.REPort.DATA**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The data array, which is the stimulus values at all the measurement points that failed the limit test. The array size is defined by the SCPI.CALCulate( <i>Ch</i> ).SElected.LIMit.REPort.POINts property.
<i>Syntax</i>	Dim <i>Data</i> As Variant <i>Data</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.LIMit.REPort.DATA
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.LIMit.REPort.POINts**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Long
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The number of the measurement points that failed the limit test. The array of stimulus values of the points can be read out by the SCPI.CALCulate( <i>Ch</i> ).SElected.LIMit.REPort.DATA property.
<i>Syntax</i>	Dim <i>Cnt</i> As Long <i>Cnt</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.LIMit.REPort.POINts
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.LIMit.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the limit test function.
<i>Allowable Values</i>	True: Limit test function ON False: Limit test function OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.LIMit.STATe <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.LIMit.STATe = True
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Limit Test &gt; Limit Test</b>

**SCPI.CALCulate(*Ch*).SELEcted.MARKer(*Mk*).ACTivate**

<i>Object Type</i>	Method
<i>Target</i>	Marker <i>Mk</i> of the active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Mk</i> : marker number 1–15, or reference marker number 16 (see Table 4 on page 68)
<i>Description</i>	Sets the active marker. If a marker is OFF this function will turn it ON. Turning ON a marker with the number from 1 to 15 will turn ON all the markers of smaller numbers. Turning ON the reference marker with number 16 does not turn ON the markers with the numbers from 1 to 15, but switches these markers to the relative measurement mode.
<i>Syntax</i>	<i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).ACTivate
<i>Equivalent Softkeys</i>	<b>Markers &gt; Active Marker &gt; Marker n</b> <b>Markers &gt; Reference Marker</b>

**Table 4. *Mk*: Marker Number**

<i>Data Type</i>	Long
<i>Description</i>	Marker number. Numbers from 1 to 15 are for regular markers, number 16 is for the reference marker.
<i>Range</i>	from 1 to 16
<i>Out of Range</i>	An error occurs. Error code: 203.
<i>Notes</i>	If the marker number is not specified, it is taken as equal to 1.

**SCPI.CALCulate(*Ch*).SElected.MARKer(*Mk*).BWIDth.DATA**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	Marker <i>Mk</i> of the active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Mk</i> : marker number 1–15, or reference marker number 16 (see Table 4 on page 68)
<i>Description</i>	The bandwidth search result. The bandwidth search can be performed relatively to the marker <i>Mk</i> , or relatively to the absolute maximum value of the trace (in this case the marker number is ignored), what is set by the SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer( <i>Mk</i> ).BWIDth.REFerence property. The array contains 4 elements: <i>Data</i> (0)      Bandwidth; <i>Data</i> (1)      Center frequency; <i>Data</i> (2)      Q value; <i>Data</i> (3)      Loss.
<i>Notes</i>	If the bandwidth search is impossible, all the read out values are 0. If the search is performed relatively to a maker, which is OFF, an error occurs (error code 204).
<i>Syntax</i>	Dim <i>Data</i> As Variant <i>Data</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer( <i>Mk</i> ).BWIDth.DATA
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.MARKer(*Mk*).BWIDth.REFerence**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The selection of the reference point for the bandwidth search function: reference marker or absolute maximum value of the trace.
<i>Range</i>	"MARKer" : Bandwidth search relative to the reference marker "MAXimum" : Bandwidth search relative to the absolute maximum of the trace
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	The value is ignored.
<i>Preset Value</i>	"MAX"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).BWIDth.REFerence app.SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).BWIDth.REFerence = "MARK"
<i>Equivalent Softkeys</i>	<b>Markers &gt; Marker Math &gt; Bandwidth Search &gt; Search Ref To</b>

**SCPI.CALCulate(*Ch*).SElected.MARKer(*Mk*).BWIDth.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the bandwidth search function.
<i>Allowable Values</i>	True: Bandwidth search function ON False: Bandwidth search function OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).BWIDth.STATe <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).BWIDth.STATe = True
<i>Equivalent Softkeys</i>	<b>Markers &gt; Marker Math &gt; Bandwidth Search &gt; Bandwidth Search</b>

**SCPI.CALCulate(*Ch*).SElected.MARKer(*Mk*).BWIDth.THReshold**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The bandwidth definition value.
<i>Range</i>	Varies depending on the trace format.
<i>Out of Range</i>	No limitation
<i>Preset Value</i>	–3
<i>Unit</i>	dB (decibel)   ° (degree)   s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).BWIDth.THReshold <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).BWIDth.THReshold = –6.0
<i>Equivalent Softkeys</i>	<b>Markers &gt; Marker Math &gt; Bandwidth Search &gt; Bandwidth Value</b>



**SCPI.CALCulate(*Ch*).SElected.MARKer(*Mk*).BWIDth.TYPE**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The type of the bandwidth search function.
<i>Range</i>	"BPASs" : Bandpass "NOTCh" : Notch
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	The value is ignored.
<i>Preset Value</i>	"BPAS"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).BWIDth.TYPE app.SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).BWIDth.TYPE = "NOTC"
<i>Equivalent Softkeys</i>	<b>Markers &gt; Marker Math &gt; Bandwidth Search &gt; Type</b>

**SCPI.CALCulate(*Ch*).SElected.MARKer(*Mk*).COUPle**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	All traces of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the marker coupling function.
<i>Allowable Values</i>	True: Marker coupling ON False: Marker coupling OFF
<i>Preset Value</i>	True
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).COUPle <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).COUPle = false
<i>Equivalent Softkeys</i>	<b>Markers &gt; Properties &gt; Marker Couple</b>

## SCPI.CALCulate(*Ch*).SElected.MARKer(*Mk*).FUNction.DOMain.STARt

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	All traces of channel <i>Ch</i> (if the marker search range coupling is set to OFF by the SCPI.CALCulate( <i>Ch</i> ).SElected.FUNction.DOMain.COUPle property), the active trace of channel <i>Ch</i> (if otherwise), <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The start value of the marker search range.
<i>Range</i>	From the stimulus current start value to the stimulus current stop value.
<i>Out of Range</i>	No limitation
<i>Preset Value</i>	3e5
<i>Unit</i>	Hz (Hertz)   s (second)   dBm (decibels above 1 milliwatt)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).FUNction.DOMain.STARt app.SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).FUNction.DOMain.STARt = 1e6
<i>Equivalent Softkeys</i>	<b>Markers &gt; Marker Search &gt; Search Start</b>

## SCPI.CALCulate(*Ch*).SElected.MARKer(*Mk*).FUNction.DOMain.STATe

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	All traces of channel <i>Ch</i> (if the marker search range coupling is set to OFF by the SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer.FUNction.DOMain.COUPLE property), the active trace of channel <i>Ch</i> (if otherwise), <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the arbitrary range when executing the marker search.
<i>Allowable Values</i>	True: Marker search range ON False: Marker search range OFF (entire sweep range)
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).FUNction.DOMain.STATe app.SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).FUNction.DOMain.STATe = True
<i>Equivalent Softkeys</i>	<b>Markers &gt; Marker Search &gt; Search Range</b>

## SCPI.CALCulate(*Ch*).SElected.MARKer(*Mk*).FUNction.DOMain.STOP

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	All traces of channel <i>Ch</i> (if the marker search range coupling is set to OFF by the SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer.FUNction.DOMain.COUPLE property), the active trace of channel <i>Ch</i> (if otherwise), <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The stop value of the marker search range.
<i>Range</i>	From the stimulus current start value to the stimulus current stop value.
<i>Out of Range</i>	No limitation
<i>Preset Value</i>	1.5e9
<i>Unit</i>	Hz (Hertz)   s (second)   dBm (decibels above 1 milliwatt)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).FUNction.DOMain.STOP <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).FUNction.DOMain.STOP = 1e6
<i>Equivalent Softkeys</i>	<b>Markers &gt; Marker Search &gt; Search Stop</b>

**SCPI.CALCulate(*Ch*).SELEcted.MARKer(*Mk*).FUNction.EXECute**

<i>Object Type</i>	Method
<i>Target</i>	Marker <i>Mk</i> of the active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Mk</i> : marker number 1–15, or reference marker number 16 (see Table 4 on page 68)
<i>Description</i>	Executes the marker search according to the specified criterion. The type of the marker search is set by the SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNction.TYPE property.
<i>Syntax</i>	<i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNction.EXECute
<i>Equivalent Softkeys</i>	<b>Markers &gt; Marker Search &gt; Maximum   Minimum</b> <b>Markers &gt; Marker Search &gt; Peak &gt; Search Peak   Search Max Peak   Search Peak Left   Search Peak Right</b> <b>Markers &gt; Marker Search &gt; Target &gt; Search Target   Search Target Left   Search Target Right</b>

**SCPI.CALCulate(*Ch*).SElected.MARKer(*Mk*).FUNction.PEXCursion**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Marker <i>Mk</i> of the active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Mk</i> : marker number 1–15, or reference marker number 16 (see Table 4 on page 68)
<i>Description</i>	The peak excursion value, when the marker search for peak is performed by the SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer( <i>Mk</i> ).FUNction.EXECute method.
<i>Range</i>	Varies depending on the trace format.
<i>Out of Range</i>	No limitation
<i>Preset Value</i>	1
<i>Unit</i>	dB (decibel)   ° (degree)   s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer( <i>Mk</i> ).FUNction.PEXCursion <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer( <i>Mk</i> ).FUNction.PEXCursion = 3.0
<i>Equivalent Softkeys</i>	<b>Markers &gt; Marker Search &gt; Peak &gt; Peak Excursion</b>

**SCPI.CALCulate(*Ch*).SELEcted.MARKer(*Mk*).FUNction.PPOLarity**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Marker <i>Mk</i> of the active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Mk</i> : marker number 1–15, or reference marker number 16 (see Table 4 on page 68)
<i>Description</i>	The peak polarity selection, when the marker search for peak is performed by the SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNction.EXECute method.
<i>Range</i>	"POSitive" : Positive polarity "NEGative" : Negative polarity "BOTH" : Both positive polarity and negative polarity
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	The value is ignored.
<i>Preset Value</i>	"POS"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNction.PPOLarity app.SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNction.PPOLarity = "neg"
<i>Equivalent Softkeys</i>	<b>Markers &gt; Marker Search &gt; Peak &gt; Peak Polarity &gt; Positive   Negative   Both</b>



**SCPI.CALCulate(*Ch*).SElected.MARKer(*Mk*).FUNction.TARGet**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Marker <i>Mk</i> of the active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Mk</i> : marker number 1–15, or reference marker number 16 (see Table 4 on page 68)
<i>Description</i>	The target value, when the marker search for target is performed by the SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer( <i>Mk</i> ).FUNction.EXECute method.
<i>Range</i>	Varies depending on the trace format.
<i>Out of Range</i>	No limitation
<i>Preset Value</i>	0
<i>Unit</i>	dB (decibel)   ° (degree)   s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer( <i>Mk</i> ).FUNction.TARGet <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer( <i>Mk</i> ).FUNction.TARGet = -10
<i>Equivalent Softkeys</i>	<b>Markers &gt; Marker Search &gt; Target &gt; Target Value</b>

**SCPI.CALCulate(*Ch*).SElected.MARKer(*Mk*).FUNction.TRACKing**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Marker <i>Mk</i> of the active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Mk</i> : marker number 1–15, or reference marker number 16 (see Table 4 on page 68)
<i>Description</i>	The ON/OFF state of the marker search tracking function.
<i>Allowable Values</i>	True: Marker search tracking ON False: Marker search tracking OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = app.SCPi.CALCulate( <i>Ch</i> ).SElected.MARKer( <i>Mk</i> ).FUNction.TRACKing app.SCPi.CALCulate( <i>Ch</i> ).SElected.MARKer( <i>Mk</i> ).FUNction.TRACKing = True
<i>Equivalent Softkeys</i>	<b>Markers &gt; Marker Search &gt; Tracking</b>

**SCPI.CALCulate(*Ch*).SElected.MARKer(*Mk*).FUNctio.n.TTRansition**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Marker <i>Mk</i> of the active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Mk</i> : marker number 1–15, or reference marker number 16 (see Table 4 on page 68)
<i>Description</i>	The selection of the type of the target transition, when the marker search for transition is performed by the SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer( <i>Mk</i> ).FUNctio.n.EXECute method.
<i>Range</i>	"POSitive" : Positive target transition "NEGative" : Negative target transition "BOTH" : Both positive target transition and negative target transition
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	The value is ignored.
<i>Preset Value</i>	"POS"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer( <i>Mk</i> ).FUNctio.n.TTRansition app.SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer( <i>Mk</i> ).FUNctio.n.TTRansition = "NEG"
<i>Equivalent Softkeys</i>	<b>Marker &gt; Marker Search &gt; Target &gt; Target Transition</b>

**SCPI.CALCulate(*Ch*).SELEcted.MARKer(*Mk*).FUNction.TYPE**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Marker <i>Mk</i> of the active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Mk</i> : marker number 1–15, or reference marker number 16 (see Table 4 on page 68)
<i>Description</i>	The selection of the type of the marker search, which is performed by the SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNction.EXECute method.
<i>Range</i>	"MAXimum" : Maximum value search "MINimum" : Minimum value search "PEAK" : Peak search "LPEak" : Peak search to the left from the marker "RPEak" : Peak search to the right from the marker "TARGet" : Target search "LTARget" : Target search to the left from the marker "RTARget" : Target search to the right from the marker
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	The value is ignored.
<i>Preset Value</i>	"MAX"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNction.TYPE app.SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).FUNction.TYPE = "MIN"
<i>Equivalent Softkeys</i>	<b>Markers &gt; Marker Search &gt; Maximum   Minimum</b>  <b>Markers &gt; Marker Search &gt; Peak &gt; Search Peak   Search Max Peak   Search Peak Left   Search Peak Right</b>  <b>Markers &gt; Marker Search &gt; Target &gt; Search Target   Search Target Left   Search Target Right</b>

**SCPI.CALCulate(*Ch*).SElected.MARKer(*Mk*).REFerence.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) ( <i>Mk</i> – arbitrary number 1 to 16)
<i>Description</i>	The ON/OFF state of the reference marker. When the reference marker is turned ON, all the values of the other markers turn to relative values.
<i>Allowable Values</i>	True: Reference marker ON False: Reference marker OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).REFerence.STATe <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).REFerence.STATe = True
<i>Equivalent Softkeys</i>	<b>Markers &gt; Reference Marker</b>

**SCPI.CALCulate(*Ch*).SELEcted.MARKer(*Mk*).SET\_**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	String
<i>Target</i>	Marker <i>Mk</i> of the active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Mk</i> : marker number 1–15, or reference marker number 16 (see Table 4 on page 68)
<i>Description</i>	The setting of the value of the specified item to the value of the position of the marker.
<i>Range</i>	"START" : Sweep start value set to the stimulus value of the marker position. "STOP" : Sweep stop value set to the stimulus value of the marker position. "CENTER" : Sweep center value set to the stimulus value of the marker position. "RLEVel" : Reference value set to the response value of the marker position.
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	The value is ignored.
<i>Syntax</i>	<code>app.SCPI.CALCulate(<i>Ch</i>).SELEcted.MARKer(<i>Mk</i>).SET_ = "STOP"</code>
<i>Equivalent Softkeys</i>	<b>Markers &gt; Marker Functions &gt; Marker→Start   Marker→Stop   Marker →Center   Marker→Ref Value</b>

**SCPI.CALCulate(*Ch*).SElected.MARKer(*Mk*).STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Marker <i>Mk</i> of the active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Mk</i> : marker number 1–15, or reference marker number 16 (see Table 4 on page 68)
<i>Description</i>	The ON/OFF state of a marker. Turning ON a marker with the number from 1 to 15 will turn ON all the markers of smaller numbers. Turning OFF a marker with the number from 1 to 15 will turn OFF all the markers of greater numbers (except for the reference marker). Turning ON/OFF the reference marker with number 16 does not turn ON/OFF the markers with the numbers from 1 to 15, but switches these markers to the relative measurement mode.
<i>Allowable Values</i>	True: Marker ON False: Marker OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer( <i>Mk</i> ).STATe <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer( <i>Mk</i> ).STATe = True
<i>Equivalent Softkeys</i>	<b>Markers &gt; Add Marker   Remove Marker</b> <b>Markers &gt; Reference Marker</b>

**SCPI.CALCulate(*Ch*).SELEcted.MARKer(*Mk*).X**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Marker <i>Mk</i> of the active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Mk</i> : marker number 1–15, or reference marker number 16 (see Table 4 on page 68)
<i>Description</i>	The stimulus value of the marker.
<i>Range</i>	From the stimulus current start value to the stimulus current stop value.
<i>Out of Value</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	Stimulus center value
<i>Unit</i>	Hz (Hertz)   s (second)   dBm (decibels above 1 milliwatt)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).X <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SELEcted.MARKer( <i>Mk</i> ).X = 1e9
<i>Equivalent Softkeys</i>	<b>Markers &gt; Edit Stimulus</b>



**SCPI.CALCulate(*Ch*).SElected.MARKer(*Mk*).Y**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	Marker <i>Mk</i> of the active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Mk</i> : marker number 1–15, or reference marker number 16 (see Table 4 on page 68)
<i>Description</i>	The response value of the marker. If the reference marker is turned ON, the values of the markers from 1 to 15 are read out as relative values to the reference marker. The array includes 2 elements: <i>Data</i> (0) real number in rectangular format, real part in polar and Smith chart formats; <i>Data</i> (1) 0 in rectangular format, imaginary part in polar and Smith chart formats.
<i>Syntax</i>	Dim <i>Data</i> As Variant <i>Data</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer( <i>Mk</i> ).Y
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.MARKer(*Mk*).COUNT**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Long
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The number of the turned ON markers.
<i>Range</i>	from 0 to 16
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	0
<i>Syntax</i>	Dim <i>MarkerCnt</i> As Long <i>MarkerCnt</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).COUNT app.SCPI.CALCulate( <i>Ch</i> ).SElected.MARKer(1).COUNT = 5
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.MATH.FUNcTion**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The selection of the math operation between the measurement data and the memory trace data. The math result replaces the data trace. If the data trace is not saved, the command is ignored.
<i>Range</i>	"DIVide" : Division $Data / Mem$ . "MULTiply" : Multiplication $Data \times Mem$ . "ADD" : Addition $Data + Mem$ . "SUBTract" : Subtraction $Data - Mem$ . "NORMal" : No math
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	An error occurs. Error code 210.
<i>Preset Value</i>	"NORM"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.MATH.FUNcTion app.SCPI.CALCulate( <i>Ch</i> ).SElected.MATH.FUNcTion= "DIV"
<i>Equivalent Softkeys</i>	<b>Trace &gt; Data Math &gt; Data/Mem   Data*Mem   Data+Mem   Data–Mem   OFF</b>

