



MEDAQLib

Micro-Epsilon
Data Acquisition Library

Application programming
interface (API) for
accessing digital sensors

Sensors

IFD2401	ILD1302	ILR110x	DT3100
IFD2431	ILD1320	ILR115x	
IFD2445	ILD1401	ILR118x	DT6100
IFD2451	ILD1402	ILR1191	DT6120
IFD2461	ILD1420		DT62xx
IFD2471	ILD17xx	ODC12xx	KSS6380
	ILD22xx	ODC2500	DT65xx
ACS7000	ILD23xx	ODC2520	
		ODC2600	

Interfaces

RS232	IF2008	IF2001/USB	CSP2008
TCP/IP	IF2004		C-Box
USB	IF2004/USB		

MEDAQLib

Micro-Epsilon Data Acqusition Library
X9751165

Application programming interface (API) for accessing digital sensors
V 4.0.2.25475

MICRO-EPSILON
MESSTECHNIK
GmbH & Co. KG
Königbacher Strasse 15

D-94496 Ortenburg

Tel. +49/8542/168-0
Fax +49/8542/168-90
e-mail info@micro-epsilon.de
www.micro-epsilon.de

Contents

1 Introduction	38
2 Installation	40
3 Accessing MEDAQLib in Visual C/C++	42
3.1 Setting up Visual Studio	42
3.1.1 Using MEDAQLib.lib, static approach	42
3.1.2 Using MEDAQLib.dll, dynamic approach	42
4 Using MEDAQLib	43
4.1 Main structure of a MEDAQLib program	43
4.2 How to call a function at sensor	44
4.3 Main structure in a LabView Sample	44
4.4 Synchronized data from two ILD2300 at IF2008	45
4.5 Block based data acquisition vs Poll	47
4.5.1 Block based data acquisition	47
4.5.2 Polling	47
4.5.3 Additional hints	47
5 Samples	48
5.1 C-Sharp Example	48
5.2 Delphi Example	48
5.3 DLL Example	48
5.4 LabView	48
5.5 Lib Example	48
5.6 IF2008 Example	48
5.7 Unicode Example	48
5.8 VBA Example	49
5.9 VB2013Example	49
5.10 X64 Example	49
5.11 SensorFinder	49
5.12 SensorTest	49
6 Function Reference	50
6.1 Create a sensor instance	50
6.2 Releasing sensor instance	53
6.3 Set parameters	53
6.4 Get parameters	60
6.5 Clear internal parameter buffer	69
6.6 Connecting to the sensor	69
6.7 Closing connection to sensor	70
6.8 Sending commands to the sensor	70
6.9 Polling data from sensor	71
6.10 Number of values available to read	73
6.11 Block wise data acquisition from sensor	73
6.12 Get additional error information	76
6.13 Get version of MEDAQLib dll	78
6.14 EnableLogging wrapper function	79
6.15 User logging functions	83
6.16 OpenSensor wrapper functions	85
6.17 ExecSCmd wrapper functions	92

7 Parameters	102
7.1 Naming conventions	102
7.2 Interface parameters	102
7.2.1 All Interfaces	102
7.2.2 RS232	108
7.2.3 IF2004	110
7.2.4 IF2004_USB	112
7.2.5 IF2008	115
7.2.6 TCP/IP	117
7.2.7 WinUSB	119
7.3 Sensor parameters	120
8 Sensor commands	121
8.1 Communication via SensorCommand	121
8.2 Sensor commands valid for each sensor	123
8.2.1 Enable_Logging	124
8.2.2 Automatic_Setup	126
8.2.3 Get_TransmittedDataInfo	126
8.2.4 Use_Defaults	128
8.2.5 SettingsChanged	128
8.2.6 Get_DrvSetting	128
8.2.7 Cmd_Generic	128
8.2.8 Clear_Buffers	129
8.2.9 DataAvail_Event	129
8.2.10 Open_DataSocket	130
8.2.11 Set_Setup	131
8.2.12 Get_Setup	132
8.2.13 Update_... (Meta command)	133
8.3 SensorFinder commands	134
8.3.1 Start_FindSensor	136
8.3.2 Get_FindSensorProgress	139
8.3.3 Get_FoundSensor	139
8.3.4 Abort_FindSensor	148
8.3.5 Set_IPConfig	148
9 Commands for ILR sensors	152
9.1 Commands for ILR110x_115x	152
9.1.1 General commands	152
9.1.1.1 General	152
9.1.1.1.1 Get_Parameters (GAP)	152
9.1.1.1.2 Get_Energy (GDB)	156
9.1.1.1.3 Get_SerialNbr (GNR)	156
9.1.1.1.4 Get_ErrorStatus (GSI)	156
9.1.1.1.5 Get_Temperature (GTE)	156
9.1.1.1.6 Get_Version (GVE)	157
9.1.1.1.7 Set_Stand-by (ISB)	157
9.1.1.1.8 Set_VisibleLaser (IVL)	157
9.1.1.2 Triggering	157
9.1.1.2.1 Set_ContinousMode (ICM)	157
9.1.1.2.2 Exec_ContMeasure (ECM)	157
9.1.1.2.3 Trg_SingleMeasure (ESM)	158
9.1.1.3 Parameter management	158
9.1.1.3.1 Save_Parameters (EPW)	158

9.1.2	Measurement	158
9.1.2.1	General	158
9.1.2.1.1	Set_Offset (IDO)	158
9.1.3	Data output	158
9.1.3.1	Switching outputs	158
9.1.3.1.1	Set_HysteresisQ1 (IH1)	158
9.1.3.1.2	Set_HysteresisQ2 (IH2)	158
9.1.3.1.3	Set_LimitQ1-1 (IL1)	159
9.1.3.1.4	Set_LimitQ2-1 (IL2)	159
9.1.3.1.5	Set_LimitQ1-2 (IL4)	159
9.1.3.1.6	Set_LimitQ2-2 (IL5)	159
9.1.3.1.7	Set_ModeQ1 (IM1)	159
9.1.3.1.8	Set_ModeQ2 (IM2)	160
9.1.3.1.9	Set_NormQ1 (IN1)	160
9.1.3.1.10	Set_NormQ2 (IN2)	160
9.1.3.2	Analog output	160
9.1.3.2.1	Set_LimitQA-1 (IL3)	160
9.1.3.2.2	Set_LimitQA-2 (IL6)	161
9.1.3.2.3	Set_NormQA (INA)	161
9.2	Commands for ILR118x	161
9.2.1	General commands	162
9.2.1.1	General	162
9.2.1.1.1	Set_Autostart (AS)	162
9.2.1.1.2	Get_Autostart (AS)	162
9.2.1.1.3	Get_Info (ID)	163
9.2.1.1.4	Get_AllParameters (PA)	163
9.2.1.1.5	Get_Temperature (TP)	166
9.2.1.1.6	Laser_Off (LF)	166
9.2.1.1.7	Laser_On (LO)	166
9.2.1.2	Tracking	166
9.2.1.2.1	DistanceTracking (DT)	166
9.2.1.2.2	DistanceTracking7m (DS)	166
9.2.1.2.3	DistanceTracking10Hz (DW)	167
9.2.1.2.4	DistanceTracking50Hz (DX)	167
9.2.1.2.5	StopTracking (<ESC>)	167
9.2.1.3	Triggering	167
9.2.1.3.1	DistanceTriggered (DF)	167
9.2.1.3.2	DistanceMeasure (DM)	167
9.2.1.3.3	Set_MeasureTime (ST)	167
9.2.1.3.4	Get_MeasureTime (ST)	167
9.2.1.3.5	Set_TriggerDelay (TD)	168
9.2.1.3.6	Get_TriggerDelay (TD)	168
9.2.1.3.7	Set_TriggerMode (TM)	169
9.2.1.3.8	Get_TriggerMode (TM)	169
9.2.1.4	Interfaces	170
9.2.1.4.1	Set_Baudrate (BR)	170
9.2.1.4.2	Get_Baudrate (BR)	170
9.2.1.5	Parameter management	170
9.2.1.5.1	Reset_Parameters (PR)	170
9.2.2	Measurement	173
9.2.2.1	Measurement value processing	173
9.2.2.1.1	Set_ScaleFactor (SF)	173
9.2.2.1.2	Get_ScaleFactor (SF)	174

9.2.2.1.3	Set_Offset (OF)	174
9.2.2.1.4	Get_Offset (OF)	174
9.2.2.1.5	CurrentDistAsOffset (SO)	175
9.2.2.1.6	Set_AverageValue (SA)	175
9.2.2.1.7	Get_AverageValue (SA)	175
9.2.3	Data output	175
9.2.3.1	General	175
9.2.3.1.1	Set_OutputFormat (SD)	175
9.2.3.1.2	Get_OutputFormat (SD)	176
9.2.3.1.3	Set_RemovalMeasVal (RM)	176
9.2.3.1.4	Get_RemovalMeasVal (RM)	177
9.2.3.1.5	Set_ErrorMode (SE)	178
9.2.3.1.6	Get_ErrorMode (SE)	178
9.2.3.2	Switching outputs	178
9.2.3.2.1	Set_AlarmStart (AC)	178
9.2.3.2.2	Get_AlarmStart (AC)	179
9.2.3.2.3	Set_AlarmHysteresis (AH)	179
9.2.3.2.4	Get_AlarmHysteresis (AH)	179
9.2.3.2.5	Set_AlarmWidth (AW)	179
9.2.3.2.6	Get_AlarmWidth (AW)	180
9.2.3.3	Analog output	180
9.2.3.3.1	Set_RangeBegin4mA (RB)	180
9.2.3.3.2	Get_RangeBegin4mA (RB)	180
9.2.3.3.3	Set_RangeEnd20mA (RE)	181
9.2.3.3.4	Get_RangeEnd20mA (RE)	181
9.3	Commands for ILR1191	181
9.3.1	General commands	182
9.3.1.1	General	182
9.3.1.1.1	Set_Autostart (AS)	182
9.3.1.1.2	Get_Autostart (AS)	183
9.3.1.1.3	Set_PilotLaser (PL)	184
9.3.1.1.4	Get_PilotLaser (PL)	184
9.3.1.1.5	Get_Info (ID)	184
9.3.1.1.6	Get_AllParameters (PA)	185
9.3.1.1.7	Get_Temperature (TP)	189
9.3.1.1.8	Get_HWDiagnosis (HW)	189
9.3.1.1.9	Trigger_ColdStart (DR)	189
9.3.1.2	Tracking	189
9.3.1.2.1	DistanceTracking (DT)	189
9.3.1.2.2	SpeedTracking (VT)	189
9.3.1.2.3	StopTracking (<ESC>)	190
9.3.1.3	Triggering	190
9.3.1.3.1	DistanceTriggered (DF)	190
9.3.1.3.2	DistanceMeasure (DM)	190
9.3.1.3.3	SpeedMeasure (VM)	190
9.3.1.3.4	Set_TriggerDelay (TD)	190
9.3.1.3.5	Get_TriggerDelay (TD)	191
9.3.1.4	Interfaces	192
9.3.1.4.1	Set_Baudrate (BR)	192
9.3.1.4.2	Get_Baudrate (BR)	192
9.3.1.5	Parameter management	193
9.3.1.5.1	Reset_Parameters (PR)	193
9.3.2	Measurement	197

9.3.2.1	General	197
9.3.2.1.1	Set_MeasFreq (MF)	197
9.3.2.1.2	Get_MeasFreq (MF)	197
9.3.2.2	Measurement value processing	197
9.3.2.2.1	Set_ScaleFactor (SF)	197
9.3.2.2.2	Get_ScaleFactor (SF)	198
9.3.2.2.3	Set_Offset (OF)	198
9.3.2.2.4	Get_Offset (OF)	198
9.3.2.2.5	CurrentDistAsOffset (SO)	199
9.3.3	Data output	199
9.3.3.1	General	199
9.3.3.1.1	Set_OutputFormat (SD)	199
9.3.3.1.2	Get_OutputFormat (SD)	200
9.3.3.1.3	Set_TerminatingChar (TE)	200
9.3.3.1.4	Get_TerminatingChar (TE)	201
9.3.3.1.5	Set_ErrorMode (SE)	201
9.3.3.1.6	Get_ErrorMode (SE)	201
9.3.3.1.7	Set_MeasureWindow (MW)	202
9.3.3.1.8	Get_MeasureWindow (MW)	202
9.3.3.1.9	Set_AverageValue (SA)	203
9.3.3.1.10	Get_AverageValue (SA)	203
9.3.3.2	Switching outputs	203
9.3.3.2.1	Set_Out1Parameters (Q1)	203
9.3.3.2.2	Get_Out1Parameters (Q1)	204
9.3.3.2.3	Set_Out2Parameters (Q2)	205
9.3.3.2.4	Get_Out2Parameters (Q2)	206
9.3.3.3	Analog output	207
9.3.3.3.1	Set_AnalogOutLimits (QA)	207
9.3.3.3.2	Get_AnalogOutLimits (QA)	207
9.3.3.4	SSI	208
9.3.3.4.1	Set_FormatSSI (SC)	208
9.3.3.4.2	Get_FormatSSI (SC)	208
10	Commands for ILD sensors	209
10.1	Commands for ILD1401	209
10.1.1	General commands	209
10.1.1.1	General	209
10.1.1.1.1	Get_Info (INFO)	209
10.1.1.1.2	Get_Version (VERSION)	210
10.1.1.1.3	Reset_Boot (BOOT)	211
10.1.2	Measurement	211
10.1.2.1	Set_Median (MEDIAN)	211
10.1.3	Data output	211
10.1.3.1	General	211
10.1.3.1.1	SaveLastMV (SAVELASTMV)	211
10.1.3.1.2	Set_OutputChannel (OUTPUTCHANNEL)	211
10.2	Commands for ILD1302/ILD1402	212
10.2.1	General commands	212
10.2.1.1	General	212
10.2.1.1.1	Get_Info (GET_INFO)	212
10.2.1.1.2	Get_Settings (GET_SETTINGS)	215
10.2.1.1.3	Reset_Boot (RESET_BOOT)	218
10.2.1.2	User level	218

10.2.1.2.1	Set_KeyLock (SET_KEYLOCK)	218
10.2.1.3	Interfaces	219
10.2.1.3.1	Set_Baudrate (SET_BAUDRATE)	219
10.2.1.4	Parameter management	219
10.2.1.4.1	Set_Default (SET_DEFAULT)	219
10.2.1.4.2	Set_SaveSettingsMode (SET_SAVE_SETTINGS_MODE)	219
10.2.1.5	Internal controller commands	220
10.2.1.5.1	Laser_Off (LASER_OFF)	220
10.2.1.5.2	Laser_On (LASER_ON)	220
10.2.1.5.3	Set_ExtInputMode (SET_EXT_INPUT_MODE)	220
10.2.1.5.4	Get_CI_Mode (GET_CI_MODE)	220
10.2.1.5.5	Set_CI_Mode_1401 (SET_CI_MODE_1401)	220
10.2.1.5.6	Set_CI_Mode_1402 (SET_CI_MODE_1402)	220
10.2.1.5.7	Set_OperationMode (SET_MODE)	221
10.2.1.5.8	Set_LaserDiode1 (SET_LD1)	221
10.2.1.5.9	Set_LaserDiode2 (SET_LD2)	221
10.2.1.5.10	Get_LaserDiodeError (GET_LDERROR)	221
10.2.1.5.11	Get_CurrentOfMonitor (GET_IMON)	222
10.2.2	Measurement	222
10.2.2.1	Set_Speed (SET_SCANRATE)	222
10.2.2.2	Set_PeakSearching (SET_PEAKSEARCHING)	222
10.2.2.3	Get_Video	222
10.2.2.4	Set_Threshold (SET_THRESHOLD)	223
10.2.2.5	Set_Av (SET_AV)	223
10.2.3	Data output	223
10.2.3.1	General	223
10.2.3.1.1	Dat_Out_Off (DAT_OUT_OFF)	223
10.2.3.1.2	Dat_Out_On (DAT_OUT_ON)	223
10.2.3.1.3	Set_ErrorHandler (SET_ANALOG_ERROR_HANDLER)	224
10.2.3.1.4	ASCII_Output (ASCII_OUTPUT)	224
10.2.3.1.5	Set_OutputType (SET_OUTPUT_CHANNEL)	224
10.2.3.1.6	Set_OutputMode (SET_OUTPUTMODE)	224
10.2.3.1.7	Set_OutputTime (SET_OUTPUTTIME_MS)	225
10.2.3.2	Analog output	225
10.2.3.2.1	Set_TeachValue (SET_TEACH_VALUE)	225
10.2.3.2.2	Reset_TeachValue (RESET_TEACH_VALUE)	225
10.3	Commands for ILD1320/ILD1420	225
10.3.1	General commands	226
10.3.1.1	General	226
10.3.1.1.1	Get_Help (HELP)	226
10.3.1.1.2	Get_Info (GETINFO)	226
10.3.1.1.3	Get_OutputInfo_RS422 (GETOUTINFO_RS422)	227
10.3.1.1.4	Set_Unit (UNIT)	229
10.3.1.1.5	Get_Unit (UNIT)	229
10.3.1.1.6	Reset_Boot (RESET)	229
10.3.1.1.7	Reset_Counter (RESETCNT)	229
10.3.1.1.8	Set_Keylock (KEYLOCK)	230
10.3.1.1.9	Get_Keylock (KEYLOCK)	230
10.3.1.1.10	Set_KeyFunction (KEYFUNC)	231
10.3.1.1.11	Get_KeyFunction (KEYFUNC)	231
10.3.1.1.12	Set_Echo (ECHO)	231
10.3.1.1.13	Get_Echo (ECHO)	231
10.3.1.1.14	Get_AllParameters (PRINT)	232

10.3.1.2 User level	242
10.3.1.2.1 Logout (LOGOUT)	242
10.3.1.2.2 Login (LOGIN)	242
10.3.1.2.3 Get_UserLevel (GETUSERLEVEL)	242
10.3.1.2.4 Set_DefaultUser (STDUSER)	243
10.3.1.2.5 Get_DefaultUser (STDUSER)	243
10.3.1.2.6 Set_Password (PASSWD)	243
10.3.1.3 Triggering	243
10.3.1.3.1 Set_TriggerMode (TRIGGER)	243
10.3.1.3.2 Get_TriggerMode (TRIGGER)	244
10.3.1.3.3 Set_TriggerMoment (TRIGGERAT)	244
10.3.1.3.4 Get_TriggerMoment (TRIGGERAT)	244
10.3.1.3.5 Set_TriggerCount (TRIGGERCOUNT)	244
10.3.1.3.6 Get_TriggerCount (TRIGGERCOUNT)	245
10.3.1.3.7 Software_Trigger (TRIGGERSW)	245
10.3.1.4 Interfaces	245
10.3.1.4.1 Set_Baudrate (BAUDRATE)	245
10.3.1.4.2 Get_Baudrate (BAUDRATE)	245
10.3.1.4.3 Set_AppLanguage (LANGUAGE)	246
10.3.1.4.4 Get_AppLanguage (LANGUAGE)	246
10.3.1.5 Parameter management	246
10.3.1.5.1 Save_InterfaceParameters (BASICSETTINGS STORE)	246
10.3.1.5.2 Load_InterfaceParameters (BASICSETTINGS READ)	246
10.3.1.5.3 Save_MeasureParameters (MEASSETTINGS STORE)	246
10.3.1.5.4 Load_MeasureParameters (MEASSETTINGS READ)	247
10.3.1.5.5 Rename_MeasureParameters (MEASSETTINGS RENAME)	247
10.3.1.5.6 Get_CurrentMeasureSetting (MEASSETTINGS CURRENT)	247
10.3.1.5.7 Get_MeasureSettingsList (MEASSETTINGS LIST)	247
10.3.1.5.8 Delete_MeasureParameters (MEASSETTINGS DELETE)	247
10.3.1.5.9 Get_MeasurePresetList (MEASSETTINGS PRESETLIST)	248
10.3.1.5.10 Set_InitialMeasureSetting (MEASSETTINGS INITIAL)	248
10.3.1.5.11 Get_InitialMeasureSetting (MEASSETTINGS INITIAL)	248
10.3.1.5.12 Set_MeasurePresetMode (MEASSETTINGS PRESETMODE)	248
10.3.1.5.13 Get_MeasurePresetMode (MEASSETTINGS PRESETMODE)	249
10.3.1.5.14 Set_Default (SETDEFAULT)	249
10.3.1.5.15 Export_Parameters (EXPORT)	249
10.3.1.5.16 Import_Parameters (IMPORT)	250
10.3.1.6 Internal controller commands	250
10.3.1.6.1 Set_MultiFunctionInputMode (MFIFUNC)	250
10.3.1.6.2 Get_MultiFunctionInputMode (MFIFUNC)	250
10.3.1.6.3 Set_MultiFunctionInputLevel (MFILEVEL)	251
10.3.1.6.4 Get_MultiFunctionInputLevel (MFILEVEL)	251
10.3.1.6.5 Set_InitialChartType (CHARTTYPE_INITIAL)	251
10.3.1.6.6 Get_InitialChartType (CHARTTYPE_INITIAL)	251
10.3.1.6.7 Set_ChartType (CHARTTYPE)	252
10.3.1.6.8 Get_ChartType (CHARTTYPE)	252
10.3.1.6.9 Set_TargetMode (TARGETMODE)	252
10.3.1.6.10 Get_TargetMode (TARGETMODE)	252
10.3.1.6.11 Get_HTTP (GETHTTP)	253
10.3.2 Measurement	253
10.3.2.1 General	253
10.3.2.1.1 Set_MeasurePeak (MEASPEAK)	253
10.3.2.1.2 Get_MeasurePeak (MEASPEAK)	254

10.3.2.1.3	Set_Samplerate (MEASRATE)	254
10.3.2.1.4	Get_Samplerate (MEASRATE)	254
10.3.2.1.5	Set_LaserPower (LASERPOW)	254
10.3.2.1.6	Get_LaserPower (LASERPOW)	255
10.3.2.1.7	Get_VideoStreamSignal	255
10.3.2.2	Video signal	256
10.3.2.2.1	Set_ROI (ROI)	256
10.3.2.2.2	Get_ROI (ROI)	256
10.3.2.3	Measurement value processing	257
10.3.2.3.1	Set_Averaging (AVERAGE)	257
10.3.2.3.2	Get_Averaging (AVERAGE)	258
10.3.2.3.3	Set_MasterValue (MASTERMV)	258
10.3.2.3.4	Get_MasterValue (MASTERMV)	259
10.3.3	Data output	259
10.3.3.1	General	259
10.3.3.1.1	Set_DataOutInterface (OUTPUT)	259
10.3.3.1.2	Get_DataOutInterface (OUTPUT)	260
10.3.3.1.3	Set_ResamplingDevice (OUTREDUCEDEVICE)	260
10.3.3.1.4	Get_ResamplingDevice (OUTREDUCEDEVICE)	260
10.3.3.1.5	Set_ResamplingCount (OUTREDUCECOUNT)	261
10.3.3.1.6	Get_ResamplingCount (OUTREDUCECOUNT)	261
10.3.3.1.7	Set_HoldLastValid (OUTHOLD)	261
10.3.3.1.8	Get_HoldLastValid (OUTHOLD)	261
10.3.3.2	Selected measurement values	262
10.3.3.2.1	Set_OutputAdditional_RS422 (OUTADD_RS422)	262
10.3.3.2.2	Get_OutputAdditional_RS422 (OUTADD_RS422)	263
10.3.3.2.3	Set_Output_RS422 (OUT_RS422)	263
10.3.3.2.4	Get_Output_RS422 (OUT_RS422)	265
10.3.3.2.5	Set_OutputVideo_RS422 (OUTVIDEO_RS422)	266
10.3.3.2.6	Get_OutputVideo_RS422 (OUTVIDEO_RS422)	266
10.3.3.3	Switching outputs	266
10.3.3.3.1	Set_ErrorOutput1 (ERROROUT1)	266
10.3.3.3.2	Get_ErrorOutput1 (ERROROUT1)	267
10.3.3.3.3	Set_ErrorLimit (ERRORLIMIT)	267
10.3.3.3.4	Get_ErrorLimit (ERRORLIMIT)	267
10.3.3.3.5	Set_ErrorHysteresis (ERRORHYSTERESIS)	268
10.3.3.3.6	Get_ErrorHysteresis (ERRORHYSTERESIS)	268
10.3.3.3.7	Set_ErrorOutHoldTime (ERROROUTHOLD)	268
10.3.3.3.8	Get_ErrorOutHoldTime (ERROROUTHOLD)	268
10.3.3.3.9	Set_ErrorLevelOut1 (ERRORLEVELOUT1)	269
10.3.3.3.10	Get_ErrorLevelOut1 (ERRORLEVELOUT1)	269
10.3.3.4	Analog output	269
10.3.3.4.1	Set_AnalogScale (ANALOGSCALE)	269
10.3.3.4.2	Get_AnalogScale (ANALOGSCALE)	270
10.4	Commands for ILD1700	270
10.4.1	General commands	271
10.4.1.1	General	271
10.4.1.1.1	Get_Info (GET_INFO)	271
10.4.1.1.2	Get_Settings (GET_SETTINGS)	274
10.4.1.1.3	Reset_Boot (RESET_BOOT)	277
10.4.1.2	User level	277
10.4.1.2.1	Set_KeyLock (SET_KEYLOCK)	277
10.4.1.3	Interfaces	277

10.4.1.3.1 Set_Baudrate (SET_BAUDRATE)	277
10.4.1.4 Parameter management	278
10.4.1.4.1 Set_Default (SET_DEFAULT)	278
10.4.1.4.2 WriteFlashZero (WriteFlashZero)	278
10.4.1.5 Internal controller commands	278
10.4.1.5.1 Laser_Off (LASER_OFF)	278
10.4.1.5.2 Laser_On (LASER_ON)	278
10.4.1.5.3 Set_Sync_TrMode (SET_SYNCMODE/TRIGGERMODE)	279
10.4.1.5.4 Set_ErrorOutput (SET_ERROROUTPUT)	279
10.4.2 Measurement	279
10.4.2.1 Set_MeasureMode	279
10.4.2.2 Set_Speed (SET_SPEED)	280
10.4.2.3 Set_VideoMode	280
10.4.2.4 Get_Video	280
10.4.2.5 Set_Av0 (SET_AV0)	280
10.4.2.6 Set_Av1 (SET_AV1)	280
10.4.2.7 Set_Av2 (SET_AV2)	280
10.4.2.8 Set_Av3 (SET_AV3)	281
10.4.2.9 Set_AvX (SET_AVX)	281
10.4.2.10 Set_Av_T (SET_AV_T)	281
10.4.2.11 Get_MeasValue (GET_MEASVALUE)	281
10.4.3 Data output	281
10.4.3.1 General	281
10.4.3.1.1 Dat_Out_Off (DAT_OUT_OFF)	281
10.4.3.1.2 Dat_Out_On (DAT_OUT_ON)	282
10.4.3.1.3 Set_ErrorHandler (SET_ERRORHANDLER)	282
10.4.3.1.4 ASCII_Output (ASCII_OUTPUT)	282
10.4.3.1.5 Set_OutputType (SET_OUTPUTTYP)	282
10.4.3.2 Switching outputs	282
10.4.3.2.1 Set_Limits (SET_LIMITS)	282
10.4.3.2.2 Set_UpperLimit_F1 (SET_UPPERLIMIT_F1)	283
10.4.3.2.3 Set_LowerLimit_F1 (SET_LOWERLIMIT_F1)	283
10.4.3.3 Analog output	283
10.4.3.3.1 Zero (SET_ZERO)	283
10.5 Commands for ILD2200	284
10.5.1 General commands	284
10.5.1.1 General	284
10.5.1.1.1 Get_Info (INFO)	284
10.5.1.1.2 Get_Settings (Get_Settings)	286
10.5.1.1.3 Get_Version	287
10.5.1.1.4 Reset_Boot (RESET)	287
10.5.1.2 User level	287
10.5.1.2.1 Set_KeyLock (SET_TASTENSPERRE)	287
10.5.1.3 Internal controller commands	288
10.5.1.3.1 Laser_Off (LASER_OFF)	288
10.5.1.3.2 Laser_On (LASER_ON)	288
10.5.2 Measurement	288
10.5.2.1 Set_Av0 (AVG 0)	288
10.5.2.2 Set_Av1 (AVG 1)	288
10.5.2.3 Set_Av2 (AVG 2)	288
10.5.2.4 Set_Av3 (AVG 3)	288
10.5.2.5 Set_AvX (AVG n)	288
10.5.2.6 Set_Av_T (AVGTYP)	288

10.5.3 Data output	289
10.5.3.1 General	289
10.5.3.1.1 Dat_Out_Off (STOP)	289
10.5.3.1.2 Dat_Out_On (START)	289
10.5.3.2 Switching outputs	289
10.5.3.2.1 Transmit_Intensity	289
10.5.3.3 Analog output	289
10.5.3.3.1 Zero (ZERO)	289
10.6 Commands for ILD2300	289
10.6.1 General commands	290
10.6.1.1 General	290
10.6.1.1.1 Get_Help (HELP)	290
10.6.1.1.2 Get_Info (GETINFO)	290
10.6.1.1.3 Get_OutputInfo_RS422 (GETOUTINFO_RS422)	291
10.6.1.1.4 Get_OutputInfo_ETH (GETOUTINFO_ETH)	293
10.6.1.1.5 Set_SyncMode (SYNC)	295
10.6.1.1.6 Get_SyncMode (SYNC)	295
10.6.1.1.7 Set_Unit (UNIT)	296
10.6.1.1.8 Get_Unit (UNIT)	296
10.6.1.1.9 Reset_Boot (RESET)	296
10.6.1.1.10 Reset_Counter (RESETCNT)	296
10.6.1.1.11 Set_Echo (ECHO)	297
10.6.1.1.12 Get_Echo (ECHO)	297
10.6.1.1.13 Get_AllParameters (PRINT)	297
10.6.1.2 User level	311
10.6.1.2.1 Logout (LOGOUT)	311
10.6.1.2.2 Login (LOGIN)	311
10.6.1.2.3 Get_UserLevel (GETUSERLEVEL)	311
10.6.1.2.4 Set_DefaultUser (STDUSER)	311
10.6.1.2.5 Get_DefaultUser (STDUSER)	311
10.6.1.2.6 Set_Password (PASSWD)	312
10.6.1.3 Triggering	312
10.6.1.3.1 Set_TriggerMode (TRIGGER)	312
10.6.1.3.2 Get_TriggerMode (TRIGGER)	312
10.6.1.3.3 Set_TriggerMoment (TRIGGERAT)	313
10.6.1.3.4 Get_TriggerMoment (TRIGGERAT)	313
10.6.1.3.5 Set_TriggerLevel (TRIGGERLEVEL)	313
10.6.1.3.6 Get_TriggerLevel (TRIGGERLEVEL)	313
10.6.1.3.7 Set_TriggerCount (TRIGGERCOUNT)	314
10.6.1.3.8 Get_TriggerCount (TRIGGERCOUNT)	314
10.6.1.3.9 Software_Trigger (TRIGGERSW)	314
10.6.1.3.10 Set_TriggerOutput (TRIGGEROUT)	314
10.6.1.3.11 Get_TriggerOutput (TRIGGEROUT)	314
10.6.1.4 Interfaces	315
10.6.1.4.1 Set_IPConfiguration (IPCONFIG)	315
10.6.1.4.2 Get_IPConfiguration (IPCONFIG)	315
10.6.1.4.3 Set_IPDataTransferMode (MEASTRANSFER)	316
10.6.1.4.4 Get_IPDataTransferMode (MEASTRANSFER)	317
10.6.1.4.5 Set_EthernetMode (ETHERMODE)	317
10.6.1.4.6 Get_EthernetMode (ETHERMODE)	317
10.6.1.4.7 Set_Baudrate (BAUDRATE)	318
10.6.1.4.8 Get_Baudrate (BAUDRATE)	318
10.6.1.5 Parameter management	318

10.6.1.5.1	Save_Parameters (STORE)	318
10.6.1.5.2	Load_Parameters (READ)	319
10.6.1.5.3	Get_ActiveParameterSet (READ)	319
10.6.1.5.4	Set_Default (SETDEFAULT)	319
10.6.2	Measurement	320
10.6.2.1	General	320
10.6.2.1.1	Set_MeasureMode (MEASMODE)	320
10.6.2.1.2	Get_MeasureMode (MEASMODE)	320
10.6.2.1.3	Set_MeasurePeak (MEASPEAK)	320
10.6.2.1.4	Get_MeasurePeak (MEASPEAK)	321
10.6.2.1.5	Get_Video (GETVIDEO)	321
10.6.2.1.6	Set_ShutterMode (SHUTTERMODE)	322
10.6.2.1.7	Get_ShutterMode (SHUTTERMODE)	322
10.6.2.1.8	Set_Samplerate (MEASRATE)	322
10.6.2.1.9	Get_Samplerate (MEASRATE)	323
10.6.2.1.10	Set_ShutterTime (SHUTTER)	323
10.6.2.1.11	Get_ShutterTime (SHUTTER)	323
10.6.2.1.12	Set_ShutterFactor (SHUTTERFACTOR)	324
10.6.2.1.13	Get_ShutterFactor (SHUTTERFACTOR)	324
10.6.2.1.14	Set_LaserPower (LASERPOW)	324
10.6.2.1.15	Get_LaserPower (LASERPOW)	324
10.6.2.2	Video signal	325
10.6.2.2.1	Set_ROI (ROI)	325
10.6.2.2.2	Get_ROI (ROI)	325
10.6.2.2.3	Set_VideoAverage (VSAVERAGE)	325
10.6.2.2.4	Get_VideoAverage (VSAVERAGE)	326
10.6.2.2.5	Set_Threshold (THRESHOLD)	326
10.6.2.2.6	Get_Threshold (THRESHOLD)	326
10.6.2.3	Material database	326
10.6.2.3.1	Get_MaterialTable (MATERIALTABLE)	326
10.6.2.3.2	Set_ActiveMaterial (MATERIAL)	327
10.6.2.3.3	Get_ActiveMaterial (MATERIAL)	327
10.6.2.3.4	Get_MaterialInfo (MATERIALINFO)	328
10.6.2.3.5	Edit_Material (MATERIALEDIT)	328
10.6.2.3.6	Delete_Material (MATERIALDELETE)	328
10.6.2.3.7	Clear_MaterialTable	329
10.6.2.4	Measurement value processing	329
10.6.2.4.1	Set_Averaging (AVERAGE)	329
10.6.2.4.2	Get_Averaging (AVERAGE)	330
10.6.2.4.3	Set_SpikeCorrection (SPIKECORR)	330
10.6.2.4.4	Get_SpikeCorrection (SPIKECORR)	331
10.6.2.4.5	Set_StatisticDepth (STATISTICDEPTH)	332
10.6.2.4.6	Get_StatisticDepth (STATISTICDEPTH)	332
10.6.2.4.7	Reset_Statistic (RESETSTATISTIC)	332
10.6.2.4.8	Set_MasterValue (MASTERMV)	332
10.6.2.4.9	Get_MasterValue (MASTERMV)	333
10.6.3	Data output	333
10.6.3.1	General	333
10.6.3.1.1	Set_DataOutInterface (OUTPUT)	333
10.6.3.1.2	Get_DataOutInterface (OUTPUT)	333
10.6.3.1.3	Set_Resampling (OUTREDUCE)	334
10.6.3.1.4	Get_Resampling (OUTREDUCE)	334
10.6.3.1.5	Set_HoldLastValid (OUTHOLD)	335

10.6.3.1.6	Get_HoldLastValid (OUTHOLD)	335
10.6.3.1.7	Set_MeasureValueCnt (GETVALUE)	335
10.6.3.1.8	Get_MeasureValueCnt (GETVALUE)	335
10.6.3.2	Selected measurement values	336
10.6.3.2.1	Set_OutputDistance_RS422 (OUTDIST_RS422)	336
10.6.3.2.2	Get_OutputDistance_RS422 (OUTDIST_RS422)	336
10.6.3.2.3	Set_OutputThickness_RS422 (OUTTHICK_RS422)	336
10.6.3.2.4	Get_OutputThickness_RS422 (OUTTHICK_RS422)	337
10.6.3.2.5	Set_OutputStatistic_RS422 (OUTSTATIC_RS422)	337
10.6.3.2.6	Get_OutputStatistic_RS422 (OUTSTATIC_RS422)	337
10.6.3.2.7	Set_OutputStatistic_ETH (OUTSTATIC_ETH)	338
10.6.3.2.8	Get_OutputStatistic_ETH (OUTSTATIC_ETH)	338
10.6.3.2.9	Set_OutputAdditional_RS422 (OUTADD_RS422)	339
10.6.3.2.10	Get_OutputAdditional_RS422 (OUTADD_RS422)	340
10.6.3.2.11	Set_OutputAdditional_ETH (OUTADD_ETH)	340
10.6.3.2.12	Get_OutputAdditional_ETH (OUTADD_ETH)	341
10.6.3.2.13	Set_OutputVideo_ETH (OUTVIDEO)	342
10.6.3.2.14	Get_OutputVideo_ETH (OUTVIDEO)	343
11	Commands for Confocal (IFD) sensors	344
11.1	Commands for IFD2401/IFD2431	344
11.1.1	General commands	344
11.1.1.1	General	344
11.1.1.1.1	Get_Status (STS)	344
11.1.1.1.2	Get_Version (VER)	346
11.1.1.1.3	Reset (RST)	347
11.1.1.2	Sensor	347
11.1.1.2.1	Set_ActiveSensor (SEN)	347
11.1.1.2.2	Get_ActiveSensor (SEN?)	347
11.1.1.2.3	Get_Range (SCA)	347
11.1.1.2.4	Get_AllRanges (LUL)	347
11.1.1.2.5	Acquire_DarkSig (DRK)	348
11.1.1.2.6	FastDark (FDK)	348
11.1.1.2.7	Set_AutoDark (ADK)	348
11.1.1.2.8	Get_AutoDark (ADK?)	349
11.1.1.3	Triggering	349
11.1.1.3.1	Continue (CTN)	349
11.1.1.3.2	Start_Trigger (TRG)	349
11.1.1.3.3	End_Trigger	349
11.1.1.3.4	SingleShot_Trg (TRE)	349
11.1.1.3.5	Set_TrgMode_Edge (TRS)	349
11.1.1.3.6	Set_TrgMode_State (TRN)	350
11.1.1.3.7	Set_ActiveEdge (TRF)	350
11.1.1.3.8	Get_ActiveEdge (TRF?)	350
11.1.1.3.9	Software_Trigger (STR)	350
11.1.1.4	Encoder	350
11.1.1.4.1	RecenterEncoder (RCD)	350
11.1.1.5	Interfaces	351
11.1.1.5.1	Set_Baudrate (BAU)	351
11.1.1.5.2	Get_Baudrate (BAU?)	351
11.1.1.6	Parameter management	352
11.1.1.6.1	Save_Setup (SSU)	352
11.1.1.7	Internal controller commands	352