**SCPI.CALCulate(*Ch*).SElected.MATH.MEMorize**

<i>Object Type</i>	Method
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Saves the measurement data to the memory trace. Automatically turns on the display of the memory trace.
<i>Syntax</i>	<code>app.SCPi.CALCulate(<i>Ch</i>).SElected.MATH.MEMorize</code>
<i>Equivalent Softkeys</i>	<b>Trace &gt; Memorize Data Trace</b>

**SCPI.CALCulate(*Ch*).SElected.MAXhold.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the Maximum Hold function.
<i>Allowable Values</i>	True: Maximum Hold ON False: Maximum Hold OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MAXhold.STATe <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MAXhold.STATe = True
<i>Equivalent Softkeys</i>	<b>Trace &gt; Max Hold</b>

**SCPI.CALCulate(*Ch*).SElected.MSTatistics.DATA**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	<p>The math statistics data array. The statistics function is applied either over the whole range (for all the trace), or within the range specified by the SCPI.CALCulate(<i>Ch</i>).SElected.MSTatistics.DOMain.STate property (the range limits are determined by two markers).</p> <p>The array includes 3 elements:</p> <p style="padding-left: 40px;"><i>Data</i>(0)      Mean value;</p> <p style="padding-left: 40px;"><i>Data</i>(1)      Standard deviation;</p> <p style="padding-left: 40px;"><i>Data</i>(2)      Peak-to-peak (difference between the maximum value and the minimum value).</p>
<i>Syntax</i>	<p>Dim <i>Data</i> As Variant</p> <p><i>Data</i> = app.SCPI.CALCulate(<i>Ch</i>).SElected.MSTatistics.DATA</p>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.MSTatistics.DOMain.MARKer.START**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Long
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The number of the marker, which specifies the start frequency of the math statistics range.
<i>Range</i>	from 1 to 16
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	1
<i>Syntax</i>	Dim <i>MkrNum</i> As Long <i>MkrNum</i> = app.SCPi.CALCulate( <i>Ch</i> ).SElected.MSTatistics.DOMain.MARKer.START app.SCPi.CALCulate( <i>Ch</i> ).SElected.MSTatistics.DOMain.MARKer.START = 3
<i>Equivalent Softkeys</i>	<b>Markers &gt; Marker Math &gt; Statistics &gt; Statistics Start</b>

**SCPI.CALCulate(*Ch*).SElected.MSTatistics.DOMain.MARKer.STOP**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Long
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The number of the marker, which specifies the stop frequency of the math statistics range.
<i>Range</i>	from 1 to 16
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	2
<i>Syntax</i>	Dim <i>Mkr</i> As Long <i>Mkr</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.MSTatistics.DOMain.MARKer.STOP app.SCPI.CALCulate( <i>Ch</i> ).SElected. MSTatistics.DOMain.MARKer.STOP = 4
<i>Equivalent Softkeys</i>	<b>Markers &gt; Marker Math &gt; Statistics &gt; Statistics Stop</b>



**SCPI.CALCulate(*Ch*).SElected.MStatistics.DOMain.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the math statistics range.
<i>Allowable Values</i>	True: Statistics range ON False: Statistics range OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected. MStatistics.DOMain.STATe app.SCPI.CALCulate( <i>Ch</i> ).SElected. MStatistics.DOMain.STATe = True
<i>Equivalent Softkeys</i>	<b>Markers &gt; Marker Math &gt; Statistics &gt; Statistics Range</b>

**SCPI.CALCulate(*Ch*).SElected.MStatistIcs.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the math statistics display.
<i>Allowable Values</i>	True: Statistics display ON False: Statistics display OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MStatistIcs.STATe <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.MStatistIcs.STATe = True
<i>Equivalent Sofitkeys</i>	<b>Markers &gt; Marker Math &gt; Statistics &gt; Statistics</b>

**SCPI.CALCulate(Ch).SElected.RLIMit.DATA**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	<p>The data array, which is the limit line for the ripple limit function. The array size is <math>1 + 4N</math>, where <math>N</math> is the number of limit line segments.</p> <p>For the <math>n</math>-th point, where <math>n</math> from 1 to <math>N</math>:</p> <p><i>Data</i>(0) the number of limit line segments <math>N</math> is the integer from 0 to 12. Setting 0 clears the limit line;</p> <p><i>Data</i>(<math>4n-3</math>) type of the <math>n</math>-th limit line segment; 0: Off 1: On</p> <p><i>Data</i>(<math>4n-2</math>) the stimulus value in the beginning point of the <math>n</math>-th segment;</p> <p><i>Data</i>(<math>4n-1</math>) the stimulus value in the end point of the <math>n</math>-th segment;</p> <p><i>Data</i>(<math>4n-0</math>) the ripple limit value of the <math>n</math>-th segment.</p>
<i>Notes</i>	If the array size is not $1 + 4N$ , where $N$ is <i>Data</i> (0), an error occurs (error code 214). If <i>Data</i> ( $4n - 3$ ) is less than 0 or more than 1, an error occurs (error code 214). When <i>Data</i> ( $4n-2$ ), <i>Data</i> ( $4n-1$ ), and <i>Data</i> ( $4n-0$ ) elements are out of allowable range, the value is set to the limit, which is closer to the specified value.
<i>Syntax</i>	<p>Dim <i>Data</i> As Variant</p> <p><i>Data</i> = app.SCPI.CALCulate(Ch).SElected.RLIMit.DATA</p> <p>app.SCPI.CALCulate(Ch).SElected.RLIMit.DATA = Array(1,1,800,900,10)</p>
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Ripple Limit &gt; Edit Ripple Limit</b>

**SCPI.CALCulate(*Ch*).SElected.RLIMit.DISPlay.LINE**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the ripple limit line display.
<i>Allowable Values</i>	True: Ripple limit line ON False: Ripple limit line OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.RLIMit.DISPlay.LINE <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.RLIMit.DISPlay.LINE = True
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Ripple Limit &gt; Ripple Limit</b>

**SCPI.CALCulate(*Ch*).SElected.RLIMit.FAIL**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Boolean
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Ripple limit test result.
<i>Allowable Values</i>	True: Fail False: Pass
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.RLIMit.FAIL
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.RLIMit.REPort.DATA**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	<p>The data array, which is the ripple limit test results. The array size is 1+3N, where N is the number of ripple limit bands.</p> <p>For the n–th point, where n from 1 to N:</p> <p><i>Data</i>(0)        N total number of the bands;</p> <p><i>Data</i>(3n–2)     n number of the band;</p> <p><i>Data</i>(3n–1)     Ripple value in the n–th band;</p> <p><i>Data</i>(3n–0)     Ripple limit test result in the n–th band: 0: Pass 1: Fail</p>
<i>Syntax</i>	<p>Dim <i>Data</i> As Variant</p> <p><i>Data</i> = <i>app</i>.SCPI.CALCulate(<i>Ch</i>).SElected.RLIMit.REPort.DATA</p>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).SElected.RLIMit.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the ripple limit test.
<i>Allowable Values</i>	True: Ripple limit test ON False: Ripple limit test OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.RLIMit.STATe <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.RLIMit.STATe = True
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Ripple Limit &gt; Ripple Test</b>

**SCPI.CALCulate(*Ch*).SElected.SMOothing.APERture**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The smoothing aperture for the smoothing function.
<i>Range</i>	from 0.01 to 20
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	1
<i>Unit</i>	%
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.SMOothing.APERture <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.SMOothing.APERture = 1.5
<i>Equivalent Softkeys</i>	<b>Response &gt; Smo Aperture</b>



**SCPI.CALCulate(*Ch*).SElected.SMOothing.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the trace smoothing function.
<i>Allowable Values</i>	True: Trace smoothing ON False: Trace smoothing OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.SMOothing.STATe <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.SMOothing.STATe = True
<i>Equivalent Sofitkeys</i>	<b>Response &gt; Smoothing</b>

**SCPI.CALCulate(*Ch*).SELEcted.TRANSform.TIME.CENTer**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The time domain center value, when the time domain transformation function is turned ON.
<i>Range</i>	Varies depending on the specified frequency range and the number of points.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	0
<i>Unit</i>	s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SELEcted.TRANSform.TIME.CENTer <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SELEcted.TRANSform.TIME.CENTer = 1e–8
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Time Domain &gt; Center</b>

**SCPI.CALCulate(*Ch*).SElected.TRANSform.TIME.IMPulse.WIDTh**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The impulse width (time domain transformation resolution), coupled with the Kaiser–Bessel window shape $\beta$ parameter. The impulse width setting changes the $\beta$ parameter, and setting of $\beta$ parameter changes the impulse width.
<i>Range</i>	Varies depending on the specified frequency range and the number of points.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	None
<i>Unit</i>	s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.IMPulse.WIDTh <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.IMPulse.WIDTh = 1e–8
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Time Domain &gt; Window &gt; Impulse Width</b> (when the transformation type is set to Bandpass or Lowpass Impulse)

**SCPI.CALCulate(*Ch*).SElected.TRANSform.TIME.KBESsel**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The $\beta$ parameter, which controls the Kaiser–Bessel window shape, when performing time domain transformation.
<i>Range</i>	from 0 to 13
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	6
<i>Unit</i>	s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.KBESsel <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.KBESsel = 13
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Time Domain &gt; Window &gt; Kaiser Beta</b>

**SCPI.CALCulate(*Ch*).SElected.TRANSform.TIME.LPFRequency**

<i>Object Type</i>	Method
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Changes the frequency range to match with the lowpass type of the time domain transformation function.
<i>Syntax</i>	<code>app.SCPI.CALCulate(<i>Ch</i>).SElected.TRANSform.TIME.LPFRequency</code>
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Time Domain &gt; Set Frequency Low Pass</b>

**SCPI.CALCulate(*Ch*).SELEcted.TRANSform.TIME.REFLection**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The selection of the reflection type for the time domain transformation function.
<i>Range</i>	"RTRip"      Round Trip "OWAY"      One Way
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	The value is ignored.
<i>Preset Value</i>	"RTR"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.CALCulate( <i>Ch</i> ).SELEcted.TRANSform.TIME.REFLection app.SCPI.CALCulate( <i>Ch</i> ).SELEcted.TRANSform.TIME.REFLection = "OWAY"
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Time Domain &gt; Reflection Type &gt; Round Trip   One Way</b>

**SCPI.CALCulate(*Ch*).SElected.TRANSform.TIME.SPAN**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The time domain span value, when the time domain transformation function is turned ON.
<i>Range</i>	Varies depending on the specified frequency range and the number of points.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	2e–8
<i>Unit</i>	s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.SPAN <i>app</i> . SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.SPAN = 1e–8
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Time Domain &gt; Span</b>

**SCPI.CALCulate(*Ch*).SElected.TRANSform.TIME.START**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The start value used for the transformation function of the time domain function.
<i>Range</i>	Varies depending on the specified frequency range and the number of points.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	$-1e-8$
<i>Unit</i>	s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <code>app.SCPi.CALCulate(<i>Ch</i>).SElected.TRANSform.TIME.START</code> <code>app.SCPi.CALCulate(<i>Ch</i>).SElected.TRANSform.TIME.START</code> = $1e-8$
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Time Domain &gt; Start</b>



**SCPI.CALCulate(*Ch*).SElected.TRANSform.TIME.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the time domain transformation function.
<i>Allowable Values</i>	True: Time domain transformation ON False: Time domain transformation OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.STATe <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.STATe = True
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Time Domain &gt; Time Domain</b>

**SCPI.CALCulate(*Ch*).SElected.TRANSform.TIME.STEP.RTIME**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The rise time of the step signal (time domain transformation resolution), coupled with the Kaiser–Bessel window shape $\beta$ parameter. The impulse width setting changes the $\beta$ parameter, and setting of $\beta$ parameter changes the impulse width.
<i>Range</i>	Varies depending on the specified frequency range and the number of points.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	None
<i>Unit</i>	s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.IMPulse.WIDTh <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.IMPulse.WIDTh = 1e–8
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Time Domain &gt; Window &gt; Impulse Width</b> (when the transformation type is set to Lowpass Step)

**SCPI.CALCulate(*Ch*).SElected.TRANSform.TIME.STIMulus**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The selection of the stimulus type for the time domain transformation function: impulse or step.
<i>Range</i>	"IMPulse"      Impulse "STEP"          Step
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	The value is ignored.
<i>Preset Value</i>	"IMP"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.STIMulus app.SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.STIMulus = "STEP"
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Time Domain &gt; Response Type &gt; Bandpass   Lowpass Step   Lowpass Impulse</b>

**SCPI.CALCulate(*Ch*).SElected.TRANSform.TIME.STOP**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The time domain stop value, when the time domain transformation function is turned ON.
<i>Range</i>	Varies depending on the specified frequency range and the number of points.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	1e–8
<i>Unit</i>	s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.STOP <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.STOP = 2e–8
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Time Domain &gt; Stop</b>

**SCPI.CALCulate(*Ch*).SElected.TRANSform.TIME.TYPE**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The selection of the transformation type for the time domain transformation function: bandpass response or direct current circuit.
<i>Range</i>	"BPASs"      Bandpass "LPASs"      Lowpass
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	The value is ignored.
<i>Preset Value</i>	"BPASs"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.TYPE app.SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.TYPE = "STEP"
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Time Domain &gt; Response Type &gt; Bandpass   Lowpass Step   Lowpass Impulse</b>

**SCPI.CALCulate(*Ch*).SElected.TRANSform.TIME.UNIT**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	The active trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The selection of the unit for the time domain transformation function.
<i>Range</i>	"SEC"            Seconds "MET"            Metre "FEET"           Feet
<i>Out of Range</i>	The value is ignored.
<i>Preset Value</i>	"SEC"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.UNIT app.SCPI.CALCulate( <i>Ch</i> ).SElected.TRANSform.TIME.UNIT = "MET"
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Time Domain &gt; Unit &gt; Time, ns   Metric, m   Imperial, ft</b>

**SCPI.CALCulate(*Ch*).TRACe(*Tr*).DATA.FDATA**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	The specified trace <i>Tr</i> of channel <i>Ch</i> , <i>Tr</i> : trace number 1–8 (see Table 3 on page 29) <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The formatted data array. The array elements contain measurements in the current format, for example, in logarithmic magnitude format (Log Mag). Also, see section “Measurement Data Arrays” on page 19. The array size is 2N, where N is the number of measurement points. For the n–th point, where n from 1 to N: <i>Data</i> (2n–2) real number in rectangular format, real part in polar and Smith chart formats; <i>Data</i> (2n–1) 0 in rectangular format, imaginary part in polar and Smith chart formats.
<i>Syntax</i>	Dim <i>Data</i> As Variant <i>Data</i> = <i>app</i> .SCPI.CALCulate( <i>Ch</i> ).Trace( <i>Tr</i> ).DATA.FDATA
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*).TRACe(*Tr*).DATA.FMEMory**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	The specified trace <i>Tr</i> of channel <i>Ch</i> , <i>Tr</i> : trace number 1–8 (see Table 3 on page 29) <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The formatted memory array. The array elements contain saved measurements in the current format, for example, in logarithmic magnitude format (Log Mag). Also, see section “Measurement Data Arrays” on page 19. The array size is 2N, where N is the number of measurement points. For the n–th point, where n from 1 to N: <i>Data</i> (2n–2) real number in rectangular format, real part in polar and Smith chart formats; <i>Data</i> (2n–1) 0 in rectangular format, imaginary part in polar and Smith chart formats.
<i>Syntax</i>	Dim <i>Data</i> As Variant <i>Data</i> = app.SCPI.CALCulate( <i>Ch</i> ). Trace( <i>Tr</i> ).DATA.FMEMory
<i>Equivalent Softkeys</i>	<b>None</b>



**SCPI.CALCulate(*Ch*). TRACe(*Tr*).DATA.SDATA**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	The specified trace <i>Tr</i> of channel <i>Ch</i> , <i>Tr</i> : trace number 1–8 (see Table 3 on page 29) <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The corrected data array. The corrected measurements are complex numbers. Also, see section “Measurement Data Arrays” on page 19. The array size is 2N, where N is the number of measurement points. For the n–th point, where n from 1 to N: <i>Data</i> (2n–2) the real part of corrected measurement; <i>Data</i> (2n–1) the imaginary part of corrected measurement.
<i>Syntax</i>	Dim <i>Data</i> As Variant <i>Data</i> = app.SCPICALCulate( <i>Ch</i> ). Trace( <i>Tr</i> ).DATA.SDATA
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.CALCulate(*Ch*). TRACe(*Tr*).DATA.SMEMory**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	The specified trace <i>Tr</i> of channel <i>Ch</i> , <i>Tr</i> : trace number 1–8 (see Table 1 on page 21) <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The corrected memory array. The corrected measurements are complex numbers. Also, see section “Measurement Data Arrays” on page 19. The array size is 2N, where N is the number of measurement points. For the n–th point, where n from 1 to N: <i>Data</i> (2n–2) the real part of corrected measurement memory; <i>Data</i> (2n–1) the imaginary part of corrected measurement memory.
<i>Syntax</i>	Dim <i>Data</i> As Variant <i>Data</i> = app.SCPI.CALCulate( <i>Ch</i> ). Trace( <i>Tr</i> ).DATA.SMEMory
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.DISPlay.ANNotation.FREQuency.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Instrument
<i>Description</i>	The ON/OFF state of the frequency or time label on the channel window.
<i>Allowable Values</i>	True: Frequency label display ON False: Frequency label display OFF
<i>Preset Value</i>	True
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.DISPlay.ANNotation.FREQuency.STATe <i>app</i> .SCPI.DISPlay. ANNotation.FREQuency.STATe = False
<i>Equivalent Softkeys</i>	<b>Display &gt; Frequency Label</b>

**SCPI.DISPlay.COLor.BACK**

<i>Object Type</i>	Property (read/write)						
<i>Data Type</i>	Variant (Long array)						
<i>Target</i>	Instrument						
<i>Description</i>	<p>The background color for trace display.</p> <p>The array contains 3 elements:</p> <table style="margin-left: 40px;"> <tr> <td><i>Data(0)</i></td> <td>Red value R;</td> </tr> <tr> <td><i>Data(1)</i></td> <td>Green value G;</td> </tr> <tr> <td><i>Data(2)</i></td> <td>Blue value B.</td> </tr> </table>	<i>Data(0)</i>	Red value R;	<i>Data(1)</i>	Green value G;	<i>Data(2)</i>	Blue value B.
<i>Data(0)</i>	Red value R;						
<i>Data(1)</i>	Green value G;						
<i>Data(2)</i>	Blue value B.						
<i>Range</i>	For all the array elements from 0 to 255.						
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.						
<i>Preset Value</i>	16, 16, 16						
<i>Syntax</i>	<p>Dim <i>Data</i> As Variant</p> <p><i>Data</i> = <i>app</i>.SCPI.DISPlay.COLor.BACK</p> <p><i>app</i>.SCPI.DISPlay.COLor.BACK = Array(0, 0, 0)</p>						
<i>Equivalent Softkeys</i>	<b>Display &gt; Color &gt; Background</b>						

**SCPI.DISPlay.COLor.GRATicule**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Variant (Long array)
<i>Target</i>	Instrument
<i>Description</i>	<p>The grid and the graticule label color for trace display.</p> <p>The array contains 3 elements:</p> <p style="padding-left: 40px;"><i>Data(0)</i>      Red value R;</p> <p style="padding-left: 40px;"><i>Data(1)</i>      Green value G;</p> <p style="padding-left: 40px;"><i>Data(2)</i>      Blue value B.</p>
<i>Range</i>	For all array elements from 0 to 255.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	<p style="padding-left: 40px;"><i>Data(0)</i>      63;</p> <p style="padding-left: 40px;"><i>Data(1)</i>      63;</p> <p style="padding-left: 40px;"><i>Data(2)</i>      63.</p>
<i>Syntax</i>	<p>Dim <i>Data</i> As Variant</p> <p><i>Data</i> = <i>app</i>.SCPI.DISPlay.COLor.GRATicule</p> <p><i>app</i>.SCPI.DISPlay.COLor. GRATicule = Array(128, 128, 128)</p>
<i>Equivalent Softkeys</i>	<b>Display &gt; Color &gt; Grid</b>

**SCPI.DISPlay.COLor.RESet**

<i>Object Type</i>	Method
<i>Target</i>	Instrument
<i>Description</i>	Restores the display settings to the default values.
<i>Syntax</i>	<code>app.SCPI.DISPlay.COLor.RESet</code>
<i>Equivalent Softkeys</i>	<b>Display &gt; Set Defaults</b>

**SCPI.DISPlay.COLor.TRACe(*Tr*).DATA**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Variant (Long array)
<i>Target</i>	Trace number <i>Tr</i> in all channels, <i>Tr</i> : trace number 1–8 (see Table 3 on page 29)
<i>Description</i>	The data trace color. The array contains 3 elements: <i>Data</i> (0) Red value R; <i>Data</i> (1) Green value G; <i>Data</i> (2) Blue value B.
<i>Range</i>	For all array elements from 0 to 255.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	Varies depending on the trace number.
<i>Syntax</i>	Dim <i>Data</i> As Variant <i>Data</i> = <i>app</i> .SCPI.DISPlay.COLor.TRACe( <i>Tr</i> ).DATA <i>app</i> .SCPI.DISPlay.COLor.TRACe( <i>Tr</i> ).DATA = Array(255, 255, 0)
<i>Equivalent Softkeys</i>	<b>Display &gt; Color &gt; Data Trace</b>

**SCPI.DISPLAY.COLOR.TRACe(*Tr*).MEMory**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Variant (Long array)
<i>Target</i>	Trace number <i>Tr</i> in all channels <i>Tr</i> : trace number 1–8 (see Table 3 on page 29)
<i>Description</i>	The memory trace color. The array contains 3 elements: <i>Data</i> (0) Red value R; <i>Data</i> (1) Green value G; <i>Data</i> (2) Blue value B.
<i>Range</i>	For all array elements from 0 to 255.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	Varies depending on the trace number.
<i>Syntax</i>	Dim <i>Data</i> As Variant <i>Data</i> = app.SCPI.DISPLAY.COLOR.TRACe( <i>Tr</i> ).MEMory app.SCPI.DISPLAY.COLOR.TRACe( <i>Tr</i> ). MEMory = Array(255, 255, 0)
<i>Equivalent Softkeys</i>	<b>Display &gt; Color &gt; Memory Trace</b>



**SCPI.DISPlay.FSIGN**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Instrument
<i>Description</i>	The ON/OFF state of the <i>Fail</i> sign display, when performing limit test or ripple limit test.
<i>Allowable Values</i>	True: <i>Fail</i> sign display ON False: <i>Fail</i> sign display OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.DISPlay.FSIGN <i>app</i> .SCPI.DISPlay.FSIGN = True
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Limit Test &gt; Fail Sign</b> <b>Analysis &gt; Ripple Limit &gt; Fail Sign</b>

**SCPI.DISPlay.IMAGe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Instrument
<i>Description</i>	The inverted color display of the data traces.
<i>Range</i>	"NORMAl" : Normal display "INVert" : Inverted color display
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	The value is ignored.
<i>Preset Value</i>	"NORM"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.DISPlay.IMAGe app.SCPI.DISPlay.IMAGe = "INV"
<i>Equivalent Softkeys</i>	<b>Display &gt; Invert Color</b>




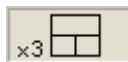
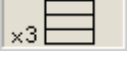
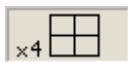
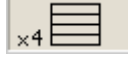
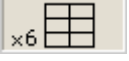
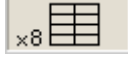
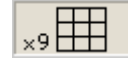
**SCPI.DISPlay.MAXimize**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Active Channel
<i>Description</i>	The ON/OFF maximize state of the active channel window.
<i>Allowable Values</i>	True: Maximize active channel state ON False: Maximize active channel state OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = app.SCPI.DISPlay.MAXimize app.SCPI.DISPlay.MAXimize = True
<i>Equivalent Softkeys</i>	<b>Channel &gt; Maximize Channel</b>

**SCPI.DISPlay.SPLit**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Long
<i>Target</i>	Instrument
<i>Description</i>	The number and layout of the channel windows on the screen. The channel window layout is in Table 5 below.
<i>Range</i>	from 1 to 10
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	1
<i>Syntax</i>	Dim Value As Long Value = app.SCPI.DISPlay.SPLit app.SCPI.DISPlay.SPLit = 2
<i>Equivalent Softkeys</i>	<b>Channel &gt; Allocate channels</b>

**Table 5. Channel Window Layout on the Screen**

1: 	2: 	3: 
4: 	5: 	6: 
7: 	8: 	9: 
10: 		

**SCPI.DISPlay.UPDATE\_.IMMEDIATE**

<i>Object Type</i>	Method
<i>Target</i>	Instrument
<i>Description</i>	Updates the display once, when the display update is set to OFF (SCPI.DISPlay.ENABLE property is set to False).
<i>Syntax</i>	<code>app.SCPI.DISPlay.UPDATE_.IMMEDIATE</code>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.DISPlay.WINDOW(*Ch*).ACTivate**

<i>Object Type</i>	Method
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Sets the active channel.
<i>Notes</i>	The channel window must be displayed. At attempt to set to the active channel the channel, which is not displayed, an error occurs.
<i>Syntax</i>	<code>app.SCPI.DISPlay.WINDOW(<i>Ch</i>).ACTivate</code>
<i>Equivalent Softkeys</i>	<b>Channel &gt; Active Channel</b> <b>Trace &gt; Active Trace</b>

**SCPI.DISPlay.WINDow(Ch).ANNotation.MARKer.ALIGn.TYPE**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The alignment mode of the marker display position of each trace, when the only active trace display feature is turned OFF (SCPI.DISPlay.WINDow(Ch).ANNotation.MARKer.SINGle.STATe property is set to False).
<i>Range</i>	"VERTical" : Vertical alignment "HORizontal" : Horizontal alignment "NONE" : No alignment
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	The value is ignored.
<i>Preset Value</i>	"NONE"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.DISPlay.WINDow( <i>Ch</i> ).ANNotation.MARKer.ALIGn.TYPE app.SCPI.DISPlay.WINDow( <i>Ch</i> ).ANNotation.MARKer.ALIGn.TYPE = "VERT"
<i>Equivalent Softkeys</i>	<b>Markers &gt; Properties &gt; Align &gt; Vertical   Horizontal   OFF</b>

**SCPI.DISPlay.WINDow(*Ch*).ANNotation.MARKeR.SINGle.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the marker display for the active trace only.
<i>Allowable Values</i>	True: Only active trace markers display ON False: Only active trace markers display OFF
<i>Preset Value</i>	True
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI SCPI.DISPlay.WINDow( <i>Ch</i> ).ANNotation.MARKeR.SINGle.STATe <i>app</i> .SCPI SCPI.DISPlay.WINDow( <i>Ch</i> ).ANNotation.MARKeR.SINGle.STATe = True
<i>Equivalent Softkeys</i>	<b>Markers &gt; Properties &gt; Active Only</b>

**SCPI.DISPlay.WINDows(Ch).MAXimize**

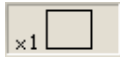

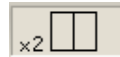

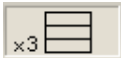
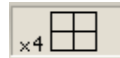
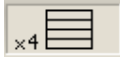
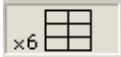
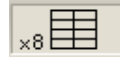
<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Active Trace of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF maximize state of the active trace for channel window.
<i>Allowable Values</i>	True: Maximize active trace state ON False: Maximize active trace state OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.DISPlay.WINDows(1).MAXimize <i>app</i> .SCPI.DISPlay.WINDows(1).MAXimize = True
<i>Equivalent Softkeys</i>	<b>Trace &gt; Maximize Trace</b>



**SCPI.DISPlay.WINDows(Ch).SPLit**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Long
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The number and layout of the trace on the channel window. The trace layout is in Table 6 below.
<i>Range</i>	From 1 to 9
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	1
<i>Syntax</i>	Dim <i>Value</i> As Long <i>Value</i> = <i>app</i> .SCPI.DISPlay.WINDows(1).SPLit <i>app</i> . <i>app</i> .SCPI.DISPlay.WINDows(1).SPLit = 2
<i>Equivalent Softkeys</i>	<b>Trace -&gt; Allocate Traces</b>

**Table 6. Trace Layout on the Channel Window**

1: 	2: 	3: 
4: 	5: 	6: 
7: 	8: 	9: 

**SCPI.DISPlay.WINDow(*Ch*).TITLe.DATA**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Channel <i>Ch</i> , <i>Ch:</i> channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The channel title label.
<i>Range</i>	up to 254 characters
<i>Preset Value</i>	""
<i>Syntax</i>	Dim <i>Text</i> As String <i>Text</i> = app.SCPI.DISPlay.WINDow( <i>Ch</i> ).TITLe.DATA app.SCPI.DISPlay.WINDow( <i>Ch</i> ).TITLe.DATA = "Example1"
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.DISPlay.WINDow(*Ch*).TITLe.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the title label display.
<i>Allowable Values</i>	True: Title label display ON False: Title label display OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.DISPlay.WINDow( <i>Ch</i> ).TITLe.STATe <i>app</i> .SCPI.DISPlay.WINDow( <i>Ch</i> ).TITLe.STATe = True
<i>Equivalent Softkeys</i>	<b>Display &gt; Title Label</b>

## SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).ANNotation.MARKer.POSition. X

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Trace <i>Tr</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Tr</i> : trace number 1–8 (see Table 3 on page 29)
<i>Description</i>	The display position of the marker value on the X-axis by a percentage of the display width.
<i>Range</i>	from 0 to 100
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	0
<i>Unit</i>	%
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.DISPlay.WINDow( <i>Ch</i> ).TRACe( <i>Tr</i> ).ANNotation.MARKer.POSition.X <i>app</i> .SCPI.DISPlay.WINDow( <i>Ch</i> ).TRACe( <i>Tr</i> ).ANNotation.MARKer.POSition.X = 50
<i>Equivalent Softkeys</i>	<b>Markers &gt; Properties &gt; Data X Position</b>

## SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).ANNotation.MARKer.POSition. Y

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Trace <i>Tr</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Tr</i> : trace number 1–8 (see Table 3 on page 29)
<i>Description</i>	The display position of the marker value on the Y-axis by a percentage of the display height.
<i>Range</i>	from 0 to 100
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	0
<i>Unit</i>	%
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.DISPlay.WINDow( <i>Ch</i> ).TRACe( <i>Tr</i> ).ANNotation.MARKer.POSition.Y <i>app</i> .SCPI.DISPlay.WINDow( <i>Ch</i> ).TRACe( <i>Tr</i> ).ANNotation.MARKer.POSition.Y = 50
<i>Equivalent Softkeys</i>	<b>Markers &gt; Properties &gt; Data Y Position</b>

**SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.AUTO**

<i>Object Type</i>	Method
<i>Target</i>	Trace <i>Tr</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Tr</i> : trace number 1–8 (see Table 3 on page 29)
<i>Description</i>	Executes the auto scale function for the trace.
<i>Syntax</i>	<code>app.SCPI.DISPlay.WINDow(<i>Ch</i>).TRACe(<i>Tr</i>).Y.SCALe.AUTO</code>
<i>Equivalent Softkeys</i>	<b>Scale &gt; Auto Scale</b>

**SCPI.DISPlay.WINdow(*Ch*).TRACe(*Tr*).Y.SCALe.PDIVision**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Trace <i>Tr</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Tr</i> : trace number 1–8 (see Table 3 on page 29)
<i>Description</i>	The trace scale. Sets the scale per division, when the data format is the rectangular format. Sets the full scale value, when the data format is the Smith chart format or the polar format.
<i>Range</i>	from 10E–18 to 1E18
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	Varies depending on the format. Logarithmic Magnitude: 10 dB/Div Phase: 40 °/Div Expand Phase: 100 °/Div Group Delay: 10e–9 s/Div Smith Chart, Polar, SWR: 1 /Div Linear Magnitude: 0.1 /Div Real part, Imaginary part: 0.2 /Div
<i>Unit</i>	dB/Div (decibel per division), °/Div (degree per division), s/Div (second per division)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.DISPlay.WINdow( <i>Ch</i> ).TRACe( <i>Tr</i> ).Y.SCALe.PDIVision <i>app</i> .SCPI.DISPlay.WINdow( <i>Ch</i> ).TRACe( <i>Tr</i> ).Y.SCALe.PDIVision = 20
<i>Equivalent Softkeys</i>	<b>Scale &gt; Scale</b>

**SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.RLEVel**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Trace <i>Tr</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Tr</i> : trace number 1–8 (see Table 3 on page 29)
<i>Description</i>	The value of the reference line (response value on the reference line). For the rectangular format only.
<i>Range</i>	from $-1E-18$ to $1E18$
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	0 (except for SWR: 1)
<i>Unit</i>	dB (decibel)   ° (degree)   s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <code>app.SCPI.DISPlay.WINDow(<i>Ch</i>).TRACe(<i>Tr</i>).Y.SCALe.RLEVel</code> <code>app.SCPI.DISPlay.WINDow(<i>Ch</i>).TRACe(<i>Tr</i>).Y.SCALe.RLEVel = 10</code>
<i>Equivalent Softkeys</i>	<b>Scale &gt; Ref Value</b>



**SCPI.DISPlay.WINDow(*Ch*).TRACe(*Tr*).Y.SCALe.RPOSITION**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Long
<i>Target</i>	Trace <i>Tr</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Tr</i> : trace number 1–8 (see Table 3 on page 29)
<i>Description</i>	The position of the reference line. For the rectangular format only.
<i>Range</i>	From 0 to the number of the scale divisions (set by the SCPI.DISPlay.WINDow( <i>Ch</i> ).Y.SCALe.DIVisions property, 10 by default).
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	5 (except for SWR: 0)
<i>Syntax</i>	Dim <i>Value</i> As Long <i>Value</i> = <i>app</i> .SCPI.DISPlay.WINDow( <i>Ch</i> ).TRACe( <i>Tr</i> ).Y.SCALe.RPOSITION <i>app</i> .SCPI.DISPlay.WINDow( <i>Ch</i> ).TRACe( <i>Tr</i> ).Y.SCALe.RPOSITION = 10
<i>Equivalent Softkeys</i>	<b>Scale &gt; Ref Position</b>

**SCPI.DISPlay.WINDow(*Ch*).Y.SCALe.DIVisions**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Long
<i>Target</i>	Channel <i>Ch</i> , <i>Ch:</i> channel number 1–9 (see Table 1 on page 21).
<i>Description</i>	The number of the vertical scale divisions. For the rectangular format only.
<i>Range</i>	from 4 to 20
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	10
<i>Resolution</i>	2
<i>Syntax</i>	Dim <i>Value</i> As Long <i>Value</i> = <code>app.SCPI.DISPlay.WINDow(<i>Ch</i>).Y.SCALe.DIVisions</code> <code>app.SCPI.DISPlay.WINDow(<i>Ch</i>).Y.SCALe.DIVisions</code> = 12
<i>Equivalent Softkeys</i>	<b>Scale &gt; Divisions</b>

**SCPI.HCOPy.DATE.STAMp**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Instrument
<i>Description</i>	The ON/OFF state of the current date and time printout in the upper right corner.
<i>Allowable Values</i>	True: Date & time printout ON False: Date & time printout OFF
<i>Preset Value</i>	True
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.HCOPy.DATE.STAMp <i>app</i> .SCPI.HCOPy.DATE.STAMp = False
<i>Equivalent Softkeys</i>	<b>System &gt; Print &gt; Print Date &amp; Time</b>

**SCPI.HCOPy.IMAGe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Instrument
<i>Description</i>	The inverted color image printout.
<i>Range</i>	"NORMal" : Normal printout "INVert" : Inverted color printout
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	The value is ignored.
<i>Preset Value</i>	"NORM"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.HCOPy.IMAGe app.SCPI.HCOPy.IMAGe = "INV"
<i>Equivalent Softkeys</i>	<b>System &gt; Print &gt; Invert Image</b>