11.1.1.7.1	Set_LampTest (SLP)	352
11.1.1.7.2	Get_LampTest (SLP?)	352
11.1.1.7.3	Set_LampTestThr (CSL)	352
11.1.1.7.4	Get_LampTestThr (CSL?)	353
11.1.1.7.5	Set_Watchdog (WDE)	353
11.1.1.7.6	Get_Watchdog (WDE?)	353
11.1.1.7.7	Set_WatchdogPrd (WDP)	353
11.1.1.7.8	Get_WatchdogPrd (WDP?)	354
11.1.2	Measurement	354
11.1.2.1	General	354
11.1.2.1.1	Set_MeasureMode (MOD)	354
11.1.2.1.2	Get_MeasureMode (MOD?)	354
11.1.2.1.3	Set_FirstPeakMode (MSP)	354
11.1.2.1.4	Get_FirstPeakMode (MSP?)	355
11.1.2.1.5	Set_SRIndex (SRA)	355
11.1.2.1.6	Get_SRIndex (SRA?)	355
11.1.2.1.7	Set_FreeSR (FRQ)	356
11.1.2.1.8	Get_FreeSR (FRQ?)	356
11.1.2.1.9	Set_Exposure (TEX)	356
11.1.2.1.10	Get_Exposure (TEX?)	357
11.1.2.1.11	Get_MinSR (FRM)	357
11.1.2.1.12	Set_DoubleFreq (DFA)	357
11.1.2.1.13	Get_DoubleFreq (DFA?)	357
11.1.2.1.14	Set_Frequencies (DFF)	358
11.1.2.1.15	Get_Frequencies (DFF?)	358
11.1.2.1.16	Get_CCD (CCD)	358
11.1.2.1.17	Get_DarkSig (SGD)	359
11.1.2.1.18	Start_Spectrum	359
11.1.2.1.19	Get_Spectrum	360
11.1.2.1.20	End_Spectrum	361
11.1.2.2	Video signal	361
11.1.2.2.1	Set_Threshold (MNP)	361
11.1.2.2.2	Get_Threshold (MNP?)	361
11.1.2.2.3	Set_Threshold1 (SPP)	361
11.1.2.2.4	Get_Threshold1 (SPP?)	362
11.1.2.2.5	Set_Threshold2 (SDP)	362
11.1.2.2.6	Get_Threshold2 (SDP?)	362
11.1.2.2.7	Set_SpectralAv (AVS)	362
11.1.2.2.8	Get_SpectralAv (AVS?)	363
11.1.2.2.9	Set_Binning (FBH)	363
11.1.2.2.10	Get_Binning (FBH?)	363
11.1.2.3	Material database	363
11.1.2.3.1	Set_RefractIndex (SRI)	363
11.1.2.3.2	Get_RefractIndex (SRI?)	364
11.1.2.3.3	Set_RefIdxFile (INF)	364
11.1.2.3.4	Get_RefIdxFile (INF?)	365
11.1.2.4	Light source	365
11.1.2.4.1	Set_LEDIntensity (LED)	365
11.1.2.4.2	Get_LEDIntensity (LED?)	366
11.1.2.4.3	Set_AutoAdaptLED (AAL)	366
11.1.2.4.4	Get_AutoAdaptLED (AAL?)	366
11.1.2.4.5	Set_AdaptLEDThr (VTH)	366
11.1.2.4.6	Get_AdaptLEDThr (VTH?)	367

11.1.2.4.7	Set_IntLightSrc (CCL)	367
11.1.2.4.8	Get_IntLightSrc (CCL?)	367
11.1.2.5	Measurement value processing	367
11.1.2.5.1	Set_Averaging (AVR)	367
11.1.2.5.2	Get_Averaging (AVR?)	368
11.1.3	Data output	368
11.1.3.1	General	368
11.1.3.1.1	Set_IntensityMode (DFI)	368
11.1.3.1.2	Get_IntensityMode (DFI?)	368
11.1.3.1.3	Set_HoldLastValid (HLV)	368
11.1.3.1.4	Get_HoldLastValid (HLV?)	369
11.1.3.1.5	Set_MissingSignal (RSP)	369
11.1.3.1.6	Get_MissingSignal (RSP?)	369
11.1.3.1.7	Set_Reverse (RVS)	369
11.1.3.1.8	Get_Reverse (RVS?)	369
11.1.3.1.9	Set_Ascii (ASC)	370
11.1.3.1.10	Set_Binary (BIN)	370
11.1.3.2	Selected measurement values	370
11.1.3.2.1	Set_OutputData (SOD)	370
11.1.3.2.2	Get_OutputData (SOD?)	370
11.1.3.3	Analog output	371
11.1.3.3.1	Set_AnalogOut (ANA)	371
11.1.3.3.2	Get_AnalogOut (ANA?)	372
11.1.3.3.3	Set_AnalogZero (SOF)	373
11.1.4	Internal commands	373
11.1.4.1	Set_DataScale (CEE)	373
11.1.4.2	Get_DataScale (CEE?)	373
11.1.4.3	Set_BarycenterSca (CEB)	373
11.1.4.4	Get_BarycenterSca (CEB?)	374
11.1.4.5	Set_BarycenterOff (CRB)	374
11.1.4.6	Get_BarycenterOff (CRB?)	374
11.1.4.7	Upload_RefIdxFile	374
11.1.4.8	Get_WhiteRef (SGW)	375
11.1.4.9	Get_NormSig (SGN)	375
11.1.4.10	Get_CalibTable (SGC)	375
11.1.4.11	Upload_CalibTable	376
11.2	Commands for IFD2445/IFD2451/IFD2461/IFD2471	376
11.2.1	General commands	377
11.2.1.1	General	377
11.2.1.1.1	Get_Help (HELP)	377
11.2.1.1.2	Get_Info (GETINFO)	377
11.2.1.1.3	Get_Temperature (GETTEMP)	378
11.2.1.1.4	Set_Echo (ECHO)	378
11.2.1.1.5	Get_Echo (ECHO)	378
11.2.1.1.6	Get_AllParameters (PRINT)	379
11.2.1.1.7	Set_SyncMode (SYNC)	409
11.2.1.1.8	Get_SyncMode (SYNC)	409
11.2.1.1.9	Reset_Boot (RESET)	409
11.2.1.2	User level	409
11.2.1.2.1	Logout (LOGOUT)	409
11.2.1.2.2	Login (LOGIN)	409
11.2.1.2.3	Get_UserLevel (GETUSERLEVEL)	410
11.2.1.2.4	Set_DefaultUser (STDUSER)	410

11.2.1.2.5	Get_DefaultUser (STDUSER)	410
11.2.1.2.6	Set_Password (PASSWD)	410
11.2.1.3	Sensor	411
11.2.1.3.1	Get_SensorTable (SENSORTABLE)	411
11.2.1.3.2	Set_ActiveSensor (SENSORHEAD)	411
11.2.1.3.3	Get_ActiveSensor (SENSORHEAD)	412
11.2.1.3.4	Get_SensorInfo (SENSORINFO)	412
11.2.1.3.5	DarkCorr (DARKCORR)	412
11.2.1.3.6	Set_DarkCorrThreshold (DARKCORRTHRES)	413
11.2.1.3.7	Get_DarkCorrThreshold (DARKCORRTHRES)	413
11.2.1.3.8	LightCorr (LIGHTCORR)	413
11.2.1.4	Triggering	413
11.2.1.4.1	Set_TriggerMode (TRIGGER)	413
11.2.1.4.2	Get_TriggerMode (TRIGGER)	414
11.2.1.4.3	Set_TriggerMoment (TRIGGERAT)	414
11.2.1.4.4	Get_TriggerMoment (TRIGGERAT)	414
11.2.1.4.5	Set_TriggerLevel (TRIGGERLEVEL)	414
11.2.1.4.6	Get_TriggerLevel (TRIGGERLEVEL)	415
11.2.1.4.7	Set_TriggerCount (TRIGGERCOUNT)	415
11.2.1.4.8	Get_TriggerCount (TRIGGERCOUNT)	415
11.2.1.4.9	Software_Trigger (TRIGGERSW)	415
11.2.1.4.10	Set_TriggerOnEncoder (TRIGGERENC)	415
11.2.1.4.11	Get_TriggerOnEncoder (TRIGGERENC)	416
11.2.1.5	Encoder	417
11.2.1.5.1	Set_EncoderInterpolation1 (ENCINTERPOL1)	417
11.2.1.5.2	Get_EncoderInterpolation1 (ENCINTERPOL1)	417
11.2.1.5.3	Set_EncoderInterpolation2 (ENCINTERPOL2)	417
11.2.1.5.4	Get_EncoderInterpolation2 (ENCINTERPOL2)	418
11.2.1.5.5	Set_EncoderInterpolation3 (ENCINTERPOL3)	418
11.2.1.5.6	Get_EncoderInterpolation3 (ENCINTERPOL3)	418
11.2.1.5.7	Set_EncoderMode1 (ENCREF1)	418
11.2.1.5.8	Get_EncoderMode1 (ENCREF1)	419
11.2.1.5.9	Set_EncoderMode2 (ENCREF2)	419
11.2.1.5.10	Get_EncoderMode2 (ENCREF2)	419
11.2.1.5.11	Set_EncoderMode3 (ENCREF3)	419
11.2.1.5.12	Get_EncoderMode3 (ENCREF3)	420
11.2.1.5.13	Set_EncoderPreload1 (ENCVALUE1)	420
11.2.1.5.14	Get_EncoderPreload1 (ENCVALUE1)	420
11.2.1.5.15	Set_EncoderPreload2 (ENCVALUE2)	420
11.2.1.5.16	Get_EncoderPreload2 (ENCVALUE2)	421
11.2.1.5.17	Set_EncoderPreload3 (ENCVALUE3)	421
11.2.1.5.18	Get_EncoderPreload3 (ENCVALUE3)	421
11.2.1.5.19	Load_Encoder1 (ENCSET)	421
11.2.1.5.20	Load_Encoder2 (ENCSET)	421
11.2.1.5.21	Load_Encoder3 (ENCSET)	421
11.2.1.5.22	EnableRef_Encoder1 (ENCRESET)	422
11.2.1.5.23	EnableRef_Encoder2 (ENCRESET)	422
11.2.1.5.24	EnableRef_Encoder3 (ENCRESET)	422
11.2.1.5.25	Set_Encoder.MaxValue1 (ENCMAX1)	422
11.2.1.5.26	Get_Encoder.MaxValue1 (ENCMAX1)	422
11.2.1.5.27	Set_Encoder.MaxValue2 (ENCMAX2)	422
11.2.1.5.28	Get_Encoder.MaxValue2 (ENCMAX2)	423
11.2.1.5.29	Set_Encoder.MaxValue3 (ENCMAX3)	423

11.2.1.5.30	Get_Encoder.MaxValue3 (ENCMAX3)	423
11.2.1.6	Interfaces	423
11.2.1.6.1	Set_IPConfiguration (IPCONFIG)	423
11.2.1.6.2	Get_IPConfiguration (IPCONFIG)	424
11.2.1.6.3	Set_IPDataTransferMode (MEASTRANSFER)	425
11.2.1.6.4	Get_IPDataTransferMode (MEASTRANSFER)	425
11.2.1.6.5	Set_EthernetMode (ETHERMODE)	426
11.2.1.6.6	Get_EthernetMode (ETHERMODE)	426
11.2.1.6.7	Set_Baudrate (BAUDRATE)	426
11.2.1.6.8	Get_Baudrate (BAUDRATE)	427
11.2.1.7	Parameter management	427
11.2.1.7.1	Save_Parameters (STORE)	427
11.2.1.7.2	Load_Parameters (READ)	427
11.2.1.7.3	Set_Default (SETDEFAULT)	428
11.2.2	Measurement	428
11.2.2.1	General	428
11.2.2.1.1	Set_MeasureMode (MEASMODE)	428
11.2.2.1.2	Get_MeasureMode (MEASMODE)	429
11.2.2.1.3	Set_MeasurePeak (MEASPEAK)	429
11.2.2.1.4	Get_MeasurePeak (MEASPEAK)	429
11.2.2.1.5	Set_ShutterMode (SHUTTERMODE)	430
11.2.2.1.6	Get_ShutterMode (SHUTTERMODE)	430
11.2.2.1.7	Set_Samplerate (MEASRATE)	430
11.2.2.1.8	Get_Samplerate (MEASRATE)	431
11.2.2.1.9	Set_ShutterTime (SHUTTER)	431
11.2.2.1.10	Get_ShutterTime (SHUTTER)	431
11.2.2.1.11	Get_Video (GETVIDEO)	432
11.2.2.1.12	Get_VideoStreamSignal	433
11.2.2.2	Video signal	434
11.2.2.2.1	Set_ROI (ROI)	434
11.2.2.2.2	Get_ROI (ROI)	435
11.2.2.2.3	Set_VideoAverage (VSAVERAGE)	435
11.2.2.2.4	Get_VideoAverage (VSAVERAGE)	436
11.2.2.2.5	Set_Threshold (THRESHOLD)	436
11.2.2.2.6	Get_Threshold (THRESHOLD)	436
11.2.2.3	Material database	436
11.2.2.3.1	Get_MaterialTable (MATERIALTABLE)	436
11.2.2.3.2	Set_ActiveMaterial (MATERIAL)	438
11.2.2.3.3	Get_ActiveMaterial (MATERIAL)	438
11.2.2.3.4	Get_MaterialInfo (MATERIALINFO)	438
11.2.2.3.5	Edit_Material_Abbe (MATERIALEDIT)	439
11.2.2.3.6	Edit_Material_Nx (MATERIALEDIT)	439
11.2.2.3.7	Delete_Material (MATERIALDELETE)	440
11.2.2.3.8	Clear_MaterialTable	440
11.2.2.4	Peak processing	440
11.2.2.4.1	Set_RefractiveCorrection (REFRACCORR)	440
11.2.2.4.2	Get_RefractiveCorrection (REFRACCORR)	441
11.2.2.4.3	Set_MaterialMultiPeak (MATERIALMP)	441
11.2.2.4.4	Get_MaterialMultiPeak (MATERIALMP)	442
11.2.2.5	Measurement value processing	442
11.2.2.5.1	Set_Averaging (AVERAGE)	442
11.2.2.5.2	Get_Averaging (AVERAGE)	443
11.2.2.5.3	Set_SpikeCorrection (SPIKECORR)	444

11.2.2.5.4	Get_SpikeCorrection (SPIKECORR)	445
11.2.2.5.5	Set_StatisticSignal (STATISTICSIGNAL)	446
11.2.2.5.6	Get_StatisticSignal (STATISTICSIGNAL)	446
11.2.2.5.7	Set_StatisticDepth (STATICDEPTH)	447
11.2.2.5.8	Get_StatisticDepth (STATICDEPTH)	447
11.2.2.5.9	Reset_Statistic (RESETSTATICISTIC)	447
11.2.2.5.10	Set_MasterSignal (MASTERSIGNAL)	447
11.2.2.5.11	Get_MasterSignal (MASTERSIGNAL)	448
11.2.2.5.12	Set_MasterValue (MASTERMV)	449
11.2.2.5.13	Get_MasterValue (MASTERMV)	449
11.2.3	Data output	449
11.2.3.1	General	449
11.2.3.1.1	Set_DataOutInterface (OUTPUT)	449
11.2.3.1.2	Get_DataOutInterface (OUTPUT)	450
11.2.3.1.3	Set_Resampling (OUTREDUCE)	450
11.2.3.1.4	Get_Resampling (OUTREDUCE)	450
11.2.3.1.5	Set_HoldLastValid (OUTHOLD)	451
11.2.3.1.6	Get_HoldLastValid (OUTHOLD)	451
11.2.3.1.7	Set_FramesPerPacket_ETH (MEASCNT_ETH)	451
11.2.3.1.8	Get_FramesPerPacket_ETH (MEASCNT_ETH)	452
11.2.3.2	Selected measurement values	452
11.2.3.2.1	Set_OutputDistance_RS422 (OUTDIST_RS422)	452
11.2.3.2.2	Get_OutputDistance_RS422 (OUTDIST_RS422)	453
11.2.3.2.3	Set_OutputDistance_ETH (OUTDIST_ETH)	454
11.2.3.2.4	Get_OutputDistance_ETH (OUTDIST_ETH)	455
11.2.3.2.5	Set_OutputThickness_RS422 (OUTTHICK_RS422)	457
11.2.3.2.6	Get_OutputThickness_RS422 (OUTTHICK_RS422)	460
11.2.3.2.7	Set_OutputThickness_ETH (OUTTHICK_ETH)	463
11.2.3.2.8	Get_OutputThickness_ETH (OUTTHICK_ETH)	466
11.2.3.2.9	Set_OutputStatistic_RS422 (OUTSTATIC_RS422)	469
11.2.3.2.10	Get_OutputStatistic_RS422 (OUTSTATIC_RS422)	469
11.2.3.2.11	Set_OutputStatistic_ETH (OUTSTATIC_ETH)	470
11.2.3.2.12	Get_OutputStatistic_ETH (OUTSTATIC_ETH)	470
11.2.3.2.13	Set_OutputAdditional_RS422 (OUTADD_RS422)	471
11.2.3.2.14	Get_OutputAdditional_RS422 (OUTADD_RS422)	472
11.2.3.2.15	Set_OutputAdditional_ETH (OUTADD_ETH)	474
11.2.3.2.16	Get_OutputAdditional_ETH (OUTADD_ETH)	475
11.2.3.2.17	Set_UnlinearizedMode (SWITCHMD2)	476
11.2.3.2.18	Get_UnlinearizedMode (SWITCHMD2)	477
11.2.3.2.19	Set_OutputVideo_ETH (OUTVIDEO)	477
11.2.3.2.20	Get_OutputVideo_ETH (OUTVIDEO)	478
11.2.3.3	Switching outputs	479
11.2.3.3.1	Set_ErrorOutput1 (ERROROUT1)	479
11.2.3.3.2	Get_ErrorOutput1 (ERROROUT1)	479
11.2.3.3.3	Set_ErrorOutput2 (ERROROUT2)	479
11.2.3.3.4	Get_ErrorOutput2 (ERROROUT2)	480
11.2.3.3.5	Set_ErrorLimit (ERRORLIMIT)	480
11.2.3.3.6	Get_ErrorLimit (ERRORLIMIT)	481
11.2.3.3.7	Set_ErrorLevel (ERRORLEVEL)	482
11.2.3.3.8	Get_ErrorLevel (ERRORLEVEL)	482
11.2.3.4	Analog output	482
11.2.3.4.1	Set_AnalogOutput (ANALOGOUT)	482
11.2.3.4.2	Get_AnalogOutput (ANALOGOUT)	483

11.2.3.4.3 Set_AnalogRange (ANALOG RANGE)	483
11.2.3.4.4 Get_AnalogRange (ANALOG RANGE)	483
11.2.3.4.5 Set_AnalogScale (ANALOG SCALE)	484
11.2.3.4.6 Get_AnalogScale (ANALOG SCALE)	484
11.2.4 Internal commands	485
11.2.4.1 Set_PeakParameter (PEAKPARAM)	485
11.2.4.2 Get_PeakParameter (PEAKPARAM)	485
11.2.4.3 Set_BoreControlParam (BORECTRL)	486
11.2.4.4 Get_BoreControlParam (BORECTRL)	486
11.2.4.5 Get_SpecCal (SPECCAL)	487
11.2.4.6 Del_LinearisationTable (DELLINTAB)	487
11.2.4.7 Save_ParametersTemp (SAVE2TMP)	487
11.2.4.8 Load_ParametersTemp (RESTORETMP)	487
11.2.4.9 Set_CalibrationDefaults (SETCALDEFAULT)	488
11.2.4.10 Get_RegHyst (REGHYST)	488
11.2.4.11 Upload_CalibTable	488
11.2.4.12 Save_DarkCorrReference (SAVEDARKCORRREF)	488
12 Commands for ODC sensors	489
12.1 Commands for ODC1202	489
12.1.1 General commands	489
12.1.1.1 General	489
12.1.1.1.1 Get_Version	489
12.1.1.2 User level	490
12.1.1.2.1 Get_Echo	490
12.1.1.3 Interfaces	490
12.1.1.3.1 Change_Baudrate	490
12.1.1.4 Parameter management	490
12.1.1.4.1 RefreshVideoThr	490
12.1.2 Measurement	491
12.1.2.1 Set_ParamSet1	491
12.1.2.2 Get_ParamSet1	493
12.1.2.3 Set_ParamSet2	496
12.1.2.4 Get_ParamSet2	498
12.1.2.5 Get_Video	501
12.1.2.6 Get_MeasValues	501
12.1.2.7 Get_Value_DataRecord	503
12.1.3 Data output	504
12.1.3.1 General	504
12.1.3.1.1 AutoDataTransfer	504
12.1.3.2 Switching outputs	505
12.1.3.2.1 Activate_Teach	505
12.1.3.3 Analog output	507
12.1.3.3.1 Reset_MinMax_Analog	507
12.2 Commands for ODC2500	507
12.2.1 General commands	508
12.2.1.1 General	508
12.2.1.1.1 Get_Info (INFO)	508
12.2.1.1.2 Reset_Boot (RESET)	509
12.2.1.2 Parameter management	509
12.2.1.2.1 Save_OptionData (SAVE_OPT_RAM_TO_FLASH)	509
12.2.1.2.2 Save_MeasProgData (SAVE_MPR_RAM_TO_FLASH)	509
12.2.2 Measurement	509

12.2.2.1	Choose_MeasProg (CHOOSE_MP)	509
12.2.2.2	Switch_Edge (SWITCH_EDGE)	510
12.2.2.3	Write_OptionData (WR_OPT_TO_RAM)	510
12.2.2.4	Read_OptionData (RD_OPT_RAM)	513
12.2.2.5	Update_OptionData	516
12.2.2.6	Write_MeasProgData (WR_MPR_TO_RAM)	519
12.2.2.7	Read_MeasProgData (RD_MPR_RAM)	522
12.2.2.8	Read_MinMax (RD_MINMAX)	525
12.2.2.9	Read_MinMaxReset (RD_MINMAX_RESET)	526
12.2.3	Data output	526
12.2.3.1	General	526
12.2.3.1.1	Dat_Out_Off (STOP)	526
12.2.3.1.2	Dat_Out_On (START)	526
12.3	Commands for ODC2520	527
12.3.1	General commands	527
12.3.1.1	General	527
12.3.1.1.1	Get_Help (HELP)	527
12.3.1.1.2	Get_Info (GETINFO)	528
12.3.1.1.3	Set_Echo (ECHO)	529
12.3.1.1.4	Get_Echo (ECHO)	529
12.3.1.1.5	Get_AllParameters (PRINT)	529
12.3.1.1.6	Set_SyncMode (SYNC)	560
12.3.1.1.7	Get_SyncMode (SYNC)	560
12.3.1.1.8	Set_Unit (UNIT)	560
12.3.1.1.9	Get_Unit (UNIT)	561
12.3.1.1.10	Reset_Boot (RESET)	561
12.3.1.2	User level	561
12.3.1.2.1	Logout (LOGOUT)	561
12.3.1.2.2	Login (LOGIN)	561
12.3.1.2.3	Get_UserLevel (GETUSERLEVEL)	561
12.3.1.2.4	Set_DefaultUser (STDUSER)	561
12.3.1.2.5	Get_DefaultUser (STDUSER)	562
12.3.1.2.6	Set_Password (PASSWD)	562
12.3.1.3	Measure distance	562
12.3.1.3.1	Get_MeasDistTable (MEASDISTTABLE)	562
12.3.1.3.2	Set_ActiveMeasDist (MEASDIST)	563
12.3.1.3.3	Get_ActiveMeasDist (MEASDIST)	563
12.3.1.3.4	Get_MeasDistInfo (MEASDISTINFO)	563
12.3.1.3.5	LightCorr (LIGHTCORR)	563
12.3.1.4	Triggering	564
12.3.1.4.1	Set_TriggerMode (TRIGGER)	564
12.3.1.4.2	Get_TriggerMode (TRIGGER)	564
12.3.1.4.3	Set_TriggerMoment (TRIGGERAT)	564
12.3.1.4.4	Get_TriggerMoment (TRIGGERAT)	565
12.3.1.4.5	Set_TriggerLevel (TRIGGERLEVEL)	565
12.3.1.4.6	Get_TriggerLevel (TRIGGERLEVEL)	565
12.3.1.4.7	Set_TriggerCount (TRIGGERCOUNT)	565
12.3.1.4.8	Get_TriggerCount (TRIGGERCOUNT)	566
12.3.1.4.9	Software_Trigger (TRIGGERSW)	566
12.3.1.5	Interfaces	566
12.3.1.5.1	Set_IPConfiguration (IPCONFIG)	566
12.3.1.5.2	Get_IPConfiguration (IPCONFIG)	567
12.3.1.5.3	Set_IPDataTransferMode (MEATRANSFER)	567

12.3.1.5.4	Get_IPDataTransferMode (MEATRANSFER)	568
12.3.1.5.5	Set_EthernetMode (ETHERMODE)	568
12.3.1.5.6	Get_EthernetMode (ETHERMODE)	568
12.3.1.5.7	Set_Baudrate (BAUDRATE)	569
12.3.1.5.8	Get_Baudrate (BAUDRATE)	569
12.3.1.6	Parameter management	569
12.3.1.6.1	Save_Parameters (STORE)	569
12.3.1.6.2	Load_Parameters (READ)	570
12.3.1.6.3	Set_Default (SETDEFAULT)	570
12.3.2	Measurement	571
12.3.2.1	General	571
12.3.2.1.1	Set_MeasureMode (MEASMODE)	571
12.3.2.1.2	Get_MeasureMode (MEASMODE)	571
12.3.2.1.3	Set_SearchDirection (SEARCHDIR)	571
12.3.2.1.4	Get_SearchDirection (SEARCHDIR)	572
12.3.2.1.5	Set_MeasureDirection (MEASDIR)	572
12.3.2.1.6	Get_MeasureDirection (MEASDIR)	572
12.3.2.1.7	Set_ExpectedEdges (EXPEDGES)	572
12.3.2.1.8	Get_ExpectedEdges (EXPEDGES)	573
12.3.2.1.9	Set_SegmentDefinition1 (DEFSEG1)	573
12.3.2.1.10	Get_SegmentDefinition1 (DEFSEG1)	573
12.3.2.1.11	Set_SegmentDefinition2 (DEFSEG2)	574
12.3.2.1.12	Get_SegmentDefinition2 (DEFSEG2)	574
12.3.2.1.13	Set_SegmentDefinition3 (DEFSEG3)	574
12.3.2.1.14	Get_SegmentDefinition3 (DEFSEG3)	575
12.3.2.1.15	Set_SegmentDefinition4 (DEFSEG4)	575
12.3.2.1.16	Get_SegmentDefinition4 (DEFSEG4)	575
12.3.2.1.17	Set_SegmentDefinition5 (DEFSEG5)	576
12.3.2.1.18	Get_SegmentDefinition5 (DEFSEG5)	576
12.3.2.1.19	Set_SegmentDefinition6 (DEFSEG6)	576
12.3.2.1.20	Get_SegmentDefinition6 (DEFSEG6)	577
12.3.2.1.21	Set_SegmentDefinition7 (DEFSEG7)	577
12.3.2.1.22	Get_SegmentDefinition7 (DEFSEG7)	577
12.3.2.1.23	Set_SegmentDefinition8 (DEFSEG8)	578
12.3.2.1.24	Get_SegmentDefinition8 (DEFSEG8)	578
12.3.2.1.25	Get_VideoStreamSignal	578
12.3.2.2	Video signal	580
12.3.2.2.1	Set_ROI (ROI)	580
12.3.2.2.2	Get_ROI (ROI)	580
12.3.2.2.3	Set_Threshold (THRESHOLD)	580
12.3.2.2.4	Get_Threshold (THRESHOLD)	581
12.3.2.3	Measurement value processing	581
12.3.2.3.1	Set_Averaging (AVERAGE)	581
12.3.2.3.2	Get_Averaging (AVERAGE)	582
12.3.2.3.3	Set_SpikeCorrection (SPIKECORR)	583
12.3.2.3.4	Get_SpikeCorrection (SPIKECORR)	583
12.3.2.3.5	Set_StatisticSignal (STATISTICSIGNAL)	584
12.3.2.3.6	Get_StatisticSignal (STATISTICSIGNAL)	585
12.3.2.3.7	Set_Statistic2Signal (STATISTIC2SIGNAL)	586
12.3.2.3.8	Get_Statistic2Signal (STATISTIC2SIGNAL)	587
12.3.2.3.9	Set_StatisticDepth (STATISTICDEPTH)	588
12.3.2.3.10	Get_StatisticDepth (STATISTICDEPTH)	588
12.3.2.3.11	Reset_Statistic (RESETSTATISTIC)	588

12.3.2.3.12 Set_EdgeFilter1 (EDGEFILTER1)	588
12.3.2.3.13 Get_EdgeFilter1 (EDGEFILTER1)	589
12.3.2.3.14 Set_EdgeFilter2 (EDGEFILTER2)	590
12.3.2.3.15 Get_EdgeFilter2 (EDGEFILTER1)	590
12.3.2.3.16 Set_EdgeFilter1Signal (EDGEFILTER1SIGNAL)	591
12.3.2.3.17 Get_EdgeFilter1Signal (EDGEFILTER1SIGNAL)	592
12.3.2.3.18 Set_EdgeFilter2Signal (EDGEFILTER2SIGNAL)	593
12.3.2.3.19 Get_EdgeFilter2Signal (EDGEFILTER2SIGNAL)	594
12.3.2.3.20 Set_MasterSignal (MASTERSIGNAL)	595
12.3.2.3.21 Get_MasterSignal (MASTERSIGNAL)	596
12.3.2.3.22 Set_MasterValue (MASTERMV)	597
12.3.2.3.23 Get_MasterValue (MASTERMV)	597
12.3.3 Data output	598
12.3.3.1 General	598
12.3.3.1.1 Set_DataOutInterface (OUTPUT)	598
12.3.3.1.2 Get_DataOutInterface (OUTPUT)	598
12.3.3.1.3 Set_Resampling (OUTREDUCE)	598
12.3.3.1.4 Get_Resampling (OUTREDUCE)	599
12.3.3.1.5 Set_HoldLastValid (OUTHOLD)	600
12.3.3.1.6 Get_HoldLastValid (OUTHOLD)	600
12.3.3.2 Selected measurement values	600
12.3.3.2.1 Set_OutputEdgeLightDark_RS422 (OUTEDGEHL_RS422)	600
12.3.3.2.2 Get_OutputEdgeLightDark_RS422 (OUTEDGEHL_RS422)	600
12.3.3.2.3 Set_OutputEdgeLightDark_ETH (OUTEDGEHL_ETH)	601
12.3.3.2.4 Get_OutputEdgeLightDark_ETH (OUTEDGEHL_ETH)	601
12.3.3.2.5 Set_OutputEdgeDarkLight_RS422 (OUTEDGELH_RS422)	601
12.3.3.2.6 Get_OutputEdgeDarkLight_RS422 (OUTEDGELH_RS422)	601
12.3.3.2.7 Set_OutputEdgeDarkLight_ETH (OUTEDGELH_ETH)	602
12.3.3.2.8 Get_OutputEdgeDarkLight_ETH (OUTEDGELH_ETH)	602
12.3.3.2.9 Set_OutputDiameter_RS422 (OUTDIA_RS422)	602
12.3.3.2.10 Get_OutputDiameter_RS422 (OUTDIA_RS422)	603
12.3.3.2.11 Set_OutputDiameter_ETH (OUTDIA_ETH)	603
12.3.3.2.12 Get_OutputDiameter_ETH (OUTDIA_ETH)	604
12.3.3.2.13 Set_OutputGap_RS422 (OUTGAP_RS422)	605
12.3.3.2.14 Get_OutputGap_RS422 (OUTGAP_RS422)	605
12.3.3.2.15 Set_OutputGap_ETH (OUTGAP_ETH)	606
12.3.3.2.16 Get_OutputGap_ETH (OUTGAP_ETH)	606
12.3.3.2.17 Set_OutputSegment_RS422 (OUTSEG_RS422)	607
12.3.3.2.18 Get_OutputSegment_RS422 (OUTSEG_RS422)	611
12.3.3.2.19 Set_OutputSegment_ETH (OUTSEG_ETH)	615
12.3.3.2.20 Get_OutputSegment_ETH (OUTSEG_ETH)	619
12.3.3.2.21 Set_OutputStatistic_RS422 (OUTSTATISTIC_RS422)	624
12.3.3.2.22 Get_OutputStatistic_RS422 (OUTSTATISTIC_RS422)	624
12.3.3.2.23 Set_OutputStatistic_ETH (OUTSTATISTIC_ETH)	625
12.3.3.2.24 Get_OutputStatistic_ETH (OUTSTATISTIC_ETH)	626
12.3.3.2.25 Set_OutputAdditional_RS422 (OUTADD_RS422)	627
12.3.3.2.26 Get_OutputAdditional_RS422 (OUTADD_RS422)	628
12.3.3.2.27 Set_OutputAdditional_ETH (OUTADD_ETH)	629
12.3.3.2.28 Get_OutputAdditional_ETH (OUTADD_ETH)	630
12.3.3.2.29 Set_OutputVideo_ETH (OUTVID_ETH)	630
12.3.3.2.30 Get_OutputVideo_ETH (OUTVID_ETH)	631
12.3.3.3 Switching outputs	632
12.3.3.3.1 Set_ErrorOutput1 (ERRORROUT1)	632

12.3.3.3.2	Get_ErrorOutput1 (ERROROUT1)	632
12.3.3.3.3	Set_ErrorOutput2 (ERROROUT2)	633
12.3.3.3.4	Get_ErrorOutput2 (ERROROUT2)	633
12.3.3.3.5	Set_ErrorLimit (ERRORLIMIT)	633
12.3.3.3.6	Get_ErrorLimit (ERRORLIMIT)	634
12.3.3.3.7	Set_ErrorLevelOut1 (ERRORLEVELOUT1)	636
12.3.3.3.8	Get_ErrorLevelOut1 (ERRORLEVELOUT1)	636
12.3.3.3.9	Set_ErrorLevelOut2 (ERRORLEVELOUT2)	636
12.3.3.3.10	Get_ErrorLevelOut2 (ERRORLEVELOUT2)	637
12.3.3.4	Analog output	637
12.3.3.4.1	Set_AnalogOutput (ANALOGOUT)	637
12.3.3.4.2	Get_AnalogOutput (ANALOGOUT)	638
12.3.3.4.3	Set_AnalogRange (ANALOGRANGE)	639
12.3.3.4.4	Get_AnalogRange (ANALOGRANGE)	639
12.3.3.4.5	Set_AnalogScale (ANALOGSCALE)	639
12.3.3.4.6	Get_AnalogScale (ANALOGSCALE)	640
12.4	Commands for ODC2600	641
12.4.1	General commands	641
12.4.1.1	General	641
12.4.1.1.1	Get_Info (INFO)	641
12.4.1.1.2	Reset_Boot (RESET)	643
12.4.1.2	Triggering	643
12.4.1.2.1	Triggermode_Reset (TRIGGERMODE RESET)	643
12.4.1.2.2	Triggermode_Trigger (TRIGGERMODE TRIGGER)	643
12.4.1.3	Parameter management	643
12.4.1.3.1	Save_OptionData (SAVE OPT RAM TO FLASH)	643
12.4.1.3.2	Save_MeasProgData (SAVE MPR RAM TO FLASH)	643
12.4.2	Measurement	643
12.4.2.1	Choose_MeasProg (CHOOSE MP)	643
12.4.2.2	Switch_Edge (SWITCH EDGE)	644
12.4.2.3	Write_OptionData (WR OPT TO RAM)	645
12.4.2.4	Read_OptionData (RD OPT RAM)	648
12.4.2.5	Update_OptionData	651
12.4.2.6	Write_MeasProgData (WR MPR TO RAM)	654
12.4.2.7	Read_MeasProgData (RD MPR RAM)	659
12.4.2.8	Set_LightRef (SET LIGHT REFERENCE TUNING)	663
12.4.2.9	Reset_LightRef (RESET LIGHT REFERENCE TUNING)	663
12.4.2.10	Read_MinMax (RD MINMAX)	663
12.4.2.11	Read_MinMax_Reset (RD MINMAX RESET)	664
12.4.3	Data output	665
12.4.3.1	General	665
12.4.3.1.1	Dat_Out_Off (STOP)	665
12.4.3.1.2	Dat_Out_On (START)	665
13	Commands for Eddy sensors	666
13.1	Commands for DT3100	666
13.1.1	Information	666
13.1.1.1	Get_Status (STS)	666
13.1.1.2	Get_Settings (SET)	667
13.1.1.3	Get_Error (ERR)	668
13.1.1.4	Get_ControllerInfo (IND)	668
13.1.1.5	Get_SensorInfo (SEN)	669
13.1.1.6	Get_CableInfo1 (CB1)	670