**SCPI.HCOPy.IMMEDIATE**

<i>Object Type</i>	Method
<i>Target</i>	Instrument
<i>Description</i>	Prints out the image displayed on the screen without previewing.
<i>Syntax</i>	<code>app.SCPI.HCOPy.IMMEDIATE</code>
<i>Equivalent Softkeys</i>	<b>System &gt; Print &gt; Print Embedded</b>

**SCPI.HCOPy.PAINt**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Instrument
<i>Description</i>	The color chart for the image printout.
<i>Range</i>	"COLor" : Color printout "GRAY" : Grayscale printout "BW" : Black&white printout
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	The value is ignored.
<i>Preset Value</i>	"BW"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.HCOPy.PAINt app.SCPI.HCOPy.PAINt = "COL"
<i>Equivalent Softkeys</i>	<b>System &gt; Print &gt; Print Color</b>

**SCPI.IEEE4882.IDN**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	String
<i>Target</i>	Instrument
<i>Description</i>	The instrument information string. The string format: "{manufacturer}, {model}, {serial number}, {software version/firmware version}".
<i>Range</i>	up to 40 characters
<i>Syntax</i>	Dim <i>ID</i> As String <i>ID</i> = <i>app.NAME</i>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.IEEE4882.RST**

<i>Object Type</i>	Method
<i>Target</i>	Instrument
<i>Description</i>	Restores the default settings of the instrument. There is difference from presetting the instrument with the SCPI.SYSTem.PRESet method – in this case the trigger mode is set to <i>Hold</i> .
<i>Syntax</i>	<i>app</i> .SCPI.IEEE4882.RST
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.IEEE4882.TRG**

<i>Object Type</i>	Method
<i>Target</i>	Instrument
<i>Description</i>	If the trigger source is set to LAN (SCPI.TRIGger.SEQuence.SOURce property is set to " <i>BUS</i> "), triggers a sweep. If the trigger source is not set to the bus (SCPI.TRIGger.SEQuence.SOURce property is not set to " <i>BUS</i> ") or the instrument is not waiting for a trigger, the method is ignored.
<i>Syntax</i>	<i>app</i> .SCPI.IEEE4882.TRG
<i>Related Commands</i>	SCPI.TRIGger.SEQuence.SOURce SCPI.INITiate( <i>Ch</i> ).CONTInuous SCPI.INITiate( <i>Ch</i> ).IMMEDIATE
<i>Equivalent Softkeys</i>	<b>None</b>



**SCPI.IEEE4882.WAI**

<i>Object Type</i>	Method
<i>Target</i>	Instrument
<i>Description</i>	Waits for the execution of all commands sent before this command.
<i>Syntax</i>	<i>app</i> .SCPI.IEEE4882.WAI
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.INITiate(Ch).CONTInuous**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the continuous trigger initiation mode. If the continuous trigger initiation mode is set to OFF, the channel turns to the hold state.
<i>Allowable Values</i>	True: Continuous trigger initiation mode ON False: Continuous trigger initiation mode OFF
<i>Preset Value</i>	True
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.INITiate( <i>Ch</i> ).CONTInuous <i>app</i> .SCPI.INITiate( <i>Ch</i> ).CONTInuous = False
<i>Notes</i>	The sweep start in continuous trigger initiation mode depends on the trigger source. If the trigger is set to internal, the sweeps will go immediately one after another. If the trigger is set otherwise, the sweep will start when the trigger signal is received.
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Trigger &gt; Continuous</b> <b>Stimulus &gt; Trigger &gt; Hold</b>

**SCPI.INITiate(*Ch*).IMMEDIATE**

<i>Object Type</i>	Method
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	<p>Sets the channel to the single trigger mode. Before this method is called, the channel must be in hold state, otherwise an error occurs (error code 213) and the method is ignored.</p> <p>On completion of the sweep, the channel goes back into the hold state.</p> <p>The method returns control before the end of the sweep.</p>
<i>Syntax</i>	<i>app</i> .SCPI.INITiate( <i>Ch</i> ).IMMEDIATE
<i>Notes</i>	The sweep start in the single trigger mode depends on the trigger source. If the trigger is set to internal, the sweep will start immediately after the method is called. If the trigger is set otherwise, the sweep will start when the trigger signal is received.
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Trigger &gt; Single</b>

**SCPI.MMEMory.COPY(*Src, Dst*)**

<i>Object Type</i>	Method
<i>Target</i>	Instrument
<i>Description</i>	Copies a file.
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.COPY( <i>Src, Dst</i> )
<i>Parameter</i>	<i>Src</i> – Source file name. String data type. <i>Dst</i> – Destination file name. String data type.
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.MMEMory.DELEte(*File*)**

<i>Object Type</i>	Method
<i>Target</i>	Instrument
<i>Description</i>	Deletes a file.
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.DELEte( <i>File</i> )
<i>Parameter</i>	<i>File</i> – File name. String data type.
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.MMEMory.LOAD.CKIT(*Ck*)**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	String
<i>Target</i>	Calibration kit <i>Ck</i> , <i>Ck</i> : calibration kit number 1–11 (see Table 7 on page 157)
<i>Description</i>	Recalls the definition file for the calibration kit. The file must be saved by the SCPI.MMEMory.STORe.CKIT( <i>Ck</i> ) property.
<i>Range</i>	up to 254 characters
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.LOAD.CKIT( <i>Ck</i> ) = <i>File</i>
<i>Notes</i>	If the full path to the file is not specified, the <i>\CalKit</i> subdirectory of the main directory will be searched for the file. The calibration kit definition file has *. <i>ckd</i> extension by default.
<i>Equivalent Softkeys</i>	<b>None</b>

**Table 7. *Ck*: Calibration Kit Number**

<i>Data Type</i>	Long
<i>Description</i>	Calibration kit number.
<i>Range</i>	from 1 to 16
<i>Out of Range</i>	An error occurs. Error code: 114.
<i>Notes</i>	If the calibration kit number is not specified, it is taken as equal to 1.

## SCPI.MMEMory.LOAD.CALibration

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	String
<i>Target</i>	User calibration of the active channel.
<i>Description</i>	Recalls the specified calibration file. The file must be saved by the SCPI.MMEMory.STORE.CALibration property.
<i>Range</i>	up to 254 characters
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.LOAD.CALibration = <i>File</i>
<i>Notes</i>	If the full path to the file is not specified, the <i>\State</i> subdirectory of the main directory will be searched for the file. The calibration files have <i>*.cal</i> extension by default.
<i>Equivalent Softkeys</i>	<b>System &gt; Recall &gt; Calibration</b>

**SCPI.MMEMory.LOAD.LIMit**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	String
<i>Target</i>	Active trace of the active channel.
<i>Description</i>	Recalls the specified limit table file. The file must be saved by the SCPI.MMEMory.STORe.LIMit property.
<i>Range</i>	up to 254 characters
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.LOAD.LIMit = <i>File</i>
<i>Notes</i>	If the full path to the file is not specified, the \Limit subdirectory of the main directory will be searched for the file. The limit table files have *.lim extension by default.
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Limit Test &gt; Edit Limit Line &gt; Restore Limit Table</b>

**SCPI.MMEMory.LOAD.RLIMit**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	String
<i>Target</i>	Active trace of the active channel.
<i>Description</i>	Recalls the ripple limit table file. The file must be saved by the SCPI.MMEMory.STORe.RLIMit property.
<i>Range</i>	up to 254 characters
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.LOAD.RLIMit = <i>File</i>
<i>Notes</i>	If the full path to the file is not specified, the <i>\Limit</i> subdirectory of the main directory will be searched for the file. The ripple limit files have <i>*.rlm</i> extension by default.
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Ripple Limit &gt; Edit Ripple Limit &gt; Restore Ripple Limit Table</b>



**SCPI.MMEMory.LOAD.SEGMent**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	String
<i>Target</i>	Active channel
<i>Description</i>	Recalls the segment table file. The file must be saved by the SCPI.MMEMory.STORE.SEGMent property.
<i>Range</i>	up to 254 characters
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.LOAD.SEGMent = <i>File</i>
<i>Notes</i>	If the full path to the file is not specified, the \Segment subdirectory of the main directory will be searched for the file. The segment files have *.seg extension by default.
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Segment Table &gt; Recall</b>

**SCPI.MMEMory.LOAD.STATe**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	String
<i>Target</i>	Instrument
<i>Description</i>	Recalls the specified instrument state file. The file must be saved by the SCPI.MMEMory.STORe.STATe property.
<i>Range</i>	up to 254 characters
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.LOAD.STATe = <i>File</i>
<i>Notes</i>	If the full path to the file is not specified, the \State subdirectory of the main directory will be searched for the file. The instrument state files have *.cfg extension by default.
<i>Equivalent Softkeys</i>	<b>System &gt; Recall &gt; State</b> <b>System &gt; Recall &gt; Calibration</b> <b>System &gt; Recall &gt; State and Calibration</b>

**SCPI.MMEMory.MDIRectory**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	String
<i>Target</i>	Instrument
<i>Description</i>	Creates a new directory (folder). Contains the full path to the folder being created.
<i>Range</i>	up to 254 characters
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.MDIRectory = <i>Path</i>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.MMEMory.STORe.CKIT(*Ck*)**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	String
<i>Target</i>	Calibration kit <i>Ck</i> , <i>Ck</i> : calibration kit number 1–11 (see Table 7 on page 157)
<i>Description</i>	Saves the definition file for the calibration kit parameters.
<i>Range</i>	up to 254 characters
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.STORe.CKIT( <i>Ck</i> ) = <i>File</i>
<i>Notes</i>	If the full path to the file is not specified, the file will be saved to the <i>\CalKit</i> subdirectory of the main directory. The calibration kit definition file has *. <i>dat</i> extension by default.
<i>Equivalent Softkeys</i>	<b>None</b>

## SCPI.MMEMory.STORe.CALibration

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	String
<i>Target</i>	User calibration of the active channel.
<i>Description</i>	Save the specified calibration file.
<i>Range</i>	up to 254 characters
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.STORe.CALibration = <i>File</i>
<i>Notes</i>	If the full path to the file is not specified, the <i>\State</i> subdirectory of the main directory will be used for the file. The calibration files have <i>*.cal</i> extension by default.
<i>Equivalent Softkeys</i>	<b>System &gt; Save &gt; Calibration</b>

**SCPI.MMEMory.STORe.FDATa**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	String
<i>Target</i>	Active trace of the active channel
<i>Description</i>	Saves the CSV formatted data into a file.
<i>Range</i>	up to 254 characters
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.STORe.FDATa = <i>File</i>
<i>Notes</i>	If the full path to the file is not specified, the file will be saved to the \CSV subdirectory of the main directory. The files have *.csv extension by default.
<i>Equivalent Softkeys</i>	<b>System &gt; Save &gt; Save Trace Data</b>

**SCPI.MMEMory.STORe.IMAGe**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	String
<i>Target</i>	Instrument
<i>Description</i>	Saves the display image in BMP or PNG format into a file.
<i>Range</i>	up to 254 characters
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.STORe.IMAGe = <i>File</i>
<i>Notes</i>	If the full path to the file is not specified, the file will be saved to the <i>\Image</i> subdirectory of the main directory. If the file has <i>*.png</i> extension, the file has PNG format, in all the other cases the file has BMP format.
<i>Equivalent Softkeys</i>	<b>System &gt; Print &gt; Print Windows</b>

**SCPI.MMEMory.STORe.LIMit**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	String
<i>Target</i>	Active trace of the active channel
<i>Description</i>	Saves the limit table into a file with the specified name.
<i>Range</i>	up to 254 characters
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.STORe.LIMit = <i>File</i>
<i>Notes</i>	If the full path to the file is not specified, the file will be saved to the <i>\Limit</i> subdirectory of the main directory. The files have <i>*.lim</i> extension by default.
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Limit Test &gt; Edit Limit Line &gt; Save Limit Table</b>



**SCPI.MMEMory.STORe.RLIMit**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	String
<i>Target</i>	Active trace of the active channel
<i>Description</i>	Saves the ripple limit table into a file with the specified name.
<i>Range</i>	up to 254 characters
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.STORe.RLIMit = <i>File</i>
<i>Notes</i>	If the full path to the file is not specified, the file will be saved to the <i>\Limit</i> subdirectory of the main directory. The ripple limit files have <i>*.rlm</i> extension by default.
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Ripple Limit &gt; Edit Ripple Limit &gt; Save Ripple Limit Table</b>

**SCPI.MMEMory.STORe.SEGMent**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	String
<i>Target</i>	Active channel
<i>Description</i>	Saves the segment table in a file with the specified name.
<i>Range</i>	up to 254 characters
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.STORe.SEGMent = <i>File</i>
<i>Notes</i>	If the full path to the file is not specified, the file will be saved to the \Segment subdirectory of the main directory. The segment files have *.seg extension by default.
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Segment Table &gt; Save</b>

**SCPI.MMEMory.STORe.SNP.DATA**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	String
<i>Target</i>	Active channel
<i>Description</i>	Saves the measured S-parameters of the active channel into a Touchstone file with the specified name. The file type (1-port or 2-port) is set by the SCPI.MMEMory.STORe.SNP.TYPE.S1P and SCPI.MMEMory.STORe.SNP.TYPE.S2P properties. 1-port type file saves one reflection parameter: S11. 2-port type file saves the two parameters: S11, S21.
<i>Range</i>	up to 254 characters
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.STORe.SNP.DATA = <i>File</i>
<i>Notes</i>	If the full path to the file is not specified, the file will be saved to the <i>\FixtureSim</i> subdirectory of the main directory. The 1-port measurement files have *.slp extension; the 2-port measurement files have *.s2p extension.
<i>Equivalent Softkeys</i>	<b>System &gt;Save &gt; Touchstone File &gt; Save</b>

**SCPI.MMEMory.STORe.SNP.FORMat**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Active channel
<i>Description</i>	The data format for the S-parameters saving by the SCPI.MMEMory.STORe.SNP.DATA property.
<i>Range</i>	"MA" : Logarithmic Magnitude / Angle format "DB" : Linear Magnitude / Angle format "RI" : Real part /Imaginary part format
<i>Out of Range</i>	The value is ignored.
<i>Preset Value</i>	"RI"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.MMEMory.STORe.SNP.FORMat app.SCPI.MMEMory.STORe.SNP.FORMat = "DB"
<i>Equivalent Softkeys</i>	<b>System &gt; Save &gt; Touchstone File &gt; Format</b>

**SCPI.MMEMory.STORe.SNP.TYPE.S1P**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Long
<i>Target</i>	Active channel
<i>Description</i>	Selects 1-port measurements and sets port numbers (selects S11 parameter) when saving the measured S-parameters by the SCPI.MMEMory.STORe.SNP.DATA property.
<i>Range</i>	from 1 to 2
<i>Out of Range</i>	An error occurs. Error code 222.
<i>Preset Value</i>	1
<i>Syntax</i>	Dim <i>Value</i> As Long <i>Value</i> = app.SCPI.MMEMory.STORe.SNP.TYPE.S1P app.SCPI.MMEMory.STORe.SNP.TYPE.S1P = 2
<i>Equivalent Softkeys</i>	<b>System &gt; Save &gt; Touchstone File &gt; Type &gt; 1–Port (s1p)</b> <b>System &gt; Save &gt; Touchstone File &gt; Select Port (s1p)</b>

**SCPI.MMEMory.STORe.SNP.TYPE.S2P**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Variant (Long array)
<i>Target</i>	Active channel
<i>Description</i>	<p>Selects 2-port measurements and sets port numbers when saving the measured S-parameters by the SCPI.MMEMory.STORe.SNP.DATA property.</p> <p>The array contains 2 elements:</p> <p style="padding-left: 40px;"><i>Data(0)</i>      First port number (reads out 1, ignored when written);</p> <p style="padding-left: 40px;"><i>Data(1)</i>      Second port number (reads out 2, ignored when written).</p>
<i>Syntax</i>	<p>Dim <i>Data</i> As Variant</p> <p><i>Data</i> = app.SCPI.MMEMory.STORe.SNP.TYPE.S2P</p> <p>app.SCPI.MMEMory.STORe.SNP.TYPE.S2P = Array(1, 2)</p>
<i>Equivalent Softkeys</i>	<b>System &gt; Save &gt; Touchstone File &gt; Type &gt; 2-Port (s2p)</b>

**SCPI.MMEMory.STORe.STATe**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	String
<i>Target</i>	Instrument
<i>Description</i>	Saves the instrument state into a file with the specified name.
<i>Range</i>	up to 254 characters
<i>Syntax</i>	<i>app</i> .SCPI.MMEMory.STORe.STATe = <i>File</i>
<i>Notes</i>	If the full path to the file is not specified, the file will be saved to the <i>\State</i> subdirectory of the main directory. The state files have *. <i>cfg</i> extension by default.
<i>Equivalent Softkeys</i>	<b>System &gt; Save &gt; State</b>

**SCPI.MMEMory.STORe.STYPe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Instrument
<i>Description</i>	Selects the type of the instrument or channel state saving by the SCPI.MMEMory.STORe.STATe or SCPI.MMEMory.STORe.CHANnel.STATe property.
<i>Range</i>	"STATe" : Measurement conditions "CSTate" : Measurement conditions and calibration tables "DSTate" : Measurement conditions and data traces "CDSTate" : Measurement conditions, calibration tables and data traces
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	An error occurs. Error code 205.
<i>Preset Value</i>	"CST"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.MMEMory.STORe.STYPe app.SCPI.MMEMory.STORe.STYPe = "STATe"
<i>Equivalent Softkeys</i>	<b>None</b>



**SCPI.OUTPUT.STATE**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Instrument
<i>Description</i>	Sets the ON/OFF state of the stimulus signal output. Measurements cannot be performed when the stimulus signal output is set to OFF.
<i>Allowable Values</i>	True: Stimulus signal output ON False: Stimulus signal output OFF
<i>Preset Value</i>	True
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = app.SCPI.OUTPUT.STATE app.SCPI.OUTPUT.STATE = False
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Power &gt; RF Out</b>

**SCPI.SENSE(*Ch*).AVERage.CLEAr**

<i>Object Type</i>	Method
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Resets the averaging data count to 0. Restarts the averaging process.
<i>Syntax</i>	<code>app.SCPI.SENSE(<i>Ch</i>).AVERage.CLEAr</code>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SENSE(*Ch*).AVERAge.COUNT**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Long
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The averaging factor, when the averaging function is set to ON by the SCPI.SENSE( <i>Ch</i> ).AVERAge.STATe property.
<i>Range</i>	from 1 to 999
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	10
<i>Syntax</i>	Dim <i>Value</i> As Long <i>Value</i> = app.SCPI.SENSE( <i>Ch</i> ).AVERAge.COUNT app.SCPI.SENSE( <i>Ch</i> ).AVERAge.COUNT = 2
<i>Equivalent Softkeys</i>	<b>Response &gt; Avg Factor</b>

**SCPI.SENSE(*Ch*).AVERAge.CURRent**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Long
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The current step of averaging.
<i>Range</i>	from 1 to Averaging Factor
<i>Syntax</i>	Dim <i>Value</i> As Long <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).AVERAge.CURRent
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SENSE(*Ch*).AVERAge.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the averaging function.
<i>Allowable Values</i>	True: Averaging ON False: Averaging OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).AVERAge.STATe <i>app</i> .SCPI.SENSE( <i>Ch</i> ).AVERAge.STATe = False
<i>Equivalent Softkeys</i>	<b>Response &gt; Averaging</b>

**SCPI.SENSE(*Ch*).BANDwidth.RESolution**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Channel <i>Ch</i> , <i>Ch:</i> channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The IF bandwidth value.
<i>Range</i>	from 10 to 30000
<i>Resolution</i>	In steps of 3. (10, 30, 100, 300, 1000, 3000, 10000, 30000)
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	10000
<i>Unit</i>	Hz (Hertz)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).BANDwidth.RESolution <i>app</i> .SCPI.SENSE( <i>Ch</i> ).BANDwidth.RESolution = 100
<i>Equivalent Softkeys</i>	<b>Response &gt; IF Bandwidth</b>

**SCPI.SENSE(*Ch*).CORRection.CLEAr**

<i>Object Type</i>	Method
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Clears the calibration coefficient table.
<i>Syntax</i>	<code>app.SCPI.SENSE(<i>Ch</i>).CORRection.CLEAr</code>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SENSE(*Ch*).CORRection.COEFFicient.DATA(*Str*, *Pt\_r*, *Pt\_s*)**

<i>Object Type</i>	Property (read)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	<p>The calibration coefficient data array set by the type of the corrected error <i>Str</i>, the number of the receiver port <i>Pt_r</i> and the number of the source port <i>Pt_s</i>,</p> <p><i>Str</i> : error type (see below)  <i>Pt_r</i> : the number of the receiver port 1–2 (see Table 2 on page 21)  <i>Pt_s</i> : the number of the source port 1–2 (see Table 2 on page 21)</p> <p>The array size is 2N, where N is the number of measurement points.  For the n–th point, where n from 1 to N:</p> <p><i>Data</i>(2n–2) real part of the calibration coefficients  <i>Data</i>(2n–1) imaginary part of the calibration coefficients</p>
<i>Parameter</i>	<p>String <i>Str</i> – corrected error type:  "ES": Source match  "ER": Reflection tracking  "ED": Directivity  "EL": Load match  "ET": Transmission tracking  "EX": Isolation</p> <p>When ES, ER, or ED is used, the numbers of the ports <i>Pt_r</i> and <i>Pt_s</i> must be the same. When EL, ET, or EX is used, the numbers of the ports <i>Pt_r</i> and <i>Pt_s</i> must be different.</p>
<i>Syntax</i>	<p>Dim <i>Data</i> As Variant</p> <p><i>Data</i> = app.SCPI.SENSE(<i>Ch</i>).CORRection.COEFFicient.DATA (<i>Str</i>, <i>Pt_r</i>, <i>Pt_s</i>)</p>
<i>Notes</i>	The written calibration coefficients become effective only after the SCPI.SENSE( <i>Ch</i> ).CORRection.COEFFicient.SAVE method is invoked.
<i>Equivalent Softkeys</i>	<b>None</b>



**SCPI.SENSE(*Ch*).CORRection.COLLEct.ACQuire.LOAD**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	Long
<i>Target</i>	Channel <i>Ch</i> , <i>Ch:</i> channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Measures the calibration data of the <i>load</i> standard for the specified port.
<i>Range</i>	Port number is 1 or 2.
<i>Out of Range</i>	An error occurs (error code: 222).
<i>Syntax</i>	<code>app.SCPI.SENSE(<i>Ch</i>).CORRection.COLLEct.ACQuire.LOAD = 1</code>
<i>Notes</i>	The property writing starts the measurement for the channel independently of the trigger initiation and trigger source settings. The function of the property writing waits for the completion of the measurement.
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Response (Open) &gt; Load (Optional)</b> <b>Calibration &gt; Response (Short) &gt; Load (Optional)</b> <b>Calibration &gt; Full 1–Port Cal &gt; Load</b> <b>Calibration &gt; One Path 2–Port Cal &gt; Load</b>

**SCPI.SENSE(*Ch*).CORRection.COLLEct.ACQuire.OPEN**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	Long
<i>Target</i>	Channel <i>Ch</i> , <i>Ch:</i> channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Measures the calibration data of the <i>open</i> standard for the specified port.
<i>Range</i>	Port number is 1 or 2.
<i>Out of Range</i>	An error occurs (error code: 222).
<i>Syntax</i>	<code>app.SCPI.SENSE(<i>Ch</i>).CORRection.COLLEct.ACQuire.OPEN= 1</code>
<i>Notes</i>	The property writing starts the measurement for the channel independently of the trigger initiation and trigger source settings. The function of the property writing waits for the completion of the measurement.
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Response (Open) &gt; Open</b> <b>Calibration &gt; Full 1–Port Cal &gt; Open</b> <b>Calibration &gt; One Path 2–Port Cal &gt; Open</b>

**SCPI.SENSE(*Ch*).CORRection.COLLEct.ACQuire.SHORt**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	Long
<i>Target</i>	Channel <i>Ch</i> , <i>Ch:</i> channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Measures the calibration data of the <i>short</i> standard for the specified port.
<i>Range</i>	Port number is 1 or 2.
<i>Out of Range</i>	An error occurs (error code: 222).
<i>Syntax</i>	<code>app.SCPI.SENSE(<i>Ch</i>).CORRection.COLLEct.ACQuire.SHORt = 1</code>
<i>Notes</i>	The property writing starts the measurement for the channel independently of the trigger initiation and trigger source settings. The function of the property writing waits for the completion of the measurement.
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Response (Short) &gt; Short</b> <b>Calibration &gt; Full 1–Port Cal &gt; Short</b> <b>Calibration &gt; One Path 2–Port Cal &gt; Short</b>

**SCPI.SENSE(*Ch*).CORRection.COLLEct.ACQuire.THRU**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	Variant (Long array)
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Measures the calibration data of the <i>thru</i> standard between the source port and the receiver port. The array contains 2 elements: <i>Data</i> (0)        the number of the receiver port; <i>Data</i> (1)        the number of the source port.
<i>Range</i>	Port number is 1 or 2. The array elements can not contain the same port numbers.
<i>Out of Range</i>	If an incorrect port number is specified, an error occurs (error code: 222). If the same port numbers are specified, an error occurs (error code: 220).
<i>Syntax</i>	<code>app.SCPi.SENSE(<i>Ch</i>).CORRection.COLLEct.ACQuire.THRU= Array(1, 2)</code>
<i>Notes</i>	The property writing starts the measurement for the channel independently of the trigger initiation and trigger source settings. The function of the property writing waits for the completion of the measurement.
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Response (Thru) &gt; Thru</b> <b>Calibration &gt; One Path 2–Port Cal &gt; Thru</b>

**SCPI.SENSE(*Ch*).CORRection.COLLEct.CKIT.LABel**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Calibration kit, selected for channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The calibration kit label.
<i>Range</i>	up to 254 characters
<i>Preset Value</i>	Varies depending on the number of the calibration kit. 1: Not Def 50Ohm 2: Not Def 75Ohm 3: 05CK10A-150 –F– 4: 05CK10A-150 –M– 5: N1.1 Type-N –F– 6: N1.1 Type-N –M– 7: Agil. 85032B –F– 8: Agil. 85032B –M– 9: Agil. 85036B –F– 10: Agil. 85036B –M– 11: N611 –F– 12: N611 –M– 13: Empty 14: Empty 15: Empty 16: Empty
<i>Syntax</i>	Dim <i>Lab</i> As String <i>Lab</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLEct.CKIT.LABel <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLEct.CKIT.LABel = "User1"
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Edit Cal Kit &gt; Label</b>

**SCPI.SENSE(*Ch*).CORRection.COLLEct.CKIT.RESet**

<i>Object Type</i>	Method
<i>Target</i>	Calibration kit, selected for channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Resets the calibration kit to the factory settings.
<i>Syntax</i>	<code>app.SCPI.SENSE(<i>Ch</i>).CORRection.COLLEct.CKIT.RESet</code>
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Restore Cal Kit</b>

**SCPI.SENSE(*Ch*).CORRection.COLLEct.CKIT.SELect**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Long
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The selected calibration kit for the channel.
<i>Range</i>	from 1 to 16
<i>Out of Range</i>	An error occurs. Error code: 222.
<i>Preset Value</i>	1
<i>Syntax</i>	Dim <i>Value</i> As Long <i>Value</i> = <code>app.SCPI.SENSE(<i>Ch</i>).CORRection.COLLEct.CKIT.SELect</code> <code>app.SCPI.SENSE(<i>Ch</i>).CORRection.COLLEct.CKIT.SELect</code> = 3
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Cal Kit &gt; Cal Kit n</b>

**Table 8. Std: Calibration Standard Number**

<i>Data Type</i>	Long
<i>Description</i>	The number of the standard.
<i>Range</i>	Varies depending on the number of the standards in the calibration kit.
<i>Out of Range</i>	If the specified standard number is greater than the number of standards in the kit, an error occurs (error code: 222).
<i>Notes</i>	If the standard number is not specified, it is taken as equal to 1.

**SCPI.SENSE(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).C0**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Standard <i>Std</i> of the calibration kit specified for channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Std</i> : standard number (see Table 8 on page 191)
<i>Description</i>	The C0 value of the open calibration standard.
<i>Range</i>	from $-1\text{E}18$ to $1\text{E}18$
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	Varies depending on the selected calibration kit and the standard.
<i>Unit</i>	$1\text{E}-15$ F (Farad)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLect.CKIT.STAN( <i>Std</i> ).C0 <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLect.CKIT.STAN( <i>Std</i> ).C0 = 100
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Edit Cal Kit &gt; Capacitance &gt; C0 [<math>10^{-15}</math> F]</b>



**SCPI.SENSE(*Ch*).CORRection.COLLEct.CKIT.STAN(*Std*).C1**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Standard <i>Std</i> of the calibration kit specified for channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Std</i> : standard number (see Table 8 on page 191)
<i>Description</i>	The C1 value of the open calibration standard.
<i>Range</i>	from –1E18 to 1E18
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	Varies depending on the selected calibration kit and the standard.
<i>Unit</i>	1E–27 F/Hz (Farad/Hertz)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLEct.CKIT.STAN( <i>Std</i> ).C1 <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLEct.CKIT.STAN( <i>Std</i> ).C1 = 100
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Edit Cal Kit &gt; Capacitance &gt; C1 [10<sup>-27</sup> F/Hz]</b>

**SCPI.SENSE(*Ch*).CORRection.COLLection.CKIT.STAN(*Std*).C2**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Standard <i>Std</i> of the calibration kit specified for channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Std</i> : standard number (see Table 8 on page 191)
<i>Description</i>	The C2 value of the open calibration standard.
<i>Range</i>	from –1E18 to 1E18
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	Varies depending on the selected calibration kit and the standard.
<i>Unit</i>	1E–36 F/Hz <sup>2</sup> (Farad/Hertz <sup>2</sup> )
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLection.CKIT.STAN( <i>Std</i> ).C2 <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLection.CKIT.STAN( <i>Std</i> ).C2 = 100
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Edit Cal Kit &gt; Capacitance &gt; C2 [10<sup>–36</sup> F/Hz<sup>2</sup>]</b>

**SCPI.SENSE(*Ch*).CORRection.COLLection.CKIT.STAN(*Std*).C3**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Standard <i>Std</i> of the calibration kit specified for channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Std</i> : standard number (see Table 8 on page 191)
<i>Description</i>	The C3 value of the open calibration standard.
<i>Range</i>	from –1E18 to 1E18
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	Varies depending on the selected calibration kit and the standard.
<i>Unit</i>	1E–45 F/Hz <sup>3</sup> (Farad/Hertz <sup>3</sup> )
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLection.CKIT.STAN( <i>Std</i> ).C3 <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLection.CKIT.STAN( <i>Std</i> ).C3 = 100
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Edit Cal Kit &gt; Capacitance &gt; C3 [10<sup>–45</sup> F/Hz<sup>3</sup>]</b>

**SCPI.SENSE(*Ch*).CORRection.COLLeCt.CKIT.STAN(*Std*).DELay**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Standard <i>Std</i> of the calibration kit specified for channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Std</i> : standard number (see Table 8 on page 191)
<i>Description</i>	The offset delay value of the calibration standard.
<i>Range</i>	from –1E18 to 1E18
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	Varies depending on the selected calibration kit and the standard.
<i>Unit</i>	s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLeCt.CKIT.STAN( <i>Std</i> ).DELay <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLeCt.CKIT.STAN( <i>Std</i> ).DELay = 93E–12
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Edit Cal Kit &gt; Offset Delay</b>

**SCPI.SENSE(*Ch*).CORRection.COLLect.CKIT.STAN(*Std*).L0**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Standard <i>Std</i> of the calibration kit specified for channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Std</i> : standard number (see Table 8 on page 191)
<i>Description</i>	The L0 value of the short calibration standard.
<i>Range</i>	from –1E18 to 1E18
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	Varies depending on the selected calibration kit and the standard.
<i>Unit</i>	1E–12 H (Henry)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLect.CKIT.STAN( <i>Std</i> ).L0 <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLect.CKIT.STAN( <i>Std</i> ).L0 = 100
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Edit Cal Kit &gt; Inductance &gt; L0 [10<sup>-12</sup> H]</b>

**SCPI.SENSE(*Ch*).CORRection.COLLeCt.CKIT.STAN(*Std*).L1**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Standard <i>Std</i> of the calibration kit specified for channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Std</i> : standard number (see Table 8 on page 191)
<i>Description</i>	The L1 value of the short calibration standard.
<i>Range</i>	from –1E18 to 1E18
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	Varies depending on the selected calibration kit and the standard.
<i>Unit</i>	1E–24 H/Hz (Henry/Hertz)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLeCt.CKIT.STAN( <i>Std</i> ).L1 <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLeCt.CKIT.STAN( <i>Std</i> ).L1 = 100
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Edit Cal Kit &gt; Inductance &gt; L1 [10<sup>-24</sup> H/Hz]</b>

**SCPI.SENSE(*Ch*).CORRection.COLLeCt.CKIT.STAN(*Std*).L2**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Standard <i>Std</i> of the calibration kit specified for channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Std</i> : standard number (see Table 8 on page 191)
<i>Description</i>	The L2 value of the short calibration standard.
<i>Range</i>	from –1E18 to 1E18
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	Varies depending on the selected calibration kit and the standard.
<i>Unit</i>	1E–33 H/Hz <sup>2</sup> (Henry/Hertz <sup>2</sup> )
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLeCt.CKIT.STAN( <i>Std</i> ).L2 <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLeCt.CKIT.STAN( <i>Std</i> ).L2 = 100
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Edit Cal Kit &gt; Inductance &gt; L2 [10<sup>–33</sup> H/Hz<sup>2</sup>]</b>

**SCPI.SENSE(*Ch*).CORRection.COLLection.CKIT.STAN(*Std*).L3**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Standard <i>Std</i> of the calibration kit specified for channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Std</i> : standard number (see Table 8 on page 191)
<i>Description</i>	The L3 value of the short calibration standard.
<i>Range</i>	from –1E18 to 1E18
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	Varies depending on the selected calibration kit and the standard.
<i>Unit</i>	1E–42 H/Hz <sup>3</sup> (Henry/Hertz <sup>3</sup> )
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLection.CKIT.STAN( <i>Std</i> ).L3 <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLection.CKIT.STAN( <i>Std</i> ).L3 = 100
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Edit Cal Kit &gt; Inductance &gt; L3 [10<sup>–42</sup> H/Hz<sup>3</sup>]</b>



**SCPI.SENSE(*Ch*).CORRection.COLLEct.CKIT.STAN(*Std*).LABel**

<i>Object Type</i>	Property (read)
<i>Data Type</i>	String
<i>Target</i>	Standard <i>Std</i> of the calibration kit specified for channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Std</i> : standard number (see Table 8 on page 191)
<i>Description</i>	The label of the calibration standard.
<i>Range</i>	up to 254 characters
<i>Preset Value</i>	Varies depending on the selected calibration kit and the standard.
<i>Syntax</i>	Dim <i>Lab</i> As String <i>Lab</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLEct.CKIT.STAN( <i>Std</i> ).LABel
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SENSE(*Ch*).CORRection.COLLEct.CKIT.STAN(*Std*).LOSS**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Standard <i>Std</i> of the calibration kit specified for channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Std</i> : standard number (see Table 8 on page 191)
<i>Description</i>	The offset loss value of the calibration standard.
<i>Range</i>	from –1E18 to 1E18
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	Varies depending on the selected calibration kit and the standard.
<i>Unit</i>	$\Omega/s$ (Ohm/second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLEct.CKIT.STAN( <i>Std</i> ).LOSS <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLEct.CKIT.STAN( <i>Std</i> ).LOSS = 700E6
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Edit Cal Kit &gt; Offset Loss</b>

**SCPI.SENSE(*Ch*).CORRection.COLLEct.CKIT.STAN(*Std*).TYPE**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	String
<i>Target</i>	Standard <i>Std</i> of the calibration kit specified for channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Std</i> : standard number (see Table 8 on page 191)
<i>Description</i>	The type of the calibration standard.
<i>Range</i>	"OPEN" : Open "SHORt" : Short "LOAD" : Load "THRU" : Thru
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLEct.CKIT.STAN( <i>Std</i> ).TYPE
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Edit Cal Kit &gt; STD Type</b>