13.1.1.7	Get_CableInfo2 (CB2)	670
13.1.1.8	DetectSensorChange (DSC)	671
13.1.1.9	Set_TextField (ETF)	671
13.1.1.10	Get_TextField (ETF?)	671
13.1.2	Parameter Management	671
13.1.2.1	Save_Settings (SSE)	671
13.1.2.2	Read_Settings (RSE)	671
13.1.2.3	Set_Defaults (DSE)	672
13.1.3	Display	673
13.1.3.1	Set_AppLanguage (LNG)	673
13.1.3.2	Get_AppLanguage (LNG?)	673
13.1.3.3	Set_AppYScale (SCA)	673
13.1.3.4	Get_AppYScale (SCA?)	674
13.1.3.5	Set_AppDispMode (DMD)	674
13.1.3.6	Get_AppDispMode (DMD?)	674
13.1.4	Measurement	675
13.1.4.1	Set_SRIndex (SRA)	675
13.1.4.2	Get_SRIndex (SRA?)	675
13.1.4.3	Get_Measure (GMD)	675
13.1.4.4	Set_MeasureCnt (VTT)	675
13.1.4.5	Get_MeasureCnt (VTT?)	675
13.1.4.6	Set_MeasureMode (MMD)	676
13.1.4.7	Get_MeasureMode (MMD?)	676
13.1.4.8	Set_AvrType (AVT)	676
13.1.4.9	Get_AvrType (AVT?)	676
13.1.4.10	Set_AvrNbr (AVN)	677
13.1.4.11	Get_AvrNbr (AVN?)	677
13.1.4.12	Get_ControllerTemp (GCT)	677
13.1.4.13	Get_SensorTemp (GST)	677
13.1.5	Linearization	678
13.1.5.1	Get_CalibState (CST)	678
13.1.5.2	Set_CalibTarget (TAR)	678
13.1.5.3	Get_CalibTarget (TAR?)	678
13.1.5.4	Get_CalibProgress (BSY)	678
13.1.5.5	Set_CalibStart (SMR)	679
13.1.5.6	Set_CalibEnd (EMR)	679
13.1.5.7	Set_CalibMid (MMR)	679
13.1.5.8	Exit_Calib (ECA)	679
13.1.5.9	Set_FactoryCalib (FCA)	679
13.1.5.10	Set_Poti (WPT)	680
13.1.5.11	Get_Poti (RPT)	680
13.1.5.12	Save_Poti (SPT)	680
13.1.6	Internal commands	681
13.1.6.1	Update_Firmware	681
13.1.6.2	Get_BootLoaderMode (BOT)	681
13.1.6.3	Get_BootLoaderVer (VBL)	681
13.1.6.4	Get_FirmwareVer (VFW)	681
13.1.6.5	Start_BootLoader (SBL)	682
13.1.6.6	Erase_Flash (EFL)	682
13.1.6.7	Write_Sensor (WDS)	682
13.1.6.8	Read_Sensor (RDS)	682
13.1.6.9	Write_AnalogEEPROM (WA1+2)	682
13.1.6.10	Read_AnalogEEPROM (RA1+2)	682

13.1.6.11	Write_SensorParam (WSP)	683
13.1.6.12	Read_SensorParam (RSP)	683
13.1.6.13	Write_AnalogParam (WP1+2)	683
13.1.6.14	Read_AnalogParam (RP1+2)	683
13.1.6.15	Exec_SysCalibOff (SCO)	683
13.1.6.16	Exec_SysCalibGain (SCG)	684
13.1.6.17	Write_Index (WRI)	684
13.1.6.18	Read_Index (RDI)	684
13.1.6.19	Write_IndexDigital (WRD)	684
13.1.6.20	Read_IndexDigital (RDD)	684
13.1.6.21	Get_ButtonState (CBT)	685
13.1.6.22	Set_Multiplexer (MUX)	685
14	Commands for Capa sensors	686
14.1	Commands for DT6100	686
14.1.1	Measurement	686
14.1.1.1	Set_SRIndex (SRA)	686
14.1.1.2	Get_SRIndex (SRA?)	687
14.1.1.3	Set_Trigger (TRG)	687
14.1.1.4	Get_Trigger (TRG?)	687
14.1.1.5	Get_Measure (GMD)	687
14.1.1.6	Set_AvrType (AVT)	688
14.1.1.7	Get_AvrType (AVT?)	688
14.1.1.8	Set_AvrNbr (AVN)	688
14.1.1.9	Get_AvrNbr (AVN?)	688
14.1.2	Linearization	689
14.1.2.1	Set_LinMode (LIN)	689
14.1.2.2	Get_LinMode (LIN?)	689
14.1.2.3	Set_LinPoint (SLP)	689
14.1.2.4	Get_LinPoint (GLP)	690
14.1.3	Information	690
14.1.3.1	Get_Status (STS)	690
14.1.3.2	Get_Version (VER)	691
14.1.4	Parameter management	691
14.1.4.1	Save_Setup (SSU)	691
14.1.4.2	Read_Setup (RSU)	691
14.1.4.3	Factory_Defaults (FDE)	692
14.1.5	Internal commands	693
14.1.5.1	Set_Coefficient (SCO)	693
14.1.5.2	Get_Coefficient (GCO)	693
14.1.6	Special commands	693
14.1.6.1	Use_Defaults	693
14.1.6.2	Get_DrvSetting	694
14.2	Commands for DT6120	694
14.2.1	Measurement	694
14.2.1.1	Set_Samplerate	694
14.2.1.2	Get_Samplerate	695
14.2.1.3	Set_Averaging	695
14.2.1.4	Get_Averaging	696
14.2.1.5	Get_Measure	697
14.2.2	Data output	697
14.2.2.1	Set_Range	697
14.2.3	Interfaces	697

14.2.3.1	Set_Baudrate	697
14.2.3.2	Get_Baudrate	698
14.2.3.3	Set_SensorAddress	698
14.2.4	Information	698
14.2.4.1	Get_SensorInfo	698
14.2.4.2	Get_ControllerInfo	699
14.2.4.3	Read_AllBlocks	700
14.3	Commands for DT6200	700
14.3.1	User Level	700
14.3.1.1	Login (LGO)	700
14.3.1.2	Logout (LGO?)	700
14.3.1.3	Set_Password (PWD)	701
14.3.2	Measurement	701
14.3.2.1	Set_SampleTime (STI)	701
14.3.2.2	Get_SampleTime (STI?)	701
14.3.2.3	Set_Trigger (TRG)	702
14.3.2.4	Get_Trigger (TRG?)	702
14.3.2.5	Get_Measure (GMD)	702
14.3.2.6	Set_AvrType (AVT)	702
14.3.2.7	Get_AvrType (AVT?)	703
14.3.2.8	Set_AvrNbr (AVN)	703
14.3.2.9	Get_AvrNbr (AVN?)	703
14.3.2.10	Set_AnalogLowPass (ALP)	703
14.3.2.11	Get_AnalogLowPass (ALP?)	704
14.3.3	Data output	704
14.3.3.1	ChannelStatus (CHS)	704
14.3.3.2	Set_Range (MRA)	704
14.3.3.3	Get_RawDataRange (MDF)	704
14.3.3.4	Get_RawDataRanges	705
14.3.4	Math	705
14.3.4.1	Set_MathFunction (SMF)	705
14.3.4.2	Get_MathFunction (GMF)	706
14.3.4.3	Clr_MathFunction (CMF)	706
14.3.5	Interfaces	707
14.3.5.1	Set_DataPort (SDP)	707
14.3.5.2	Get_DataPort (GDP)	707
14.3.5.3	Set_IPConfiguration (IPS)	707
14.3.5.4	Get_IPConfiguration (IPS?)	708
14.3.5.5	Set_EthernetMode (IFC)	708
14.3.5.6	Get_EthernetMode (IFC?)	709
14.3.5.7	Set_AppLanguage (LNG)	709
14.3.5.8	Get_AppLanguage (LNG?)	709
14.3.6	Linearization	709
14.3.6.1	Set_LinMode (LIN)	709
14.3.6.2	Get_LinMode (LIN?)	710
14.3.6.3	Set_LinPoint (SLP)	710
14.3.6.4	Get_LinPointOffline (SLP)	711
14.3.6.5	Get_LinPoint (GLP)	711
14.3.7	Information	712
14.3.7.1	Get_Status (STS)	712
14.3.7.2	Get_Version (VER)	713
14.3.7.3	Get_ChannelInfo (CHI)	713
14.3.7.4	Get_ChannelInfos	714

14.3.7.5	Get_ControllerInfo (COI)	715
14.3.8	Internal commands	715
14.3.8.1	Update_Firmware	715
14.4	Commands for KSS6380	716
14.4.1	Measurement	716
14.4.1.1	Set_SRIndex (SRA)	716
14.4.1.2	Get_SRIndex (SRA?)	717
14.4.1.3	Set_Trigger (TRG)	717
14.4.1.4	Get_Trigger (TRG?)	717
14.4.1.5	Get_Measure (GMD)	717
14.4.1.6	Set_AvrType (AVT)	717
14.4.1.7	Get_AvrType (AVT?)	718
14.4.1.8	Set_AvrNbr (AVN)	718
14.4.1.9	Get_AvrNbr (AVN?)	718
14.4.2	Data output	718
14.4.2.1	ChannelStatus (CHS)	718
14.4.2.2	Set_Channel (CHT)	719
14.4.2.3	Get_Channel (CHT?)	719
14.4.3	Math	719
14.4.3.1	Set_MathFunction (SMF)	719
14.4.3.2	Get_MathFunction (GMF)	720
14.4.3.3	Clr_MathFunction (CMF)	720
14.4.4	Information	721
14.4.4.1	Get_Status (STS)	721
14.4.4.2	Get_Version (VER)	722
14.4.5	Parameter management	722
14.4.5.1	Save_Setup (SSU)	722
14.4.5.2	Read_Setup (RSU)	722
14.4.5.3	Factory_Defaults (FDE)	722
14.4.6	Internal commands	723
14.4.6.1	Set_TempCoeff (STC)	723
14.4.6.2	Get_TempCoeff (GTC)	723
14.4.6.3	Get_UnlinEddyVal (GUE)	724
14.4.6.4	Set_LinPoint (SLP)	724
14.4.6.5	Get_LinPoint (GLP)	725
14.4.7	Special commands	726
14.4.7.1	Use_Defaults	726
14.4.7.2	Get_DrvSetting	726
14.5	Commands for DT6500	726
14.5.1	User Level	727
14.5.1.1	Logout (LGO)	727
14.5.1.2	Login (LGI)	727
14.5.1.3	Set_Password (PWD)	727
14.5.2	Measurement	728
14.5.2.1	Set_SRIndex (SRA)	728
14.5.2.2	Get_SRIndex (SRA?)	728
14.5.2.3	Set_SampleTime (STI)	729
14.5.2.4	Get_SampleTime (STI?)	729
14.5.2.5	Set_Trigger (TRG)	729
14.5.2.6	Get_Trigger (TRG?)	730
14.5.2.7	Get_Measure (GMD)	730
14.5.2.8	Set_AvrType (AVT)	730
14.5.2.9	Get_AvrType (AVT?)	730

14.5.2.10 Set_AvrNbr (AVN)	731
14.5.2.11 Get_AvrNbr (AVN?)	731
14.5.3 Data output	731
14.5.3.1 ChannelStatus (CHS)	731
14.5.3.2 Set_Channel (CHT)	731
14.5.3.3 Get_Channel (CHT?)	732
14.5.3.4 Set_Range (MRA)	732
14.5.3.5 Get_RawDataRange (MDF)	732
14.5.3.6 Get_RawDataRanges	733
14.5.4 Display	733
14.5.4.1 Set_Display (DIS)	733
14.5.4.2 Get_Display (DIS?)	734
14.5.5 Math	734
14.5.5.1 Set_MathFunction (SMF)	734
14.5.5.2 Get_MathFunction (GMF)	735
14.5.5.3 Clr_MathFunction (CMF)	735
14.5.6 Interfaces	735
14.5.6.1 Set_DataPort (SDP)	735
14.5.6.2 Get_DataPort (GDP)	736
14.5.6.3 Set_IPConfiguration (IPS)	736
14.5.6.4 Get_IPConfiguration (IPS?)	737
14.5.6.5 Set_EthernetMode (IFC)	737
14.5.6.6 Get_EthernetMode (IFC?)	738
14.5.6.7 Set_AppLanguage (LNG)	738
14.5.6.8 Get_AppLanguage (LNG?)	738
14.5.7 Linearization	738
14.5.7.1 Set_LinMode (LIN)	738
14.5.7.2 Get_LinMode (LIN?)	739
14.5.7.3 Set_LinPoint (SLP)	739
14.5.7.4 Set_LinPointOffline (SLP)	740
14.5.7.5 Get_LinPoint (GLP)	740
14.5.8 Information	741
14.5.8.1 Get_Status (STS)	741
14.5.8.2 Get_Version (VER)	743
14.5.8.3 Get_ChannelInfo (CHI)	743
14.5.8.4 Get_ChannelInfos	744
14.5.8.5 Get_ControllerInfo (COI)	745
14.5.9 Parameter management	745
14.5.9.1 Save_Setup (SSU)	745
14.5.9.2 Read_Setup (RSU)	745
14.5.9.3 Factory_Defaults (FDE)	746
14.5.10 Internal commands	747
14.5.10.1 Update_Firmware	747
14.5.11 Special commands	748
14.5.11.1 Use_Defaults	748
14.5.11.2 Get_DrvSetting	748
15 Commands for Color (ACS) sensors	749
15.1 Commands for ACS7000	749
15.1.1 General commands	749
15.1.1.1 General	749
15.1.1.1.1 Get_Help (HELP)	749
15.1.1.1.2 Get_Info (GETINFO)	750

15.1.1.1.3	Set_Echo (ECHO)	751
15.1.1.1.4	Get_Echo (ECHO)	751
15.1.1.1.5	Get_AllParameters (PRINT)	751
15.1.1.1.6	Set_SyncMode (SYNC)	771
15.1.1.1.7	Get_SyncMode (SYNC)	771
15.1.1.1.8	Reset_Boot (RESET)	771
15.1.1.1.9	Set_Keylock (KEYLOCK)	771
15.1.1.1.10	Get_Keylock (KEYLOCK)	772
15.1.1.2	User Level	772
15.1.1.2.1	Logout (LOGOUT)	772
15.1.1.2.2	Login (LOGIN)	772
15.1.1.2.3	Get_UserLevel (GETUSERLEVEL)	773
15.1.1.2.4	Set_DefaultUser (STDUSER)	773
15.1.1.2.5	Get_DefaultUser (STDUSER)	773
15.1.1.2.6	Set_Password (PASSWD)	773
15.1.1.3	Sensor	774
15.1.1.3.1	Set_Observer (OBSERVER)	774
15.1.1.3.2	Get_Observer (OBSERVER)	774
15.1.1.3.3	Set_LightSource (LQSRC)	774
15.1.1.3.4	Get_LightSource (LQSRC)	775
15.1.1.3.5	Set_LEDControl (LEDCTRL)	775
15.1.1.3.6	Get_LEDControl (LEDCTRL)	775
15.1.1.3.7	Set_LEDIntensityColdWhite (LEDKW)	776
15.1.1.3.8	Get_LEDIntensityColdWhite (LEDKW)	776
15.1.1.3.9	Set_LEDIntensityGreen (LEDGR)	776
15.1.1.3.10	Get_LEDIntensityGreen (LEDGR)	776
15.1.1.3.11	Set_LEDIntensityWarmWhite (LEDWW)	776
15.1.1.3.12	Get_LEDIntensityWarmWhite (LEDWW)	777
15.1.1.3.13	Set_LEDIntensityViolet (LEDUV)	777
15.1.1.3.14	Get_LEDIntensityViolet (LEDUV)	777
15.1.1.3.15	AutoLEDAdjustment (AUTOLEDADJ)	777
15.1.1.3.16	DarkCorr (DARKCORR)	777
15.1.1.3.17	LightCorr (LIGHTCORR)	777
15.1.1.4	Triggering	778
15.1.1.4.1	Set_TriggerMode (TRIGGER)	778
15.1.1.4.2	Get_TriggerMode (TRIGGER)	778
15.1.1.4.3	Set_TriggerLevel (TRIGGERLEVEL)	778
15.1.1.4.4	Get_TriggerLevel (TRIGGERLEVEL)	778
15.1.1.4.5	Set_TriggerCount (TRIGGERCOUNT)	779
15.1.1.4.6	Get_TriggerCount (TRIGGERCOUNT)	779
15.1.1.4.7	Software_Trigger (TRIGGERSW)	779
15.1.1.5	Interfaces	779
15.1.1.5.1	Set_IPConfiguration (IPCONFIG)	779
15.1.1.5.2	Get_IPConfiguration (IPCONFIG)	780
15.1.1.5.3	Set_IPDataTransferMode (MEATRANSFER)	781
15.1.1.5.4	Get_IPDataTransferMode (MEATRANSFER)	781
15.1.1.5.5	Set_EthernetMode (ETHERMODE)	782
15.1.1.5.6	Get_EthernetMode (ETHERMODE)	782
15.1.1.5.7	Set_Baudrate (BAUDRATE)	782
15.1.1.5.8	Get_Baudrate (BAUDRATE)	783
15.1.1.5.9	Set_DigitalOutColorFormat (COLOROUT_FORMAT)	783
15.1.1.5.10	Get_DigitalOutColorFormat (COLOROUT_FORMAT)	783
15.1.1.5.11	Set_DigitalOutBinFormat (BIN_FORMAT)	784

15.1.1.5.12 Get_DigitalOutBinFormat (BIN_FORMAT)	784
15.1.1.6 Parameter management	784
15.1.1.6.1 Save_Parameters (STORE)	784
15.1.1.6.2 Load_Parameters (READ)	784
15.1.1.6.3 Set_Default (SETDEFAULT)	785
15.1.2 Measurement	785
15.1.2.1 General	785
15.1.2.1.1 Set_MeasureMode (MEASMODE)	785
15.1.2.1.2 Get_MeasureMode (MEASMODE)	786
15.1.2.1.3 Set_ShutterMode (SHUTTERMODE)	786
15.1.2.1.4 Get_ShutterMode (SHUTTERMODE)	786
15.1.2.1.5 Get_Video (GETVIDEO)	786
15.1.2.1.6 Set_Samplerate (MEASRATE)	787
15.1.2.1.7 Get_Samplerate (MEASRATE)	788
15.1.2.1.8 Set_DeltaMode (DELTAMODE)	788
15.1.2.1.9 Get_DeltaMode (DELTAMODE)	788
15.1.2.1.10 Set_DeltaKL (DELTA_KL)	788
15.1.2.1.11 Get_DeltaKL (DELTA_KL)	789
15.1.2.1.12 Set_DeltaKC (DELTA_KC)	789
15.1.2.1.13 Get_DeltaKC (DELTA_KC)	789
15.1.2.1.14 Set_DeltaKH (DELTA_KH)	789
15.1.2.1.15 Get_DeltaKH (DELTA_KH)	789
15.1.2.2 Color database	790
15.1.2.2.1 Get_ColorTable (COLORTABLE)	790
15.1.2.2.2 Edit_Color (COLORNEW)	792
15.1.2.2.3 Set_ColorDescription (COLORDESCR)	793
15.1.2.2.4 Get_ColorDescription (COLORDESCR)	793
15.1.2.2.5 Set_ColorThresholds (THRESHOLDS)	794
15.1.2.2.6 Get_ColorThresholds (THRESHOLDS)	794
15.1.2.2.7 Set_ColorSpace (COLORSPACE)	795
15.1.2.2.8 Get_ColorSpace (COLORSPACE)	795
15.1.2.2.9 Move_Color (MOVECOLOR)	795
15.1.2.2.10 Reset_ColorMapping (RESETMAPPING)	795
15.1.2.2.11 Delete_Color (COLORDELETE)	796
15.1.2.2.12 Clear_ColorTable	796
15.1.2.3 Measurement value processing	796
15.1.2.3.1 Set_VideoAverage (VSAVERAGE)	796
15.1.2.3.2 Get_VideoAverage (VSAVERAGE)	796
15.1.2.3.3 Set_Averaging (AVERAGE)	797
15.1.2.3.4 Get_Averaging (AVERAGE)	798
15.1.2.3.5 Set_StatisticSignal (STATISTICSIGNAL)	799
15.1.2.3.6 Get_StatisticSignal (STATISTICSIGNAL)	799
15.1.2.3.7 Set_StatisticDepth (STATISTICDEPTH)	800
15.1.2.3.8 Get_StatisticDepth (STATISTICDEPTH)	800
15.1.2.3.9 Reset_Statistic (RESETSTATISTIC)	800
15.1.3 Data output	801
15.1.3.1 General	801
15.1.3.1.1 Set_DataOutInterface (OUTPUT)	801
15.1.3.1.2 Get_DataOutInterface (OUTPUT)	801
15.1.3.1.3 Set_Resampling (OUTREDUCE)	801
15.1.3.1.4 Get_Resampling (OUTREDUCE)	802
15.1.3.1.5 Set_HoldLastValid (OUTHOLD)	802
15.1.3.1.6 Get_HoldLastValid (OUTHOLD)	802

15.1.3.2 Selected measurement values	803
15.1.3.2.1 Set_OutputVideo_ETH (OUTVIDEO)	803
15.1.3.2.2 Get_OutputVideo_ETH (OUTVIDEO)	803
15.1.3.2.3 Set_OutputColor_ETH (OUTCOLOR_ETH)	804
15.1.3.2.4 Get_OutputColor_ETH (OUTCOLOR_ETH)	805
15.1.3.2.5 Set_OutputColor_RS422 (OUTCOLOR_RS422)	806
15.1.3.2.6 Get_OutputColor_RS422 (OUTCOLOR_RS422)	807
15.1.3.2.7 Set_DistanceMode (DISTANCEMODE)	808
15.1.3.2.8 Get_DistanceMode (DISTANCEMODE)	808
15.1.3.2.9 Set_OutputDistance_ETH (OUTDIST_ETH)	808
15.1.3.2.10 Get_OutputDistance_ETH (OUTDIST_ETH)	811
15.1.3.2.11 Set_OutputDistance_RS422 (OUTDIST_RS422)	813
15.1.3.2.12 Get_OutputDistance_RS422 (OUTDIST_RS422)	814
15.1.3.2.13 Set_OutputStatistic_ETH (OUTSTATISTIC_ETH)	814
15.1.3.2.14 Get_OutputStatistic_ETH (OUTSTATISTIC_ETH)	815
15.1.3.2.15 Set_OutputStatistic_RS422 (OUTSTATISTIC_RS422)	815
15.1.3.2.16 Get_OutputStatistic_RS422 (OUTSTATISTIC_RS422)	816
15.1.3.2.17 Set_DigitalOutDistance (OUTDIST_COLOROUT)	816
15.1.3.2.18 Get_DigitalOutDistance (OUTDIST_COLOROUT)	817
15.1.3.2.19 Set_OutputStatus_ETH (OUTSTATUS_ETH)	817
15.1.3.2.20 Get_OutputStatus_ETH (OUTSTATUS_ETH)	818
15.1.3.2.21 Set_OutputStatus_RS422 (OUTSTATUS_RS422)	820
15.1.3.2.22 Get_OutputStatus_RS422 (OUTSTATUS_RS422)	821
16 Commands for Interfaces	824
16.1 Commands for ETH_IF1032	824
16.1.1 User Level	824
16.1.1.1 Logout (LGO)	824
16.1.1.2 Login (LGI)	824
16.1.1.3 Set_Password (PWD)	824
16.1.2 Measurement	825
16.1.2.1 Set_SampleTime (STI)	825
16.1.2.2 Get_SampleTime (STI?)	825
16.1.2.3 Set_Trigger (TRG)	825
16.1.2.4 Get_Trigger (TRG?)	826
16.1.2.5 Set_AvrType (AVT)	826
16.1.2.6 Get_AvrType (AVT?)	826
16.1.2.7 Set_AvrNbr (AVN)	826
16.1.2.8 Get_AvrNbr (AVN?)	827
16.1.3 Data output	827
16.1.3.1 ChannelStatus (CHS)	827
16.1.3.2 Get_RawDataRange (MDF)	827
16.1.3.3 Get_RawDataRanges	828
16.1.4 Interfaces	828
16.1.4.1 Set_DataPort (SDP)	828
16.1.4.2 Get_DataPort (GDP)	828
16.1.4.3 Set_IPConfiguration (IPS)	829
16.1.4.4 Get_IPConfiguration (IPS?)	829
16.1.4.5 Set_EthernetMode (IFC)	830
16.1.4.6 Get_EthernetMode (IFC?)	830
16.1.4.7 Set_AppLanguage (LNG)	830
16.1.4.8 Get_AppLanguage (LNG?)	831
16.1.5 Sensor interface	831

16.1.5.1	Set_SensorInterface (SIF)	831
16.1.5.2	Get_SensorInterface (SIF?)	831
16.1.5.3	Set_SensorBaudrate (SBR)	831
16.1.5.4	Get_SensorBaudrate (SBR?)	832
16.1.5.5	Set_SensorAddress (SAD)	832
16.1.5.6	Get_SensorAddress (SAD?)	832
16.1.5.7	Set_AnalogRange (ARA)	833
16.1.5.8	Get_AnalogRange (ARA?)	833
16.1.5.9	Set_AnalogOffset (AOF)	833
16.1.5.10	Get_AnalogOffset (AOF?)	834
16.1.5.11	Set_AnalogUnit (AUN)	834
16.1.5.12	Get_AnalogUnit (AUN?)	835
16.1.5.13	Set_AnalogMathFunction (AMF)	836
16.1.5.14	Get_AnalogMathFunction (AMF?)	836
16.1.6	Information	837
16.1.6.1	Get_Status (STS)	837
16.1.6.2	Get_Version (VER)	837
16.1.6.3	Get_ChannelInfo (CHI)	838
16.1.6.4	Get_ChannelInfos	838
16.1.6.5	Get_ControllerInfo (COI)	839
16.1.7	Internal commands	840
16.1.7.1	Update_Firmware	840
16.2	Commands for USBAdapter_IF2004	840
16.2.1	Special commands	841
16.2.1.1	Set_SyncMaster	841
16.2.1.2	Set_Algorithm	841
16.2.1.3	Get_Algorithm	841
16.2.1.4	Set_SelectedSensors	842
16.2.1.5	Get_SelectedSensors	842
16.2.1.6	Set_SensorChannelBaudrate	842
16.2.1.7	Get_SensorChannelBaudrate	843
16.2.2	Parameter management	843
16.2.2.1	Save_Parameters	843
16.2.2.2	Load_Parameters	843
16.2.3	Information	843
16.2.3.1	Get_FPGAVersion	843
16.2.3.2	Get_HWRevision	844
16.2.3.3	Get_Option	844
16.2.4	Timer	844
16.2.4.1	Set_TimerFrequency	844
16.2.4.2	Get_TimerFrequency	845
16.2.5	RS422 (Sensor)	845
16.2.5.1	Inputs	845
16.2.5.1.1	Use_Gate	845
16.2.5.1.2	Get_RxValue	846
16.2.5.2	Outputs	846
16.2.5.2.1	Set_TxDSource	846
16.2.5.2.2	Get_TxDSource	847
16.2.5.3	Trigger	847
16.2.5.3.1	Set_TrgSource	847
16.2.5.3.2	Get_TrgSource	847
16.2.6	RS422 (Computer)	848
16.2.6.1	Set_DataMode	848

16.2.6.2	Get_DataMode	848
16.2.6.3	Set_RS422OutSource	848
16.2.6.4	Get_RS422OutSource	849
16.2.6.5	Set_RS422Baudrate	849
16.2.6.6	Get_RS422Baudrate	849
16.2.7	Switching inputs	850
16.2.7.1	Set_ExtTriggerInSource	850
16.2.7.2	Get_ExtTriggerInSource	850
16.2.7.3	Set_ExtTriggerInDirection	850
16.2.7.4	Get_ExtTriggerInDirection	851
16.2.7.5	Get_ExtTriggerInValue	851
16.2.8	Switching outputs	851
16.2.8.1	Set_ExtTriggerOutSource	851
16.2.8.2	Get_ExtTriggerOutSource	852
16.2.9	LED	852
16.2.9.1	Set_LEDMode	852
16.2.9.2	Get_LEDMode	853
16.3	Commands for PCICardIF2004	854
16.3.1	Special commands	854
16.3.1.1	Use_Defaults	854
16.3.1.2	Get_DrvSetting	855
16.3.1.3	Set_SyncMaster	856
16.3.2	General	856
16.3.2.1	Get_FPGAVersion	856
16.3.2.2	IF2004_SystemReset	856
16.3.2.3	IF2004_SensorReset12	856
16.3.2.4	IF2004_SensorReset34	857
16.3.3	Encoder	857
16.3.3.1	Actions	857
16.3.3.1.1	Enc_ClearCounter	857
16.3.3.1.2	Enc_SetLoadReg	857
16.3.3.1.3	Enc_LoadCounter	857
16.3.3.1.4	Enc_LatchCounter	857
16.3.3.1.5	Enc_GetLatchReg	857
16.3.3.1.6	Enc_UnlockTraceC	858
16.3.3.1.7	Enc_CheckRef	858
16.3.3.1.8	Enc_IsFirstRef	858
16.3.3.1.9	Enc_IsSecondRef	858
16.3.4	RS422	858
16.3.4.1	Inputs	858
16.3.4.1.1	IF2004_CheckGate	858
16.4	Commands for PCICard_IF2008	859
16.4.1	Special commands	859
16.4.1.1	Use_Defaults	859
16.4.1.2	Get_DrvSetting	860
16.4.1.3	Set_SyncMaster	861
16.4.1.4	Set_PowerSwitch	862
16.4.2	Information	862
16.4.2.1	Get_FPGAVersion	862
16.4.2.2	Get_HWRevision	862
16.4.2.3	Is_Channel56Available	862
16.4.2.4	Is_ADCAvailable	863
16.4.2.5	Is_DigitalIOAvailable	863

16.4.2.6	Get_PowerError	863
16.4.3	Timer	863
16.4.3.1	Set_TimerFrequency	863
16.4.4	Encoder	864
16.4.4.1	Settings	864
16.4.4.1.1	Set_EncoderInterpolation	864
16.4.4.1.2	Get_EncoderInterpolation	865
16.4.4.1.3	Set_EncoderDirection	866
16.4.4.1.4	Get_EncoderDirection	866
16.4.4.1.5	Set_EncoderMode	866
16.4.4.1.6	Get_EncoderMode	867
16.4.4.1.7	Set_EncoderLatchSource	867
16.4.4.1.8	Get_EncoderLatchSource	868
16.4.4.2	Actions	869
16.4.4.2.1	Clear_Encoder	869
16.4.4.2.2	Set_EncoderPreload	869
16.4.4.2.3	Load_Encoder	869
16.4.4.2.4	Latch_Encoder	869
16.4.4.2.5	Get_EncoderValue	870
16.4.4.2.6	EnableRef_Encoder	870
16.4.4.2.7	Get_EncoderReference	870
16.4.5	RS422	871
16.4.5.1	Inputs	871
16.4.5.1.1	Use_Gate	871
16.4.5.1.2	Get_RxValue	872
16.4.5.2	Outputs	872
16.4.5.2.1	Set_TxDSource	872
16.4.5.2.2	Get_TxDSource	872
16.4.5.2.3	Set_TxValue	872
16.4.5.2.4	Get_TxValue	873
16.4.5.3	Trigger	873
16.4.5.3.1	Set_TrgSource	873
16.4.5.3.2	Get_TrgSource	873
16.4.5.3.3	Set_TrgValue	874
16.4.5.3.4	Get_TrgValue	874
16.4.6	Switching inputs	874
16.4.6.1	Set_DigitalInLatchSource	874
16.4.6.2	Get_DigitalInLatchSource	875
16.4.6.3	Get_DigitalInValue	875
16.4.7	Switching outputs	875
16.4.7.1	Set_DigitalOutSource	875
16.4.7.2	Get_DigitalOutSource	876
16.4.7.3	Set_DigitalOutValue	876
16.4.7.4	Get_DigitalOutValue	876
16.4.8	Analog inputs	877
16.4.8.1	Set_ADCLatchSource	877
16.4.8.2	Get_ADCLatchSource	877
16.4.8.3	Get_ADCValue	878
16.5	Commands for CSP2008	879
16.5.1	Parameter management	879
16.5.1.1	Load_Parameters (LOADSETUP)	879
16.5.1.2	Save_Parameters (STORESETUP)	879
16.5.2	Mastering	880

16.5.2.1	Set_MasterValue (SWMASTER)	880
16.5.2.2	Reset_MasterValue (SWREMASTER)	880
16.5.3	Software trigger	880
16.5.3.1	Set_SoftwareTriggerEnable (SWTRIGGERENABLE)	880
16.5.3.2	Get_SoftwareTriggerEnable (SWTRIGGERENABLE)	880
16.5.3.3	Set_SoftwareTrigger (SWTRIGGER)	880
16.5.3.4	Get_SoftwareTrigger (SWTRIGGER)	881
16.5.4	Special commands	881
16.5.4.1	Use_Defaults	881
16.5.4.2	Get_DrvSetting	882
16.6	Commands for C-Box	882
16.6.1	General commands	883
16.6.1.1	General	883
16.6.1.1.1	Get_Info (GETINFO)	883
16.6.1.1.2	Set_SyncMode (SYNC)	884
16.6.1.1.3	Get_SyncMode (SYNC)	884
16.6.1.1.4	Reset_Boot (RESET)	885
16.6.1.1.5	Get_ScanStatus (SCANSTATUS)	885
16.6.1.1.6	Get_AllParameters (PRINT)	885
16.6.1.2	Triggering	896
16.6.1.2.1	Set_TriggerMode (TRIGGER)	896
16.6.1.2.2	Get_TriggerMode (TRIGGER)	896
16.6.1.2.3	Set_TriggerLevel (TRIGGERLEVEL)	897
16.6.1.2.4	Get_TriggerLevel (TRIGGERLEVEL)	897
16.6.1.2.5	Set_TriggerCount (TRIGGERCOUNT)	897
16.6.1.2.6	Get_TriggerCount (TRIGGERCOUNT)	898
16.6.1.2.7	Software_Trigger (TRIGGERSW)	898
16.6.1.3	Interfaces	898
16.6.1.3.1	Set_IPConfiguration (IPCONFIG)	898
16.6.1.3.2	Get_IPConfiguration (IPCONFIG)	899
16.6.1.3.3	Set_IPDataTransferMode (MEASTRANSFER)	899
16.6.1.3.4	Get_IPDataTransferMode (MEASTRANSFER)	900
16.6.1.3.5	Set_Baudrate (BAUDRATE)	900
16.6.1.3.6	Get_Baudrate (BAUDRATE)	900
16.6.1.3.7	Set_AppLanguage (LANGUAGE)	901
16.6.1.3.8	Get_AppLanguage (LANGUAGE)	901
16.6.1.4	Sensors	901
16.6.1.4.1	Scan_Sensor1 (SCAN1)	901
16.6.1.4.2	Scan_Sensor2 (SCAN2)	901
16.6.1.4.3	Get_Info1 (GETINFO1)	902
16.6.1.4.4	Get_Info2 (GETINFO2)	902
16.6.1.4.5	Set_LaserPower1 (LASERPOW1)	902
16.6.1.4.6	Get_LaserPower1 (LASERPOW1)	902
16.6.1.4.7	Set_LaserPower2 (LASERPOW2)	902
16.6.1.4.8	Get_LaserPower2 (LASERPOW2)	903
16.6.1.4.9	Set_Averaging1 (AVERAGE1)	903
16.6.1.4.10	Get_Averaging1 (AVERAGE1)	904
16.6.1.4.11	Set_Averaging2 (AVERAGE2)	904
16.6.1.4.12	Get_Averaging2 (AVERAGE2)	905
16.6.1.4.13	SensorCommand1 (TUNNEL1)	905
16.6.1.4.14	SensorCommand2 (TUNNEL2)	906
16.6.1.4.15	Get_SensorInstance	906
16.6.1.5	Parameter management	906

16.6.1.5.1	Save_Parameters (STORE)	906
16.6.1.5.2	Load_Parameters (READ)	907
16.6.1.5.3	Set_Default (SETDEFAULT)	907
16.6.2	Measurement	907
16.6.2.1	General	907
16.6.2.1.1	Set_MeasureMode (MEASMODE)	907
16.6.2.1.2	Get_MeasureMode (MEASMODE)	908
16.6.2.1.3	Set_Samplerate (MEASRATE)	908
16.6.2.1.4	Get_Samplerate (MEASRATE)	908
16.6.2.2	Measurement value processing	908
16.6.2.2.1	Set_Averaging (AVERAGE)	908
16.6.2.2.2	Get_Averaging (AVERAGE)	909
16.6.2.2.3	Set_MasterValue (MASTERMV)	910
16.6.2.2.4	Get_MasterValue (MASTERMV)	911
16.6.3	Data output	911
16.6.3.1	General	911
16.6.3.1.1	Set_DataOutInterface (OUTPUT)	911
16.6.3.1.2	Get_DataOutInterface (OUTPUT)	911
16.6.3.1.3	Set_Resampling (OUTREDUCE)	912
16.6.3.1.4	Get_Resampling (OUTREDUCE)	912
16.6.3.1.5	Set_HoldLastValid (OUTHOLD)	913
16.6.3.1.6	Get_HoldLastValid (OUTHOLD)	913
16.6.3.1.7	Set_OutputScale_RS422_USB (OUTSCALE_RS422_USB)	914
16.6.3.1.8	Get_OutputScale_RS422_USB (OUTSCALE_RS422_USB)	914
16.6.3.2	Selected measurement values	915
16.6.3.2.1	Set_Output_RS422 (OUT_RS422)	915
16.6.3.2.2	Get_Output_RS422 (OUT_RS422)	916
16.6.3.2.3	Set_Output_USB (OUT_USB)	916
16.6.3.2.4	Get_Output_USB (OUT_USB)	917
16.6.3.2.5	Set_Output_ETH (OUT_ETH)	918
16.6.3.2.6	Get_Output_ETH (OUT_ETH)	919
16.6.3.2.7	Set_OutputAdditional (OUT_ADDITIONAL)	920
16.6.3.2.8	Get_OutputAdditional (OUT_ADDITIONAL)	920
16.6.3.3	Analog output	920
16.6.3.3.1	Set_AnalogOutput (ANALOGOUT)	920
16.6.3.3.2	Get_AnalogOutput (ANALOGOUT)	921
16.6.3.3.3	Set_AnalogRange (ANALOGRANGE)	921
16.6.3.3.4	Get_AnalogRange (ANALOGRANGE)	922
16.6.3.3.5	Set_AnalogScale (ANALOGSCALE)	922
16.6.3.3.6	Get_AnalogScale (ANALOGSCALE)	922
16.6.4	Internal commands	923
16.6.4.1	Get_FirmwareVersion	923
16.6.4.2	Prepare_UpdateFirmware	923
16.6.4.3	Start_UpdateFirmware	924
16.6.4.4	Get_UpdateFirmwareProgress	924
16.6.4.5	Generate_Firmware	924
Index		925

1 Introduction

MEDAQLib is a software library for easy data acquisition and communication with digital Micro-Epsilon sensors. The library is independent from any communication protocol or hardware interface, i.e. all sensors or controllers are accessed from your program in the same way independent whether via TCP/IP or USB or serial communication.