**SCPI.SENSE(*Ch*).CORRection.COLLection.CKIT.STAN(*Std*).Z0**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Standard <i>Std</i> of the calibration kit specified for channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Std</i> : standard number (see Table 8 on page 191)
<i>Description</i>	The offset Z0 value of the calibration standard.
<i>Range</i>	from –1E18 to 1E18
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	50 or 75, depending on the selected calibration kit.
<i>Unit</i>	$\Omega$ (Ohm)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLection.CKIT.STAN( <i>Std</i> ).Z0 <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLection.CKIT.STAN( <i>Std</i> ).Z0 = 50
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Edit Cal Kit &gt; Offset Z0</b>

**SCPI.SENSE(*Ch*).CORRection.COLLEct.CLEAr**

<i>Object Type</i>	Method
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Clears the measurement values of the calibration standards.
<i>Syntax</i>	<code>app.SCPI.SENSE(<i>Ch</i>).CORRection.COLLEct.CLEAr</code>
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Calibrate &gt; Response (Open)   Response (Short)   Response (Thru)   One Path 2–Port Cal   Full 1–Port Cal &gt; Cancel</b>

**SCPI.SENSE(Ch).CORRection.COLLEct.ECAL.INFOrmation**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	String
<i>Target</i>	AutoCal module ( <i>Ch</i> – arbitrary number 1 to 9)
<i>Description</i>	<p>Gets information about the AutoCal Module connected to the Network Analyzer in a string with comma separated fields.</p> <p>Autocal Module Information:</p> <ul style="list-style-type: none"> <li>Model Name,</li> <li>Serial Number,</li> <li>Current Temperature of AutoCal Module,</li> </ul> <p>Selected Characterization Information:</p> <ul style="list-style-type: none"> <li>Characterization Name,</li> <li>Characterization Date and Time,</li> <li>Min Frequency,</li> <li>Max Frequency,</li> <li>Number of Points,</li> <li>Characterization Temperature,</li> <li>PortA Connector,</li> <li>PortB Connector,</li> <li>PortA Adapter,</li> <li>PortB Adapter,</li> <li>Analyzer,</li> <li>Location,</li> <li>Operator.</li> </ul>
<i>Syntax</i>	<p>Dim <i>ID</i> As String</p> <p><i>ID</i> = app.SCPi.SENSE(<i>Ch</i>).CORRection.COLLEct.ECAL.INFOrmation</p>
<i>Equivalent Softkeys</i>	<b>Calibration &gt; AutoCal &gt; Characterization Info</b>

**SCPI.SENSE(*Ch*).CORRection.COLLect.ECAL.ORIentation.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	AutoCal ( <i>Ch</i> – arbitrary number 1 to 9)
<i>Description</i>	The ON/OFF state of the Auto-Orientation function used when executing AutoCal.
<i>Allowable Values</i>	True: Auto-Orientation function ON False: Auto-Orientation function OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = app.SCPI.SENSE( <i>Ch</i> ).CORRection.COLLect.ECAL.ORIentation.STATe app.SCPI.SENSE( <i>Ch</i> ).CORRection.COLLect.ECAL.ORIentation.STATe = False
<i>Equivalent Softkeys</i>	<b>Calibration &gt; AutoCal &gt; Orientation &gt; Auto</b>

**SCPI.SENSE(*Ch*).CORRection.COLLEct.ECAL.PATH(*Pt*)**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Long
<i>Target</i>	AutoCal ( <i>Ch</i> – arbitrary number 1 to 9)
<i>Description</i>	Sets or reads out the AutoCal module port number which is connected to a selected Network Analyzer port ( <i>Pt</i> ).
<i>Allowable Values</i>	1: Port A of AutoCal Module 2: Port B of AutoCal Module
<i>Syntax</i>	Dim <i>Value</i> As Long <i>Value</i> = app.SCPI.SENSE( <i>Ch</i> ).CORRection.COLLEct.ECAL.PATH( <i>Pt</i> ) app.SCPI.SENSE( <i>Ch</i> ).CORRection.COLLEct.ECAL.PATH( <i>Pt</i> ) = 2
<i>Equivalent Softkeys</i>	<b>Calibration &gt; AutoCal &gt; Orientation &gt; Port 1 - Port A   Port 1 - Port B</b>



**SCPI.SENSE(*Ch*).CORRection.COLLection.ECAL.SOLT1**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	Long
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 22)
<i>Description</i>	Executes 1-port calibration of the specified port of selected channel ( <i>Ch</i> ) using the AutoCal module.
<i>Syntax</i>	Dim <i>Port</i> As Long <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLection.ECAL.SOLT1 = <i>Port</i>
<i>Equivalent Softkeys</i>	<b>Calibration &gt; AutoCal &gt; 1-Port AutoCal   2-Port AutoCal</b>

**SCPI.SENSE(*Ch*).CORRection.COLLEct.ECAL.UCHar**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	AutoCal ( <i>Ch</i> – arbitrary number 1 to 9)
<i>Description</i>	Sets or reads out the Characteristic used when executing AutoCal (factory or user characterization).
<i>Range</i>	"CHAR0" : Factory characterization "CHAR1" : User characterization 1 "CHAR2" : User characterization 2 "CHAR3" : User characterization 3
<i>Preset Value</i>	CHAR0
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.SENSE( <i>Ch</i> ).CORRection.COLLEct.ECAL.UCHar app.SCPI.SENSE( <i>Ch</i> ).CORRection.COLLEct.ECAL.UCHar = "CHAR0"
<i>Equivalent Softkeys</i>	<b>Calibration &gt; AutoCal &gt; Characterization &gt; Factory   User 1   User 2   User 3</b>

**SCPI.SENSE(*Ch*).CORRection.COLLEct.METHod.ERESponse**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	Variant (Long array)
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Selects the ports and sets the <i>one path 2–port calibration</i> type for the calculation of the calibration coefficients on completion of the calibration executed by the SCPI.SENSE( <i>Ch</i> ).CORRection.COLLEct.SAVE method.  The array contains 2 elements:  <i>Data</i> (0)          the number of the receiver port;  <i>Data</i> (1)          the number of the source port.
<i>Range</i>	Port number is 1 or 2. The array elements can not contain the same port numbers.
<i>Out of Range</i>	If an incorrect port number is specified, an error occurs (error code: 222). If the same port numbers are specified, an error occurs (error code: 220).
<i>Syntax</i>	<code>app.SCPI.SENSE(<i>Ch</i>).CORRection.COLLEct.METHod.ERESponse = Array(2, 1)</code>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SENSE(*Ch*).CORRection.COLLeCt.METHod.RESPonse.OPEN**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	Long
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Selects the port and sets the <i>response calibration (Open)</i> type for the calculation of the calibration coefficients on completion of the calibration executed by the SCPI.SENSE( <i>Ch</i> ).CORRection.COLLeCt.SAVE method.
<i>Range</i>	from 1 to 2
<i>Out of Range</i>	An error occurs. Error code: 222.
<i>Syntax</i>	<code>app.SCPI.SENSE(<i>Ch</i>).CORRection.COLLeCt.METHod.RESPonse.OPEN = 1</code>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SENSE(*Ch*).CORRection.COLLEct.METHod.RESPonse.SHORt**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	Long
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Selects the port and sets the <i>response calibration (Short)</i> type for the calculation of the calibration coefficients on completion of the calibration executed by the SCPI.SENSE( <i>Ch</i> ).CORRection.COLLEct.SAVE method.
<i>Range</i>	from 1 to 2
<i>Out of Range</i>	An error occurs. Error code: 222.
<i>Syntax</i>	<code>app.SCPI.SENSE(<i>Ch</i>).CORRection.COLLEct.METHod.RESPonse.SHORt = 1</code>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SENSE(*Ch*).CORRection.COLLEct.METHod.RESPonse.THRU**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	Variant (Long array)
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	<p>Selects the ports and sets the <i>response calibration (Thru)</i> type for the calculation of the calibration coefficients on completion of the calibration executed by the SCPI.SENSE(<i>Ch</i>).CORRection.COLLEct.SAVE method.</p> <p>The array contains 2 elements:</p> <p style="padding-left: 40px;"><i>Data(0)</i> the number of the receiver port;</p> <p style="padding-left: 40px;"><i>Data(1)</i> the number of the source port.</p>
<i>Range</i>	Port number is 1 or 2. Array elements can not contain the same port numbers.
<i>Out of Range</i>	If an incorrect port number is specified, an error occurs (error code: 222). If the same port numbers are specified, an error occurs (error code: 220).
<i>Syntax</i>	<code>app.SCPI.SENSE(<i>Ch</i>).CORRection.COLLEct.METHod.RESPonse.THRU = Array(2, 1)</code>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SENSE(*Ch*).CORRection.COLLeCt.METHod.SOLT1**

<i>Object Type</i>	Property (write only)
<i>Data Type</i>	Long
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Selects the port and sets the <i>full 1–port calibration</i> type for the calculation of the calibration coefficients on completion of the calibration executed by the SCPI.SENSE( <i>Ch</i> ).CORRection.COLLeCt.SAVE method.
<i>Range</i>	from 1 to 2
<i>Out of Range</i>	An error occurs. Error code: 222.
<i>Syntax</i>	<code>app.SCPI.SENSE(<i>Ch</i>).CORRection.COLLeCt.METHod.SOLT1 = 1</code>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SENSE(*Ch*).CORRection.COLLEct.METHod.TYPE**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	String
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The calibration type selected for calculating of the calibration coefficients on completion of the calibration executed by the SCPI.SENSE( <i>Ch</i> ).CORRection.COLLEct.SAVE method.
<i>Range</i>	"RESPO" : Response (Open) "RESPS" : Response (Short) "RESPT" : Response (Thru) "SOLT1" : Full 1–port calibration "1PATH" : One path 2–port calibration "NONE" : Not defined
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.SENSE( <i>Ch</i> ).CORRection.COLLEct.METHod.TYPE
<i>Equivalent Softkeys</i>	<b>None</b>



**SCPI.SENSE(*Ch*).CORRection.COLLect.SAVE**

<i>Object Type</i>	Method
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	<p>Calculates the calibration coefficients from the calibration standards measurements depending on the selected calibration type.</p> <p>On completion of the method, all the calibration standards measurements are cleared and the error correction automatically turns ON.</p> <p>At the attempt to execute this method before all the needed standards are measured, an error occurs and the method is ignored.</p>
<i>Syntax</i>	<i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.COLLect.SAVE
<i>Related Commands</i>	<p>Calibration type selection:</p> <p>SCPI.SENSE(<i>Ch</i>).CORRection.COLLect.METHod.RESPonse.OPEN  SCPI.SENSE(<i>Ch</i>).CORRection.COLLect.METHod.RESPonse.SHORT  SCPI.SENSE(<i>Ch</i>).CORRection.COLLect.METHod.RESPonse.THURU  SCPI.SENSE(<i>Ch</i>).CORRection.COLLect.METHod.ERESponse  SCPI.SENSE(<i>Ch</i>).CORRection.COLLect.METHod.SOLT1</p> <p>Calibration standards measurement:</p> <p>SCPI.SENSE(<i>Ch</i>).CORRection.COLLect.ACQUIRE.LOAD  SCPI.SENSE(<i>Ch</i>).CORRection.COLLect.ACQUIRE.OPEN  SCPI.SENSE(<i>Ch</i>).CORRection.COLLect.ACQUIRE.SHORT  SCPI.SENSE(<i>Ch</i>).CORRection.COLLect.ACQUIRE.THURU</p>
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Response (Open)   Response (Short)   Response (Thru)   One Path 2–Port Cal   Full 1–Port Cal &gt; Apply</b>

**SCPI.SENSE(*Ch*).CORRection.EXTension.PORT(*Pt*).FREQUency(*Ls*)**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Port <i>Pt</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Pt</i> : port number 1–2 (see Table 2 on page 21)
<i>Description</i>	The value of the frequency at the point number <i>Ls</i> for calculation of the loss for the port extension function, <i>Ls</i> : point number 1–2 (see Table 9 on page 218)
<i>Range</i>	from 2E4 to 4.8E9
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	1e9
<i>Unit</i>	Hz (Hertz)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.PORT( <i>Pt</i> ).FREQUency( <i>Ls</i> ) <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.PORT( <i>Pt</i> ).FREQUency( <i>Ls</i> ) = 100E6
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Port Extensions &gt; Loss &gt; Freq1   Freq2</b>

**Table 9. *Ls*: point number**

<i>Data Type</i>	Long
<i>Description</i>	Point number for setting the losses in the functions extension of port
<i>Range</i>	1 or 2
<i>Out of Range</i>	An error occurs. Error code: 222

**SCPI.SENSE(*Ch*).CORRection.EXTension.PORT(*Pt*).INCLude(*Ls*).STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Port <i>Pt</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Pt</i> : port number 1–2 (see Table 2 on page 21)
<i>Description</i>	The ON/OFF state of the loss compensation at the point number <i>Ls</i> to calculate the loss for the port extension function, <i>Ls</i> : point number 1–2 (see Table 9 on page 218).
<i>Allowable Values</i>	True: Loss compensation ON False: Loss compensation OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = app.SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.PORT( <i>Pt</i> ).INCLude( <i>Ls</i> ).STATe app.SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.PORT( <i>Pt</i> ).INCLude( <i>Ls</i> ).STATe = True
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Port Extensions &gt; Loss &gt; Loss1   Loss2</b>

**SCPI.SENSE(*Ch*).CORRection.EXTension.PORT(*Pt*).LDC**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Port <i>Pt</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Pt</i> : port number 1–2 (see Table 2 on page 21)
<i>Description</i>	The loss value at DC for the loss calculation of the port extension function.
<i>Range</i>	from -200 to 200
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	0
<i>Unit</i>	dB (decibel)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.PORT( <i>Pt</i> ).LDC <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.PORT( <i>Pt</i> ).LDC = 10
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Port Extensions &gt; Loss &gt; Loss at DC</b>

**SCPI.SENSE(*Ch*).CORRection.EXTension.PORT(*Pt*).LOSS(*Ls*)**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Port <i>Pt</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Pt</i> : port number 1–2 (see Table 2 on page 21)
<i>Description</i>	The loss value at the point number <i>Ls</i> to calculate the loss for the port extension function, <i>Ls</i> : point number 1–2 (see Table 9 on page 218)
<i>Range</i>	from -200 to 200
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	0
<i>Unit</i>	dB (decibel)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.PORT( <i>Pt</i> ).LOSS( <i>Ls</i> ) <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.PORT( <i>Pt</i> ).LOSS( <i>Ls</i> ) = 10
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Port Extensions &gt; Loss &gt; Loss 1   Loss 2</b>

**SCPI.SENSE(*Ch*).CORRection.EXTension.PORT(*Pt*).TIME**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Port <i>Pt</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Pt</i> : port number 1–2 (see Table 2 on page 21)
<i>Description</i>	The electrical delay value for the port extension function.
<i>Range</i>	from -10 to 10
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	0
<i>Unit</i>	s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.PORT( <i>Pt</i> ).TIME <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.PORT( <i>Pt</i> ).TIME = 1e-9
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Port Extensions &gt; Extension Port1   Extension Port2</b>

**SCPI.SENSE(*Ch*).CORRection.EXTension.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the port extension function.
<i>Allowable Values</i>	True: Port extension function ON False: Port extension function OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.STATe <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.EXTension.STATe = True
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Port Extensions &gt; Extension</b>

**SCPI.SENSE(Ch).CORRection.IMPedance.INPut.MAGNitude**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Instrument
<i>Description</i>	The system impedance Z0.
<i>Range</i>	from 0.001 to 1000
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	50
<i>Unit</i>	$\Omega$ (Ohm)
<i>Syntax</i>	Dim Value As Double Value = app.SCPI.SENSE.CORRection.IMPedance.INPut.MAGNitude app.SCPI.SENSE.CORRection.IMPedance.INPut.MAGNitude = 75
<i>Equivalent Softkeys</i>	<b>Calibration &gt; System Z0</b>



**SCPI.SENSE(*Ch*).CORRection.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the error correction.
<i>Allowable Values</i>	True: Error correction ON False: Error correction OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.STATe <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.STATe = True
<i>Equivalent Softkeys</i>	<b>Calibration &gt; Correction</b>

**SCPI.SENSE(*Ch*).CORRection.TRANSform.TIME.FREQuency**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The Value of Frequency specified for Cable Loss in Time Domain measurements.
<i>Range</i>	from 1.0 to 10000.0
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	1000.0
<i>Unit</i>	MHz
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.TRANSform.TIME.FREQuency <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.TRANSform.TIME.FREQuency = 2000.0
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Time Domain &gt; Cable Correction &gt; Frequency</b>

**SCPI.SENSE(*Ch*).CORRection.TRANSform.TIME.LOSS**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Channel <i>Ch</i> , <i>Ch:</i> channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The Value of Cable Loss for Time Domain measurements.
<i>Range</i>	from 0.0 to 1000.0
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	0.0
<i>Unit</i>	dB/m
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.TRANSform.TIME.LOSS <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.TRANSform.TIME.LOSS = 0.25
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Time Domain &gt; Cable Correction &gt; Cable Loss</b>

**SCPI.SENSE(*Ch*).CORRection.TRANSform.TIME.RVELOCITY**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Channel <i>Ch</i> , <i>Ch:</i> channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The Value of Velocity Factor for Time Domain measurements.
<i>Range</i>	from 0.01 to 1.0
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	1.0
<i>Unit</i>	None
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.TRANSform.TIME.RVELOCITY <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.TRANSform.TIME.RVELOCITY = 0.68
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Time Domain &gt; Cable Correction &gt; Velocity Factor</b>

**SCPI.SENSE(*Ch*).CORRection.TRANSform.TIME.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	The channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of Cable Loss correction in Time Domain measurements.
<i>Allowable Values</i>	True: Cable Loss correction ON False: Cable Loss correction OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.TRANSform.TIME.STATe <i>app</i> .SCPI.SENSE( <i>Ch</i> ).CORRection.TRANSform.TIME.STATe = True
<i>Equivalent Softkeys</i>	<b>Analysis &gt; Time Domain &gt; Cable Correction &gt; Cable Correction</b>