Currently the following sensors are supported:

ILR sensors

[optoNCDT ILR 110x](#), [optoNCDT ILR 115x](#), [optoNCDT ILR 118x](#), [optoNCDT ILR 1191](#)

ILD sensors

[optoNCDT 1302](#), [optoNCDT 1320](#), [optoNCDT 1401](#), [optoNCDT 1402](#), [optoNCDT 1420](#), [optoNCDT 1700](#), [optoNCDT 2200](#) (including [optoNCDT 2220](#)), [optoNCDT 2300](#)

IFD sensors

[confocalDT 2401](#), [confocalDT 2431](#), [confocalDT 2445](#), [confocalDT 2451](#), [confocalDT 2461](#), [confocalDT 2471](#)

ODC sensors

[optoCONTROL 1202](#) (including [optoCONTROL 1220](#)), [optoCONTROL 2500](#), [optoCONTROL 2520](#), [optoCONTROL 2600](#)

Eddy sensors

[eddyNCDT 3100](#)

Capa sensors

[capaNCDT 6100](#), [capaNCDT 6120](#), [capaNCDT 6200](#) (including [capaNCDT 6220](#) and [6230](#)), [combiSENSOR 6380](#), [capaNCDT 6500](#) (including [capaNCDT 6530](#))

Color sensors

[colorCONTROL ACS7000](#)

Interface module

[Eth_IF1032](#)

USB adapters

[IF2004_USB](#) adapter

PCI cards

[IF2004](#) and [IF2008](#) PCI card

Universal controller

[CSP2008](#) and C-Box

The sensors can be accessed through interfaces [RS232](#), USB ([IF2001_USB](#) (RS422) and RS232 high level interface), USB (via [WinUSB](#)), [IF2004](#) and [IF2008](#) PCI card (RS422), [IF2004_USB](#) adapter (RS422) and [TCP/IP](#).

The software consists of a DLL to be imported into your data acquisition project. As programming languages C/C++, Visual Basic, Delphi and many other languages are supported. For C/C++ an additional include file is provided to get you started.

1 Introduction

More than one sensor can be used over different interfaces at the same time (from one or several applications).

The functions of MEDAQLib are thread-safe, so you can call any function at the same time from concurrent threads.

Additionally Micro-Epsilon provides ICONNECT having special modules to access all the features of these sensors. ICONNECT is a programming environment giving you the possibility to develop applications without the need to "real programming".

2 Installation

MEDAQLib comes as setup executable. After installation to your destination folder the following directory structure and files will be created:

Release/MEDAQLib.dll

the 32 bit driver dll.

Release/MEDAQLib.lib

linker library for Visual C/C++ projects.

Release/MEDAQLib.NET.dll

the 32 bit assembly for using MEDAQLib in each .NET language.

Release/SensorTest.exe

simple 32 bit command line test program for any sensor (see Samples/[SensorTest](#)).

Release-x64/MEDAQLib.dll

the 64 bit version of driver dll.

Release-x64/MEDAQLib.lib

linker library 64 bit version for Visual C/C++ projects.

Release/MEDAQLib.NET.dll

the 64 bit assembly for using MEDAQLib in each .NET language.

Release-x64/SensorTest.exe

simple 64 bit command line test program for any sensor (see Samples/[SensorTest](#))

MEDAQLib.h

include file for C/C++ projects.

MEDAQLib.bas

module for easy access MEDAQLib from VB6 or VBA.

Documentation/MEDAQLib.pdf

the reference guide you are currently reading.

Documentation/Version History.txt

changes between the different released versions.

Documentation/Medaqlib-Quick_Reference.pdf

the quick start reference in english language.

Documentation/Medaqlib-Schnellstartanleitung.pdf

the quick start reference in german language.

Samples

examples showing usage of MEDAQLib.

Nothing is installed into any Windows system directory. Nothing is changed in Windows registry.

In addition the following drivers are also copied to installation folder (but not installed yet):

2 Installation

Driver/WinUSB for accessing the [SENSOR_IFD2401/SENSOR_IFD2431](#) and [CONTROLLER_CBOX](#) over [WinUSB](#).

Driver/IF2004 for accessing sensors over the [PCI_CARD_IF2004](#) PCI card.

Driver/IF2008 for accessing sensors over the [PCI_CARD_IF2008](#) PCI card.

Driver/FTDI the driver for [USB_ADAPTER_IF2004](#) and the IF2001_USB (RS422).

To install a driver follow the driver wizard and manually select the directory.
These drivers are also included in the driver CD supplied with your hardware.

3 Accessing MEDAQLib in Visual C/C++

3.1 Setting up Visual Studio

For accessing MEDAQLib from Visual Studio copy the file MEDAQLib.h into your project directory, add it to your project (Project->Add Files) and include it in your C/C++ source file as follows:

```
#include "MEDAQLib.h"
```

Now you are able to compile your project accessing MEDAQLib. For linking in MEDAQLib.dll there are two possibilities:

3.1.1 Using MEDAQLib.lib, static approach

Copy the file MEDAQLib.lib into your project directory and add it to your project (Project->Add Files). Now you are able to also link your project.

3.1.2 Using MEDAQLib.dll, dynamic approach

Do not link your project against MEDAQLib.lib. For accessing the functions you have to load the MEDAQLib.dll using

```
HINSTANCE hInstance= LoadLibrary ("MEDAQLib.dll");
```

and get the functions pointers like

```
CREATESENSORINSTANCE pCreateSensorInstance= (CREATESENSORINSTANCE) GetProcAddress
(hInstance, "CreateSensorInstance");

RELEASESENSORINSTANCE pReleaseSensorInstance= (RELEASESENSORINSTANCE) GetProcAddress
(hInstance, "ReleaseSensorInstance");
```

Please check any return codes after calling these functions, which is not shown here. Calling a function then is done like

```
DWORD result= pCreateSensorInstance (SENSOR_IID2300);
```

4 Using MEDAQLib

In the following some programming examples in different programming languages are shown to give a first introduction into the main working structure of a MEDAQLib based program. Please note that these code examples contain no error handling, so they are not meant for production usage.

4.1 Main structure of a MEDAQLib program

As a first step the information about sensor and its interface have to be given to MEDAQLib. Also, it is a good idea to enable logfile writing by MEDAQLib.

```
// Tell MEDAQLib about sensor type to be used.
HANDLE hSensor = CreateSensorInstance(SENSOR_ILD2300);

// Tell MEDAQLib about interface to be used.
SetParameterString(hSensor, "IP_Interface", "TCP/IP");
SetParameterString(hSensor, "IP_RemoteAddr", "169.254.168.150");

// Enable Logfile writing
SetParameterInt (hSensor, "IP_EnableLogging", 1);
SetParameterString(hSensor, "IP_LogFile", "c:/MEDAQLib-log.txt");

// Try to open communication to sensor via interface specified.
err= OpenSensor(hSensor);
```

After having successfully opened communication with the sensor it is possible to acquire data from sensor. The following sample shows continuous data acquistion from sensor. It will always ask for 200 values.

```
// In the following we try to always get 200 values from sensor.
//
#define EXPECTED_BLOCK_SIZE 200; // please adjust to your setting.

while(!bDone)
{
    // Check whether there's enough data to read in.
    int currentlyAvailable = 0;
    err= DataAvail(hSensor, &currentlyAvailable);

    if (currentlyAvailable > EXPECTED_BLOCK_SIZE)
    {
        // Allocate memory to get raw and scaled data into.
        int rawData[EXPECTED_BLOCK_SIZE];
        double scaledData[EXPECTED_BLOCK_SIZE];

        // Set additional parameters for TransferData;
        int expectedBlockSize = EXPECTED_BLOCK_SIZE;
        int gotBlockSize = 0;

        // Fetch data from MEDAQLib's internal buffer.
        err= TransferData(hSensor, rawData, scaledData, expectedBlockSize, &gotBlockSize);

        // Now expectedBlockSize should be equal to gotBlockSize.
        // rawData contains original values from sensor, scaledData contains scaled data.
        // Do your computation on data ....
    }

    Sleep(10); // Sleep 10ms, allow other things to happen, ....
}
```

At end of program the communication channel and all MEDAQLib internal resources have to be returned to system.

```
// Close down by closing interface and releasing sensor instance.
CloseSensor(hSensor);
ReleaseSensorInstance(hSensor);
```

4.2 How to call a function at sensor

The following sample code shows how to call a specific function at a sensor. For the example we assume the connection to the sensor was already established by having called `CreateSensorInstance` and `OpenSensor`.

The example is trying to get information about the current measurement mode of an ILD2300:

```
// Get measure mode of sensor.
int iMeasureMode; /* will receive current measure mode */
SetParameterString(hSensor, "S_Command", "Get_MeasureMode");
err= SensorCommand(hSensor);
GetParameterInt(hSensor, "SA_MeasureMode", &iMeasureMode);
```

First the command has to be given to `S_Command`. The call to `SensorCommand` executes the command at the sensor itself. After having executed the command at the sensor the application is able to receive result values (sensor answers, `SA_xxx`) from MEDAQLib.

To set a value at sensor again first give the command to `S_Command`. Then the parameters for this call (sensor parameters, `SP_xxx`) have to be given to MEDAQLib. Only after this information is given to MEDAQLib a successful execution of the function at the sensor is possible:

```
// And set it to distance mode ( = 0, see Set_MeasureMode)
SetParameterString(hSensor, "S_Command", "Set_MeasureMode");
SetParameterInt(hSensor, "SP_MeasureMode", 0/* = Distance */);
err= SensorCommand(hSensor);
```

These two examples above showed the principal functions to be called. For functions expecting a single argument only MEDAQLib provides short cut functions, so the above code may be written shorter like follows:

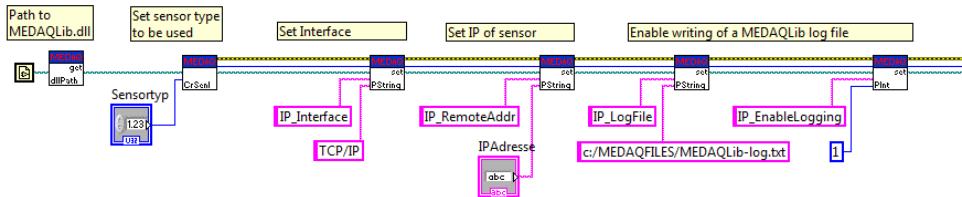
```
err= ExecSCmdGetInt(hSensor, "Get_MeasureMode", "SA_MeasureMode", &iMeasureMode);
err= SetIntExecSCmd(hSensor, "Set_MeasureMode", "SP_MeasureMode", 0/* = Distance */);
```

4.3 Main structure in a LabView Sample

The following sample shows how to implement the basic MEDAQLib application using LabView. Please note again that any error handling is omitted for clarity of structure.

The first LabView picture shows how to tell MEDAQLib which sensor type to use by calling `CreateSensorInstance`, how to specify the communication interface and how to enable log file creation.

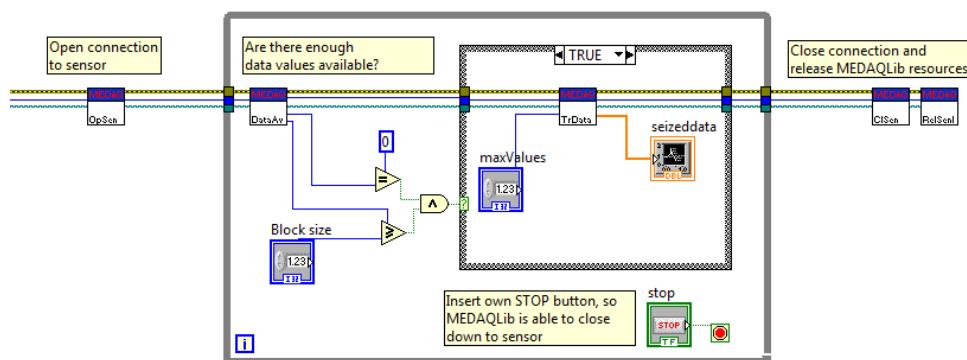
4 Using MEDAQLib



Up to this point only information is given to MEDAQLib, no communication is established to a sensor.

The second picture starts with a call to `OpenSensor`. Now the settings from the first picture are used to try to establish a connection to the sensor using the interface specified. And only now is possible for MEDAQLib to detect any errors in the settings given.

The right part shows a `DataAvail` and `TransferData` loop to get data from sensor. At the end the interface used is closed by calling `CloseSensor` and the MEDAQLib internal resources are released by calling function `ReleaseSensorInstance`.



The FALSE part of the switch structure is empty and not shown. The additional STOP button is added as on pressing the LabView red stop button gives MEDAQLib no possibility to release any sensor related interface resulting then in a sensor meaning it is still connected to some application. On the next start it will refuse a new connection. Please note that also not implemented is any sleep/wait for new data as this is application dependent.

4.4 Synchronized data from two ILD2300 at IF2008

This example synchronizes two `SENSOR_ILD2300` sensors by `IF2008` card and reads data.

Please ensure that both sensors has equal settings.

For readability this example does not handle errors. Please add error handling to your code.

Create sensor and `IF2008` card instances and open them.

4 Using MEDAQLib

```

DWORD err;
DWORD sensor1= CreateSensorInstByName ("ILD2300");
DWORD sensor2= CreateSensorInstByName ("ILD2300");
DWORD ifCard= CreateSensorInstByName ("IF2008");
err= OpenSensorIF2008 (sensor1, 0, 0); /* First card, first channel */
err= OpenSensorIF2008 (sensor2, 0, 1); /* First card, second channel */
err= OpenSensorIF2008 (ifCard, 0, -1); /* First card, no data channel */

```

Now the sensors must be synchronized. The synchronization master is the **IF2008** card.

```

/* Set both sensors the same samplerate */
double samplerate= 10; // kHz
err= SetDoubleExecSCmd (sensor1, "Set_Samplerate", "SP_Measrate", samplerate);
err= SetDoubleExecSCmd (sensor2, "Set_Samplerate", "SP_Measrate", samplerate);

/* Set both sensors to synchronization slave */
SetParameterInt (sensor1, "SP_SyncMode", 1); /* Slave */
SetParameterInt (sensor1, "SP_SyncTermination", 1); /* On */
err= ExecSCmd (sensor1, "Set_SyncMode");
SetParameterInt (sensor2, "SP_SyncMode", 1); /* Slave */
SetParameterInt (sensor2, "SP_SyncTermination", 1); /* On */
err= ExecSCmd (sensor2, "Set_SyncMode");

// Set up the IF2008 card timer 1 frequency, convert from kHz to Hz
SetParameterInt (ifCard, "SP_TimerNumber", 1);
SetParameterDouble (ifCard, "SP_TimerFrequency", samplerate*1000.0);
SetParameterDouble (ifCard, "SP_TimerRatio", 0.5);
err= ExecSCmd (ifCard, "Set_TimerFrequency");

// and put it out at sync outputs
SetParameterInt (ifCard, "SP_TrgChannel1", 1); /* Timer 1 */
SetParameterInt (ifCard, "SP_TrgChannel2", 1); /* Timer 1 */
SetParameterInt (ifCard, "SP_TrgChannel3", 1); /* Timer 1 */
SetParameterInt (ifCard, "SP_TrgChannel4", 1); /* Timer 1 */
SetParameterInt (ifCard, "SP_TrgChannel5", 1); /* Timer 1 */
SetParameterInt (ifCard, "SP_TrgChannel6", 1); /* Timer 1 */
err= ExecSCmd (ifCard, "Set_TrgSource");

```

At the moment where you want to start data acquisition, clear all buffers so new data from sensors has same time base.

```

// Clear all buffers of all sensors
err= SetIntExecSCmd (sensor1, "Clear_Buffers", "SP_AllDevices", 1);

```

Then you can read synchronized data in a loop.

```

int avail[2], maxValues, read[2];
int raw[2][10000];
double scaled[2][10000];
while (true)
{
    err= DataAvail (sensor1, &avail[0]);
    err= DataAvail (sensor2, &avail[1]);
    maxValues= min (avail[0], avail[1]);
    if (maxValues>_countof (scaled)) /* if this happens each */
        { /* time, reduce sleep time or increase buffers, */
        maxValues= _countof (scaled); /* otherwise internal */
    } /* MEDAQLib buffer will overflow */

    err= TransferData (sensor1, &raw[0], &scaled[0], maxValues, &read[0]);
    err= TransferData (sensor2, &raw[1], &scaled[1], maxValues, &read[1]);
    assert (read[0]==read[1]); /* must be equal */

    /* Process data in raw and scaled ... */
    Sleep (100); /* Sleep until some new data has arrived */
}

```

4.5 Block based data acquisition vs Poll

MEDAQLib continuously collects all data from sensor in an own background thread or timer and stores this data into an internal ring buffer. No function call from user to MEDAQLib does interrupt reading the data, so no data is lost. For reading the data from MEDAQLib to customer application, MEDAQLib supports two operation modes:

4.5.1 Block based data acquisition

The measurement values are collected and retrieved by customer application in larger data blocks by calling [TransferData](#). The advantage of this method is that the transfer cycles do not have to be so fast and often. Because of this normally there is no loss of data.

4.5.2 Polling

Using function [Poll](#) only the very last measurement value is returned to the caller. So the response time is very short, but you may miss some data on not polling often enough.

4.5.3 Additional hints

Both modes may be used simultaneously because polling for data does not change any internal buffer of MEDAQLib.

The function [DataAvail](#) can be used to retrieve the number of available values in buffer.

To wait for enough data, the user can call [DataAvail](#) in a loop (combined with a Sleep function) or he can use the [DataAvail_Event](#).

5 Samples

For real examples please take a look at the given examples in subdirectory samples. They show you different possibilities on using MEDAQLib:

5.1 C-Sharp Example

This example on disk shows how to access an [SENSOR_ILD2300](#) via [TCP/IP](#) from C#. Before using it please change the IP address to suit your settings.

5.2 Delphi Example

This example on disk shows how to access an [SENSOR_ILD1402](#) via [RS232](#) from Delphi. Before using it please change the COM port number to suit your settings.

5.3 DLL Example

This example on disk shows how to access an [SENSOR_ILD2200](#) via [IF2004](#) hardware card from Visual C++. In addition it shows how to access the functions from MEDAQLib without using MEDAQLib.lib, but using the dynamic approach using LoadLibrary and GetProcAddress.

5.4 LabView

This example on disk shows how to access an [SENSOR_ILD1700](#) via [RS232](#) from LabView 8.6 (and exported to 8.5 and 8.2). Before using it please change the COM port number to suit your settings. Several VI's with base functionality are included which can be used for own applications.

5.5 Lib Example

This example on disk shows how to access an [SENSOR_ILD1700](#) via [RS232](#) from Visual C++. Before using it please change the COM port number to suit your settings. In contrast to the DLLExample it links against the MEDAQLib.lib library directly.

5.6 IF2008 Example

This example on disk shows how to access [SENSOR_ILD2200](#) sensors via [IF2008](#) card and an encoder from Visual C++. Before using it please change the internal settings to suit your settings. It shows how to poll data and retrieve data synchronised.

5.7 Unicode Example

This example on disk shows how to access an [SENSOR_IFD2401](#) via USB ([WinUSB](#)) from Visual C++. It uses the MEDAQLib dynamically and can be compiled for ANSI and UNICODE. To wait for data it uses event handling.

5.8 VBA Example

This example on disk shows how to access an [SENSOR_ODC2600](#) via [RS232](#) from Excel speaking Visual Basic for Applications. Before using it please change the COM port number to suit your settings. To see the code please right click the "Table 1" tab at the bottom of Excel. In the context menu select "Show code". The second example (VBAExample2.xls) does the same but with sensor [SENSOR_ILD1700](#).

5.9 VB2013Example

This example on disk shows how to access an [SENSOR_ILD1420](#) via [RS232](#) using Visual Basic 2013.

5.10 X64 Example

This example on disk shows how to access an [SENSOR_ILD2300](#) via [TCP/IP](#) from Visual C++ (64 Bit). Before using it please change the IP address to suit your settings. Additionally the usage of [Get_TransmittedDataInfo](#) is shown.

5.11 SensorFinder

This example on disk is a C command line utility which search for sensor [SENSOR_ILD2300](#) (using [SensorFinder commands](#)) and (if found over [TCP/IP](#)) change it's IP address (using [Set_IPConfig](#)).

5.12 SensorTest

This example on disk is a C command line utility which search/open any sensor, show information about transmitted data and continuously retrieves data.

6 Function Reference

6.1 Create a sensor instance

Name: CreateSensorInstance

CreateSensorInstance

Description:

Creates an instance for the sensor which will be used in further function calls.

Attention! Please don't forget to call [ReleaseSensorInstance](#) at the end. Otherwise there will be memory and handle leaks.

Declaration:

```
DWORD CreateSensorInstance (ME_SENSOR sensor);
```

Parameter: ME_SENSOR sensor

sensor

Direction: [IN]

Valid values:

- SENSOR_ILR110x_115x (19) - optoNCDT ILR
- SENSOR_ILR118x (20) - optoNCDT ILR
- SENSOR_ILR1191 (21) - optoNCDT ILR
- SENSOR_ILD1302 (24) - optoNCDT
- SENSOR_ILD1320 (41) - optoNCDT
- SENSOR_ILD1401 (1) - optoNCDT
- SENSOR_ILD1402 (23) - optoNCDT
- SENSOR_ILD1420 (42) - optoNCDT
- SENSOR_ILD1700 (2) - optoNCDT
- SENSOR_ILD2200 (5) - optoNCDT
- SENSOR_ILD2300 (29) - optoNCDT
- SENSOR_IFD2401 (12) - confocalDT
- SENSOR_IFD2431 (13) - confocalDT
- SENSOR_IFD2445 (39) - confocalDT
- SENSOR_IFD2451 (30) - confocalDT
- SENSOR_IFD2461 (44) - confocalDT
- SENSOR_IFD2471 (26) - confocalDT
- SENSOR_ODC1202 (25) - optoCONTROL
- SENSOR_ODC2500 (8) - optoCONTROL
- SENSOR_ODC2520 (37) - optoCONTROL
- SENSOR_ODC2600 (9) - optoCONTROL
- SENSOR_DT3100 (28) - eddyNCDT
- SENSOR_DT6100 (16) - capaNCDT
- SENSOR_DT6120 (40) - capaNCDT
- CONTROLLER_DT6200 (33) - capaNCDT
- CONTROLLER_KSS6380 (18) - capaNCDT
- CONTROLLER_DT6500 (15) - capaNCDT
- SENSOR_ACST000 (35) - colorCONTROL
- SENSOR_LLT27xx - scanCONTROL and gapCONTROL (31), only for SensorFinder commands, [OpenSensor](#) will fail
- ETH_IF1032 (34) - Interface module Ethernet/EtherCAT
- USB_ADAPTER_IF2004 (36) - USB adapter IF2004
- PCI_CARD_IF2004 (10) - PCI card IF2004
- PCI_CARD_IF2008 (22) - PCI card IF2008

[CONTROLLER_CSP2008](#) (32) - Universal controller
[CONTROLLER_CBOX](#) (38) - External controller C-Box

Description: Type of sensor used.

Returns: Number of the created sensor instance or zero, if parameter sensor is not valid.

Name: CreateSensorInstByName

CreateSensorInstByName

Description:

Creates an instance for the sensor which will be used in further function calls.

Attention! Please don't forget to call [ReleaseSensorInstance](#) at the end. Otherwise there will be memory and handle leaks.

Declaration:

```
DWORD CreateSensorInstByName (LPCSTR sensorName);
```

Parameter: LPCSTR sensorName

sensorName

Direction: [IN]

Valid values:

- 'ILR110', 'ILR115' matches [SENSOR_ILR110x_115x](#)
- 'ILR118' matches [SENSOR_ILR118x](#)
- 'ILR1191' matches [SENSOR_ILR1191](#)
- 'ILD1302', 'optoNCDT 1302' matches [SENSOR_ILD1302](#)
- 'ILD1320', 'optoNCDT 1320' matches [SENSOR_ILD1320](#)
- 'ILD1401', 'optoNCDT 1401' matches [SENSOR_ILD1401](#)
- 'ILD1402', 'optoNCDT 1402' matches [SENSOR_ILD1402](#)
- 'ILD1420', 'optoNCDT 1420' matches [SENSOR_ILD1420](#)
- 'ILD17', 'optoNCDT 17' matches [SENSOR_ILD1700](#)
- 'ILD22', 'optoNCDT 22' matches [SENSOR_ILD2200](#)
- 'ILD23', 'optoNCDT 23' matches [SENSOR_ILD2300](#)
- 'IFC2401', 'IFD2401', 'confocalDT 2401' matches [SENSOR_IFD2401](#)
- 'IFC2431', 'IFD2431', 'confocalDT 2431' matches [SENSOR_IFD2431](#)
- 'IFC2445', 'IFD2445', 'confocalDT 2445' matches [SENSOR_IFD2445](#)
- 'IFC2451', 'IFD2451', 'confocalDT 2451' matches [SENSOR_IFD2451](#)
- 'IFC2461', 'IFD2461', 'confocalDT 2461' matches [SENSOR_IFD2461](#)
- 'IFC2471', 'IFD2471', 'confocalDT 2471' matches [SENSOR_IFD2471](#)
- 'ODC1202', 'optoCONTROL 1202', 'ODC1220', 'optoCONTROL 1220' matches [SENSOR_ODC1202](#)
- 'ODC2500', 'optoCONTROL 2500' matches [SENSOR_ODC2500](#)
- 'ODC2520', 'optoCONTROL 2520' matches [SENSOR_ODC2520](#)
- 'ODC2600', 'optoCONTROL 2600' matches [SENSOR_ODC2600](#)
- 'DT3100', 'eddyNCDT 3100' matches [SENSOR_DT3100](#)
- 'DT6100', 'capaNCDT 6100' matches [SENSOR_DT6100](#)
- 'DT6120', 'capaNCDT 6120' matches [SENSOR_DT6120](#)
- 'DT62', 'capaNCDT 62' matches [CONTROLLER_DT6200](#)
- 'KSS6380', 'capaNCDT 6380' matches [CONTROLLER_KSS6380](#)
- 'DT65', 'capaNCDT 65' matches [CONTROLLER_DT6500](#)
- 'ACS7000', 'colorCONTROL 7000' matches [SENSOR_ACS7000](#)
- 'LLT27', 'scanCONTROL 27', 'gapCONTROL 27' matches [SENSOR_LLT27xx](#)

'IF1032' matches [ETH_IF1032](#)
 'IF2004 USB' matches [USB_ADAPTER_IF2004](#)
 'IF2004' matches [PCI_CARD_IF2004](#)
 'IF2008' matches [PCI_CARD_IF2008](#)
 'CSP2008' matches [CONTROLLER_CSP2008](#)
 'C-Box' matches [CONTROLLER_CBOX](#)

Description: Any of the values above are tokenized (delimited at spaces) and searched (case insensitive) in parameter sensorName.
 This will guarantee a high match rate. Only if all of the tokens are contained, the sensorName is valid.

Returns: Number of the created sensor instance or zero, if parameter sensorName cannot be interpreted.

Name: CreateSensorInstByNameU

CreateSensorInstByNameU

Description:

Creates an instance for the sensor which will be used in further function calls (Unicode version).

Attention! Please don't forget to call [ReleaseSensorInstance](#) at the end. Otherwise there will be memory and handle leaks.

Declaration:

```
DWORD CreateSensorInstByNameU (LPCWSTR sensorName);
```

Parameter: LPCWSTR sensorName

sensorName

Direction: [IN]

Valid values:

'ILR110', 'ILR115' matches [SENSOR_ILR110x_115x](#)
 'ILR118' matches [SENSOR_ILR118x](#)
 'ILR1191' matches [SENSOR_ILR1191](#)
 'ILD1302', 'optoNCDT 1302' matches [SENSOR_ILD1302](#)
 'ILD1320', 'optoNCDT 1320' matches [SENSOR_ILD1320](#)
 'ILD1401', 'optoNCDT 1401' matches [SENSOR_ILD1401](#)
 'ILD1402', 'optoNCDT 1402' matches [SENSOR_ILD1402](#)
 'ILD1420', 'optoNCDT 1420' matches [SENSOR_ILD1420](#)
 'ILD17', 'optoNCDT 17' matches [SENSOR_ILD1700](#)
 'ILD22', 'optoNCDT 22' matches [SENSOR_ILD2200](#)
 'ILD23', 'optoNCDT 23' matches [SENSOR_ILD2300](#)
 'IFC2401', 'IFD2401', 'confocalDT 2401' matches [SENSOR_IFD2401](#)
 'IFC2431', 'IFD2431', 'confocalDT 2431' matches [SENSOR_IFD2431](#)
 'IFC2445', 'IFD2445', 'confocalDT 2445' matches [SENSOR_IFD2445](#)
 'IFC2451', 'IFD2451', 'confocalDT 2451' matches [SENSOR_IFD2451](#)
 'IFC2461', 'IFD2461', 'confocalDT 2461' matches [SENSOR_IFD2461](#)
 'IFC2471', 'IFD2471', 'confocalDT 2471' matches [SENSOR_IFD2471](#)
 'ODC1202', 'optoCONTROL 1202', 'ODC1220', 'optoCONTROL 1220' matches [SENSOR_ODC1202](#)
 'ODC2500', 'optoCONTROL 2500' matches [SENSOR_ODC2500](#)
 'ODC2520', 'optoCONTROL 2520' matches [SENSOR_ODC2520](#)
 'ODC2600', 'optoCONTROL 2600' matches [SENSOR_ODC2600](#)
 'DT3100', 'eddyNCDT 3100' matches [SENSOR_DT3100](#)
 'DT6100', 'capaNCDT 6100' matches [SENSOR_DT6100](#)

'DT6120', 'capaNCDT 6120' matches **SENSOR_DT6120**
 'DT62', 'capaNCDT 62' matches **CONTROLLER_DT6200**
 'KSS6380', 'capaNCDT 6380' matches **CONTROLLER_KSS6380**
 'DT65', 'capaNCDT 65' matches **CONTROLLER_DT6500**
 'ACS7000', 'colorCONTROL 7000' matches **SENSOR_ACS7000**
 'LLT27', 'scanCONTROL 27', 'gapCONTROL 27' matches **SENSOR_LLTT27xx**
 'IF1032' matches **ETH_IF1032**
 'IF2004 USB' matches **USB_ADAPTER_IF2004**
 'IF2004' matches **PCI_CARD_IF2004**
 'IF2008' matches **PCI_CARD_IF2008**
 'CSP2008' matches **CONTROLLER_CSP2008**
 'C-Box' matches **CONTROLLER_CBOX**

Description: Any of the values above are tokenized (delimited at spaces) and searched (case insensitive) in parameter `sensorName`.

This will guarantee a high match rate. Only if all of the tokens are contained, the `sensorName` is valid.

Returns: Number of the created sensor instance or zero, if parameter `sensorName` cannot be interpreted.

6.2 Releasing sensor instance

Name: `ReleaseSensorInstance`

ReleaseSensorInstance

Description:

Free the specified sensor instance.

Declaration:

```
ERR_CODE ReleaseSensorInstance (DWORD instanceHandle);
```

Parameter: `DWORD instanceHandle`

`instanceHandle`

Direction: [IN]

Description: Number of the sensor instance, previously returned by `CreateSensorInstance`.

Returns:

`ERR_NOERROR` (0) on success.

`ERR_INSTANCE_NOT_EXIST` (-24) if `instanceHandle` is not valid.

6.3 Set parameters

Before connecting to the sensor ([OpenSensor](#)) or sending commands to the sensor ([SensorCommand](#)) parameters for both functions can be specified. Each parameter can be set by one of following functions.

Name: `SetParameterInt`

SetParameterInt

Description:

Set a 4 Byte integer parameter.

Declaration:

```
ERR_CODE SetParameterInt (DWORD instanceHandle, LPCSTR paramName, int
paramValue);
```

Parameter: DWORD instanceHandle instanceHandle
Direction: [IN]
Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR paramName paramName
Direction: [IN]
Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.
Description: Name of the parameter as string.

Parameter: int paramValue paramValue
Direction: [IN]
Description: Value of the parameter.

Returns:

[ERR_NOERROR](#) (0) on success.
[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

Name: SetParameterIntU **SetParameterIntU**

Description:
 Set a 4 Byte integer parameter (Unicode version).

Declaration:

```
ERR_CODE SetParameterIntU (DWORD instanceHandle, LPCWSTR paramName,
int paramValue);
```

Parameter: DWORD instanceHandle instanceHandle
Direction: [IN]
Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR paramName paramName
Direction: [IN]
Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.
Description: Name of the parameter as unicode string.

Parameter: int paramValue paramValue
Direction: [IN]
Description: Value of the parameter.

Returns:

[ERR_NOERROR](#) (0) on success.
[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

Name: SetParameterDWORD_PTR

**SetParameterDWORD_-
PTR**
Description:

Set a 4 Byte (Win32) or 8 Byte (Win64) unsigned integer parameter.

Declaration:

```
ERR_CODE SetParameterDWORD_PTR (DWORD instanceHandle, LPCSTR paramName,
                               DWORD_PTR paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.

Description: Name of the parameter as string.

Parameter: DWORD_PTR paramValue

paramValue

Direction: [IN]

Description: Value of the parameter.

Returns:

[ERR_NOERROR](#) (0) on success.

[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

Name: SetParameterDWORD_PTRU

**SetParameterDWORD_-
PTRU**
Description:

Set a 4 Byte (Win32) or 8 Byte (Win64) unsigned integer parameter (Unicode version).

Declaration:

```
ERR_CODE SetParameterDWORD_PTRU (DWORD instanceHandle, LPCWSTR paramName,
                                 DWORD_PTR paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.

Description: Name of the parameter as unicode string.

Parameter: DWORD_PTR paramValue

paramValue

Direction: [IN]

Description: Value of the parameter.

Returns:

[ERR_NOERROR](#) (0) on success.

[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

Name: SetParameterDouble

SetParameterDouble
Description:

Set a 8 Byte double parameter.

Declaration:

```
ERR_CODE SetParameterDouble (DWORD instanceHandle, LPCSTR paramName,
                           double paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.

Description: Name of the parameter as string.

Parameter: double paramValue

paramValue

Direction: [IN]

Description: Value of the parameter.

Returns:

[ERR_NOERROR](#) (0) on success.

[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

Name: SetParameterDoubleU

SetParameterDoubleU
Description:

Set a 8 Byte double parameter (Unicode version).

Declaration:

```
ERR_CODE SetParameterDoubleU (DWORD instanceHandle, LPCWSTR paramName,
                            double paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.

Description: Name of the parameter as unicode string.

Parameter: double paramValue

paramValue

Direction: [IN]

Description: Value of the parameter.

Returns:

[ERR_NOERROR](#) (0) on success.

[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

Name: SetParameterString

SetParameterString
Description:

Set a string (pointer to a character array) parameter.

Declaration:

```
ERR_CODE SetParameterString (DWORD instanceHandle, LPCSTR paramName,
                           LPCSTR paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.

Description: Name of the parameter as string.

Parameter: LPCSTR paramValue

paramValue

Direction: [IN]

Description: Value of the parameter.

Returns:

[ERR_NOERROR](#) (0) on success.

[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

Name: SetParameterStringU

SetParameterStringU
Description:

Set a string (pointer to a character array) parameter (Unicode version).

Declaration:

```
ERR_CODE SetParameterStringU (DWORD instanceHandle, LPCWSTR paramName,
                            LPCWSTR paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.

Description: Name of the parameter as unicode string.

Parameter: LPCWSTR paramValue

paramValue

Direction: [IN]

Description: Value (as unicode) of the parameter.

Returns:

[ERR_NOERROR](#) (0) on success.

[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

Name: SetParameterBinary

SetParameterBinary
Description:

Set binary data (pointer to a character array) to MEDAQLib.

Declaration:

```
ERR_CODE SetParameterBinary (DWORD instanceHandle, LPCSTR paramName,
                           char *paramValue, DWORD len);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.

Description: Name of the parameter as string. Useful for the parameter SP_CmdStr (in [Cmd_Generic](#))

Parameter: char * paramValue

paramValue

Direction: [IN]

Description: Value of the parameter.

Parameter: DWORD len

len

Direction: [IN]

Description: Length of the binary data in bytes.

Returns:

[ERR_NOERROR](#) (0) on success.

[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

Name: SetParameterBinaryU

SetParameterBinaryU
Description:

Set binary data (pointer to a character array) to MEDAQLib (Unicode version).

Declaration:

```
ERR_CODE SetParameterBinary (DWORD instanceHandle, LPCWSTR paramName,
                           char *paramValue, DWORD len);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.

Description: Name of the parameter as unicode string. Useful for the parameter SP_CmdStr (in [Cmd_Generic](#))

Parameter: char * paramValue paramValue

Direction: [IN]

Description: Value of the parameter.

Parameter: DWORD len len

Direction: [IN]

Description: Length of the binary data in bytes.

Returns:

[ERR_NOERROR](#) (0) on success.

[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

Name: SetParameters [SetParameters](#)

Description:

Set a list of parameters at once.

Declaration:

```
ERR_CODE SetParameters (DWORD instanceHandle, LPCSTR parameterList);
```

Parameter: DWORD instanceHandle instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR parameterList parameterList

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.

The parameters must be separated by white spaces (space, tabulator, carriage return or line feed).

Parameter name and value must be separated by equality sign (=).

Attention! No space is allowed here.

Attention! No space is allowed here.

If parameter value only contains a sign and digits, it is interpreted as integer. If there is a decimal point and/or exponent additionally it is interpreted as double. Otherwise it is interpreted as string parameter.

If the string value should contain spaces, the whole string must be quoted by "..." or '...'.

Description: List of the parameters as string.

Returns:

[ERR_NOERROR](#) (0) on success.

[ERR_WRONG_PARAMETER](#) (-18) if the parameter list has syntax errors.

[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

Name: SetParametersU

SetParametersU
Description:

Set a list of parameters at once (Unicode version).

Declaration:

```
ERR_CODE SetParametersU (DWORD instanceHandle, LPCWSTR parameterList);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR parameterList

parameterList

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.

The parameters must be separated by white spaces (space, tabulator, carriage return or line feed).

Parameter name and value must be separated by equality sign (=).

Attention! No space is allowed here.

If parameter value only contains a sign and digits, it is interpreted as integer. If there is a decimal point and/or exponent additionally it is interpreted as double. Otherwise it is interpreted as string parameter.

If the string value should contain spaces, the whole string must be quoted by "..." or '...'.

Description: List of the parameters as unicode string.

Returns:

[ERR_NOERROR](#) (0) on success.

[ERR_WRONG_PARAMETER](#) (-18) if the parameter list has syntax errors.

[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

6.4 Get parameters

After sending a command to a sensor ([SensorCommand](#)) the answer can be retrieved at same manner as setting parameters.

Name: GetParameterInt

GetParameterInt
Description:

Get a 4 Byte integer parameter.

Declaration:

```
ERR_CODE GetParameterInt (DWORD instanceHandle, LPCSTR paramName, int
 *paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR paramName
Direction: [IN]
Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.
Description: Name of the parameter as string.

paramName

Parameter: int * paramValue
Direction: [OUT]
Description: Pointer to a variable retrieving the parameter

paramValue

Returns:

[ERR_NOERROR](#) (0) on success.
[ERR_WRONG_PARAMETER](#) (-18) if the pointer to paramValue is NULL or the datatype does not match.
[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.
[ERR_NOT_FOUND](#) (-25) if the parameter (specified by paramName) is not found.

Name: GetParameterIntU

GetParameterIntU

Description:
 Get a 4 Byte integer parameter (Unicode version).

Declaration:

```
ERR_CODE GetParameterIntU (DWORD instanceHandle, LPCWSTR paramName,
                           int *paramValue);
```

Parameter: DWORD instanceHandle
Direction: [IN]
Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

instanceHandle

Parameter: LPCWSTR paramName
Direction: [IN]
Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.
Description: Name of the parameter as unicode string.

paramName

Parameter: int * paramValue
Direction: [OUT]
Description: Pointer to a variable retrieving the parameter

paramValue

Returns:

[ERR_NOERROR](#) (0) on success.
[ERR_WRONG_PARAMETER](#) (-18) if the pointer to paramValue is NULL or the datatype does not match.
[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.
[ERR_NOT_FOUND](#) (-25) if the parameter (specified by paramName) is not found.

Name: GetParameterDWORD_PTR

**GetParameterDWORD_-
PTR**
Description:

Get a 4 Byte (Win32) or 8 Byte (Win64) unsigned integer parameter.

Declaration:

```
ERR_CODE GetParameterDWORD_PTR (DWORD instanceHandle, LPCSTR paramName,
                               DWORD_PTR *paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by
[CreateSensorInstance](#).

Parameter: LPCSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#)
 and [Sensor commands](#) and following.

Description: Name of the parameter as string.

Parameter: DWORD_PTR * paramValue

paramValue

Direction: [OUT]

Description: Pointer to a variable retrieving the parameter

Returns:

[ERR_NOERROR](#) (0) on success.

[ERR_WRONG_PARAMETER](#) (-18) if the pointer to paramValue is NULL or the
 datatype does not match.

[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

[ERR_NOT_FOUND](#) (-25) if the parameter (specified by paramName) is not
 found.

Name: GetParameterDWORD_PTRU

**GetParameterDWORD_-
PTRU**
Description:

Get a 4 Byte (Win32) or 8 Byte (Win64) unsigned integer parameter (Unicode
 version).

Declaration:

```
ERR_CODE GetParameterDWORD_PTRU (DWORD instanceHandle, LPCWSTR paramName,
                                 DWORD_PTR *paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by
[CreateSensorInstance](#).

Parameter: LPCWSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#)
 and [Sensor commands](#) and following.

Description: Name of the parameter as unicode string.

Parameter: DWORD_PTR * paramValue

paramValue

Direction: [OUT]

Description: Pointer to a variable retrieving the parameter

Returns:

ERR_NOERROR (0) on success.

ERR_WRONG_PARAMETER (-18) if the pointer to paramValue is NULL or the datatype does not match.

ERR_INSTANCE_NOT_EXIST (-24) if instanceHandle is not valid.

ERR_NOT_FOUND (-25) if the parameter (specified by paramName) is not found.

Name: GetParameterDouble

GetParameterDouble

Description:

Get a 8 Byte double parameter.

Declaration:

```
ERR_CODE GetParameterDouble (DWORD instanceHandle, LPCSTR paramName,
                           double *paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by CreateSensorInstance.

Parameter: LPCSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.

Description: Name of the parameter as string.

Parameter: double * paramValue

paramValue

Direction: [OUT]

Description: Pointer to a variable retrieving the parameter

Returns:

ERR_NOERROR (0) on success.

ERR_WRONG_PARAMETER (-18) if the pointer to paramValue is NULL or the datatype does not match.

ERR_INSTANCE_NOT_EXIST (-24) if instanceHandle is not valid.

ERR_NOT_FOUND (-25) if the parameter (specified by paramName) is not found.

Name: GetParameterDoubleU

GetParameterDoubleU
Description:

Get a 8 Byte double parameter (Unicode version).

Declaration:

```
ERR_CODE GetParameterDoubleU (DWORD instanceHandle, LPCWSTR paramName,
    double *paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.

Description: Name of the parameter as unicode string.

Parameter: double * paramValue

paramValue

Direction: [OUT]

Description: Pointer to a variable retrieving the parameter

Returns:

[ERR_NOERROR](#) (0) on success.

[ERR_WRONG_PARAMETER](#) (-18) if the pointer to paramValue is NULL or the datatype does not match.

[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

[ERR_NOT_FOUND](#) (-25) if the parameter (specified by paramName) is not found.

Name: GetParameterString

GetParameterString
Description:

Get a string (character array) parameter.

Declaration:

```
ERR_CODE GetParameterString (DWORD instanceHandle, LPCSTR paramName,
    LPSTR paramValue, DWORD *maxLen);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.

Description: Name of the parameter as string.

Parameter: LPSTR paramValue paramValue

Direction: [OUT]

Description: Pointer to a variable (character buffer) retrieving the parameter.

Parameter: DWORD * maxLen maxLen

Direction: [IN/OUT]

Description: The buffer must be allocated by the application. The size of the buffer is specified at maxLen. If the resulting string is larger than maxLen, it is truncated and **ERR_NOMEMORY** (-19) is returned. The real length of the string (maybe truncated) is returned in maxLen too. If paramValue is NULL, the length of the containing string is returned in maxLen.

Returns:

ERR_NOERROR (0) on success.

ERR_WRONG_PARAMETER (-18) if the pointer to maxLen is NULL or the datatype does not match.

ERR_NOMEMORY (-19) if the buffer is too short to hold the complete answer.

ERR_INSTANCE_NOT_EXIST (-24) if instanceHandle is not valid.

ERR_NOT_FOUND (-25) if the parameter (specified by paramName) is not found.

Name: GetParameterStringU

GetParameterStringU

Description:

Get a string (character array) parameter (Unicode version).

Declaration:

```
ERR_CODE GetParameterStringU (DWORD instanceHandle, LPCWSTR paramName,
    LPWSTR paramValue, DWORD *maxLen);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.

Description: Name of the parameter as unicode string.

Parameter: LPWSTR paramValue

paramValue

Direction: [OUT]

Description: Pointer to a variable (unicode buffer) retrieving the parameter.

Parameter: DWORD * maxLen

maxLen

Direction: [IN/OUT]

Description: The buffer must be allocated by the application. The size of the buffer (in characters, not bytes) is specified at maxLen. If the resulting string is larger than maxLen, it is truncated and **ERR_NOMEMORY** (-19) is returned. The real length of the string (maybe truncated) is returned in maxLen too.

If paramValue is NULL, the length of the containing string is returned in maxLen.

Returns:

ERR_NOERROR (0) on success.
ERR_WRONG_PARAMETER (-18) if the pointer to maxLen is NULL or the datatype does not match.
ERR_NOMEMORY (-19) if the buffer is too short to hold the complete answer.
ERR_INSTANCE_NOT_EXIST (-24) if instanceHandle is not valid.
ERR_NOT_FOUND (-25) if the parameter (specified by paramName) is not found.

Name: GetParameterBinary

GetParameterBinary
Description:

Get binary data from MEDAQLib.

Declaration:

```
ERR_CODE GetParameterBinary (DWORD instanceHandle, LPCSTR paramName,
                           char *paramValue, DWORD *maxLen);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.

Description: Name of the parameter as string.

Parameter: char * paramValue

paramValue

Direction: [OUT]

Description: Pointer to a variable (character buffer) retrieving the parameter.

Parameter: DWORD * maxLen

maxLen

Direction: [IN/OUT]

Description: The buffer must be allocated by the application. The size of the buffer (in bytes) is specified at maxLen. If the resulting binary data is larger than maxLen, it is truncated and **ERR_NOMEMORY** (-19) is returned. The real length of the binary data (maybe truncated) is returned in maxLen too.

If paramValue is NULL, the length of the containing binary data is returned in maxLen.

Returns:

ERR_NOERROR (0) on success.
ERR_WRONG_PARAMETER (-18) if the pointer to maxLen is NULL or the datatype does not match.
ERR_NOMEMORY (-19) if the buffer is too short to hold the complete answer.
ERR_INSTANCE_NOT_EXIST (-24) if instanceHandle is not valid.
ERR_NOT_FOUND (-25) if the parameter (specified by paramName) is not found.

Name: GetParameterBinaryU

GetParameterBinaryU
Description:

Get binary data from MEDAQLib (Unicode version).

Declaration:

```
ERR_CODE GetParameterBinaryU (DWORD instanceHandle, LPCWSTR paramName,
    char *paramValue, DWORD *maxLen);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Interface parameters](#) and [Sensor commands](#) and following.

Description: Name of the parameter as unicode string.

Parameter: char * paramValue

paramValue

Direction: [OUT]

Description: Pointer to a variable (character buffer) retrieving the parameter.

Parameter: DWORD * maxLen

maxLen

Direction: [IN/OUT]

Description: The buffer must be allocated by the application. The size of the buffer (in bytes) is specified at maxLen. If the resulting binary data is larger than maxLen, it is truncated and [ERR_NOMEMORY](#) (-19) is returned. The real length of the binary data (maybe truncated) is returned in maxLen too.

If paramValue is NULL, the length of the containing binary data is returned in maxLen.

Returns:

[ERR_NOERROR](#) (0) on success.

[ERR_WRONG_PARAMETER](#) (-18) if the pointer to maxLen is NULL or the datatype does not match.

[ERR_NOMEMORY](#) (-19) if the buffer is too short to hold the complete answer.

[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

[ERR_NOT_FOUND](#) (-25) if the parameter (specified by paramName) is not found.

Name: GetParameters

GetParameters
Description:

Get all available parameters at once.

Declaration:

```
ERR_CODE GetParameters (DWORD instanceHandle, LPSTR parameterList,
    DWORD *maxLen);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPSTR parameterList parameterList

Direction: [OUT]

Valid value: The parameters are separated by a space. Parameter name and value is separated by equality sign (=). Integer parameters only contains a sign and digits. Double parameters contains a decimal point and/or exponent additionally. String parameters are always quoted by "...".

Description: List of the parameters as string.

Parameter: DWORD * maxLen maxLen

Direction: [IN/OUT]

Description: The buffer must be allocated by the application. The size of the buffer is specified at maxLen. If the resulting string is larger than maxLen, it is truncated and [ERR_NOMEMORY](#) (-19) is returned. The real length of the string (maybe truncated) is returned in maxLen too. If parameterList is NULL, the length of the containing string is returned in maxLen.

Returns:

[ERR_NOERROR](#) (0) on success.

[ERR_WRONG_PARAMETER](#) (-18) if the pointer to maxLen is NULL.

[ERR_NOMEMORY](#) (-19) if the buffer is too short to hold the complete answer.

[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

Name: GetParametersU GetParametersU

Description:

Get all available parameters at once (Unicode version).

Declaration:

```
ERR_CODE GetParametersU (DWORD instanceHandle, LPWSTR parameterList,
                        DWORD *maxLen);
```

Parameter: DWORD instanceHandle instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPWSTR parameterList parameterList

Direction: [OUT]

Valid value: The parameters are separated by a space. Parameter name and value is separated by equality sign (=). Integer parameters only contains a sign and digits. Double parameters contains a decimal point and/or exponent additionally. String parameters are always quoted by "...".

Description: List of the parameters as unicode string.

Parameter: DWORD * maxLen maxLen

Direction: [IN/OUT]

Description: The buffer must be allocated by the application. The size of the buffer (in characters, not bytes) is specified at maxLen. If the resulting string is larger than maxLen, it is truncated and [ERR_NOMEMORY](#) (-19) is returned. The real length of the string (maybe truncated) is returned in maxLen too.

If parameterList is NULL, the length of the containing string is returned in maxLen.

Returns:

`ERR_NOERROR` (0) on success.
`ERR_WRONG_PARAMETER` (-18) if the pointer to `maxLen` is NULL.
`ERR_NOMEMORY` (-19) if the buffer is too short to hold the complete answer.
`ERR_INSTANCE_NOT_EXIST` (-24) if `instanceHandle` is not valid.

6.5 Clear internal parameter buffer

Name: `ClearAllParameters`
ClearAllParameters
Description:

Before building up a new command using the [Set parameters](#) functions, the internal parameter buffer should be cleared. This avoids parameter mismatch between several commands.

Declaration:

```
ERR_CODE ClearAllParameters (DWORD instanceHandle);
```

Parameter: `DWORD instanceHandle`
`instanceHandle`
Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Returns:

`ERR_NOERROR` (0) on success.
`ERR_INSTANCE_NOT_EXIST` (-24) if `instanceHandle` is not valid.

6.6 Connecting to the sensor

Name: `OpenSensor`
OpenSensor
Description:

Establish the connection to the sensor.

After connecting to the sensor, all internal parameters starting with `SP_`, `CP_` and `IP_` are cleared (like at function [ClearAllParameters](#)).

Attention! Please don't forget call [CloseSensor](#) at the end. Otherwise it can come to unpredictable behaviour when resources leave open.

Declaration:

```
ERR_CODE OpenSensor (DWORD instanceHandle);
```

Parameter: `DWORD instanceHandle`
`instanceHandle`
Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Returns:

ERR_NOERROR (0) on success.
 ERR_CANNOT_OPEN (-2) if interface cannot be opened.
 ERR_APPLYING_PARAMS (-4) if parameters cannot applied to driver.
 ERR_INTERFACE_NOT_SUPPORTED (-10) if the specified interface (parameter "IP_Interface") is not supported.
 ERR_ALREADY_OPEN (-11) if the connection is already open.
 ERR_CANNOT_CREATE_INTERFACE (-12) if the interface cannot be created.
 ERR_WRONG_PARAMETER (-18) if a parameter is not valid (e.g. out of range).
 ERR_NOMEMORY (-19) if there is not enough memory to allocate MEDAQLib ring buffer.
 ERR_INSTANCE_NOT_EXIST (-24) if instanceHandle is not valid.
 ERR_WARNING (-26) if data interface could not be opened.
 If first bit of IP_AutomaticMode is set (1), any error returned by [SensorCommand](#).
 Any other asynchronous error happened at internal threads or callbacks or timers.

6.7 Closing connection to sensor

Name: CloseSensor

CloseSensor
Description:

Close the connection to the sensor.

Declaration:

```
ERR_CODE CloseSensor (DWORD instanceHandle);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Returns:

ERR_NOERROR (0) on success.
 ERR_HW_COMMUNICATION (-7) if TCP/IP connection could not be closed.
 ERR_INSTANCE_NOT_EXIST (-24) if instanceHandle is not valid.

6.8 Sending commands to the sensor

Name: SensorCommand

SensorCommand
Description:

Before sending a command to the sensor the command and parameters must be specified with [Set parameters](#) functions. After the command is executed the answer can be read using [Get parameters](#) functions.

At start of this function, all internal parameters starting with SA_, and IA_ are cleared (like at function [ClearAllParameters](#)).

At end of this function, all internal parameters starting with SP_, CP_, and IP_ are cleared (like at function [ClearAllParameters](#)).

Declaration:

```
ERR_CODE SensorCommand (DWORD instanceHandle);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Returns:

ERR_NOERROR (0) on success.
 ERR_FUNCTION_NOT_SUPPORTED (-1) if a function was called which is not supported by sensor or hardware interface..
 ERR_CANNOT_OPEN (-2) for [SensorFinder commands](#).
 ERR_NOT_OPEN (-3) if connection is not established
 ERR_APPLYING_PARAMS (-4) if interface parameters changed by command (e.g. change baudrate) and cannot be applied.
 ERR_CLEARUNG_BUFFER (-6) for command [Clear_Buffers](#) or if IP_ClearSendBuffer or IP_ClearReceiveBuffer is specified the driver could not be cleared or pending operations on USB could not be canceled.
 ERR_HW_COMMUNICATION (-7) if communication with sensor failed.
 ERR_TIMEOUT_READING_FROM_SENSOR (-8) if no answer is received within answer time.
 ERR_READING_SENSOR_DATA (-9) if reading sensor answer over USB failed.
 ERR_INTERFACE_NOT_SUPPORTED (-10) for internal command Open_DataSocket.
 ERR_CANNOT_CREATE_INTERFACE (-12) for [SensorFinder commands](#).
 ERR_NO_SENSORDATA_AVAILABLE (-13) for sensor command Get_Spectrum of SENSOR_IFD2401 and SENSOR_IFD2431.
 ERR_UNKNOWN_SENSOR_COMMAND (-14) if parameter [S_Command](#) is not valid.
 ERR_UNKNOWN_SENSOR_ANSWER (-15) if sensor answer is not an answer to a command.
 ERR_SENSOR_ANSWER_ERROR (-16) if sensor answer cannot be interpreted or sensor returned an error number.
 ERR_SENSOR_ANSWER_TOO_SHORT (-17) if sensor answer is too short.
 ERR_WRONG_PARAMETER (-18) if a parameter is not valid (e.g. out of range).
 ERR_NOMEMORY (-19) if there is not enough memory to allocate a buffer.
 ERR_INSTANCE_NOT_EXIST (-24) if instanceHandle is not valid.
 ERR_WARNING (-26) for [SensorFinder commands](#).
 ERR_SENSOR_ANSWER_WARNING (-27) if sensor returned a warning.
 Any other asynchronous error happened at interal threads or callbacks or timers.

6.9 Polling data from sensor

Name: Poll

Poll

Description:

Get the latest values from sensor.

Declaration:

```
ERR_CODE Poll (DWORD instanceHandle, int *rawData, double *scaledData,
               int maxValues);
```

Parameter: DWORD instanceHandle instanceHandle
Direction: [IN]
Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: int * rawData rawData
Direction: [OUT]
Description: Pointer to value (or array of values) to retrieve latest data frame from sensor as raw values.
 If the pointer is null, no data is transferred for this parameter.
 The meaning of the raw values is described at each sensor section.

Parameter: double * scaledData scaledData
Direction: [OUT]
Description: Pointer to value (or array of values) to retrieve latest data frame from sensor scaled by sensor range.
 If the pointer is null, no data is transferred for this parameter.
 The meaning of the scaled values is described at each sensor section.
 Please see parameter [IP_ScaleErrorValues](#) for scaling error values.

Parameter: int maxValues maxValues
Direction: [IN]
Description: Length of rawData and scaledData. Some sensors can measure more than one value per measure cycle (e.g. IFD's which measures distance, intensity, ... or ODC's which can measure multiple segments). In this case maxValues can be set up to frame size (values per measure cycle).
 If maxValues is smaller than frame size only the first maxValues values are transferred.
 If maxValues is greater than frame size, values of more than one frame are transferred.

Returns:

[ERR_NOERROR](#) (0) on success.
[ERR_NOT_OPEN](#) (-3) if connection is not established
[ERR_NO_SENSORDATA_AVAILABLE](#) (-13) if either the receive buffer does not contain any value from sensor. This error can occur if you poll for data immediately after connecting or after a sensor command. In this case, wait some time until data arrives at MEDAQLib. Or this error can occur, if MEDAQLib doesn't can interpret data from sensor, because some information is missing. In this case, call the function as described at error text of [GetError](#).
[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.
 If the internal ring buffer overflows (because Poll does not remove data), no warning is output.
 Any other asynchronous error happened at internal threads or callbacks or timers.

6.10 Number of values available to read

Name: DataAvail

DataAvail

Description:

The values available returned may be divided by frame size to calculate the number of frames available. A frame is the set of values transmitted by the sensor for each one measurement.

Declaration:

```
ERR_CODE DataAvail (DWORD instanceHandle, int *avail);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: int * avail

avail

Direction: [OUT]

Description: Pointer to value to retrieve number of values available from sensor

Returns:

[ERR_NOERROR](#) (0) on success.

[ERR_NOT_OPEN](#) (-3) if connection is not established.

[ERR_TIMEOUT_READING_FROM_SENSOR](#) (-8) if no data frame received for at least one second or three data cycles.

[ERR_NO_SENSORDATA_AVAILABLE](#) (-13) This error can occur, if MEDAQLib doesn't can interpret data from sensor, because some information is missing. In this case, call the function as described at error text of [GetError](#).

[ERR_WRONG_PARAMETER](#) (-18) if parameter avail is NULL.

[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

Any other asynchronous error happened at interal threads or callbacks or timers.

6.11 Block wise data acquisition from sensor

Name: TransferData

TransferData

Description:

Transfer the data from driver to application.

Declaration:

```
ERR_CODE TransferData (DWORD instanceHandle, int *rawData, double *scaledData,
                      int maxValues, int *read);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: int * rawData rawData
Direction: [OUT]
Description: Pointer to array of values to retrieve data from sensor as raw values.
 If the pointer is null, no data is transferred for this parameter. If rawData and scaledData are null all buffers used are emptied.
 The meaning of the raw values is described at each sensor section.

Parameter: double * scaledData scaledData
Direction: [OUT]
Description: Pointer to array of values to retrieve data frame from sensor scaled by sensor range.
 If the pointer is null, no data is transferred for this parameter. If rawData and scaledData are null all buffers used are emptied.
 The meaning of the scaled values is described at each sensor section.
 Please see parameter [IP_ScaleErrorValues](#) for scaling error values.

Parameter: int maxValues maxValues
Direction: [IN]
Description: Length of rawData and scaledData. It should be a multiple of frame size, otherwise the rest of the last frame is lost, because TransferData always starts with frame start.

Parameter: int * read read
Direction: [OUT]
Description: Will receive the real number of data values transferred. This value returned will be less than the expected maxValues if not enough data is available.

Returns:

[ERR_NOERROR](#) (0) on success.
[ERR_NOT_OPEN](#) (-3) if connection is not established.
[ERR_NO_SENSORDATA_AVAILABLE](#) (-13) This error can occur, if MEDAQLib doesn't can interpret data from sensor, because some information is missing. In this case, call the function as described at error text of [GetError](#).
[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.
 Any other asynchronous error happened at interal threads or callbacks or timers.

Name: TransferDataTS **TransferDataTS**
Description:
 Same as [TransferData](#) but with an additional parameter to retrieve timestamp of data.
Declaration:

```
ERR_CODE TransferDataTS (DWORD instanceHandle, int *rawData, double
    *scaledData, int maxValues, int *read, double *timestamp);
```

Parameter: DWORD instanceHandle instanceHandle
Direction: [IN]
Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: int * rawData	rawData
Direction: [OUT]	
Description: Pointer to array of values to retrieve data from sensor as raw values.	
Parameter: double * scaledData	scaledData
Direction: [OUT]	
Description: Pointer to array of values to retrieve data frame from sensor scaled by sensor range.	
If rawData is null or scaledData is null, no data is transferred for this parameter. If both parameters are null all buffers used are emptied.	
The meaning of the raw and scaled values is described at each sensor section. Please see parameter IP_ScaleErrorValues for scaling error values.	
Parameter: int maxValues	maxValues
Direction: [IN]	
Description: Length of rawData and scaledData. It should be a multiple of frame size, otherwise the rest of the last frame is lost, because TransferDataTS always starts with frame start.	
Parameter: int * read	read
Direction: [OUT]	
Description: Will receive the real number of data values transferred. This value returned will be less than the expected maxValues if not enough data is available.	
Parameter: double * timestamp	timestamp
Direction: [OUT]	
Description: Will receive the timestamp of the first (oldest) value in data array. It is in milli seconds starting at 01.01.1970 01:00.	
It can be used to synchronize data from different sensors.	
Each time when the internal ring buffer is cleared (e.g. at sensor command), it is resetted and when next data block arrives it is calculated by actual time minus acquire duration of data block (at sensor).	
At TransferData(TS) it is automatically increased (by adding duration of block, calculated from expected datarate). But if data is discarded (because of the ring buffer overflow or synchronization lost), it is not incremented and therefore no longer correct. In this case, call sensor command Clear_Buffers .	

Returns:

- [ERR_NOERROR](#) (0) on success.
- [ERR_NOT_OPEN](#) (-3) if connection is not established.
- [ERR_NO_SENSORDATA_AVAILABLE](#) (-13) This error can occur, if MEDAQLib doesn't can interpret data from sensor, because some information is missing. In this case, call the function as described at error text of [GetError](#).
- [ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.
- Any other asynchronous error happened at internal threads or callbacks or timers.

6.12 Get additional error information

Name: GetError

GetError

Description:

Get the extended error text of last error in MEDAQLib.

This function can be used after an error return from any MEDAQLib function.

Declaration:

```
ERR_CODE GetError (DWORD instanceHandle, LPSTR errText, DWORD maxLen);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPSTR errText

errText

Direction: [OUT]

Description: String buffer to get extended error string.

Parameter: DWORD maxLen

maxLen

Direction: [IN]

Description: Length of string buffer. If the error text is longer as maxLen it is truncated. The string is null terminated.

Returns:

[ERR_NOERROR](#) (0) if no error occurred previously.

[ERR_WRONG_PARAMETER](#) (-18) if the pointer to errText is NULL and maxLen is not 0.

[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

Any other error value reported by previous function or happened asynchronous at interal threads or callbacks or timers.

Name: GetErrorU

GetErrorU

Description:

Get the last error in driver (Unicode version). Rest of functionality is identical to [GetError](#).

Declaration:

```
ERR_CODE GetErrorU (DWORD instanceHandle, LPWSTR errText, DWORD maxLen);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPWSTR errText

errText

Direction: [OUT]

Description: Unicode string buffer to get extended error string.

Parameter: DWORD maxLen

maxLen

Direction: [IN]

Description: Length of string buffer (in characters, not bytes). If the error text is longer as maxLen it is truncated. The string is null terminated.

Returns:

- ERR_NOERROR** (0) on success.
- ERR_WRONG_PARAMETER** (-18) if the pointer to errText is NULL and maxLen is not 0.
- ERR_INSTANCE_NOT_EXIST** (-24) if instanceHandle is not valid.
- Any other error value reported by previous function or happened asynchronous at interal threads or callbacks or timers.

Following errors can occur in MEDAQLib:

ERR_NOERROR (0)

Function was successful, no error occurred.

ERR_FUNCTION_NOT_SUPPORTED (-1)

A function was called which is not supported by sensor or hardware interface.

ERR_CANNOT_OPEN (-2)

The hardware interface could not be opened.

ERR_NOT_OPEN (-3)

The connection to sensor is not established.

ERR_APPLYING_PARAMS (-4)

Internal parameters could not be applied.

ERR_SEND_CMD_TO_SENSOR (-5)

Internal error message, never returned at any MEDAQLib function.

ERR_CLEARUNG_BUFFER (-6)

Hardware or driver buffers failed to clear.

ERR_HW_COMMUNICATION (-7)

Communication with sensor failed or hardware interface detected asynchronous errors or a hardware interface was closed unexpected.

ERR_TIMEOUT_READING_FROM_SENSOR (-8)

A timeout occurred while reading from sensor.

ERR_READING_SENSOR_DATA (-9)

Reading sensor answer from sensor failed or asynchronous read operation failed or **DataAvail_Event** could not be set or transmitted data from sensor changed unexpected.

ERR_INTERFACE_NOT_SUPPORTED (-10)

The specified hardware interface does not exist or is not available for this sensor.

ERR_ALREADY_OPEN (-11)

The connection to sensor is already open.

ERR_CANNOT_CREATE_INTERFACE (-12)

The specified hardware interface cannot be created.

ERR_NO_SENSORDATA_AVAILABLE (-13)

Sensor data should be read but is not available.

ERR_UNKNOWN_SENSOR_COMMAND (-14)

The specified command is not available for the sensor.

ERR_UNKNOWN_SENSOR_ANSWER (-15)

The answer received from sensor is not known by MEDAQLib.

ERR_SENSOR_ANSWER_ERROR (-16)

The sensor returned an error message.

ERR_SENSOR_ANSWER_TOO_SHORT (-17)

The answer received from sensor is not complete.

ERR_WRONG_PARAMETER (-18)

An input parameter is not valid or the datatype does not match.

ERR_NOMEMORY (-19)

The answer does not fit into the available buffer or there is too less system memory to allocate a buffer.

ERR_NO_ANSWER RECEIVED (-20)

Internal error message, never returned at any MEDAQLib function.

ERR_SENSOR_ANSWER_DOES_NOT_MATCH_COMMAND (-21)

Internal error message, never returned at any MEDAQLib function.

ERR_BAUDRATE_TOO_LOW (-22)

Baudrate is too low for this command.

ERR_OVERFLOW (-23)

An asynchronous overflow occured in any hardware or driver buffer.

ERR_INSTANCE_NOT_EXIST (-24)

The specified instance handle is not valid.

ERR_NOT_FOUND (-25)

The parameter (specified by paramName) is not found.

ERR_WARNING (-26)

A warning occured in MEDAQLib, e.g. Ethernet packet counter mismatch, discarding data because of changed parameters, ring buffer overflow (only at [DataAvail](#) or [TransferData](#)), failure at setting IP configuration, ...

ERR_SENSOR_ANSWER_WARNING (-27)

The sensor returned a warning.

6.13 Get version of MEDAQLib dll

Name: GetDLLVersion

GetDLLVersion

Description:

Retrieves the version of the MEDAQLib dll.

This function can be called at any time, no sensor instance is needed. The version is stored in versionStr and is limited to length of maxLen (should be at least 11 bytes).

Declaration:

```
ERR_CODE GetDLLVersion (LPSTR versionStr, DWORD maxLen);
```

Parameter: LPSTR versionStr

versionStr

Direction: [OUT]

Description: String buffer to get version info.

Parameter: DWORD maxLen
Direction: [IN]
Description: Length of string buffer. If the version info is longer as maxLen it is truncated. The string is null terminated.

maxLen

Returns:

[ERR_NOERROR](#) (0) on success.

Name: GetDLLVersionU

GetDLLVersionU
Description:

Retrieves the version of the MEDAQLib dll (Unicode version). Rest of functionality is identical to [GetDLLVersion](#).

Declaration:

```
ERR_CODE GetDLLVersionU (LPWSTR versionStr, DWORD maxLen);
```

Parameter: LPWSTR versionStr
Direction: [OUT]
Description: Unicode string buffer to get version info.

versionStr

Parameter: DWORD maxLen
Direction: [IN]
Description: Length of string buffer (in characters, not bytes). If the version info is longer as maxLen it is truncated. The string is null terminated.

maxLen

Returns:

[ERR_NOERROR](#) (0) on success.

6.14 EnableLogging wrapper function

Wrapper functions for a set of [Set parameters](#) functions and executes the sensor command [EnableLogging](#). This usage of this functions makes the code shorter and more readable.

Name: EnableLogging

EnableLogging
Description:

Set the parameters to enable logging.

Declaration:

```
ERR_CODE EnableLogging (DWORD instanceHandle, BOOL enableLogging, int
    logType, int logLevel, LPCSTR logFile, BOOL logAppend, BOOL logFlush,
    int logSplitSize);
```

Parameter: DWORD instanceHandle
Direction: [IN]
Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

instanceHandle

Parameter: BOOL enableLogging	enableLogging
Direction: [IN]	
Valid values:	
0= FALSE	
1= TRUE	
Description:	This parameter enables or disables logging to file for debugging purposes.
Parameter: int logType	logType
Direction: [IN]	
Valid values:	
A bit combination of following values:	
1= HIGH_TYPE (User <--> MEDAQLib)	
2= MIDDLE_TYPE (Sensor layer <--> Interface layer)	
4= LOW_TYPE (MEDAQLib <--> Hardware driver)	
8= ERROR_TYPE (Any errors reported by MEDAQLib)	
16= DRIVER_TYPE (Hardware driver <--> Sys driver)	
32= APPL_TYPE (Application specific, see LogToFile and LogToFileU)	
Description:	This parameter specifies the type of messages to log.
Parameter: int logLevel	logLevel
Direction: [IN]	
Valid values:	
A bit combination of following values:	
1= EMERGENCY_LEVEL (logging emerging errors)	
2= CRITICAL_LEVEL (logging critical errors)	
4= ERROR_LEVEL (logging errors which occurs)	
8= WARNING_LEVEL (logging warnings from MEDAQLib)	
16= NOTICE_LEVEL (logging notices)	
32= TRACE_LEVEL (logging function calls)	
64= DATA_LEVEL (logging data in binary mode)	
Description:	This parameter specifies the kind of event to log.
Parameter: LPCSTR logFile	logFile
Direction: [IN]	
Description:	File name of log file.
If it is empty or ends with '\' or '/', an automatic generated name ('TCLogFile_%yyyy-%MM-%dd_%hh-%mm-%ss.txt') is appended.	
Many placeholders (...) can be used for automatic name generation.	
Important ones are:	
%h= hour in 24 hours format (0, 1, ..., 9, 10, ..., 23)	
%hh= hour in 24 hours format (00, 01, ..., 09, 10, ..., 23)	
%H= hour in 12 hours format (1, 2, ..., 9, 10, 11, 12)	
%HH= hour in 12 hours format (01, 02, ..., 09, 10, 11, 12)	
%m= minute (0, 1, ..., 9, 10, ..., 59)	
%mm= minute (00, 01, ..., 09, 10, ..., 59)	
%s= second (0, 1, ..., 9, 10, ..., 59)	
%ss= second (00, 01, ..., 09, 10, ..., 59)	
%Ms= millisecond (000, 001, ..., 999)	
%Us= microsecond (000000, 000001, ..., 999999)	
%PP= output for morning, afternoon for US-American time format (AM, PM)	
%d= day (1, 2, ..., 9, 10, ..., 31)	
%dd= day (01, 02, ..., 09, 10, ..., 31)	

%M= month (1, 2, ..., 9, 10, 11, 12)
 %MM= month (01, 02, ..., 09, 10, 11, 12)
 %yy= year (70, 71, ..., 99, 00, 01, ..., 38)
 %yyyy= year (1970, 1971, ..., 1999, 2000, 2001, ..., 2038)
 %DoW= day of week (0= Sunday, 1= Monday, ..., 6= Saturday)
 %DoY= day of year (0, 1, ..., 9, 10, ..., 364, if a leap year 365)
 %WoY= week of year (0, 1, ..., 9, 10, ..., 52), starting with 0= the week with the first Sunday in year (US format)

Parameter: BOOL logAppend logAppend

Direction: [IN]

Valid values:

0= FALSE
1= TRUE

Description: This parameter specifies if the logfile should be cleared at opening or if the new data should be appended to file.

Parameter: BOOL logFlush logFlush

Direction: [IN]

Valid values:

0= FALSE
1= TRUE

Description: This parameter specifies if the logfile should be flushed after each output. In this case, it is sure that all information is stored to logfile before proceeding. But depending on the storage device it can slow down the MEDAQLib.

Parameter: int logSplitSize logSplitSize

Direction: [IN]

Valid values:

Minimum: 0
Maximum: 2147483647 (INT_MAX)

Unit: KB (1024 Bytes)

Description: If this parameter is greater than 0, logfile is closed and reopened when this size is reached. If the file name contains placeholders (%...), a new name is generated before opening. Otherwise, the same file is opened again and if appending is off, the old content is overwritten.

Returns:

Any error value which can be returned by the wrapped functions [SetParameterInt](#), [SetParameterString](#) and [SensorCommand](#).

Name: EnableLoggingU

EnableLoggingU

Description:

Set the parameters to enable logging (Unicode version).

Declaration:

```
ERR_CODE EnableLoggingU (DWORD instanceHandle, BOOL enableLogging,
int logType, int logLevel, LPCWSTR logFile, BOOL logAppend, BOOL
logFlush, int logSplitSize);
```

Parameter: DWORD instanceHandle instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: BOOL enableLogging	enableLogging
Direction: [IN]	
Valid values:	
0= FALSE	
1= TRUE	
Description:	This parameter enables or disables logging to file for debugging purposes.
Parameter: int logType	logType
Direction: [IN]	
Valid values:	
A bit combination of following values:	
1= HIGH_TYPE (User <--> MEDAQLib)	
2= MIDDLE_TYPE (Sensor layer <--> Interface layer)	
4= LOW_TYPE (MEDAQLib <--> Hardware driver)	
8= ERROR_TYPE (Any errors reported by MEDAQLib)	
16= DRIVER_TYPE (Hardware driver <--> Sys driver)	
32= APPL_TYPE (Application specific, see LogToFile and LogToFileU)	
Description:	This parameter specifies the type of messages to log.
Parameter: int logLevel	logLevel
Direction: [IN]	
Valid values:	
A bit combination of following values:	
1= EMERGENCY_LEVEL (logging emerging errors)	
2= CRITICAL_LEVEL (logging critical errors)	
4= ERROR_LEVEL (logging errors which occurs)	
8= WARNING_LEVEL (logging warnings from MEDAQLib)	
16= NOTICE_LEVEL (logging notices)	
32= TRACE_LEVEL (logging function calls)	
64= DATA_LEVEL (logging data in binary mode)	
Description:	This parameter specifies the kind of event to log.
Parameter: LPCWSTR logFile	logFile
Direction: [IN]	
Description:	File name of log file as unicode string.
If it is empty or ends with '\' or '/', an automatic generated name ('TCLogFile_%yyyy-%MM-%dd_%hh-%mm-%ss.txt') is appended.	
Many placeholders (...) can be used for automatic name generation.	
Important ones are:	
%h= hour in 24 hours format (0, 1, ..., 9, 10, ..., 23)	
%hh= hour in 24 hours format (00, 01, ..., 09, 10, ..., 23)	
%H= hour in 12 hours format (1, 2, ..., 9, 10, 11, 12)	
%HH= hour in 12 hours format (01, 02, ..., 09, 10, 11, 12)	
%m= minute (0, 1, ..., 9, 10, ..., 59)	
%mm= minute (00, 01, ..., 09, 10, ..., 59)	
%s= second (0, 1, ..., 9, 10, ..., 59)	
%ss= second (00, 01, ..., 09, 10, ..., 59)	
%Ms= millisecond (000, 001, ..., 999)	
%Us= microsecond (000000, 000001, ..., 999999)	
%PP= output for morning, afternoon for US-American time format (AM, PM)	
%d= day (1, 2, ..., 9, 10, ..., 31)	
%dd= day (01, 02, ..., 09, 10, ..., 31)	

%M= month (1, 2, ..., 9, 10, 11, 12)
 %MM= month (01, 02, ..., 09, 10, 11, 12)
 %yy= year (70, 71, ..., 99, 00, 01, ..., 38)
 %yyyy= year (1970, 1971, ..., 1999, 2000, 2001, ..., 2038)
 %DoW= day of week (0= Sunday, 1= Monday, ..., 6= Saturday)
 %DoY= day of year (0, 1, ..., 9, 10, ..., 364, if a leap year 365)
 %WoY= week of year (0, 1, ..., 9, 10, ..., 52), starting with 0= the week with the first Sunday in year (US format)

Parameter: BOOL logAppend logAppend

Direction: [IN]

Valid values:

- 0= FALSE
- 1= TRUE

Description: This parameter specifies if the logfile should be cleared at opening or if the new data should be appended to file.

Parameter: BOOL logFlush logFlush

Direction: [IN]

Valid values:

- 0= FALSE
- 1= TRUE

Description: This parameter specifies if the logfile should be flushed after each output. In this case, it is sure that all information is stored to logfile before proceeding. But depending on the storage device it can slow down the MEDAQLib.

Parameter: int logSplitSize logSplitSize

Direction: [IN]

Valid values:

- Minimum:** 0
- Maximum:** 2147483647 (INT_MAX)

Unit: KB (1024 Bytes)

Description: If this parameter is greater than 0, logfile is closed and reopened when this size is reached. If the file name contains placeholders (%...), a new name is generated before opening. Otherwise, the same file is opened again and if appending is off, the old content is overwritten.

Returns:

Any error value which can be returned by the wrapped functions [SetParameterIntU](#), [SetParameterStringU](#) and [SensorCommand](#).

6.15 User logging functions

This functions allow the user to add own lines to MEDAQLib Logfile. It can be used for debugging purposes.

Name: LogToFile LogToFile

Description:

- Add a line to MEDAQLib Logfile.

Declaration:

```
ERR_CODE LogToFile (DWORD instanceHandle, int logLevel, LPCSTR location,
LPCSTR message, ...);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: int logLevel

logLevel

Direction: [IN]

Valid values:

- 1= EMERGENCY_LEVEL
- 2= CRITICAL_LEVEL
- 4= ERROR_LEVEL
- 8= WARNING_LEVEL
- 16= NOTICE_LEVEL
- 32= TRACE_LEVEL
- 64= DATA_LEVEL

Description: This parameter specifies the level for the line to log.