**SCPI.SENSE(*Ch*).CORRection.TYPE(*Tr*)**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Variant array)
<i>Target</i>	Trace <i>Tr</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Tr</i> : trace number 1–8 (see Table 3 on page 29)
<i>Description</i>	The information about the applied calibration type and the port numbers for the specified trace. The array contains 3 elements: <i>Data</i> (0) calibration type (see below); <i>Data</i> (1) the number of the receiver port to be calibrated; <i>Data</i> (2) the number of the source port to be calibrated.
<i>Range</i>	Calibration type in the element <i>Data</i> (0): "RESPO" : Response (Open) "RESPS" : Response (Short) "RESPT" : Response (Thru) "SOLT1" : Full 1–port calibration "1PATH" : One path 2–port calibration "NONE" : Not defined
<i>Syntax</i>	Dim <i>CallInfo</i> As Variant <i>CallInfo</i> = app.SCPI.SENSE( <i>Ch</i> ).CORRection.TYPE( <i>Tr</i> )
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SENSE(*Ch*).FREQUENCY.CENTER**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The stimulus center value of the sweep range for linear or logarithmic sweep types.
<i>Range</i>	From the minimum value to the maximum value of the analyzer frequency range.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	The center value of the analyzer frequency range.
<i>Unit</i>	Hz (Hertz)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).FREQUENCY.CENTER <i>app</i> .SCPI.SENSE( <i>Ch</i> ).FREQUENCY.CENTER = 1E9
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Center</b>

**SCPI.SENSE(*Ch*).FREQUENCY.CW**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The value of the fixed frequency for the power sweep. Before this method is called, the sweep type must be set to power sweep, otherwise the method is ignored.
<i>Range</i>	From the minimum value to the maximum value of the analyzer frequency range.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	The center value of the analyzer frequency range.
<i>Unit</i>	Hz (Hertz)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).FREQUENCY.CW <i>app</i> .SCPI.SENSE( <i>Ch</i> ).FREQUENCY.CW = 1E9
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Power &gt; CW Freq</b>



**SCPI.SENSE(*Ch*).FREQUENCY.DATA**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The array of the measurement points frequency for linear, logarithmic or segment sweep type. The array size is N, where N is the number of measurement points. For the n–th point, where n from 1 to N: $Data(n-1)$ the frequency value at the n–th measurement point.
<i>Syntax</i>	Dim <i>Data</i> As Variant <i>Data</i> = app. SCPI.SENSE( <i>Ch</i> ).FREQUENCY.DATA
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SENSE(*Ch*).FREQUENCY.SPAN**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The stimulus span value of the sweep range for linear or logarithmic sweep types.
<i>Range</i>	From 0 to the maximum span of the analyzer frequency range.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	The maximum span of the analyzer frequency range.
<i>Unit</i>	Hz (Hertz)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).FREQUENCY.SPAN <i>app</i> .SCPI.SENSE( <i>Ch</i> ).FREQUENCY.SPAN = 1E9
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Span</b>

**SCPI.SENSE(*Ch*).FREQUENCY.START**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The stimulus start value of the sweep range for linear or logarithmic sweep types.
<i>Range</i>	From the minimum value to the maximum value of the analyzer frequency range.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	The minimum value of the analyzer frequency range.
<i>Unit</i>	Hz (Hertz)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).FREQUENCY.START <i>app</i> .SCPI.SENSE( <i>Ch</i> ).FREQUENCY.START = 1E6
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Start</b>

**SCPI.SENSE(*Ch*).FREQUENCY.STOP**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The stimulus stop value of the sweep range for linear or logarithmic sweep types.
<i>Range</i>	From the minimum value to the maximum value of the analyzer frequency range.
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	The maximum value of the analyzer frequency range.
<i>Unit</i>	Hz (Hertz)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).FREQUENCY.STOP <i>app</i> .SCPI.SENSE( <i>Ch</i> ).FREQUENCY.STOP = 1E8
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Stop</b>

**SCPI.SENSE(*Ch*).ROSCillator.SOURce**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Selects the internal or external source of the reference frequency of 10 MHz.
<i>Range</i>	"INTernal" : Internal source of the reference frequency "EXTernal" : External source of the reference frequency
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	An error occurs. Error code: 224.
<i>Preset Value</i>	"INT"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.SENSE( <i>Ch</i> ).ROSCillator.SOURce app.SCPI.SENSE( <i>Ch</i> ).ROSCillator.SOURce = "EXT"
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SENSE(*Ch*).SEGMENT.DATA**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	<p>The array of the segment sweep table.</p> <p>The array has the following format:</p> <pre>{ &lt;Buf&gt;, &lt;Flag1&gt;, &lt;Flag2&gt;, &lt;Flag3&gt;, &lt;Flag4&gt;, &lt;Flag5&gt;, &lt;N&gt;,   &lt;Start(1)&gt;, &lt;Stop(1)&gt;, &lt;NOP(1)&gt; [, &lt;IFBW(1)&gt;] [, &lt;Pow(1)&gt;] [, &lt;Del(1)&gt;] [, &lt;Time(1)&gt;],   &lt;Start(2)&gt;, &lt;Stop(2)&gt;, &lt;NOP(2)&gt; [, &lt;IFBW(2)&gt;] [, &lt;Pow(2)&gt;] [, &lt;Del(2)&gt;] [, &lt;Time(2)&gt;],   ...   &lt;Start(N)&gt;, &lt;Stop(N)&gt;, &lt;NOP(N)&gt; [, &lt;IFBW(N)&gt;] [, &lt;Pow(N)&gt;] [, &lt;Del(N)&gt;] [, &lt;Time(N)&gt;] }</pre> <p>&lt;Buf&gt; : Always 5,      &lt;Flag1&gt; : Stimulus start setting (0 – start/stop, 1 – center/span),      &lt;Flag2&gt; : Setting of the &lt;IFBW&gt; field (0 – disabled, 1 – enabled),      &lt;Flag3&gt; : Setting of the &lt;Pow&gt; field (0 – disabled, 1 – enabled),      &lt;Flag4&gt; : Setting of the &lt;Del&gt; field (0 – disabled, 1 – enabled),      &lt;Flag5&gt; : Setting of the &lt;Time&gt; field (0 – disabled, 1 – enabled),      &lt;N&gt; : Number of segments,      &lt;Start <i>n</i>&gt; : Start value of the <i>n</i>–th segment,      &lt;Stop <i>n</i>&gt; : Stop value of the <i>n</i>–th segment,      &lt;NOP <i>n</i>&gt; : Number of points of the <i>n</i>–th segment,      &lt;IFBW <i>n</i>&gt; : IF bandwidth of the <i>n</i>–th segment (if enabled),      &lt;Pow <i>n</i>&gt; : Power of the <i>n</i>–th segment (if enabled),      &lt;Del <i>n</i>&gt; : Measurement delay of the <i>n</i>–th segment (if enabled),      &lt;Time <i>n</i>&gt; : Reserved for future use (if enabled).</p>
<i>Syntax</i>	<p>Dim <i>Data</i> As Variant</p> <p><i>Data</i> = app.SCPi.SENSE(<i>Ch</i>).SEGMENT.DATA</p> <p>app.SCPi.SENSE(<i>Ch</i>).SEGMENT.DATA = <i>Data</i></p>
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Segment Table</b>

**SCPI.SENSE(*Ch*).SWEp.POINT.TIME**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The value of the delay before measurement in each measurement point.
<i>Range</i>	from 0 to 0.3
<i>Resolution</i>	5E-6
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	0
<i>Unit</i>	s (second)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).SWEp.POINT.TIME <i>app</i> .SCPI.SENSE( <i>Ch</i> ).SWEp.POINT.TIME = 5E-6
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Meas Delay</b>

**SCPI.SENSE(*Ch*).SWEep.POINts**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Long
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The number of measurement points.
<i>Range</i>	from 2 to 16001 (Planar TR1300) from 2 to 200001 (TR5048, TR7530)
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	201
<i>Syntax</i>	Dim <i>Value</i> As Long <i>Value</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).SWEep.POINts <i>app</i> .SCPI.SENSE( <i>Ch</i> ).SWEep.POINts = 1001
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Points</b>



**SCPI.SENSE(*Ch*).SWEep.REVerse.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The ON/OFF state of the reverse scan.
<i>Allowable Values</i>	True: Reserse Scan ON False: Reverse Scan OFF
<i>Preset Value</i>	"False"
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.SENSE( <i>Ch</i> ).SWEep.REVerse.STATe <i>app</i> .SCPI.SENSE( <i>Ch</i> ).SWEep.REVerse.STATe = True
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Sweep Type &gt; Reverse Scan</b>

**SCPI.SENSE(*Ch*).SWEep.TYPE**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	Sets the sweep type.
<i>Range</i>	"LINear" : Linear frequency sweep "LOGarithmic" : Logarithmic frequency sweep "SEGMENT" : Segment frequency sweep "POWER " : Power sweep
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	An error occurs. Error code: 206.
<i>Preset Value</i>	"LIN"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.SENSE( <i>Ch</i> ).SWEep.TYPE app.SCPI.SENSE( <i>Ch</i> ).SWEep.TYPE = "LOG"
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Sweep Type</b>

**SCPI.SERVICE.CHANnel(*Ch*).ACTIVE**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Long
<i>Target</i>	Instrument
<i>Description</i>	The number of the active channel.
<i>Syntax</i>	Dim <i>Value</i> As Long <i>Value</i> = <i>app</i> .SCPI.SERVICE.CHANnel(1).ACTIVE
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SERVICE.CHANnel(*Ch*).COUNT**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Long
<i>Target</i>	Instrument
<i>Description</i>	The maximum number of the channels.
<i>Syntax</i>	Dim <i>Value</i> As Long <i>Value</i> = <i>app</i> .SCPI.SERVICE.CHANnel(1).COUNT
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SERVICE.CHANNEL(*Ch*).TRACE.ACTIVE**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Long
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The active trace number of the channel.
<i>Syntax</i>	Dim <i>Value</i> As Long <i>Value</i> = <i>app</i> .SCPI.SERVICE.CHANNEL( <i>Ch</i> ).TRACE.ACTIVE
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SERVICE.CHANNEL(*Ch*).TRACE.COUNT**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Long
<i>Target</i>	Instrument
<i>Description</i>	The maximum number of the traces in the channel.
<i>Syntax</i>	Dim <i>Value</i> As Long <i>Value</i> = <i>app</i> .SCPI.SERVICE.CHANNEL(1).TRACE.COUNT
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SERVICE.PORT.COUNT**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Long
<i>Target</i>	Instrument
<i>Description</i>	The number of the ports.
<i>Syntax</i>	Dim <i>Value</i> As Long <i>Value</i> = app.SCPI.SERVICE.PORT.COUNT
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SERVICE.SWEep.FREQuency.MAXimum**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Double
<i>Target</i>	Instrument
<i>Description</i>	The upper limit of the measurement frequency.
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = app.SCPI.SERVICE.SWEep.FREQuency.MAXimum
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SERVICE.SWEep.FREqency.MINimum**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Double
<i>Target</i>	Instrument
<i>Description</i>	The lower limit of the measurement frequency.
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SERVICE.SWEep.FREqency.MINimum
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SERVICE.SWEep.POINts**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Double
<i>Target</i>	Instrument
<i>Description</i>	The maximum number of the measurement points.
<i>Syntax</i>	Dim <i>Value</i> As Long <i>Value</i> = <i>app</i> .SCPI.SERVICE.SWEep.POINts
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SOURce(Ch).POWer.CENTer**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The center value of the power sweep range.
<i>Range</i>	From the minimum value to the maximum value of the analyzer power range
<i>Resolution</i>	0.05
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	Center value of the analyzer power range.
<i>Unit</i>	dBm (decibels above 1 milliwatt)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SOURce( <i>Ch</i> ).POWer.CENTer <i>app</i> .SCPI.SOURce( <i>Ch</i> ).POWer.CENTer = -5
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Center</b>

**SCPI.SOURce(Ch).POWer.LEVel.IMMEDIATE.AMPLitude**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The power level for the frequency sweep.
<i>Range</i>	From the minimum value to the maximum value of the analyzer power range
<i>Resolution</i>	0.05
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	0
<i>Unit</i>	dBm (decibels above 1 milliwatt)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = app.SCPi.SOURce( <i>Ch</i> ).POWer.LEVel.IMMEDIATE.AMPLitude app.SCPi.SOURce( <i>Ch</i> ).POWer.LEVel.IMMEDIATE.AMPLitude = -10
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Power &gt; Output Power</b>



**SCPI.SOURce(Ch).POWer.LEVel.SLOPe.DATA**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Channel <i>Ch</i> , <i>Ch:</i> channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The power slope value for the frequency sweep.
<i>Range</i>	from 0 to 2
<i>Resolution</i>	0.05
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	0
<i>Unit</i>	dB/GHz (decibel/gigahertz)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SOURce( <i>Ch</i> ).POWer.LEVel.SLOPe.DATA <i>app</i> .SCPI.SOURce( <i>Ch</i> ).POWer.LEVel.SLOPe.DATA = 0.2
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Power &gt; Power Slope</b>

**SCPI.SOURce(Ch).POWer.PORT(Pt).CORRection.DATA**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Variant (Double array)
<i>Target</i>	Port <i>Pt</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Pt</i> : port number 1–2 (see Table 2 on page 21)
<i>Description</i>	The power correction table. The array size is 1+2N, where N is the number of the table rows. For the n–th point, where n from 1 to N: <i>Data</i> (0) the number of the table rows N integer from 0 to 10001; <i>Data</i> (2n–1) the frequency of the n–th row of the table; <i>Data</i> (2n) power correction value of the n–th row of the table.
<i>Syntax</i>	Dim <i>Data</i> As Variant <i>Data</i> = app.SCPI.SOURce(Ch).POWer.PORT(Pt).CORRection.DATA
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SOURce(Ch).POWER.PORT(Pt).CORRection.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Port <i>Pt</i> of channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21) <i>Pt</i> : port number 1–2 (see Table 2 on page 21)
<i>Description</i>	The ON/OFF state of the power correction function.
<i>Allowable Values</i>	True: Power correction ON False: Power correction OFF
<i>Preset Value</i>	True
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.SOURce( <i>Ch</i> ).POWER.PORT( <i>Pt</i> ).CORRection.STATe <i>app</i> .SCPI.SOURce( <i>Ch</i> ).POWER.PORT( <i>Pt</i> ).CORRection.STATe = True
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SOURce(Ch).POWer.SPAN**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Channel <i>Ch</i> , <i>Ch:</i> channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The power span for the power sweep.
<i>Range</i>	From 0 to the maximum span of the analyzer power range.
<i>Resolution</i>	0.05
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	The maximum span of the analyzer power range.
<i>Unit</i>	dBm (decibels above 1 milliwatt)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SOURce( <i>Ch</i> ).POWer.SPAN <i>app</i> .SCPI.SOURce( <i>Ch</i> ).POWer.SPAN = 50
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Span</b>

**SCPI.SOURce(Ch).POWER.START**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Channel <i>Ch</i> , <i>Ch:</i> channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The power sweep start for the power sweep.
<i>Range</i>	From the minimum value to the maximum value of the analyzer power range.
<i>Resolution</i>	0.05
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	The minimum value of the analyzer power range.
<i>Unit</i>	dBm (decibels above 1 milliwatt)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SOURce( <i>Ch</i> ).POWER.START <i>app</i> .SCPI.SOURce( <i>Ch</i> ).POWER.START = -5
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Start</b>

**SCPI.SOURce(Ch).POWER.STOP**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Double
<i>Target</i>	Channel <i>Ch</i> , <i>Ch</i> : channel number 1–9 (see Table 1 on page 21)
<i>Description</i>	The power sweep stop for the power sweep.
<i>Range</i>	From the minimum value to the maximum value of the analyzer power range.
<i>Resolution</i>	0.05
<i>Out of Range</i>	Sets the value of the limit, which is closer to the specified value.
<i>Preset Value</i>	The maximum value of the analyzer power range.
<i>Unit</i>	dBm (decibels above 1 milliwatt)
<i>Syntax</i>	Dim <i>Value</i> As Double <i>Value</i> = <i>app</i> .SCPI.SOURce( <i>Ch</i> ).POWER.STOP <i>app</i> .SCPI.SOURce( <i>Ch</i> ).POWER.STOP = 0
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Stop</b>

**SCPI.SYSTem.COMMunicate.ECAL.TEMPerature.SENSor**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	Double
<i>Target</i>	AutoCal module
<i>Description</i>	Reads out the temperature of the AutoCal module connected to the instrument.
<i>Unit</i>	°C (Celsius)
<i>Syntax</i>	Dim Value As Double Value = app.SCPi.SYSTem.COMMunicate.ECAL.TEMPerature.SENSor
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SYSTem.CORRection.STATe**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Instrument
<i>Description</i>	The ON/OFF state of the system error correction.
<i>Allowable Values</i>	True: System error correction ON False: System error correction OFF
<i>Preset Value</i>	True
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.SYSTem.CORRection.STATe <i>app</i> .SCPI.SYSTem.CORRection.STATe = False
<i>Equivalent Softkeys</i>	<b>None</b>



**SCPI.SYSTem.DATE**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Variant (Long array)
<i>Target</i>	Instrument
<i>Description</i>	<p>The current date.</p> <p>The array consists of three elements:</p> <p><i>Data(0)</i>      year from 1900 to 2100;</p> <p><i>Data(1)</i>      month from 1 to 12;</p> <p><i>Data(2)</i>      day from 1 to 31.</p>
<i>Syntax</i>	<p>Dim <i>Data</i> As Variant</p> <p><i>Data</i> = <i>app</i>.SCPI.SYSTem.DATE</p> <p><i>app</i>. <i>app</i>.SCPI.SYSTem.DATE = Array(2009, 9, 9)</p>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SYSTem.HIDe**

<i>Object Type</i>	Method
<i>Target</i>	Instrument
<i>Description</i>	Make the main form of a application invisible.
<i>Syntax</i>	<i>app.SCPi.SYSTem.HIDe</i>
<i>Related Commands</i>	<i>SCPI.SYSTem.SHOW</i>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SYSTem.LOCal**

<i>Object Type</i>	Method
<i>Target</i>	Instrument
<i>Description</i>	Sets the instrument to the local operation mode, when all the keys on the front panel, mouse and the touch screen are active.
<i>Syntax</i>	<i>app</i> .SCPI.SYSTem.LOCal
<i>Related Commands</i>	SCPI.SYSTem.RWLock
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SYSTem.PRESet**