Parameter: LPCSTR location

location

Direction: [IN]

Description: Location in source code where the log line is generated. Is shown in LogFile.

Parameter: LPCSTR message

message

Direction: [IN]

Description: Logging message. This parameter can be used at same as format string at C printf function (e.g. "%d, %f, %s").

Parameter: variable argument list

variable argument list

Direction: [IN]

Description: Depending on message string, additional parameters must be specified.

Returns:

[ERR_NOERROR](#) (0) on success.

[ERR_INSTANCE_NOT_EXIST](#) (-24) if instanceHandle is not valid.

Name: LogToFileU

LogToFileU
Description:

Add a line to MEDAQLib Logfile (Unicode version).

Declaration:

```
ERR_CODE LogToFileU (DWORD instanceHandle, int logLevel, LPCWSTR location,
LPCWSTR message, ...);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: int logLevel logLevel

Direction: [IN]

Valid values:

- 1= EMERGENCY_LEVEL
- 2= CRITICAL_LEVEL
- 4= ERROR_LEVEL
- 8= WARNING_LEVEL
- 16= NOTICE_LEVEL
- 32= TRACE_LEVEL
- 64= DATA_LEVEL

Description: This parameter specifies the level for the line to log.

Parameter: LPCWSTR location location

Direction: [IN]

Description: Location in source code where the log line is generated. Is shown in LogFile.

Parameter: LPCWSTR message message

Direction: [IN]

Description: Logging message. This parameter can be used at same as format string at C printf function (e.g. "%d, %f, %s").

Parameter: variable argument list variable argument list

Direction: [IN]

Description: Depending on message string, additional parameters must be specified.

Returns:

ERR_NOERROR (0) on success.

ERR_INSTANCE_NOT_EXIST (-24) if instanceHandle is not valid.

6.16 OpenSensor wrapper functions

Wrapper functions for [OpenSensor](#) to open a specific interface. This usage of this functions makes the code shorter and more readable. This functions can be used in combination with any [Set parameters](#) functions, which must be called before.

Name: OpenSensorRS232 [OpenSensorRS232](#)

Description:

Set the parameters for [RS232](#) serial interface before calling [OpenSensor](#).

Declaration:

```
ERR_CODE OpenSensorRS232 (DWORD instanceHandle, LPCSTR port);
```

Parameter: DWORD instanceHandle instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR port port

Direction: [IN]

Valid values:

"COM1"
"COM2"

...

Description: Name of the serial interface. Before opening the interface using CreateFile, the string is prefixed with "\\.\\".

Returns:

Any error value which can be returned by the wrapped functions [SetParameterString](#) and [OpenSensor](#).

Name: OpenSensorRS232U

OpenSensorRS232U

Description:

Set the parameters for [RS232](#) serial interface before calling [OpenSensor](#) (Unicode version).

Declaration:

```
ERR_CODE OpenSensorRS232U (DWORD instanceHandle, LPCWSTR port);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR port

port

Direction: [IN]

Valid values:

"COM1"
"COM2"

...

Description: Name of the serial interface as unicode string. Before opening the interface using CreateFile, the string is prefixed with "\\.\\".

Returns:

Any error value which can be returned by the wrapped functions [SetParameterStringU](#) and [OpenSensor](#).

Name: OpenSensorIF2004

OpenSensorIF2004

Description:

Set the parameters for [IF2004](#) interface card before calling [OpenSensor](#).

Declaration:

```
ERR_CODE OpenSensorIF2004 (DWORD instanceHandle, int cardInstance,
                           int channelNumber);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: int cardInstance cardInstance

Direction: [IN]

Valid values:

Minimum: 0

Maximum: 15

Description: Instance number of the IF2004 interface card. The cards are enumerated by the OS and the only way to distinguish is the card instance number. It does not change at least there are no changes at the PCI bus.

Parameter: int channelNumber channelNumber

Direction: [IN]

Valid values:

Minimum: PCI_CARD_IF2004: -1, otherwise 0

Maximum: 3

Description: Channel number on IF2004 Interface card. If the Encoder on the IF2004 card should be used to store values synchronously to a sensor, the channel number 3 is reserved for it. Otherwise (-1) the FIFO cannot be used for Encoder. Sensors can be carried on each channel.

Returns:

Any error value which can be returned by the wrapped functions [SetParameterInt](#) and [OpenSensor](#).

Name: OpenSensorIF2004_USB

OpenSensorIF2004_USB

Description:

Set the parameters for USB adpater IF2004 before calling [OpenSensor](#).

Declaration:

```
ERR_CODE OpenSensorIF2004_USB (DWORD instanceHandle, int deviceInstance,
                               LPCSTR serialNumber, LPCSTR port, int channelNumber);
```

Parameter: DWORD instanceHandle instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: int deviceInstance deviceInstance

Direction: [IN]

Valid values:

Minimum: 0

Maximum: 255

Description: Instance number of the USB adapter IF2004. The devices are enumerated by the OS. It may change if any devices are plugged or unplugged at USB bus.

Parameter: LPCSTR serialNumber serialNumber

Direction: [IN]

Valid values:

Minimum: "0000001"

Maximum: "9999999"

Description: Serial number of the USB adapter (optional). If not specified, deviceInstance is used. Leading zeros can be dismissed.

Parameter: LPCSTR port

port

Direction: [IN]

Valid values:

"COM1"

"COM2"

...

Description: Name of the serial interface part of USB adapter, e.g. COM1, COM2, ... (optional). If not specified, serialNumber or deviceInstance is used.

Parameter: int channelNumber

channelNumber

Direction: [IN]

Valid values:

Minimum: USB_ADAPTER_IF2004: -1, otherwise 0

Maximum: 4

Description: Channel number on USB adapter IF2004. If the Digital inputs on the USB adapter IF2004 should be used to store values synchronously to a sensor, the channel number 4 is reserved for it. -1 means, no data channel is written to FIFO and cannot be read using [TransferData](#) or [Poll](#). This mode can be used if USB adapter IF2004 should only be parametrized.

Returns:

Any error value which can be returned by the wrapped functions [SetParameterInt](#), [SetParameterString](#) and [OpenSensor](#).

Name: OpenSensorIF2004_USBU

OpenSensorIF2004_USBU

Description:

Set the parameters for USB adapter IF2004 before calling [OpenSensor](#) (Unicode version).

Declaration:

```
ERR_CODE OpenSensorIF2004_USBU (DWORD instanceHandle, int deviceInstance,
    LPCWSTR serialNumber, LPCWSTR port, int channelNumber);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: int deviceInstance

deviceInstance

Direction: [IN]

Valid values:

Minimum: 0

Maximum: 255

Description: Instance number of the USB adapter IF2004. The devices are enumerated by the OS. It may change if any devices are plugged or unplugged at USB bus.

Parameter: LPCWSTR serialNumber serialNumber

Direction: [IN]

Valid values:

Minimum: "0000001"

Maximum: "9999999"

Description: Serial number of the USB adapter (optional). If not specified, deviceInstance is used. Leading zeros can be dismissed.

Parameter: LPCWSTR port port

Direction: [IN]

Valid values:

"COM1"

"COM2"

...

Description: Name of the serial interface part of USB adapter, e.g. COM1, COM2, ... (optional). If not specified, serialNumber or deviceInstance is used.

Parameter: int channelNumber channelNumber

Direction: [IN]

Valid values:

Minimum: USB_ADAPTER_IF2004: -1, otherwise 0

Maximum: 4

Description: Channel number on USB adapter IF2004. If the Digital inputs on the USB adapter IF2004 should be used to store values synchronously to a sensor, the channel number 4 is reserved for it. -1 means, no data channel is written to FIFO and cannot be read using [TransferData](#) or [Poll](#). This mode can be used if IF2008 should only be parametrized.

Returns:

Any error value which can be returned by the wrapped functions [SetParameterInt](#), [SetParameterStringU](#) and [OpenSensor](#).

Name: OpenSensorIF2008 OpenSensorIF2008

Description:

Set the parameters for IF2008 interface card before calling [OpenSensor](#).

Declaration:

```
ERR_CODE OpenSensorIF2008 (DWORD instanceHandle, int cardInstance,
                           int channelNumber);
```

Parameter: DWORD instanceHandle instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: int cardInstance cardInstance

Direction: [IN]

Valid values:

Minimum: 0

Maximum: 15

Description: Instance number of the IF2008 interface card. The cards are enumerated by the OS and the only way to distinguish is the card instance number. It does not change at least there are no changes at the PCI bus.

Parameter: int channelNumber channelNumber

Direction: [IN]

Valid values:

Condition: PCI_CARD_IF2008

-1= No data acquisition

6= Encoder 1

7= Encoder 2

8= Digital IN

9= Digital RxD

10= ADC 1 (Analog/Digital converter)

11= ADC 2 (Analog/Digital converter)

Valid values:

Condition: otherwise (sensors)

0= Sensor 1 (Base Board, Connector X1)

1= Sensor 2 (Base Board, Connector X1)

2= Sensor 3 (Base Board, Connector X2)

3= Sensor 4 (Base Board, Connector X2)

4= Sensor 5 (Extension Board, Connector X1)

5= Sensor 6 (Extension Board, Connector X1)

Description: Channel number on IF2008 Interface card.

Attention! Sensor 5 and 6 are only available if IF2008E extension card is installed. Digital IN is only available if IF2008E extension card or IF2008IO extension slot is installed. ADC is only available if IF2008E extension card is installed.

-1 means, no data channel is written to FIFO and cannot be read using [TransferData](#) or [Poll](#). This mode can be used if IF2008 should only be parametrized.

Returns:

Any error value which can be returned by the wrapped functions [SetParameterInt](#) and [OpenSensor](#).

Name: OpenSensorTCPIP [OpenSensorTCPIP](#)

Description:

Set the parameters for TCP/IP ethernet interface before calling [OpenSensor](#).

Declaration:

```
ERR_CODE OpenSensorTCPIP (DWORD instanceHandle, LPCSTR remoteAddr);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR remoteAddr

remoteAddr

Direction: [IN]

Description: IP address of the remote sensor (TCP server).

Returns:

Any error value which can be returned by the wrapped functions [SetParameterString](#) and [OpenSensor](#).

Name: OpenSensorTCPIPU

OpenSensorTCPIPU
Description:

Set the parameters for TCP/IP ethernet interface before calling [OpenSensor](#) (Unicode version).

Declaration:

```
ERR_CODE OpenSensorTCPIPU (DWORD instanceHandle, LPCWSTR remoteAddr);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR remoteAddr

remoteAddr

Direction: [IN]

Description: IP address of the remote sensor (TCP server) as unicode string.

Returns:

Any error value which can be returned by the wrapped functions [SetParameterStringU](#) and [OpenSensor](#).

Name: OpenSensorWinUSB

OpenSensorWinUSB
Description:

Set the parameters for USB interface via [WinUSB](#) before calling [OpenSensor](#).

Declaration:

```
ERR_CODE OpenSensorWinUSB (DWORD instanceHandle, int deviceInstance);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: int deviceInstance

deviceInstance

Direction: [IN]

Valid values:

Minimum: 0

Maximum: 255

Description: Instance number of the USB device. The devices are enumerated by the OS and the only way to distinguish is the device instance number. It does not change at least there are no changes at the USB bus (plug/unplug devices).

Returns:

Any error value which can be returned by the wrapped functions [SetParameterInt](#) and [OpenSensor](#).

6.17 ExecSCmd wrapper functions

Wrapper functions for a set of [Set parameters](#) and [Get parameters](#) functions and [SensorCommand](#). This usage of this functions makes the code shorter and more readable. This functions can be used in combination with any [Set parameters](#) functions, which must be called before.

Name: ExecSCmd

ExecSCmd

Description:

Set the sensor command name and executes the sensor command.

Declaration:

```
ERR_CODE ExecSCmd (DWORD instanceHandle, LPCSTR sensorCommand);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR sensorCommand

sensorCommand

Direction: [IN]

Description: Name of the sensor command (used for parameter [S_Command](#)).

Returns:

Any error value which can be returned by the wrapped functions [SetParameterString](#) and [SensorCommand](#).

Name: ExecSCmdU

ExecSCmdU

Description:

Set the sensor command name and executes the sensor command (Unicode version).

Declaration:

```
ERR_CODE ExecSCmdU (DWORD instanceHandle, LPCWSTR sensorCommand);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR sensorCommand

sensorCommand

Direction: [IN]

Description: Name of the sensor command (used for parameter [S_Command](#)) as unicode string.

Returns:

Any error value which can be returned by the wrapped functions [SetParameterStringU](#) and [SensorCommand](#).

Name: SetIntExecSCmd

SetIntExecSCmd
Description:

Set the sensor command name and an integer parameter and executes the sensor command.

Declaration:

```
ERR_CODE SetIntExecSCmd (DWORD instanceHandle, LPCSTR sensorCommand,
                         LPCSTR paramName, int paramInt);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR sensorCommand

sensorCommand

Direction: [IN]

Description: Name of the sensor command (used for parameter [S_Command](#)).

Parameter: LPCSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Sensor commands](#) and following.

Description: Name of the parameter as string.

Parameter: int paramInt

paramValue

Direction: [IN]

Description: Value of the parameter.

Returns:

Any error value which can be returned by the wrapped functions [SetParameterString](#), [SetParameterInt](#) and [SensorCommand](#).

Name: SetIntExecSCmdU

SetIntExecSCmdU
Description:

Set the sensor command name and an integer parameter and executes the sensor command (Unicode version).

Declaration:

```
ERR_CODE SetIntExecSCmdU (DWORD instanceHandle, LPCWSTR sensorCommand,
                           LPCWSTR paramName, int paramInt);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR sensorCommand

sensorCommand

Direction: [IN]

Description: Name of the sensor command (used for parameter [S_Command](#)) as unicode string.

Parameter: LPCWSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Sensor commands](#) and following.

Description: Name of the parameter as unicode string.

Parameter: int paramValue

paramValue

Direction: [IN]

Description: Value of the parameter.

Returns:

Any error value which can be returned by the wrapped functions [SetParameterStringU](#), [SetParameterIntU](#) and [SensorCommand](#).

Name: SetDoubleExecSCmd

SetDoubleExecSCmd

Description:

Set the sensor command name and an double parameter and executes the sensor command.

Declaration:

```
ERR_CODE SetDoubleExecSCmd (DWORD instanceHandle, LPCSTR sensorCommand,
    LPCSTR paramName, double paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR sensorCommand

sensorCommand

Direction: [IN]

Description: Name of the sensor command (used for parameter [S_Command](#)).

Parameter: LPCSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Sensor commands](#) and following.

Description: Name of the parameter as string.

Parameter: double paramValue

paramValue

Direction: [IN]

Description: Value of the parameter.

Returns:

Any error value which can be returned by the wrapped functions [SetParameterStringU](#), [SetParameterDouble](#) and [SensorCommand](#).

Name: SetDoubleExecSCmdU

SetDoubleExecSCmdU
Description:

Set the sensor command name and an double parameter and executes the sensor command (Unicode version).

Declaration:

```
ERR_CODE SetDoubleExecSCmdU (DWORD instanceHandle, LPCWSTR sensorCommand,
    LPCWSTR paramName, double paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR sensorCommand

sensorCommand

Direction: [IN]

Description: Name of the sensor command (used for parameter [S_Command](#)) as unicode string.

Parameter: LPCWSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Sensor commands](#) and following.

Description: Name of the parameter as unicode string.

Parameter: double paramValue

paramValue

Direction: [IN]

Description: Value of the parameter.

Returns:

Any error value which can be returned by the wrapped functions [SetParameterStringU](#), [SetParameterDoubleU](#) and [SensorCommand](#).

Name: SetStringExecSCmd

SetStringExecSCmd
Description:

Set the sensor command name and an string parameter and executes the sensor command.

Declaration:

```
ERR_CODE SetStringExecSCmd (DWORD instanceHandle, LPCSTR sensorCommand,
    LPCSTR paramName, LPCSTR paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR sensorCommand

sensorCommand

Direction: [IN]

Description: Name of the sensor command (used for parameter [S_Command](#)).

Parameter: LPCSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Sensor commands](#) and following.

Description: Name of the parameter as string.

Parameter: LPCSTR paramValue

paramValue

Direction: [IN]

Description: Value of the parameter.

Returns:

Any error value which can be returned by the wrapped functions [SetParameterString](#) and [SensorCommand](#).

Name: SetStringExecSCmdU

SetStringExecSCmdU

Description:

Set the sensor command name and an string parameter and executes the sensor command (Unicode version).

Declaration:

```
ERR_CODE SetStringExecSCmdU (DWORD instanceHandle, LPCWSTR sensorCommand,
    LPCWSTR paramName, LPCWSTR paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR sensorCommand

sensorCommand

Direction: [IN]

Description: Name of the sensor command (used for parameter [S_Command](#)) as unicode string.

Parameter: LPCWSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Sensor commands](#) and following.

Description: Name of the parameter as unicode string.

Parameter: LPCWSTR paramValue

paramValue

Direction: [IN]

Description: Value of the parameter as unicode string.

Returns:

Any error value which can be returned by the wrapped functions [SetParameterStringU](#) and [SensorCommand](#).

Name: ExecSCmdGetInt

ExecSCmdGetInt
Description:

Set the sensor command name, executes the sensor command and get a integer parameter.

Declaration:

```
ERR_CODE ExecSCmdGetInt (DWORD instanceHandle, LPCSTR sensorCommand,
    LPCSTR paramName, int *paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR sensorCommand

sensorCommand

Direction: [IN]

Description: Name of the sensor command (used for parameter [S_Command](#)).

Parameter: LPCSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Sensor commands](#) and following.

Description: Name of the parameter as string.

Parameter: int * paramValue

paramValue

Direction: [OUT]

Description: Pointer to a variable retrieving the parameter

Returns:

Any error value which can be returned by the wrapped functions [SetParameterString](#), [SensorCommand](#) and [GetParameterInt](#).

Name: ExecSCmdGetIntU

ExecSCmdGetIntU
Description:

Set the sensor command name, executes the sensor command and get a integer parameter (Unicode version).

Declaration:

```
ERR_CODE ExecSCmdGetIntU (DWORD instanceHandle, LPCWSTR sensorCommand,
    LPCWSTR paramName, int *paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR sensorCommand

sensorCommand

Direction: [IN]

Description: Name of the sensor command (used for parameter [S_Command](#)) as unicode string.

Parameter: LPCWSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Sensor commands](#) and following.

Description: Name of the parameter as unicode string.

Parameter: int * paramValue

paramValue

Direction: [OUT]

Description: Pointer to a variable retrieving the parameter

Returns:

Any error value which can be returned by the wrapped functions [SetParameterStringU](#), [SensorCommand](#) and [GetParameterIntU](#).

Name: ExecSCmdGetDouble

ExecSCmdGetDouble

Description:

Set the sensor command name, executes the sensor command and get a double parameter.

Declaration:

```
ERR_CODE ExecSCmdGetDouble (DWORD instanceHandle, LPCSTR sensorCommand,
    LPCSTR paramName, double *paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR sensorCommand

sensorCommand

Direction: [IN]

Description: Name of the sensor command (used for parameter [S_Command](#)).

Parameter: LPCSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Sensor commands](#) and following.

Description: Name of the parameter as string.

Parameter: double * paramValue

paramValue

Direction: [OUT]

Description: Pointer to a variable retrieving the parameter

Returns:

Any error value which can be returned by the wrapped functions [SetParameterString](#), [SensorCommand](#) and [GetParameterDouble](#).

Name: ExecSCmdGetDoubleU

ExecSCmdGetDoubleU
Description:

Set the sensor command name, executes the sensor command and get a double parameter (Unicode version).

Declaration:

```
ERR_CODE ExecSCmdGetDoubleU (DWORD instanceHandle, LPCWSTR sensorCommand,
    LPCWSTR paramName, double *paramValue);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR sensorCommand

sensorCommand

Direction: [IN]

Description: Name of the sensor command (used for parameter [S_Command](#)) as unicode string.

Parameter: LPCWSTR paramName

paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Sensor commands](#) and following.

Description: Name of the parameter as unicode string.

Parameter: double * paramValue

paramValue

Direction: [OUT]

Description: Pointer to a variable retrieving the parameter

Returns:

Any error value which can be returned by the wrapped functions [SetParameterStringU](#), [SensorCommand](#) and [GetParameterDoubleU](#).

Name: ExecSCmdGetString

ExecSCmdGetString
Description:

Set the sensor command name, executes the sensor command and get a string parameter.

Declaration:

```
ERR_CODE ExecSCmdGetString (DWORD instanceHandle, LPCSTR sensorCommand,
    LPCSTR paramName, LPSTR paramValue, DWORD *maxLen);
```

Parameter: DWORD instanceHandle

instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCSTR sensorCommand

sensorCommand

Direction: [IN]

Description: Name of the sensor command (used for parameter [S_Command](#)).

Parameter: LPCSTR paramName paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Sensor commands](#) and following.

Description: Name of the parameter as string.

Parameter: LPSTR paramValue paramValue

Direction: [OUT]

Description: Pointer to a variable (character buffer) retrieving the parameter.

Parameter: DWORD * maxLen maxLen

Direction: [IN/OUT]

Description: The buffer must be allocated by the application. The size of the buffer is specified at maxLen. If the resulting string is larger than maxLen, it is truncated and [ERR_NOMEMORY](#) (-19) is returned. The real length of the string (maybe truncated) is returned in maxLen too. If paramValue is NULL, the length of the containing string is returned in maxLen.

Returns:

Any error value which can be returned by the wrapped functions [SetParameterString](#), [SensorCommand](#) and [GetParameterString](#).

Name: ExecSCmdGetStringU

ExecSCmdGetStringU

Description:

Set the sensor command name, executes the sensor command and get a string parameter (Unicode version).

Declaration:

```
ERR_CODE ExecSCmdGetStringU (DWORD instanceHandle, LPCWSTR sensorCommand,
    LPCWSTR paramName, LPWSTR paramValue, DWORD *maxLen);
```

Parameter: DWORD instanceHandle instanceHandle

Direction: [IN]

Description: Number of the sensor instance, previously returned by [CreateSensorInstance](#).

Parameter: LPCWSTR sensorCommand sensorCommand

Direction: [IN]

Description: Name of the sensor command (used for parameter [S_Command](#)) as unicode string.

Parameter: LPCWSTR paramName paramName

Direction: [IN]

Valid value: All available values are listed at chapter [Sensor commands](#) and following.

Description: Name of the parameter as unicode string.

Parameter: LPWSTR paramValue paramValue

Direction: [OUT]

Description: Pointer to a variable (wide character buffer) retrieving the parameter.

Parameter: DWORD * maxLen

maxLen

Direction: [IN/OUT]

Description: The buffer must be allocated by the application. The size of the buffer (in characters, not bytes) is specified at maxLen. If the resulting string is larger than maxLen, it is truncated and **ERR_NOMEMORY** (-19) is returned. The real length of the string (maybe truncated) is returned in maxLen too.

If paramValue is NULL, the length of the containing string is returned in maxLen.

Returns:

Any error value which can be returned by the wrapped functions [SetParameterStringU](#), [SensorCommand](#) and [GetParameterStringU](#).

7 Parameters

7.1 Naming conventions

Parameters for opening the interface (Interface parameters, IP_...) are used in the Function [OpenSensor](#).

Parameters for communicating with the sensor (Sensor parameters, SP_...) and answer from sensor (Sensor answer, SA_...) are used in Function [SensorCommand](#).

Direction in parameters is defined as follows:

Down: From application to driver or sensor.

Up: From sensor or driver to application.

Parameters without a default value are obligatory and must be specified. For the other parameters, the default value is used if not specified.

When a parameter contains two periods (e.g: SP_X1..16), it stands for a sequence of parameters.

7.2 Interface parameters

An interface is the hardware device directly opened by MEDAQLib to establish connection to the sensor. It communicates directly with the sensor.

Attention! The [CONTROLLER_CSP2008](#) is not an interface, it is treated as sensor (resp. controller). The sensors behind cannot be accessed by MEDAQLib.

7.2.1 All Interfaces

Parameter: String IP_Interface

IP_Interface

Direction: Down

Valid values:

- "RS232"
- "IF2004_USB"
- "IF2004"
- "IF2008"
- "TCP/IP"
- "WinUSB"

Description: Interface type where the sensor is plugged on. If the IF2001_USB (RS422) is used "RS232" must be set, because a RS232 interface is emulated.

With an RS422/RS232 to Ethernet converter additional sensors can be connected via "TCP/IP".

Parameter: int IP_AutomaticMode

IP_AutomaticMode

Direction: Down

Valid values:

- A bit combination of following values:
- "First bit (1)= Retrieve sensor information to setup MEDAQLib"
- "Second bit (2)= Activate data output at sensor"

Default: 1

Description: First bit (1) allows MEDAQLib to retrieve information from sensor automatically if needed. Normally this happens at [OpenSensor](#), but in some cases, it can happen after [SensorCommand](#), too.

Second bit (2) allows MEDAQLib to change sensor interface parameters (if needed) so is outputs data. This only happens at [OpenSensor](#).

Parameter: int IP_ScaleErrorValues IP_ScaleErrorValues

Direction: Down

Valid values:

- 1= last valid value
- 2= set to fixed value
- 3= set to negative error value

Default: 2

Description: If sensor values are not valid, they cannot be scaled. So invalid values can be set to the last valid value or can be set to a fixed value (see [IP_FixedErrorHandler](#)) or can be set to a negative error value:

For [SENSOR_ODC2500](#) and [SENSOR_ODC2600](#):

-10	DSP No edge
-11	DSP At the beginning of the picture
-12	DSP At the end of the picture
-13	DSP Dark - bright edge
-14	DSP Bright - dark edge
-15	DSP Min. number of edges
-16	DSP Max. number of edges
-17	DSP Invalid measuring program
-18	DSP Segment 1st edge >= 2nd edge
-19	DSP Segment number of edges < last edge
-20	DSP Invalid working distance
-22	ARM Laser off
-23	ARM Invalid float
-24	ARM DMA setup error
-90	ASCII mode at IF2004 not supported

For all ILD sensors, except [SENSOR_ILD2300](#) (not every sensor supports all errors):

-1	F1 bad objekt (no objekt cognizable)
-2	F2 out of range + (to near at sensor)
-3	F3 out of range - (to far from sensor)
-4	F4 poor target (objekt not evaluable)
-5	F5 Laser off (external laser off)
-6	Measured object moves towards sensor. SENSOR_ILD1700 : Sensor in trigger mode Trigger and pulses comes to fast.
-7	Measured object moves away from sensor

For [SENSOR_ILD2300](#), [SENSOR_IFD2445](#), [SENSOR_IFD2451](#), [SENSOR_IFD2461](#), [SENSOR_IFD2471](#) and [SENSOR_ACS7000](#) (not every sensor supports all errors):

-1	Scaling error RS422 interface underflow
-2	Scaling errors RS422 interface overflow
-3	Too much data for selected baudrate
-4	No peak/edge available
-5	Peak is in front of the measuring range
-6	Peak is after the measuring range
-7	Measurement cannot be calculated
-8	Measurement cannot be evaluated, global error
-9	Peak is too wide
-10	Laser beam is off

For **SENSOR_ILR118x**:

For error values, scaled values are set to 0, raw values are set to positive error values:

15	E15 - Excessively poor reflexes. Distance sensor (Front edge) against target < 0.1m.
16	E16 - Excessively strong reflexes.
17	E17 - Too much steady light (for example sun).
18	E18 - Only in DX mode (50 Hz): Too much difference between measured and pre-calculated value.
19	E19 - Only in DX mode (50 Hz): Target motion speed > 10 m/s.
23	E23 - Temperature below -10 °C
24	E24 - Temperature above +60 °C
31	E31 - Faulty EEPROM checksum, hardware error.
51	E51 - Failure to set avalanche voltage of diode laser. 1. straylight or 2. hardware error.
52	E52 - Laser current too high / laser defective.
53	E53 - One or more parameters in the EEPROM not set (Consequence: Division by 0).
54	E54 - Hardware error (PLL).
55	E55 - Hardware error.
61	E61 - Used parameter is inadmissible, invalid command sent.
62	E62 - 1. Hardware error 2. wrong value in interface communication (Parity error SIO).
63	E63 - SIO overflow.
64	E64 - Framing-Error SIO.

For **SENSOR_ILR1191**:

For error values, scaled values are set to 0, raw values are set to positive error values:

2	E02 - No target.
4	E04 - Laser defect.

For **SENSOR_ACS7000**:

-1	RS422 scaling underflow
-2	RS422 scaling overflow
-3	Too much data for this baudrate

Parameter: double IP_FixedErrorValue

IP_FixedErrorValue

Direction: Down

Valid values:

Minimum: -1.79769e+308 (-DBL_MAX)

Maximum: 1.79769e+308 (DBL_MAX)

Default: -1.79769e+308 (-DBL_MAX)

Description: If IP_ScaleErrorValues is set to fixed value, this value is returned in case of an error.

Parameter: int IP_RingBufferSize

IP_RingBufferSize

Direction: Down

Valid values:

Minimum: 512

Maximum: 1073741824 (1 GB)

Unit: Bytes

Default: 1048576 (1 MB)

Description: Data (values and sensor answer) is collected from sensor into a ring buffer. From there it is converted (value) or interpreted (sensor answer). If the ring buffer size is too small and data is not transferred fast enough an overflow occur. The buffer must be large enough to hold a complete sensor answer (especially IFD's with command Get_CCD or Get_DarkSig, ...).

Parameter: int IP_EnableLogging

IP_EnableLogging

Direction: Down

Valid values:

0= FALSE

1= TRUE

Default: 0

Description: This parameter enables or disables logging to file for debugging purposes.

Parameter: int IP_LogType

IP_LogType

Direction: Down

Valid values:

A bit combination of following values:

1= HIGH_TYPE (User <--> MEDAQLib)

2= MIDDLE_TYPE (Sensor layer <--> Interface layer)

4= LOW_TYPE (MEDAQLib <--> Hardware driver)

8= ERROR_TYPE (Any errors reported by MEDAQLib)

16= DRIVER_TYPE (Hardware driver <--> Sys driver)

32= APPL_TYPE (Application specific, see [LogFile](#) and [LogFileU](#))

Default: 2147483647 (INT_MAX)

Description: This parameter specifies the type of messages to log.

Parameter: int IP_LogLevel IP_LogLevel

Direction: Down

Valid values:

A bit combination of following values:

- 1= EMERGENCY_LEVEL (logging emerging errors)
- 2= CRITICAL_LEVEL (logging critical errors)
- 4= ERROR_LEVEL (logging errors which occurs)
- 8= WARNING_LEVEL (logging warnings from MEDAQLib)
- 16= NOTICE_LEVEL (logging notices)
- 32= TRACE_LEVEL (logging function calls)
- 64= DATA_LEVEL (logging data in binary mode)

Default: 2147483647 (INT_MAX)

Description: This parameter specifies the kind of event to log.

Parameter: String IP_LogFile IP_LogFile

Direction: Down

Default:

Description: File name of log file.

If it is empty or ends with '\' or '/', an automatic generated name ('TCLogFile_%yyyy-%MM-%dd_%hh-%mm-%ss.txt') is appended.

Many placeholders (...) can be used for automatic name generation.

Important ones are:

- %h= hour in 24 hours format (0, 1, ..., 9, 10, ..., 23)
- %hh= hour in 24 hours format (00, 01, ..., 09, 10, ..., 23)
- %H= hour in 12 hours format (1, 2, ..., 9, 10, 11, 12)
- %HH= hour in 12 hours format (01, 02, ..., 09, 10, 11, 12)
- %m= minute (0, 1, ..., 9, 10, ..., 59)
- %mm= minute (00, 01, ..., 09, 10, ..., 59)
- %s= second (0, 1, ..., 9, 10, ..., 59)
- %ss= second (00, 01, ..., 09, 10, ..., 59)
- %Ms= millisecond (000, 001, ..., 999)
- %Us= microsecond (000000, 000001, ..., 999999)
- %PP= output for morning, afternoon for US-American time format (AM, PM)
- %d= day (1, 2, ..., 9, 10, ..., 31)
- %dd= day (01, 02, ..., 09, 10, ..., 31)
- %M= month (1, 2, ..., 9, 10, 11, 12)
- %MM= month (01, 02, ..., 09, 10, 11, 12)
- %yy= year (70, 71, ..., 99, 00, 01, ..., 38)
- %yyyy= year (1970, 1971, ..., 1999, 2000, 2001, ..., 2038)
- %DoW= day of week (0= Sunday, 1= Monday, ..., 6= Saturday)
- %DoY= day of year (0, 1, ..., 9, 10, ..., 364, if a leap year 365)
- %WoY= week of year (0, 1, ..., 9, 10, ..., 52), starting with 0= the week with the first Sunday in year (US format)

Parameter: int IP_LogAppend IP_LogAppend

Direction: Down

Valid values:

0= FALSE

1= TRUE

Default: 1

Description: This parameter specifies if the logfile should be cleared at opening or if the new data should be appended to file.

Parameter: int IP_LogFlush IP_LogFlush

Direction: Down

Valid values:

0= FALSE
1= TRUE

Default: 0

Description: This parameter specifies if the logfile should be flushed after each output. In this case, it is sure that all information is stored to logfile before proceeding. But depending on the storage device it can slow down the MEDAQLib.

Example how to enable MEDAQLib logging in your application:

```
DWORD instance= CreateSensorInstance (...);  
SetParameterString (instance, "IP_Interface", "...");  
/* Set any other interface parameters */  
SetParameterInt (instance, "IP_EnableLogging", TRUE);  
SetParameterString (instance, "IP_LogFile", "C:\SensorLog.txt");  
DWORD err= OpenSensor (instance);
```

Alternatively you can use the function EnableLogging:

```
DWORD instance= CreateSensorInstance (...);  
EnableLogging (instance, TRUE, INT_MAX, INT_MAX, "C:\SensorLog.txt", TRUE,  
FALSE, 0);  
DWORD err= OpenSensor (instance);
```

Parameter: int IP_LogSplitSize IP_LogSplitSize

Direction: [IN]

Valid values:

Minimum: 0
Maximum: 2147483647 (INT_MAX)

Unit: KB (1024 Bytes)

Default: 0

Description: If this parameter is greater than 0, logfile is closed and reopened when this size is reached. If the file name contains placeholders (%...), a new name is generated before opening. Otherwise, the same file is opened again and if appending is off, the old content is overwritten.

Parameter: int IP_MaxPacketSize IP_MaxPacketSize

Direction: Down

Valid values:

Minimum: 1
Maximum: 2147483647 (INT_MAX)

Unit: Bytes

Default: SENSOR_ILD1401: 1, otherwise INT_MAX

Description: Maximum size of a block which can be transferred to the sensor at once. Because of a small receive FIFO in ILD1401 only 1 Byte after another can be sent to sensor with a break between.

Parameter: int IP_PacketDelay IP_PacketDelay

Direction: Down

Valid values:

Minimum: 0
Maximum: 2147483647 (INT_MAX)

Unit: ms

Default: SENSOR_ILD1401: 1 (RS232) or 3 (TCP/IP), otherwise 0

Description: Break time between sending two blocks to a sensor (see IP_MaxPacketSize).

Parameter: int IP_TimerResolution	IP_TimerResolution
Direction: Down	
Unit: ms	
Valid values:	
-1= Do not set timer resolution.	
0= Use greatest possible accuracy.	
1..255= Resolution in milliseconds.	
Unit: ms	
Default: -1	
Description: Timer resolution (for Windows scheduler, set by timeBeginPeriod).	
For some sensors which sends large amount of data in small packets, it could be useful to set lower timer resolution. This will instruct the scheduler to execute the acquisition thread more often.	
Parameter: int CP_BaseLevelTimeout	CP_BaseLevelTimeout
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 2147483647 (INT_MAX)	
Unit: ms	
Default: 0	
Description: This global timeout is added to the special timeout (specified by CP_SensorAnswerTimeout or internally) when waiting for any sensor answer at SensorCommand .	

7.2.2 RS232

Following sensors supports this interface:

[SENSOR_ILD1401](#), [SENSOR_ODC1202](#), [SENSOR_ODC2500](#), [SENSOR_ODC2600](#), [SENSOR_IFD2401](#), [SENSOR_IFD2431](#), [SENSOR_ILR110x_115x](#), [SENSOR_ILR118x](#), [SENSOR_ILR1191](#) (native).
[SENSOR_ILD1302](#), [SENSOR_ILD1320](#), [SENSOR_ILD1402](#), [SENSOR_ILD1420](#), [SENSOR_ILD1700](#), [SENSOR_ILD2200](#), [SENSOR_ILD2300](#), [SENSOR_IFD2445](#), [SENSOR_IFD2451](#), [SENSOR_IFD2461](#), [SENSOR_IFD2471](#), [SENSOR_ODC2520](#), [SENSOR_ACS7000](#), [CONTROLLER_CSP2008](#), [CONTROLLER_CBOX](#) (additional, e.g. [IF2001_USB](#) (RS422) and RS232 high level interface).
[SENSOR_DT6120](#) (additional, e.g. RS485/USB converter and RS232 high level interface).

Parameter: String IP_Port	IP_Port
Direction: Down	
Valid values:	
"COM1"	
"COM2"	
...	
Description: Name of the serial interface. Before opening the interface using CreateFile, the string is prefixed with "\.\\".	

Parameter: int IP_Baudrate IP_Baudrate

Direction: Down

Valid values:

SENSOR_ILD1401: 38400
 SENSOR_ILD1302, SENSOR_ILD1402: 115200, 57600, 38400, 19200,
 9600
 SENSOR_ILD1320, SENSOR_ILD1420: 1000000, 921600, 691200, 460800,
 256000, 230400, 128000, 115200, 56000, 19200, 9600
 SENSOR_ILD1700: 115200, 57600, 19200, 9600
 SENSOR_ILD2200: 691200, 1250000
 SENSOR_ILD2300: 3500000, 3000000, 2500000, 2000000, 1500000,
 921600, 691200, 460800, 230400, 115200, 9600
 SENSOR_IFD2401, SENSOR_IFD2431: 460800, 230400, 115200, 57600,
 38400, 19200, 9600
 SENSOR_IFD2445, SENSOR_IFD2451, SENSOR_IFD2461, SENSOR_IFD2471:
 4000000, 3500000, 2000000, 1500000, 921600, 691200, 460800,
 230400, 115200, 9600
 SENSOR_ODC1202: 115200, 57600, 38400, 19200, 9600
 SENSOR_ODC2500, SENSOR_ODC2600: 115200, 38400, 19200, 9600
 SENSOR_ODC2520: 4000000, 3500000, 3000000, 2500000, 2000000,
 1500000, 921600, 691200, 460800, 230400, 115200, 9600
 SENSOR_ILR110x_115x: 57600, 38400, 19200, 9600, 4800
 SENSOR_ILR118x: 38400, 19200, 9600, 4800
 SENSOR_ILR1191: 460800, 230400, 115200, 57600, 38400, 19200,
 9600
 CONTROLLER_CSP2008: 115200, 691200
 SENSOR_DT6120: 921600, 460800, 230400, 115200, 9600
 SENSOR_ACS7000: 3500000, 2000000, 1500000, 921600, 691200,
 460800, 230400, 115200, 9600
 CONTROLLER_CBOX: 8000000, 4000000, 3500000, 3000000, 2500000,
 2000000, 1500000, 921600, 691200, 460800, 230400, 115200, 9600

Unit: Baud

Default:

SENSOR_ILD1401, SENSOR_ILR118x: 9600,
 SENSOR_ODC1202: 19200,
 SENSOR_ILD1302, SENSOR_ILD1402, SENSOR_ILD1700, SENSOR_IFD2401,
 SENSOR_IFD2431, SENSOR_IFD2445, SENSOR_IFD2451, SENSOR_IFD2461,
 SENSOR_IFD2471, SENSOR_ILR1191, SENSOR_ACS7000, SENSOR_ODC2500,
 SENSOR_ODC2520, SENSOR_ODC2600, SENSOR_DT6120, CONTROLLER_CSP2008,
 CONTROLLER_CBOX: 115200,
 SENSOR_ILD2200, SENSOR_ILD2300: 691200,
 SENSOR_ILR110x_115x: 38400

Description: Speed of the RS422 serial connection.

Parameter: int IP_Stopbits IP_Stopbits

Direction: Down

Valid values:

SENSOR_ILR110x_115x, SENSOR_ODC2500, SENSOR_ODC2600: 0=
 ONESTOPBIT, 2= TWOSTOPBITS
 otherwise 1= ONE5STOPBITS

Default:

SENSOR_ODC2500, SENSOR_ODC2600: 2,
 otherwise: 0

Description: Number of stop bits of the serial connection.

Parameter: int IP_Parity IP_Parity

Direction: Down

Valid values:

SENSOR_ILR110x_115x, SENSOR_DT6120: 2= EVENPARITY
SENSOR_ODC2500, SENSOR_ODC2600: 0= NOPARITY, 1= ODD-PARITY, 2= EVENPARITY
 otherwise: 0= NOPARITY

Default:

SENSOR_ILR110x_115x, SENSOR_DT6120: 2,
 otherwise: 0

Description: Parity of the serial connection.

Parameter: int IP_ByteSize IP_ByteSize

Direction: Down

Valid values:

SENSOR_ILR110x_115x: 7, 8
 otherwise: 8

Unit: Bit

Default:

SENSOR_ILR110x_115x: 7,
 otherwise: 8

Description: Number of data bits per byte of the serial connection.

Parameter: int IP_SensorAddress IP_SensorAddress

Direction: Down

Valid for sensor:

SENSOR_DT6120

Valid values:

Minimum: 0
Maximum: 126

Default: 126

Description: If the interface is RS485 (with RS232 software emulation), up to 32 slave devices (sensors) can be connected. Therefore a sensor address is required.

7.2.3 IF2004

Following sensors supports this interface:

SENSOR_ILD1302, SENSOR_ILD1320, SENSOR_ILD1402, SENSOR_ILD1420, SENSOR_ILD1700, SENSOR_ILD2200, SENSOR_ILD2300, SENSOR_IFD2445, SENSOR_IFD2451, SENSOR_IFD2461, SENSOR_IFD2471, SENSOR_ODC2500, SENSOR_ODC2520, SENSOR_ODC2600, SENSOR_ACS7000, CONTROLLER_CSP2008, CONTROLLER_CBOX, PCI_CARD_IF2004 (native).

Parameter: int IP_CardInstance IP_CardInstance

Direction: Down

Valid values:

Minimum: 0
Maximum: 15

Default: 0

Description: Instance number of the IF2004 interface card. The cards are enumerated by the OS and the only way to distinguish is the card instance number. It does not change at least there are no changes at the PCI bus.

Parameter: int IP_ChannelNumber IP_ChannelNumber

Direction: Down

Valid values:

Minimum: PCI_CARD_IF2004: -1, otherwise 0

Maximum: 3

Default: PCI_CARD_IF2004: 3, otherwise obligatory

Description: Channel number on IF2004 Interface card. If the Encoder on the IF2004 card should be used to store values synchronously to a sensor, the channel number 3 is reserved for it. Otherwise (-1) the FIFO cannot be used for Encoder. Sensors can be carried on each channel.

Parameter: int IP_UseGate IP_UseGate

Direction: Down

Valid values:

0= FALSE

1= TRUE

Default: 0

Description: The gate input of the card (5V TTL signal) can be used to lock or free the FIFO for data from sensors. This parameter affects always two channels (0+1 or 2+3) because they are on the same connector. The encoder can be locked for FIFO too.

Parameter: int IP_Baudrate IP_Baudrate

Direction: Down

Valid values:

SENSOR_ILD1302, SENSOR_ILD1402, SENSOR_ILD1700: 115200

SENSOR_ILD2200: 691200, 1250000

SENSOR_ILD1320, SENSOR_ILD1420, SENSOR_ILD2300, SENSOR_IFD2445,
 SENSOR_IFD2451, SENSOR_IFD2461, SENSOR_IFD2471, SENSOR_ACS7000,
 CONTROLLER_CSP2008, CONTROLLER_CBOX, SENSOR_ODC2500,
 SENSOR_ODC2520 and SENSOR_ODC2600: 115200, 691200

Unit: Baud

Default:

SENSOR_ILD1302, SENSOR_ILD1402, SENSOR_ILD1700, SENSOR_IFD2445,
 SENSOR_IFD2451, SENSOR_IFD2461, SENSOR_IFD2471, SENSOR_ODC2520,
 SENSOR_ACS7000, CONTROLLER_CSP2008, CONTROLLER_CBOX: 115200,
 SENSOR_ILD2200, SENSOR_ILD2300, SENSOR_ODC2500, SENSOR_ODC2600:
 691200

Description: Speed of the RS422 serial connection. Only the ODC sensors can be used with different baud rates.

Parameter: int IP_SyncMasterChannel IP_SyncMasterChannel

Direction: Down

Valid values:

-1= No synchronisation master

0= Sensor channel 1 (Base Board, Connector 1/2)

1= Sensor channel 2 (Base Board, Connector 1/2)

2= Sensor channel 3 (Base Board, Connector 3/4)

3= Sensor channel 4 (Base Board, Connector 3/4)

Default: -1

Description: Channel number to synchronise with.

Synchronisation is done at driver layer.

Only sensor channels can be synchronized. To synchronize IF2004 internal channels (Encoder), set the latch source for this channel to the

desired master.

If synchronisation is active, a value for master channel and each slave channel is buffered before all values are transferred to MEDAQLib at once. If at a slave channel arrives two values while at master channel no value is available, the older value at the slave channel is deleted.

If at master channel arrives two values while at slave channel no value is available the last value at slave channel is duplicated. But if no last value is available (there was never data on slave channel before), the oldest value at master channel and all other slaves is deleted.

This ensures that the maximum time difference between a value at master and slave channel is one value at master channel, even if the datarates of the sensors are completely different.

To ensure that synchronised values have no offset (e.g. at start or after a sensor command) it is recommended to call ClearBuffer with SP_AllDevices set to true after opening all sensors and after each sensor command.

7.2.4 IF2004_USB

Following sensors supports this interface:

[SENSOR_ILD1401](#) (for sensor ILD1402 in compatibility mode).

[SENSOR_ILD1302](#), [SENSOR_ILD1320](#), [SENSOR_ILD1402](#), [SENSOR_ILD1420](#), [SENSOR_ILD1700](#), [SENSOR_ILD2200](#), [SENSOR_ILD2300](#), [SENSOR_ODC2500](#), [SENSOR_ODC2520](#), [SENSOR_ODC2600](#), [PCI_CARD_IF2008](#), [SENSOR_IFD2401](#), [SENSOR_IFD2431](#), [SENSOR_IFD2445](#), [SENSOR_IFD2451](#), [SENSOR_IFD2461](#), [SENSOR_IFD2471](#), [SENSOR_ACST000](#), [SENSOR_ILR110x_115x](#), [SENSOR_ILR118x](#), [SENSOR_ILR1191](#), [CONTROLLER_CSP2008](#), [CONTROLLER_CBOX](#) (native).

Parameter: int IP_DeviceInstance

IP_DeviceInstance

Direction: Down

Valid values:

Minimum: 0

Maximum: 255

Default: 0

Description: Instance number of the USB device. The devices are enumerated by the OS and the only way to distinguish is the device instance number. It does not change at least there are no changes at the USB bus (plug/unplug devices).

Parameter: String IP_SerialNumber

IP_SerialNumber

Direction: Down

Valid values:

Minimum: "0000001"

Maximum: "9999999"

Description: Serial number of the USB adapter (optional). If not specified, devicelInstance is used. Leading zeros can be dismissed.

Parameter: String IP_Port

IP_Port

Direction: Down

Valid values:

"COM1"

"COM2"

...

Description: Name of the serial interface part of USB adapter, e.g. COM1, COM2, ... (optional). If not specified, SerialNumber or DevicelInstance is used.

Parameter: int IP_ChannelNumber IP_ChannelNumber

Direction: Down

Valid values:

Condition: [USB_ADAPTER_IF2004](#)

-1 = No data acquisition

4 = Digital IN

Valid values:

Condition: otherwise (sensors)

0 = Sensor channel 1 (Connector 1/2)

1 = Sensor channel 2 (Connector 1/2)

2 = Sensor channel 3 (Connector 3/4)

3 = Sensor channel 4 (Connector 3/4)

Description: Channel number on USB adapter IF2004. If the Digital inputs on the USB adapter IF2004 should be used to store values synchronously to a sensor, the channel number 4 is reserved for it. -1 means, no data channel is written to FIFO and cannot be read using [TransferData](#) or [Poll](#). This mode can be used if USB adapter IF2004 should only be parametrized.

Parameter: int IP_Baudrate IP_Baudrate

Direction: Down

Valid values:

[SENSOR_ILD1401](#): 38400

[SENSOR_ILD1302](#), [SENSOR_ILD1402](#): 115200, 57600, 38400, 19200, 9600

[SENSOR_ILD1320](#), [SENSOR_ILD1420](#): 1000000, 921600, 691200, 460800, 256000, 230400, 128000, 115200, 56000, 19200, 9600

[SENSOR_ILD1700](#): 115200, 57600, 19200, 9600

[SENSOR_ILD2200](#): 691200, 1250000

[SENSOR_ILD2300](#): 3500000, 3000000, 2500000, 2000000, 1500000, 921600, 691200, 460800, 230400, 115200, 9600

[SENSOR_IFD2401](#), [SENSOR_IFD2431](#): 460800, 230400, 115200, 57600, 38400, 19200, 9600

[SENSOR_IFD2445](#), [SENSOR_IFD2451](#), [SENSOR_IFD2461](#), [SENSOR_IFD2471](#): 4000000, 3500000, 2000000, 1500000, 921600, 691200, 460800, 230400, 115200, 9600

[SENSOR_ODC2500](#), [SENSOR_ODC2600](#): 115200, 38400, 19200, 9600

[SENSOR_ODC2520](#): 4000000, 3500000, 3000000, 2500000, 2000000, 1500000, 921600, 691200, 460800, 230400, 115200, 9600

[CONTROLLER_CSP2008](#): 115200, 691200

[SENSOR_ILR110x_115x](#): 57600, 38400, 19200, 9600, 4800

[SENSOR_ILR118x](#): 38400, 19200, 9600, 4800, 2400

[SENSOR_ILR1191](#): 460800, 230400, 115200, 57600, 38400, 19200, 9600

[SENSOR_ACS7000](#): 3500000, 2000000, 1500000, 921600, 691200, 460800, 230400, 115200, 9600

[CONTROLLER_CBOX](#): 8000000, 4000000, 3500000, 3000000, 2500000, 2000000, 1500000, 921600, 691200, 460800, 230400, 115200, 9600

Unit: Baud

Default:

[SENSOR_ILR118x](#): 9600,

[SENSOR_ILD1401](#), 38400,

[SENSOR_ILD1302](#), [SENSOR_ILD1402](#), [SENSOR_ILD1700](#), [SENSOR_IFD2401](#),

SENSOR_IFD2431, SENSOR_IFD2445, SENSOR_IFD2451, SENSOR_IFD2461,
 SENSOR_IFD2471, SENSOR_ODC2520, SENSOR_ILR1191, SENSOR_ACS7000,
 CONTROLLER_CSP2008, CONTROLLER_CBOX: 115200,
 SENSOR_ILD2200, SENSOR_ILD2300, SENSOR_ODC2500, SENSOR_ODC2600:
 691200,
 SENSOR_ILR110x_115x: 57600

Description: Baudrate of the RS422 serial connection.

Parameter: int IP_Parity

IP_Parity

Direction: Down

Valid values:

SENSOR_ILR110x_115x: 2= EVENPARITY
 SENSOR_ODC2500, SENSOR_ODC2600: 0= NOPARITY, 2= EVEN-
 PARITY
 otherwise: 0= NOPARITY

Default: SENSOR_ILR110x_115x: 2, otherwise 0

Description: Parity of the RS422 serial connection.

Parameter: int IP_SyncMasterChannel

IP_SyncMasterChannel

Direction: Down

Valid values:

-1= No synchronisation master
 0= Sensor channel 1 (Connector 1/2)
 1= Sensor channel 2 (Connector 1/2)
 2= Sensor channel 3 (Connector 3/4)
 3= Sensor channel 4 (Connector 3/4)
 4= Digital IN

Default: -1

Description: Channel number to synchronise with.

Synchronisation is done at driver layer.

If synchronisation is active, a whole data frame for master channel and each slave channel is buffered before all frames are transferred to MEDAQLib at once.

If at a slave channel arrives two whole frames while at master channel no whole frame is available, the older frame at the slave channel is deleted.

If at master channel arrives two frames while at slave channel no whole frame is available the last complete frame at slave channel is duplicated.

But if no last complete frame is available (there was never data on slave channel before), the oldest frame at master channel and all other slaves is deleted.

This ensures that the maximum time difference between a value at master and slave channel is one value at master channel, even is the datarates of the sensors are completely different.

To ensure that synchronised values has no offset (e.g. at start or after a sensor command) it is recommended to call ClearBuffer with SP_AllDevices set to true after opening all sensors and after each sensor command.

Parameter: int IA_Baudrate

IA_Baudrate

Direction: Up

Unit: Baud

Description: Baudrate cannot be set to any value. So it is set to the next matching baudrate the USB adapter IF2004 supports. This real baudrate is returned in this parameter.

7.2.5 IF2008

Following sensors supports this interface:

[SENSOR_ILD1401](#) (for sensor ILD1402 in compatibility mode).
[SENSOR_ILD1302](#), [SENSOR_ILD1320](#), [SENSOR_ILD1402](#), [SENSOR_ILD1420](#), [SENSOR_ILD1700](#), [SENSOR_ILD2200](#), [SENSOR_ILD2300](#), [SENSOR_ODC2500](#), [SENSOR_ODC2520](#), [SENSOR_ODC2600](#), [PCI_CARD_IF2008](#), [SENSOR_IFD2401](#), [SENSOR_IFD2431](#), [SENSOR_IFD2445](#), [SENSOR_IFD2451](#), [SENSOR_IFD2461](#), [SENSOR_IFD2471](#), [SENSOR_AC7000](#), [SENSOR_ILR110x_115x](#), [SENSOR_ILR118x](#), [SENSOR_ILR1191](#), [CONTROLLER_CSP2008](#), [CONTROLLER_CBOX](#) (native).

Parameter: int IP_CardInstance IP_CardInstance

Direction: Down

Valid values:

Minimum: 0

Maximum: 15

Default: 0

Description: Instance number of the IF2008 interface card. The cards are enumerated by the OS and the only way to distinguish is the card instance number. It does not change at least there are no changes at the PCI bus.

Parameter: int IP_ChannelNumber IP_ChannelNumber

Direction: Down

Valid values:

Condition: [PCI_CARD_IF2008](#)

-1= No data acquisition

6= Encoder 1

7= Encoder 2

8= Digital IN

9= Digital RxD

10= ADC 1 (Analog/Digital converter)

11= ADC 2 (Analog/Digital converter)

Valid values:

Condition: otherwise (sensors)

0= Sensor channel 1 (Base Board, Connector X1)

1= Sensor channel 2 (Base Board, Connector X1)

2= Sensor channel 3 (Base Board, Connector X2)

3= Sensor channel 4 (Base Board, Connector X2)

4= Sensor channel 5 (Extension Board, Connector X1)

5= Sensor channel 6 (Extension Board, Connector X1)

Description: Channel number on IF2008 Interface card.

Attention! Sensor channel 5 and 6 are only available if IF2008E extension card is installed. Digital IN is only available if IF2008E extension card or IF2008IO extension slot is installed. ADC is only available if IF2008E extension card is installed.

-1 means, no data channel is written to FIFO and cannot be read using [TransferData](#) or [Poll](#). This mode can be used if IF2008 should only be parametrized.

7 Parameters

Parameter: int IP_Baudrate IP_Baudrate

Direction: Down

Valid values:

SENSOR_ILD1401: 38400
 SENSOR_ILD1302, SENSOR_ILD1402: 115200, 57600, 38400, 19200,
 9600
 SENSOR_ILD1320, SENSOR_ILD1420: 1000000, 921600, 691200, 460800,
 256000, 230400, 128000, 115200, 56000, 19200, 9600
 SENSOR_ILD1700: 115200, 57600, 19200, 9600
 SENSOR_ILD2200: 691200, 1250000
 SENSOR_ILD2300: 3500000, 3000000, 2500000, 2000000, 1500000,
 921600, 691200, 460800, 230400, 115200, 9600
 SENSOR_IFD2401, SENSOR_IFD2431: 460800, 230400, 115200, 57600,
 38400, 19200, 9600
 SENSOR_IFD2445, SENSOR_IFD2451, SENSOR_IFD2461, SENSOR_IFD2471:
 4000000, 3500000, 2000000, 1500000, 921600, 691200, 460800,
 230400, 115200, 9600
 SENSOR_ODC2500, SENSOR_ODC2600: 115200, 38400, 19200, 9600
 SENSOR_ODC2520: 4000000, 3500000, 3000000, 2500000, 2000000,
 1500000, 921600, 691200, 460800, 230400, 115200, 9600
 CONTROLLER_CSP2008: 115200, 691200
 SENSOR_ILR110x_115x: 57600, 38400, 19200, 9600, 4800
 SENSOR_ILR118x: 38400, 19200, 9600, 4800, 2400
 SENSOR_ILR1191: 460800, 230400, 115200, 57600, 38400, 19200,
 9600
 SENSOR_ACS7000: 3500000, 2000000, 1500000, 921600, 691200,
 460800, 230400, 115200, 9600
 CONTROLLER_CBOX: 8000000, 4000000, 3500000, 3000000, 2500000,
 2000000, 1500000, 921600, 691200, 460800, 230400, 115200, 9600

Unit: Baud

Default:

SENSOR_ILR118x: 9600,
 SENSOR_ILD1401, 38400,
 SENSOR_ILD1302, SENSOR_ILD1402, SENSOR_ILD1700, SENSOR_IFD2401,
 SENSOR_IFD2431, SENSOR_IFD2445, SENSOR_IFD2451, SENSOR_IFD2461,
 SENSOR_IFD2471, SENSOR_ODC2520, SENSOR_ILR1191, SENSOR_ACS7000,
 CONTROLLER_CSP2008, CONTROLLER_CBOX: 115200,
 SENSOR_ILD2200, SENSOR_ILD2300, SENSOR_ODC2500, SENSOR_ODC2600:
 691200,
 SENSOR_ILR110x_115x: 38400

Description: Speed of the RS422 serial connection.

Parameter: int IP_Parity IP_Parity

Direction: Down

Valid values:

SENSOR_ILR110x_115x: 2= EVENPARITY
 SENSOR_ODC2500, SENSOR_ODC2600: 0= NOPARITY, 2= EVEN-
 PARITY
 otherwise: 0= NOPARITY

Default: SENSOR_ILR110x_115x: 2, otherwise 0

Description: Parity of the RS422 serial connection.

Parameter: int IP_SyncMasterChannel

IP_SyncMasterChannel

Direction: Down

Valid values:

- 1= No synchronisation master
- 0= Sensor channel 1 (Base Board, Connector X1)
- 1= Sensor channel 2 (Base Board, Connector X1)
- 2= Sensor channel 3 (Base Board, Connector X2)
- 3= Sensor channel 4 (Base Board, Connector X2)
- 4= Sensor channel 5 (Extension Board, Connector X1)
- 5= Sensor channel 6 (Extension Board, Connector X1)
- 6= Encoder 1
- 7= Encoder 2
- 8= Digital IN
- 9= RxD
- 10= ADC 1
- 11= ADC 2

Default: -1

Description: Channel number to synchronise with.

Synchronisation is done at driver layer and is only supported for IF2008 cards with FPGA version 4 or higher.

Only sensor channels can be synchronized. To synchronize IF2008 internal channels (Encoder's, Digital IN, RxD or ADC's), set the latch source for this channel to the desired master.

If synchronisation is active, a whole data frame for master channel and each slave channel is buffered before all frames are transferred to MEDAQLib at once.

If at a slave channel arrives two whole frames while at master channel no whole frame is available, the older frame at the slave channel is deleted.

If at master channel arrives two frames while at slave channel no whole frame is available the last complete frame at slave channel is duplicated.

But if no last complete frame is available (there was never data on slave channel before), the oldest frame at master channel and all other slaves is deleted.

This ensures that the maximum time difference between a value at master and slave channel is one value at master channel, even is the datarates of the sensors are completely different.

To ensure that synchronised values has no offset (e.g. at start or after a sensor command) it is recommended to call ClearBuffer with SP_AllDevices set to true after opening all sensors and after each sensor command.

Parameter: int IA_Baudrate

IA_Baudrate

Direction: Up

Unit: Baud

Description: Baudrate cannot be set to any value. So it is set to the next matching baudrate the IF2008 card supports. This real baudrate is returned in this parameter.

7.2.6 TCP/IP

Following sensors supports this interface:

SENSOR_ILD1401, SENSOR_ILD1302, SENSOR_ILD1320, SENSOR_ILD1402, SENSOR_ILD1420, SENSOR_ILD1700, SENSOR_ILD2200, SENSOR_ODC1202, SENSOR_ODC2500, SENSOR_ODC2600, SENSOR_IFD2401, SENSOR_IFD2431, SENSOR_ILR110x_115x, SENSOR_ILR118x, SENSOR_ILR1191 (additional, e.g. RS232/RS422

to TCP/IP comm server and RS232 high level interface).

SENSOR_ILD2300, SENSOR_IFD2445, SENSOR_IFD2451, SENSOR_IFD2461, SENSOR_IFD2471, SENSOR_ODC2520, SENSOR_DT3100, SENSOR_DT6100, SENSOR_ACS7000, CONTROLLER_KSS6380, CONTROLLER_DT6200, CONTROLLER_DT6500, CONTROLLER_CSP2008, CONTROLLER_CBOX, ETH_IF1032 (native).

Parameter: String IP_RemoteAddr IP_RemoteAddr

Direction: Down

Valid value: IP address

Description: IP address of the remote sensor (TCP server). It has to be set in any case because the default address is not valid!

Parameter: int IP_RemotePort IP_RemotePort

Direction: Down

Valid values:

Minimum: 1

Maximum: 65535

Default: **SENSOR_ILD2300, SENSOR_IFD2445, SENSOR_IFD2451, SENSOR_IFD2461, SENSOR_IFD2471, SENSOR_ODC2520, SENSOR_ACS7000, CONTROLLER_CSP2008, CONTROLLER_CBOX:** 23 (Telnet), otherwise 10001

Description: TCP port of the remote sensor (server). For **SENSOR_ILD2300, SENSOR_IFD2445, SENSOR_IFD2451, SENSOR_IFD2461, SENSOR_IFD2471, SENSOR_ODC2520, SENSOR_ACS7000, CONTROLLER_CSP2008** and **CONTROLLER_CBOX** this parameter is used for command port. The command port 23 (Telnet) is fix. Both interfaces are generated by the driver.

Parameter: int IP_RemoteDataProtocol IP_RemoteDataProtocol

Direction: Down

Valid for sensor:

SENSOR_ILD2300

SENSOR_IFD2445

SENSOR_IFD2451

SENSOR_IFD2461

SENSOR_IFD2471

SENSOR_ODC2520

CONTROLLER_CSP2008

CONTROLLER_CBOX

SENSOR_ACS7000

Valid values:

 -1 = Query automatic (not for **CONTROLLER_CSP2008**)

 0 = TCP server

 1 = TCP client (not for **CONTROLLER_CBOX**)

 2 = UDP sender (not for **CONTROLLER_CBOX**)

Default: **CONTROLLER_CSP2008:** 0, otherwise -1

Description: IP Protocol used for data transfer. Local host automatically takes opposite.

Parameter: int IP_PortDetection IP_PortDetection

Direction: Down

Valid for sensor:

CONTROLLER_DT6200

CONTROLLER_DT6500

ETH_IF1032

Valid values:

- 0= Set port(s) manually ([CONTROLLER_DT6200](#), [CONTROLLER_DT6500](#) + [ETH_IF1032](#))
- 1= Query all ports automatically (only [CONTROLLER_DT6500](#))
- 2= Query data port automatically ([CONTROLLER_DT6200](#) + [ETH_IF1032](#))

Default: [CONTROLLER_DT6200](#) + [ETH_IF1032](#): 2, [CONTROLLER_DT6500](#):
1

Description: [CONTROLLER_DT6500](#) exists in two variations. The first one has one port for commands and data (always 10001), the new one has a command port (23, Telnet) and a data port (default 10001). If ports should be set manually, for first version both ports ([IP_RemotePort](#) and [IP_DataPort](#)) must be set to the same value (port at sensor). For the new version, [IP_RemotePort](#) must be set to port 23 and [IP_DataPort](#) must be set to data port at sensor. If ports should be queried automatically, [IP_RemotePort](#) and [IP_DataPort](#) is ignored. In this case, MEDAQLib first tries to open port 23. If successful, data port is queried and opened. If not, port 10001 used as command and data port.

At [CONTROLLER_DT6200](#) and [ETH_IF1032](#), command port is always 23 (Telnet) and data port can be set manually or automatically queried.

Parameter: int IP_DataPort

IP_DataPort

Direction: Down

Valid for sensor:

- [SENSOR_ILD2300](#)
- [SENSOR_IFD2445](#)
- [SENSOR_IFD2451](#)
- [SENSOR_IFD2461](#)
- [SENSOR_IFD2471](#)
- [SENSOR_ODC2520](#)
- [CONTROLLER_DT6200](#)
- [CONTROLLER_DT6500](#)
- [CONTROLLER_CSP2008](#)
- [CONTROLLER_CBOX](#)
- [ETH_IF1032](#)
- [SENSOR_ACS7000](#)

Valid values:

Minimum: 0

Maximum: [CONTROLLER_CSP2008](#): 32384, otherwise 65535

Default: [CONTROLLER_DT6500](#): 10001, otherwise 1024

Description: If remote data protocol is TCP server this is the remote data port, otherwise it is the local data port. If remote data protocol is set to query automatic, this parameter is ignored. If this parameter is 0, the data port is not opened.

7.2.7 WinUSB

Following sensors supports this interface:

[SENSOR_IFD2401](#), [SENSOR_IFD2431](#) and [CONTROLLER_CBOX](#) (native).

Parameter: int IP_DeviceInstance

IP_DeviceInstance

Direction: Down

Valid values:

Minimum: 0

Maximum: 255

Default: 0

Description: Instance number of the USB device. The devices are enumerated by the OS and the only way to distinguish is the device instance number. It does not change at least there are no changes at the USB bus (plug/unplug devices).

Parameter: int IP_UsbReadBufCnt

IP_UsbReadBufCnt

Direction: Down

Valid values:

Minimum: 2

Maximum: 32

Default: 8

Description: Number of buffers for read operations on USB.

Parameter: int IP_UsbReadBufSize

IP_UsbReadBufSize

Direction: Down

Valid values:

Minimum: 2

Maximum: 1048576

Unit: Bytes

Default: otherwise 512

Description: Buffer size for read operations on USB. The value is always ceiled to the next even number (2, 4, 6, ..., 1048574, 1048576).

Parameter: int CP_ThreadPriority

CP_ThreadPriority

Direction: Down

Valid values:

-15= THREAD_PRIORITY_IDLE

-2= THREAD_PRIORITY_LOWEST

-1= THREAD_PRIORITY_BELOW_NORMAL

0= THREAD_PRIORITY_NORMAL

1= THREAD_PRIORITY_ABOVE_NORMAL

2= THREAD_PRIORITY_HIGHEST

15= THREAD_PRIORITY_TIME_CRITICAL

Default: THREAD_PRIORITY_TIME_CRITICAL

Description: Priority of the USB read thread. It should be as high as possible to avoid data loss.

7.3 Sensor parameters

The avialable sensor commands and it's parameters are described at the specific sensor sections.

8 Sensor commands

8.1 Communication via SensorCommand

The function [SensorCommand](#) always must have one obligatory parameter:

Parameter: String S_Command S_Command
Direction: Down
Valid for sensor: all
Valid value: See commands for sensors.
Description: The command to execute is specified by this parameter. Some commands are for the driver, the most commands are processed by the sensor.

The following parameters are used for each call to [SensorCommand](#) so they are described only once:

Parameter: int CP_SensorAnswerTimeout CP_SensorAnswerTimeout
Direction: Down
Valid for sensor: all
Valid values:
Minimum: 0
Maximum: 2147483647 (INT_MAX)
Unit: ms
Default: Depending on sensor and command.
Description: This special timeout is added to the global timeout (specified by [CP_BaseLevelTimeout](#)) when waiting for the complete answer from sensor.

Parameter: String SA_CompleteAnswer SA_CompleteAnswer
Direction: Up
Valid for sensor: all
Description: The raw (not interpreted) answer (to a command) from the sensor is stored here.

Parameter: int SA_ErrorNumber SA_ErrorNumber
Direction: Up
Valid for sensor:
 SENSOR_ILD1302
 SENSOR_ILD1320
 SENSOR_ILD1401
 SENSOR_ILD1402
 SENSOR_ILD1420
 SENSOR_ILD1700
 SENSOR_ILD2200
 SENSOR_ILD2300
 SENSOR_IFD2445
 SENSOR_IFD2451
 SENSOR_IFD2461
 SENSOR_IFD2471
 SENSOR_ODC2500
 SENSOR_ODC2520
 SENSOR_ODC2600

SENSOR_ACS7000
 CONTROLLER_CSP2008
 CONTROLLER_CBOX

Valid values:

Minimum: 0
 Maximum: 255

Description: Error number returned by any sensor if a command was not successful.

Parameter: String SA_ErrorText

SA_ErrorText

Direction: Up

Valid for sensor:

SENSOR_ILR118x
 SENSOR_ILR1191
 SENSOR_ILD1302
 SENSOR_ILD1320
 SENSOR_ILD1402
 SENSOR_ILD1420
 SENSOR_ILD1700
 SENSOR_ILD2200
 SENSOR_ILD2300
 SENSOR_IFD2401
 SENSOR_IFD2431
 SENSOR_IFD2445
 SENSOR_IFD2451
 SENSOR_IFD2461
 SENSOR_IFD2471
 SENSOR_ODC2500
 SENSOR_ODC2520
 SENSOR_ODC2600
 SENSOR_DT3100
 SENSOR_DT6100
 SENSOR_DT6120
 CONTROLLER_DT6200
 CONTROLLER_DT6500
 CONTROLLER_CSP2008
 CONTROLLER_CBOX
 SENSOR_ACS7000
 ETH_IF1032

Description: Clear text for specific error number.

Following parameters affects the driver and interface (when communicating with the sensor):

Parameter: int IP_ClearRingBuffer

IP_ClearRingBuffer

Direction: Down

Valid Interface: all

Valid for sensor: all

Valid values:

0= FALSE
 1= TRUE

Default: 1

Description: Clears the ring buffer before sending the command to the sensor.

The containing data is discarded. So the next data in the ring buffer is the sensor answer. For **IF2004** card, the ring buffer is cleared after reading the answer too.

Parameter: int IP_ClearSendBuffer	IP_ClearSendBuffer
Direction: Down	
Valid Interface: all	
Valid for sensor: all	
Valid values:	
0= FALSE	
1= TRUE	
Default: 1	
Description: Clears the send buffer (PurgeComm (PURGE_TXCLEAR)) before sending the command to the sensor.	
Parameter: int IP_ClearReceiveBuffer	IP_ClearReceiveBuffer
Direction: Down	
Valid Interface:	
RS232	
IF2004	
IF2004_USB	
IF2008	
Valid for sensor: all	
Valid values:	
0= FALSE	
1= TRUE	
Default: 1	
Description: Clears the receive buffer (PurgeComm (PURGE_RXCLEAR) for RS232, reading the FIFO for IF2004, IF2004_USB and IF2008) before sending the command to the sensor.	

This example shows how to set measure speed of [SENSOR_ILD1700](#).