<i>Object Type</i>	Method
<i>Target</i>	Instrument
<i>Description</i>	Resets the instrument to the factory settings. The difference from the SCPI.IEEE4882.RST: method is that the trigger is set to the <i>Continuous</i> trigger mode.
<i>Syntax</i>	<code>app.SCPi.SYSTem.PRESet</code>
<i>Equivalent Softkeys</i>	<b>System &gt; Preset &gt; Apply</b>

**SCPI.SYSTem.REMote**

<i>Object Type</i>	Method
<i>Target</i>	Instrument
<i>Description</i>	Sets the instrument to the remote operation mode, when all the keys on the front panel, mouse and the touch screen are not active, except for one key labeled Return to Local. Pushing this button will reset the instrument to the local operation mode.
<i>Syntax</i>	<i>app.</i> SCPI.SYSTem.REMote
<i>Related Commands</i>	<i>SCPI.SYSTem.LOCal</i>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SYSTem.RWLock**

<i>Object Type</i>	Method
<i>Target</i>	Instrument
<i>Description</i>	Sets the instrument to the remote operation mode, when all the keys on the front panel, mouse and the touch screen are not active. Only SCPI.SYSTem.LOCal or SCPI.SYSTem.REMote command can release this remote operation mode.
<i>Syntax</i>	<i>app.</i> SCPI.SYSTem.RWLock
<i>Related Commands</i>	SCPI.SYSTem.LOCal SCPI.SYSTem.REMote
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SYSTem.SHOW**

<i>Object Type</i>	Method
<i>Target</i>	Instrument
<i>Description</i>	Return the main form of a application to visible state.
<i>Syntax</i>	<i>app.</i> SCPI.SYSTem.SHOW
<i>Related Commands</i>	<i>SCPI.SYSTem.HIDe</i>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.SYSem.TERMinate**

<i>Object Type</i>	Method
<i>Target</i>	Instrument
<i>Description</i>	Close the COM server application.
<i>Syntax</i>	<i>app.</i> SCPI.SYSem.TERMinate
<i>Equivalent Softkeys</i>	<b>System &gt; Exit &gt; Ok</b>



**SCPI.SYSTem.TIME**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Variant (Long array)
<i>Target</i>	Instrument
<i>Description</i>	<p>The current time.</p> <p>The array consists of three elements:</p> <p style="padding-left: 40px;"><i>Data(0)</i>        hours from 0 to 23;</p> <p style="padding-left: 40px;"><i>Data(1)</i>        minutes from 0 to 59;</p> <p style="padding-left: 40px;"><i>Data(2)</i>        seconds from 0 to 59.</p>
<i>Syntax</i>	<p>Dim <i>Data</i> As Variant</p> <p><i>Data</i> = app.SCPI.SYSTem.TIME</p> <p>app. app.SCPI.SYSTem.TIME = Array(15, 20, 30)</p>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.TRIGger.INPut.POLarity**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Instrument
<i>Description</i>	Selects the polarity of the external trigger.
<i>Range</i>	"POSitive" : Positive edge "NEGative" : Negative edge
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	An error occurs. Error code: 205.
<i>Preset Value</i>	"NEG"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.TRIGger.INPut.POLarity app.SCPI.TRIGger.INPut.POLarity = "POS"
<i>Related Commands</i>	SCPI.TRIGger.SEQuence.SOURce
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Trigger &gt; Ext Trig Polarity &gt; Negative edge   Positive edge</b>

**SCPI.TRIGger.SEQuence.IMMediate**

<i>Object Type</i>	Method
<i>Target</i>	Instrument
<i>Description</i>	<p>Generates a trigger, independently of the trigger source setting (except for the <i>External</i>). If the trigger source is set to <i>External</i>, an error occurs (error code 221) and the command is ignored.</p> <p>If the instrument is not in the waiting for a trigger state (sweep is in progress or all the channels are set to <i>Hold</i>), an error occurs (error code 211) and the command is ignored.</p> <p>The method returns control before the end of the sweep.</p>
<i>Syntax</i>	<code>app.SCPI.TRIGger.SEQuence.IMMediate</code>
<i>Related Commands</i>	<p>SCPI.TRIGger.SEQuence.SOURce  SCPI.INITiate(<i>Ch</i>).CONTInuous  SCPI.INITiate(<i>Ch</i>).IMMediate</p>
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.TRIGger.SEQuence.POINt**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	Boolean
<i>Target</i>	Instrument
<i>Description</i>	The ON/OFF state of the point trigger feature. When the point trigger feature is turned ON, the external trigger response is single point. When the point trigger feature is turned OFF, the external trigger response is single sweep.
<i>Allowable Values</i>	True: point trigger feature ON False: point trigger feature OFF
<i>Preset Value</i>	False
<i>Syntax</i>	Dim <i>Status</i> As Boolean <i>Status</i> = <i>app</i> .SCPI.TRIGger.SEQuence.POINt <i>app</i> .SCPI.TRIGger.SEQuence.POINt = True
<i>Related Commands</i>	SCPI.TRIGger.SEQuence.SOURce
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Trigger &gt; Ext Trig Event &gt; On Sweep   On Point</b>

**SCPI.TRIGger.SEQuence.SINGle**

<i>Object Type</i>	Method
<i>Target</i>	Instrument
<i>Description</i>	<p>Generates a trigger, independently of the trigger source setting (except for the <i>External</i>). If the trigger source is set to <i>External</i>, an error occurs (error code 221) and the command is ignored.</p> <p>If the instrument is not in the waiting for a trigger state (sweep is in progress or all the channels are set to <i>Hold</i>), an error occurs (error code 211) and the command is ignored.</p> <p>The method does not return control before the end of the sweep (waiting for the completion of the sweep of all the channels).</p>
<i>Syntax</i>	<i>app</i> .SCPI.TRIGger.SEQuence.SINGle
<i>Related Commands</i>	SCPI.TRIGger.SEQuence.SOURce SCPI.INITiate( <i>Ch</i> ).CONTinuous SCPI.INITiate( <i>Ch</i> ).IMMEDIATE
<i>Equivalent Softkeys</i>	<b>None</b>

**SCPI.TRIGger.SEQuence.SOURce**

<i>Object Type</i>	Property (read/write)
<i>Data Type</i>	String
<i>Target</i>	Instrument
<i>Description</i>	Selects the sweep trigger source.
<i>Range</i>	"INTernal" : Internal "EXTernal" : External "BUS" : Bus
<i>Notes</i>	The short format of the parameter is indicated by upper case letters. There is no distinction between upper and lower case letters when the property is written. When the property is read out, the short format is indicated by upper case letters.
<i>Out of Range</i>	An error occurs. Error code: 205.
<i>Preset Value</i>	"INT"
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = app.SCPI.TRIGger.SEQuence.SOURce app.SCPI.TRIGger.SEQuence.SOURce = "BUS"
<i>Related Commands</i>	SCPI.TRIGger.SEQuence.IMMediate SCPI.TRIGger.SEQuence.SINGle SCPI.IEEE4882.TRG
<i>Equivalent Softkeys</i>	<b>Stimulus &gt; Trigger &gt; Trigger Source &gt; Internal   External</b>

**SCPI.TRIGger.SEQence.STATe**

<i>Object Type</i>	Property (read only)
<i>Data Type</i>	String
<i>Target</i>	Instrument
<i>Description</i>	Returns the status of the scanner.
<i>Range</i>	"HOLD" : scanning is stopped "WAIT" : waiting for the start "MEAS" : is scanning
<i>Notes</i>	The short format of the parameter is indicated by upper case letters.
<i>Out of Range</i>	An error occurs. Error code: 200.
<i>Syntax</i>	Dim <i>Param</i> As String <i>Param</i> = <i>app</i> .SCPI.TRIGger.SEQence.STATe
<i>Related Commands</i>	SCPI.TRIGger.SEQence.IMMediate SCPI.TRIGger.SEQence.SINGle SCPI.IEEE4882.TRG SCPI.IEEE4882.RST
<i>Equivalent Softkeys</i>	<b>None</b>

## Appendix 1. Error Codes

114	"Header suffix out of range"
200	"Execution error"
211	"Trigger ignored"
213	"Init ignored"
220	"Parameter Error"
222	"Data out of range"
224	"Illegal parameter value"
201	"Invalid channel index"
202	"Invalid trace index"
203	"Invalid marker index"
204	"Marker is not active"
205	"Invalid save type specifier"
206	"Invalid sweep type specifier"
207	"Invalid trigger source specifier"
208	"Invalid measurement parameter specifier"
209	"Invalid format specifier"
210	"Invalid data math specifier"
214	"Invalid limit data"
215	"Invalid segment data"
216	"Invalid standard type specifier"
217	"Invalid conversion specifier"
218	"Invalid gating shape specifier"
219	"Invalid gating type specifier"
300	"Device-specific error"
302	"Status reporting system error"



### Example 1. Instrument Information String Readout

The following program reads out and displays on the screen the instrument information string – the Name property of the COM object. The string contains the following fields:

*Manufacturer, Model, Serial Number, Software Version/Firmware Version*

For example:

Copper Mountain Technologies, TR5048, 00000001, 0.7.9/ 2.0

```
Dim app As Object
Sub Example1()
Set app = CreateObject("TR5048.Application")
ID = app.Name
MsgBox ("Information string read out: " + ID)
End Sub
```

## Example 2. Checking the Instrument Ready State

Normally, the user control program starts when the analyzer executable module is running, the instrument booting is completed, and the instrument is ready for use. In some cases, it is recommended to check if the instrument is ready for use. The instrument may be not ready for use if it is not connected to PC via USB cable. Moreover, if the analyzer executable module has not been started in advance, the *CreateObject* function will automatically start the application and then within about 10 seconds the instrument booting will be in progress. The instrument will not be ready for use until the booting is completed. The *Ready* property is used to check if the instrument is ready for use.

The following program checks the *Ready* property right after a COM object has been created. If the analyzer executable module has been started in advance and the booting is completed, “*Analyzer is ready*” will be displayed. If the *Ready* property value is *False*, 10 second delay is activated for the case the analyzer executable module has been started by the COM object creation. In 10 seconds the program rechecks the *Ready* property. If the value is *True*, “*Analyzer is ready*” will be displayed, if otherwise, “*Analyzer is not ready*” will be displayed, what means the instrument is not turned on or it is not connected to PC via USB cable.

```
Dim app As Object
Sub Example2()
Set app = CreateObject("TR5048.Application")
If app.Ready = False Then
    Application.Wait (Now + TimeValue("0:00:10"))
    If app.Ready = False Then
        MsgBox ("Analyzer is not ready")
        Exit Sub
    End If
End If
MsgBox ("Analyzer is ready")
End Sub
```

### Example 3. Setting the Measurement Parameters

The following program shows the setting of some measurement parameters. First, the instrument is reset to the factory settings. Then the following parameters are set:

- Two channel windows are opened and allocated one above the other.
- The number of traces is set to 2 in the first channel window.
- For the first channel the stimulus parameters are set as follows: the frequency range from 100 MHz to 1.2 GHz, the number of measurement points 401.
- For the second channel the stimulus parameters are set as follows: the frequency range from 800 MHz to 900 MHz, the number of points 51, IF bandwidth 100 Hz, stimulus power -10 dBm.
- In the first channel window: S11 measurement is set for the trace 1, S21 measurement is set for the trace 2. The Smith chart format is set for the both traces.
- In the second channel window: S21 measurement and logarithmic magnitude format are set for the single trace. Then the auto scale function is called for this trace.

```
Dim app As Object

Public Sub Example3()
Set app = CreateObject("TR5048.Application")

app.SCPI.SYSTem.PRESet

app.SCPI.DISPlay.Split = 2
app.SCPI.Calculate(1).Parameter.Count = 2

app.SCPI.SENSE(1).Frequency.Start = 100000000
app.SCPI.SENSE(1).Frequency.STOP = 1200000000
app.SCPI.SENSE(1).SWEep.Points = 401

app.SCPI.SENSE(2).Frequency.Start = 800000000
app.SCPI.SENSE(2).Frequency.STOP = 900000000
app.SCPI.SENSE(2).SWEep.Points = 51
app.SCPI.SENSE(2).BANDwidth.RESolution = 100
app.SCPI.Source(2).Power.LEVel.IMMediate.AMPlitude = -10

app.SCPI.Calculate(1).Parameter(1).DEFine = "S11"
app.SCPI.Calculate(1).Parameter(2).DEFine = "S21"
app.SCPI.Calculate(1).Parameter(1).Select
app.SCPI.Calculate(1).Selected.Format = "SMIT"
app.SCPI.Calculate(1).Parameter(2).Select
app.SCPI.Calculate(1).Selected.Format = "SMIT"
```

```
app.SCPI.Calculate(2).Parameter(1).DEFine = "S21"  
app.SCPI.Calculate(2).Parameter(1).Select  
app.SCPI.Calculate(2).Selected.Format = "MLOG"  
app.SCPI.DISPlay.Window(2).TRACe(1).Y.SCALe.AUTO  
  
End Sub
```

#### Example 4. Measurement Data Acquisition

The following program shows data array acquisition with further writing into a file. The program also shows the method of a sweep triggering and waiting for the sweep completion.

Three variables  $F$ ,  $M$ ,  $P$  are declared in the second string of the code. They are used for arrays of frequency values (Hz), magnitude values (dB), and phase values (degree) respectively.

After the instrument has been reset to the factory settings, two operators are used for the sweep triggering and waiting for the sweep completion:

```
app.SCPI.TRIGger.SEquence.Source = "BUS"  
app.SCPI.TRIGger.SEquence.Single
```

The first operator sets the LAN bus command or the COM/DCOM interface command as a trigger source. It aborts the sweep and switches the instrument to waiting for a trigger. The second operator is used for a new sweep triggering and waiting for the sweep completion.

---

Note	Unlike the <i>SCPI.TRIGger.SEquence.IMMEDIATE</i> and <i>SCPI.IEEE4882.TRG</i> commands, which are completed immediately after a trigger generation, the <i>SCPI.TRIGger.SEquence.Single</i> command is not completed until the end of the sweep. Using the <i>SCPI.TRIGger.SEquence.Single</i> command is the simplest way to set the waiting for the sweep completion.
------	--

---

On completion of the sweep, three arrays are read out: frequency values, magnitude values and phase values. Before the magnitude and phase arrays are read out, the corresponding trace format is set.

The array size of frequency  $F$  is equal to the number of measurement points, and the array size of magnitude  $M$  and phase  $P$  is equal to the double number of measurement points (see section 11 “Measurement Data Arrays”). In rectangular formats (for magnitude and phase) the measurement data are real numbers located in even cells of the array. Odd cells of the array contain 0.

On completion of the program, the frequency, magnitude and phase values for each measurement point are written string by string into the file named *TESTFILE*.

```

Dim app As Object
Dim F, M, P

Public Sub Example4()
Set app = CreateObject("TR5048.Application")

app.SCPI.SYSTEM.PRESet

app.SCPI.TRIGger.SEQuence.Source = "BUS"
app.SCPI.TRIGger.SEQuence.Single

F = app.SCPI.SENSE(1).Frequency.Data

app.SCPI.Calculate(1).Selected.Format = "MLOG"
M = app.SCPI.Calculate(1).Selected.Data.FDATA

app.SCPI.Calculate(1).Selected.Format = "PHASe"
P = app.SCPI.Calculate(1).Selected.Data.FDATA

Open "TESTFILE" For Output As #1

For i = LBound(F) To UBound(F)
    Print #1, F(i), M(i * 2), P(i * 2)
Next i

Close #1
End Sub

```

## Example 5. Program Written in C++

The following C++ program represents an example of the measurement parameter setting, as well as acquisition and display of the measurement data array. The program also shows a method of the sweep triggering and waiting for the sweep completion.

```
//-----  
// Simple example of using COM object of TR5048.exe application.  
// This example is console application. GUI is not used in this example to  
// simplify the program. Error processing is very restricted too.  
#include "stdafx.h"  
//-----  
// Generate description of COM object of TR5048.exe application.  
#import "TR5048.exe" no_namespace  
//-----  
int _tmain(int argc, _TCHAR* argv[])  
{  
    ITR5048Ptr pNWA;           // Pointer to COM object of TR5048.exe  
    CComVariant Data;         // Variable for measurement data  
    // Init COM subsystem  
    HRESULT hr = CoInitialize(NULL);  
    if(hr != S_OK) return -1;  
    // Create COM object  
    hr = pNWA.CreateInstance(__uuidof(TR5048));  
    if(hr != S_OK) return -1;  
    // Preset network analyzer  
    pNWA->SCPI->SYSTEM->PRESet();  
    // Set frequency start to 1 GHz  
    pNWA->SCPI->SENSE[1]->FREQUENCY->START = 1e8;  
    // Set frequency stop to 1.2 GHz  
    pNWA->SCPI->SENSE[1]->FREQUENCY->STOP = 1.2e8;  
    // Set number of measurement points to 51  
    pNWA->SCPI->SENSE[1]->SWEEP->POINTS = 51;  
    // Set measured parameter to S21  
    pNWA->SCPI->CALCULATE[1]->PARAMETER[1]->DEFINE = "S21";  
    // Set trigger source to GPIB/LAN bus or COM interface  
    pNWA->SCPI->TRIGGER->SEQUENCE->SOURCE = "bus";  
    // Trigger measurement and wait  
    pNWA->SCPI->TRIGGER->SEQUENCE->SINGLE();  
    // Get measurement data (array of complex numbers)  
    Data = pNWA->SCPI->CALCULATE[1]->SELECTED->DATA->FDATA;  
    // Display measurement data.  
    // Data is array of NOP * 2 (number of measurement points).  
    // Where n is an integer between 0 and NOP - 1.  
    // Data(n*2) : Primary value at the n-th measurement point.  
    // Data(n*2+1) : Secondary value at the n-th measurement point. Always 0  
    // when the data format is not the Smith chart or the polar.  
    CComSafeArray<double> mSafeArray;  
    if (mSafeArray.Attach(Data.parray) == S_OK)  
    {  
        for (unsigned int n = 0; n < mSafeArray.GetCount() / 2; ++n)  
        {  
            printf("%.9E\t%.9E\n",  
                mSafeArray.GetAt(n*2),  
                mSafeArray.GetAt(n*2+1));  
        }  
        mSafeArray.Detach();  
    }  
    printf("Press ENTER to exit.\n");  
    getc(stdin);  
    // Release COM object  
    pNWA.Release();  
    CoUninitialize();  
    return 0;  
}
```