```
SetParameterString (instance, "S_Command", "Set_Speed");
SetParameterInt (instance, "SP_Speed", 1); /* 1 is 1.25 kHz */
err= SensorCommand (instance);
/* error handling, if err!=ERR_NOERROR */
```

Alternativly you can use the wrapper function [SetIntExecSCmd](#).

```
err= SetIntExecSCmd (instance, "Set_Speed", "SP_Speed", 1);
/* error handling, if err!=ERR_NOERROR */
```

The next example shows how to get information from [SENSOR_IFD2401](#).

```
int sensor;
SetParameterString (instance, "S_Command", "Get_Status");
err= SensorCommand (instance);
/* error handling, if err!=ERR_NOERROR */
GetParameterInt (instance, "SA_Sensor", &sensor);
```

Or you can use the wrapper function [ExecSCmdGetInt](#).

```
int sensor;
err= ExecSCmdGetInt (instance, "Get_Status", "SA_Sensor", &sensor);
/* error handling, if err!=ERR_NOERROR */
```

8.2 Sensor commands valid for each sensor

Commands are stored in Parameter [S_Command](#). Following chapters describe commands and the parameters for each command.

8.2.1 Enable_Logging

This command can control logging to file for debugging purposes at any time.

Parameter: int IP_EnableLogging IP_EnableLogging

Direction: Down

Valid values:

0= FALSE

1= TRUE

Default: 0

Description: This parameter enables or disables logging to file for debugging purposes.

Parameter: int IP_LogType IP_LogType

Direction: Down

Valid values:

A bit combination of following values:

1= HIGH_TYPE (User <--> MEDAQLib)

2= MIDDLE_TYPE (Sensor layer <--> Interface layer)

4= LOW_TYPE (MEDAQLib <--> Hardware driver)

8= ERROR_TYPE (Any errors reported by MEDAQLib)

16= DRIVER_TYPE (Hardware driver <--> Sys driver)

32= APPL_TYPE (Application specific, see [LogFile](#) and [LogFileU](#))

Default: 2147483647 (INT_MAX)

Description: This parameter specifies the type of messages to log.

Parameter: int IPLogLevel IPLogLevel

Direction: Down

Valid values:

A bit combination of following values:

1= EMERGENCY_LEVEL (logging emerging errors)

2= CRITICAL_LEVEL (logging critical errors)

4= ERROR_LEVEL (logging errors which occurs)

8= WARNING_LEVEL (logging warnings from MEDAQLib)

16= NOTICE_LEVEL (logging notices)

32= TRACE_LEVEL (logging function calls)

64= DATA_LEVEL (logging data in binary mode)

Default: 127

Description: This parameter specifies the kind of event to log.

Parameter: String IPLogFile IPLogFile

Direction: Down

Default:

Description: File name of log file.

If it is empty or ends with '\' or '/', an automatic generated name ('TCLLogFile_%yyyy-%MM-%dd_%hh-%mm-%ss.txt') is appended.

Many placeholders (...) can be used for automatic name generation.

Important ones are:

%h= hour in 24 hours format (0, 1, ..., 9, 10, ..., 23)

%hh= hour in 24 hours format (00, 01, ..., 09, 10, ..., 23)

%H= hour in 12 hours format (1, 2, ..., 9, 10, 11, 12)

%HH= hour in 12 hours format (01, 02, ..., 09, 10, 11, 12)

%m= minute (0, 1, ..., 9, 10, ..., 59)

%mm= minute (00, 01, ..., 09, 10, ..., 59)

```
%s= second (0, 1, ..., 9, 10, ..., 59)
%ss= second (00, 01, ..., 09, 10, ..., 59)
%Ms= millisecond (000, 001, ..., 999)
%Us= microsecond (000000, 000001, ..., 999999)
%PP= output for morning, afternoon for US-American time format (AM,
PM)
%d= day (1, 2, ..., 9, 10, ..., 31)
%dd= day (01, 02, ..., 09, 10, ..., 31)
%M= month (1, 2, ..., 9, 10, 11, 12)
%MM= month (01, 02, ..., 09, 10, 11, 12)
%yy= year (70, 71, ..., 99, 00, 01, ..., 38)
%yyyy= year (1970, 1971, ..., 1999, 2000, 2001, ..., 2038)
%DoW= day of week (0= Sunday, 1= Monday, ..., 6= Saturday)
%DoY= day of year (0, 1, ..., 9, 10, ..., 364, if a leap year 365)
%WoY= week of year (0, 1, ..., 9, 10, ..., 52), starting with 0= the week
with the first Sunday in year (US format)
```

Parameter: int IP_LogAppend

IP_LogAppend

Direction: Down

Valid values:

0= FALSE
1= TRUE

Default: 1

Description: This parameter specifies if the logfile should be cleared at opening or if the new data should be appended to file. It has only an effect if a new log file is opened.

Parameter: int IP_LogFlush

IP_LogFlush

Direction: Down

Valid values:

0= FALSE
1= TRUE

Default: 0

Description: This parameter specifies if the logfile should be flushed after each output. In this case, it is sure that all information is stored to logfile before proceeding. But depending on the storage device it can slow down the MEDAQLib.

Parameter: int IP_LogSplitSize

IP_LogSplitSize

Direction: [IN]

Valid values:

Minimum: 0
Maximum: 2147483647 (INT_MAX)

Unit: KB (1024 Bytes)

Default: 0

Description: If this parameter is greater than 0, logfile is closed and reopened when this size is reached. If the file name contains placeholders (%...), a new name is generated before opening. Otherwise, the same file is opened again and if appending is off, the old content is overwritten.

8.2.2 Automatic_Setup

This command has two purposes.

On the one hand it setups MEDAQlib by retrieving required information from sensor.
On the other hand, it setups the sensor so it sends measurement data back to MEDAQLib.

Parameter: int CP_AutomaticMode

CP_AutomaticMode

Direction: Down

Valid values:

A bit combination of following values:

"First bit (1)= Retrieve sensor information to setup MEDAQLib"

"Second bit (2)= Activate data output at sensor"

Default: 1

Description: First bit (1) allows MEDAQLib to retrieve information from sensor automatically if needed.

Second bit (2) allows MEDAQLib to change sensor interface parameters (if needed) so is outputs data.

8.2.3 Get_TransmittedDataInfo

Many sensors can transmit more than one value at each measurement (e.g. Distance + Intensity). All transmitted values are enclosed in a data frame. [TransferData](#) (and also [Poll](#)) returns data in an array, arranged frame by frame.

This function can be called to determine, how many and which kind of values are in a data frame.

Attention! The transmited data can change if sensor configuration has changed, e.g. by sending a sensor command.

Parameter: int IA_MaxValuesPerFrame

IA_MaxValuesPerFrame

Direction: Up

Valid for sensor: all

Description: Maximum number of values which can be transferred from sensor for one measurement.

Parameter: int IA_ValuesPerFrame

IA_ValuesPerFrame

Direction: Up

Valid for sensor: all

Valid values:

Minimum: 0

Maximum: IA_MaxValuesPerFrame

Description: If MEDAQLib does not know, which data is transferred from sensor, this value can be 0. All following parameters are postfixed by a number from 1 to this value.

Parameter: int IA_Index1..x

IA_Index1..x

Direction: Up

Valid for sensor: all

Valid values:

Minimum: 0

Maximum: as many values the sensor can transfer in one frame

Description: This value is for identification. Each value in a frame has unique index, which does never change, even if sensor configuration is changed.

Parameter: String IA_Raw_Name1..x	IA_Raw_Name1..x
Direction: Up	
Valid for sensor: all	
Description: The name of the value in the raw data.	
Parameter: String IA_Scaled_Name1..x	IA_Scaled_Name1..x
Direction: Up	
Valid for sensor: all	
Description: The name of the value in the scaled data.	
Parameter: double IA_Raw_RangeMin1..x	IA_Raw_RangeMin1..x
Direction: Up	
Valid for sensor: all	
Description: The minimum range of the value in the raw data.	
Parameter: double IA_Raw_RangeMax1..x	IA_Raw_RangeMax1..x
Direction: Up	
Valid for sensor: all	
Description: The maximum range of the value in the raw data.	
Parameter: double IA_Scaled_RangeMin1..x	IA_Scaled_RangeMin1..x
Direction: Up	
Valid for sensor: all	
Description: The minimum range of the value in the scaled data.	
Parameter: double IA_Scaled_RangeMax1..x	IA_Scaled_RangeMax1..x
Direction: Up	
Valid for sensor: all	
Description: The maximum range of the value in the scaled data.	
Parameter: String IA_Raw_Unit1..x	IA_Raw_Unit1..x
Direction: Up	
Valid for sensor: all	
Description: The unit of the value in the raw data, e.g. mm.	
Parameter: String IA_Scaled_Unit1..x	IA_Scaled_Unit1..x
Direction: Up	
Valid for sensor: all	
Description: The unit of the value in the scaled data, e.g. mm.	
Parameter: double IA_Raw_Datarate1..x	IA_Raw_Datarate1..x
Direction: Up	
Valid for sensor: all	
Description: The datarate of the value in the raw data. Datarates of all values are always the same.	
Parameter: double IA_Scaled_Datarate1..x	IA_Scaled_Datarate1..x
Direction: Up	
Valid for sensor: all	
Description: The datarate of the value in the scaled data. Datarates of all values are always the same.	

8.2.4 Use_Defaults

This command tells the driver to use default values to operate with sensor data. If some parameters are not specified they are not changed. The sensor is not affected by this command.

The following parameters are interface parameters and already described at chapter [Interface parameters](#):

[IP_AutomaticMode](#), [IP_ScaleErrorValues](#), [IP_FixedErrorValue](#), [IP_RingBufferSize](#) and [CP_BaseLevelTimeout](#). They are valid for each sensor and interface and can be changed using this command (not only when opening the driver).

Parameters only valid for a specific sensor are not described here but at command [Use_Defaults](#) of specific sensor chapter.

8.2.5 SettingsChanged

Checks if sensor or driver settings have changed since last call of [Get_DrvSetting](#).

Parameter: int IP_ResetFlag

IP_ResetFlag

Direction: Down

Valid values:

0= FALSE
1= TRUE

Description: If TRUE, the flag is resetted. Otherwise the flag is resetted at [Get_DrvSetting](#).

Parameter: int IA_SettingsChanged

IA_SettingsChanged

Direction: Up

Valid values:

0= FALSE
1= TRUE

Description: TRUE when settings have changed.

8.2.6 Get_DrvSetting

Returns the current settings of the driver used for operating with sensor data. It is the opposite of [Use_Defaults](#). The sensor is not affected by this command.

The following parameters returned, are already described at chapter [Interface parameters](#) (but instead of IP_... for interface parameter is is called now IA_... for interface answer):

[IA_AutomaticMode](#), [IA_ScaleErrorValues](#), [IA_FixedErrorValue](#), [IA_RingBufferSize](#) and [CA_BaseLevelTimeout](#).

They are returned for each sensor and interface.

Parameters only valid for a specific sensor are not described here but at command [Get_DrvSetting](#) of specific sensor chapter.

8.2.7 Cmd_Generic

With Cmd_Generic, any data can be sent to the sensor.

Parameter: String SP_CmdStr

SP_CmdStr

Direction: Down

Description: The command string as it is sent to the sensor. E.g. for ILD sensors it always starts with "+++\\rILD1" ('\r' is carriage return, 0x0d).

Parameter: int CP_SensorAnswerTimeout	CP_SensorAnswerTimeout
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 2147483647 (INT_MAX)	
Unit: ms	
Default: 500	
Description: This special timeout is added to the global timeout (specified by CP_BaseLevelTimeout) when waiting for the complete answer from sensor.	
Parameter: int CP_InitAfterSensorCommand	CP_InitAfterSensorCommand
Direction: Down	
Valid values:	
0= No	
1= Yes	
Default: 0	
Description: This parameter tells MEDAQLib if it should update internal states after processing generic sensor commands. Some sensor commands (as if IP_AutomaticMode is 1) are called in this case.	
Parameter: String SA_CompleteAnswer	SA_CompleteAnswer
Direction: Up	
Description: The raw (not interpreted) answer (to a command) from the sensor is stored here.	

8.2.8 Clear_Buffers

This command is executed by the driver and does not affect the sensor. It erases the ring buffer and the input and output buffer of the interface.

Parameter: int SP_AllDevices	SP_AllDevices
Direction: Down	
Valid values:	
0= FALSE	
1= TRUE	
Default: 0	
Description: If it is set to 1, not only this instance but all instances created from this driver are cleared. It can be used to synchronize data acquisition from several sensors because after clearing all buffers the next data from all sensors have same timestamp.	

8.2.9 DataAvail_Event

This command registers an event which will be set when new data is available. The event can be used to wait for new data in your application.

Parameter: int IP_EventOnAvailableValues	IP_EventOnAvailableValues
Direction: Down	
Valid values:	
Minimum: -1	
Maximum: 2147483647 (INT_MAX)	
Default: -1	
Description: If it is set to -1, the event is not set by this condition. Otherwise the event is set if at least so many values are available as specified here.	

Parameter: double IP_EventOnBufferFillsize IP_EventOnBufferFillsize

Direction: Down

Valid values:

Minimum: 0.0

Maximum: 1.0

Default: 1.0

Description: If it is set to 1.0, the event is not set by this condition.

Otherwise the event is set if the ring buffer fill size is at least as high as specified here (0.0 means empty, 1.0 means full).

For setting the ring buffer size please see parameter [IP_RingBufferSize](#).

Parameter: DWORD_PTR IA_DataAvailEvent IA_DataAvailEvent

Direction: Up

Description: The event handle for data avail.

If IP_EventOnAvailableValues is -1 (default) and IP_EventOnBufferFillsize is 1.0 (default), the event is released and the value is NULL.

Example how to use the event:

```

/* At your main initialisation function: */
DWORD err;
err= SetIntExecSCmd (instance, "DataAvail_Event", "IP_EventOnAvailableValues", 1024);
/* error handling, if err!=ERR_NOERROR */
HANDLE event= NULL;
GetParameterDWORD_PTR (instance, "IA_DataAvailEvent", (DWORD_PTR *)&event);

/* At your working thread or function: */
/* Maybe the event was already set before, but we are not interested in old data,
so clear old data and reset event. */
DWORD err= ExecSCmd (instance, "Clear_Buffers");
/* error handling, if err!=ERR_NOERROR */
ResetEvent (event);

while (true)
{
  /* timeout is 1000 ms */
  if (WaitForSingleObject (event, 1000)==WAIT_OBJECT_0)
    err= TransferData (instance, raw, scaled, 1024, &read);
  /* error handling, verify that read==1024, process data */
}

```

8.2.10 Open_DataSocket

This command is only available for sensors with are connected via TCP/IP and different command and data port: [SENSOR_ILD2300](#), [SENSOR_IFD2445](#), [SENSOR_IFD2451](#), [SENSOR_IFD2461](#), [SENSOR_IFD2471](#), [SENSOR_ODC2520](#), [SENSOR_ACS7000](#), [CONTROLLER_CSP2008](#), [CONTROLLER_CBOX](#), [ETH_IF1032](#), [CONTROLLER_DT6500](#) (only DT6530) and [CONTROLLER_DT6200](#).

Parameter: int IP_RemoteDataProtocol

IP_RemoteDataProtocol

Direction: Down

Valid for sensor:

[SENSOR_ILD2300](#)

[SENSOR_IFD2445](#)

[SENSOR_IFD2451](#)

[SENSOR_IFD2461](#)

[SENSOR_IFD2471](#)

SENSOR_ODC2520
 CONTROLLER_CSP2008
 CONTROLLER_CBOX
 SENSOR_ACS7000

Valid values:

- 1= Query automatic (not for CONTROLLER_CSP2008)
- 0= TCP server
- 1= TCP client (not for CONTROLLER_CBOX)
- 2= UDP sender (not for CONTROLLER_CBOX)

Description: IP Protocol used for data transfer. Local host automatically takes opposite.

Parameter: int IP_PortDetection

IP_PortDetection

Direction: Down

Valid for sensor:

CONTROLLER_DT6200
 CONTROLLER_DT6500
 ETH_IF1032

Valid values:

- 0= Set port(s) manually (CONTROLLER_DT6200, CONTROLLER_DT6500 + ETH_IF1032)
- 1= Query all ports automatically (only CONTROLLER_DT6500)
- 2= Query data port automatically (CONTROLLER_DT6200 + ETH_IF1032)

Description: If ports should be queried automatically, IP_DataPort local setting is ignored and data port is queried automatically.

Parameter: int IP_DataPort

IP_DataPort

Direction: Down

Valid for sensor:

SENSOR_ILD2300
 SENSOR_IFD2445
 SENSOR_IFD2451
 SENSOR_IFD2461
 SENSOR_IFD2471
 SENSOR_ODC2520
 CONTROLLER_DT6200
 CONTROLLER_DT6500
 CONTROLLER_CSP2008
 CONTROLLER_CBOX
 ETH_IF1032
 SENSOR_ACS7000

Valid values:

- Minimum:** 0
Maximum: CONTROLLER_CSP2008: 32384, otherwise 65535

Description: If remote data protocol is TCP server this is the remote data port, otherwise it is the local data port. If remote data protocol is set to query automatic, this parameter is ignored. If this parameter is 0, the data port is not opened.

8.2.11 Set_Setup

This command import a parameter set to a sensor.

Parameter: String SP_SetupRecord SP_SetupRecord

Direction: Down

Description: Parameter set which should be imported to sensor. Format is same as at [SensorTest](#).

First line must contain a header. Version and sensor type will be checked.. Following lines will be processed one by one. Comments (starting with #) will be ignored.

8.2.12 Get_Setup

This command exports all sensor settings.

It is only available for sensors [SENSOR_ILD1320](#), [SENSOR_ILD1420](#), [SENSOR_ILD2300](#), [SENSOR_IFD2445](#), [SENSOR_IFD2451](#), [SENSOR_IFD2461](#), [SENSOR_IFD2471](#), [SENSOR_ODC2500](#), [SENSOR_ODC2520](#), [SENSOR_ODC2600](#), [SENSOR_DT3100](#), [SENSOR_DT6100](#), [CONTROLLER_DT6200](#), [CONTROLLER_KSS6380](#), [CONTROLLER_DT6500](#), [ETH_IF1032](#), [CONTROLLER_CBOX](#) and [SENSOR_ACS7000](#).

Parameter: int SP_IncludeSavedSetups SP_IncludeSavedSetups

Direction: Down

Valid for sensor:

- [SENSOR_ILD1320](#) - include all existing parameter sets
- [SENSOR_ILD1420](#) - include all existing parameter sets
- [SENSOR_ILD2300](#) - include all existing parameter sets
- [SENSOR_ODC2500](#) - include OptionData and UserMeasProgData
- [SENSOR_ODC2520](#) - include all existing parameter sets
- [SENSOR_ODC2600](#) - include OptionData and UserMeasProgData
- [SENSOR_IFD2445](#) - include all existing parameter sets
- [SENSOR_IFD2451](#) - include all existing parameter sets
- [SENSOR_IFD2461](#) - include all existing parameter sets
- [SENSOR_IFD2471](#) - include all existing parameter sets
- [SENSOR_DT3100](#) - include saved settings
- [SENSOR_DT6100](#) - include saved setup
- [CONTROLLER_KSS6380](#) - include saved setup
- [CONTROLLER_DT6500](#) - include saved setup (not DT6530)
- [CONTROLLER_CBOX](#) - include all parameter sets
- [SENSOR_ACS7000](#) - include all existing parameter sets

Valid values:

0 = No

1 = Yes

Default: 0

Description: Specifies if saved settings should be exported, too.

Parameter: int SP_IncludeAdditionalParameters SP_IncludeAdditionalParameters

Direction: Down

Valid for sensor:

- [SENSOR_ILD2300](#) - includes Material Table
- [SENSOR_IFD2445](#) - includes Material Table
- [SENSOR_IFD2451](#) - includes Material Table
- [SENSOR_IFD2461](#) - includes Material Table
- [SENSOR_IFD2471](#) - includes Material Table
- [CONTROLLER_ESC4912](#) - includes Linearisation Modes
- [CONTROLLER_DT6200](#) - include Measure Ranges, Linearisation Modes and Linearisation Points

CONTROLLER_DT6500 - include Measure Ranges (only DT6530), Linearisation Modes and Linearisation Points (only DT6530)

SENSOR_ACS7000 - includes Color Table

Valid values:

0= No

1= Yes

Default: 0

Description: Specifies if additional parameters should be exported, too.

Parameter: int SP_IncludeInterfaceParameters

SP_IncludeInterfaceParameters

Direction: Down

Valid for sensor:

SENSOR_ILD1320

SENSOR_ILD1420

SENSOR_ILD2300

SENSOR_IFD2445

SENSOR_IFD2451

SENSOR_IFD2461

SENSOR_IFD2471

SENSOR_ODC2500

SENSOR_ODC2520

SENSOR_ODC2600

CONTROLLER_DT6200

CONTROLLER_DT6500 (only DT6530)

ETH_IF1032

CONTROLLER_CBOX

SENSOR_ACS7000

Valid values:

0= No

1= Yes

Default: 0

Description: Specifies if hardware interface parameters should be exported, too.

Attention!

Interface parameters at sensor are changed immediately, so connection may be lost at **Set_Setup** (if parameters are changed for currently used interface). Connection may be lost at **Get_Setup** too (if **SP_IncludeSavedSetup** is 1 and interface parameters differs from current setup (RAM) and a saved setup).

Parameter: String SA_SetupRecord

SA_SetupRecord

Direction: Up

Description: Parameter set with exported settings. Format is same as at **SensorTest**.

First line contains a header (version, sensor type, date and time of export). Following lines contains all sensor commands and it's parameters which are needed to setup a sensor with same settings.

8.2.13 Update_... (Meta command)

This command is a meta command for any sensors with **Get_Setup** / **Set_Setup** command pairs.

It simplifies changing just a few parameters of a larger parameter set.

First, the command name is changed from Update_... to Get_... and this function is called (including the rest of the parameter set, too). The answer parameters from sensor (SA_...) are renamed to sensor parameters (SP_...) and the parameter set (given by the user) is copied over the working parameter set. The outcome of this is a new parameter set, where all parameters are current sensor settings except the parameters which are specified by the user. Then the command name is changed to Set_... again and the command is send to sensor.

This command has no obligatory parameters, only parameters which should be changed at sensor must be specified.

It is also possible, to call an Update_... command, where the matching Get_... command needs own parameters, too (e.g. channel number). Just specify this parameter before calling Update_... command.

Following example shows how to setup **SENSOR_ILD2300** to output additional counter:

```
SetParameterString (instance, "S_Command", "Update_OutputAdditional_ETH");
SetParameterInt (instance, "SP_OutputAdditionalCounter_ETH", 1);
err= SensorCommand (instance);
/* error handling, if err!=ERR_NOERROR */
```

Another command which updates a mathematic channel at DT6200::

```
SetParameterDouble (instance, "SP_FactorCh3", 1.19);
err= SetIntExecSCmd (instance, "Update_MathFunction", "SP_Chан", 2);
/* error handling, if err!=ERR_NOERROR */
```

8.3 SensorFinder commands

MEDAQLib supports commands to detect sensors at many different interfaces with any interface settings. Following table shows the sensors which can be detected and the possible interfaces.

SENSOR_ILR110x_115x	RS232 (IF2001_USB), IF2008, IF2004_USB
SENSOR_ILR118x (01) - RS232	RS232
SENSOR_ILR118x (02) - RS422	RS232 (IF2001_USB), IF2008, IF2004_USB
SENSOR_ILR1191 (01) - RS232	RS232
SENSOR_ILR1191 (02) - RS422	RS232 (IF2001_USB), IF2008, IF2004_USB
SENSOR_ODC1202	RS232
SENSORILD1302	RS232 (IF2001_USB), IF2004, IF2008, IF2004_USB
SENSORILD1320	RS232 (IF2001_USB), IF2004, IF2008, IF2004_USB
SENSORILD1401	RS232
SENSORILD1402	RS232, IF2004, IF2008, IF2004_USB
SENSORILD1402 (Comp. mode)	RS232 (IF2001_USB), IF2008, IF2004_USB
SENSORILD1420	RS232 (IF2001_USB), IF2004, IF2008, IF2004_USB
SENSORILD1700	RS232 (IF2001_USB), IF2004, IF2008, IF2004_USB
SENSORILD2200	RS232 (IF2001_USB), IF2004, IF2008, IF2004_USB
SENSORILD2300	RS232 (IF2001_USB), IF2004, TCP/IP (local subnet), IF2008, IF2004_USB
SENSORIFD2401	RS232, WinUSB, IF2008, IF2004_USB
SENSORIFD2431	RS232, WinUSB, IF2008, IF2004_USB
SENSORIFD2445	RS232 (IF2001_USB), IF2004, TCP/IP (local subnet), IF2008, IF2004_USB
SENSORIFD2451	RS232 (IF2001_USB), IF2004, TCP/IP (local subnet), IF2008, IF2004_USB
SENSORIFD2461	RS232 (IF2001_USB), IF2004, TCP/IP (local subnet), IF2008, IF2004_USB
SENSORIFD2471	RS232 (IF2001_USB), IF2004, TCP/IP (local subnet), IF2008, IF2004_USB
SENSORACS7000	RS232 (IF2001_USB), IF2004, TCP/IP (local subnet), IF2008, IF2004_USB

SENSOR_ODC2500	RS232 (native, IF2001_USB), IF2004, IF2008, IF2004_USB
SENSOR_ODC2520	RS232 (IF2001_USB), IF2004, TCP/IP (local subnet), IF2008, IF2004_USB
SENSOR_ODC2600	RS232 (native, IF2001_USB), IF2004, IF2008, IF2004_USB
SENSOR_DT3100	TCP/IP (local subnet)
SENSOR_DT6100	TCP/IP (local subnet)
SENSOR_DT6120	RS232 (RS485/USB converter)
CONTROLLER_KSS6380	TCP/IP (local subnet)
CONTROLLER_DT6200	TCP/IP (local subnet)
CONTROLLER_DT6500	TCP/IP (local subnet)
PCI_CARD_IF2004	IF2004 (channel 3)
USB_ADAPTER_IF2004	IF2004_USB (channel 5)
PCI_CARD_IF2008	IF2008 (channel 6 to 11)
SENSOR_LLT27xx	TCP/IP (local subnet)
SENSOR_ACS7000	RS232 (IF2001_USB), IF2004, TCP/IP (local subnet), IF2008, IF2004_USB
CONTROLLER_CSP2008	TCP/IP (local subnet), on RS422 the CSP does not support detection
CONTROLLER_CBOX	RS232 (IF2001_USB), IF2004, TCP/IP (local subnet), IF2008, IF2004_USB, WinUSB
ETH_IF1032	TCP/IP (local subnet)

Because not every sensor does support same interface parameters as the interface at computer, there may be combinations which won't work.

Scanning for sensors runs asynchronous (each physical interface, e.g. COM1, COM2, IF2008[Card1, Channel1], IF2008[Card1, Channel2], ... in an own thread). First, this will ensure the sensors are found as fast as possible, and second, your calling application is not blocked and can do anything else, e.g. showing an progress bar. To do a scan, first create a sensor instance. You can create it for any sensor, but it's beneficial to use the sensor you want to search for. Now you can use folling sensor commands (internally processed by MEDAQLib) to perform a scan.

8.3.1 Start_FindSensor

This command starts a new scan for a sensor.

Parameter: int IP_SensorType

IP_SensorType

Direction: Down

Valid values:

- SENSOR_ILR110x_115x (19)
- SENSOR_ILR118x (20)
- SENSOR_ILR1191 (21)
- SENSORILD1302 (24)
- SENSORILD1320 (41)
- SENSORILD1401 (1)

SENSOR_ILD1402 (23)
 SENSOR_ILD1420 (42)
 SENSOR_ILD1700 (2)
 SENSOR_ILD2200 (5)
 SENSOR_ILD2300 (29)
 SENSOR_IFD2401 (12)
 SENSOR_IFD2431 (13)
 SENSOR_IFD2445 (39)
 SENSOR_IFD2451 (30)
 SENSOR_IFD2461 (44)
 SENSOR_IFD2471 (26)
 SENSOR_ODC1202 (25)
 SENSOR_ODC2500 (8)
 SENSOR_ODC2520 (37)
 SENSOR_ODC2600 (9)
 SENSOR_DT3100 (28)
 SENSOR_DT6100 (16)
 SENSOR_DT6120 (40)
 CONTROLLER_DT6200 (33)
 CONTROLLER_DT6500 (15)
 CONTROLLER_KSS6380 (18)
 PCI_CARD_IF2004 (10)
 PCI_CARD_IF2008 (22)
 SENSOR_LLT27xx, scanCONTROL and gapCONTROL (31)
 SENSOR_ACS7000 (35)
 CONTROLLER_CSP2008 (32)
 CONTROLLER_CBOX (38)
 ETH_IF1032 (34)
 USB_ADAPTER_IF2004 (36)

Default: Automatic

Description: Type of sensor to scan for. If this parameter is not specified, the value specified by [CreateSensorInstance](#) it is automatically taken.

Parameter: int IP_FindSimilarSensors

IP_FindSimilarSensors

Direction: Down

Valid values:

0= FALSE
1= TRUE

Default: 0

Description: When this parameter is true, the sensor name is not checked at sensor answer. So any sensor using same protocol can be found. This is an internal parameter. It should not be used by the customer.

Parameter: int IP_ScanHWRS232

IP_ScanHWRS232

Direction: Down

Valid values:

0= FALSE
1= TRUE

Default: 1

Description: The sensor is scanned on serial interface [RS232](#) (also IF2001 - USB (RS422) or RS485/USB converter).

Parameter: int IP_ScanHWIF2004	IP_ScanHWIF2004
Direction: Down	
Valid values:	
0= FALSE	
1= TRUE	
Default: 1	
Description: The sensor is scanned on PCI card IF2004 .	
Parameter: int IP_ScanHWIF2004_USB	IP_ScanHWIF2004_USB
Direction: Down	
Valid values:	
0= FALSE	
1= TRUE	
Default: 1	
Description: The sensor is scanned on USB adapter IF2004_USB .	
Parameter: int IP_ScanHWTCPPIP	IP_ScanHWTCPPIP
Direction: Down	
Valid values:	
0= FALSE	
1= TRUE	
Default: 1	
Description: The sensor is scanned on network protocoll TCP/IP .	
Parameter: int IP_ScanHWWinUSB	IP_ScanHWWinUSB
Direction: Down	
Valid values:	
0= FALSE	
1= TRUE	
Default: 1	
Description: The sensor is scanned on generic USB driver WinUSB .	
Parameter: int IP_ScanHWIF2008	IP_ScanHWIF2008
Direction: Down	
Valid values:	
0= FALSE	
1= TRUE	
Default: 1	
Description: The sensor is scanned on PCI card IF2008 .	
Parameter: int IP_QuickScanRS485	IP_QuickScanRS485
Direction: Down	
Valid values:	
0= All possible addresses are scanned (secure, but slow).	
1= Just broadcast address is scanned (only works if only one sensor is plugged at RS485 bus).	
Default: 0	
Description: This parameter is only used for sensors which supports RS485 interfaces.	
At RS485, several sensors can be connected, but just one is allowed to send data at same time. Therefore the fast scan only works if only one sensor is connected, otherwise there would be a data collision.	
If you are not sure, how much sensors are plugged, do not set this parameter. In this case the secure scan is processed.	

8.3.2 Get_FindSensorProgress

Get the progress of an active scan.

Parameter: double IA_Progress	IA_Progress
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 100.0	
Unit: %	
Description: Progress of the scan.	
Parameter: int IA_Found	IA_Found
Direction: Up	
Description: Number of sensors found until now.	

8.3.3 Get_FoundSensor

Get information about a found sensor.

Parameter: int IP_Index	IP_Index
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: Number of sensors found -1	
Description: Index (0 based) of sensor to get information.	
Parameter: int IP_SensorType	IP_SensorType
Direction: Up	
Valid values:	
SENSOR_ILR110x_115x (19)	
SENSOR_ILR118x (20)	
SENSOR_ILR1191 (21)	
SENSOR_ILD1302 (24)	
SENSOR_ILD1320 (41)	
SENSOR_ILD1401 (1)	
SENSOR_ILD1402 (23)	
SENSOR_ILD1420 (42)	
SENSOR_ILD1700 (2)	
SENSOR_ILD2200 (5)	
SENSOR_ILD2300 (29)	
SENSOR_IFD2401 (12)	
SENSOR_IFD2431 (13)	
SENSOR_IFD2445 (39)	
SENSOR_IFD2451 (30)	
SENSOR_IFD2461 (44)	
SENSOR_IFD2471 (26)	
SENSOR_ODC1202 (25)	
SENSOR_ODC2500 (8)	
SENSOR_ODC2520 (37)	
SENSOR_ODC2600 (9)	
SENSOR_DT3100 (28)	
SENSOR_DT6100 (16)	

```

SENSOR_DT6120 (40)
CONTROLLER_DT6200 (33)
CONTROLLER_DT6500 (15)
CONTROLLER_KSS6380 (18)
PCI_CARD_IF2004 (10)
PCI_CARD_IF2008 (22)
SENSOR_LLT27xx, scanCONTROL and gapCONTROL (31)
SENSOR_ACS7000 (35)
CONTROLLER_CSP2008 (32)
CONTROLLER_CBOX (38)
ETH_IF1032 (34)
USB_ADAPTER_IF2004 (36)
0

```

Description: Type of found sensor. Is the same as searched for if sensor matches or 0 if sensor is similar to the sensor searched for.

Parameter: String IP_SensorTypeName

IP_SensorTypeName

Direction: Up

Valid values:

```

ILR110x_115x
ILR118x
ILR1191
ILD1302
ILD1320
ILD1401
ILD1402
ILD1420
ILD1700
ILD2200
ILD2300
IFD2401
IFD2431
IFD2445
IFD2451
IFD2461
IFD2471
ODC1202
ODC2500
ODC2520
ODC2600
DT3100
DT6100
DT6120
DT6200
DT6500
KSS6380
PCICardIF2004
PCICardIF2008
LLT27xx
CSP2008
ETH_IF1032
ACS7000
USBAdapterIF2004

```

C-Box
Unknown

Description: Type name of found sensor.

Parameter: int IP_InterfaceIdx IP_InterfaceIdx

Direction: Up

Valid values:

RS232 (1)
IF2004_USB (2)
TCP/IP (3)
IF2008 (6)
WinUSB (7)
IF2004_USB (8)

Description: Index of the interface, on which the sensor was detected.

Parameter: String IP_Interface IP_Interface

Direction: Up

Valid values:

RS232
IF2004
TCP/IP
IF2008
WinUSB
IF2004_USB

Description: Name of the interface, on which the sensor was detected.

Parameter: String IP_Port IP_Port

Direction: Up

Valid Interface:

RS232

Valid values:

"COM1"
"COM2"

...

Description: Name of the serial interface.".

Parameter: int IP_Baudrate IP_Baudrate

Direction: Up

Valid Interface:

RS232

Valid values:

Minimum: 0

Unit: Baud

Description: Baudrate of the serial connection.

Parameter: int IP_Stopbits IP_Stopbits

Direction: Up

Valid Interface:

RS232

Valid values:

0= ONESTOPBIT
1= ONE5STOPBITS
2= TWOSTOPBITS

Description: Number of stop bits of the serial connection.

Parameter: int IP_Parity IP_Parity

Direction: Up

Valid Interface:

RS232

Valid values:

0= NOPARITY
1= ODDPARITY
2= EVENPARITY
3= MARKPARITY
4= SPACEPARITY

Description: Parity of the serial connection.

Parameter: int IP_ByteSize IP_ByteSize

Direction: Up

Valid Interface:

RS232

Valid values:

7
8

Unit: Bit

Description: Number of data bits per byte of the serial connection.

Parameter: int IP_CardInstance IP_CardInstance

Direction: Up

Valid Interface:

IF2004

Valid values:

Minimum: 0
Maximum: 15

Description: Instance number of the IF2004 Interface card.

Parameter: int IP_ChannelNumber IP_ChannelNumber

Direction: Up

Valid Interface:

IF2004

Valid values:

Minimum: 0
Maximum: 3

Description: Channel number on IF2004 Interface card. If [PCI_CARD_IF2004](#) is searched, the channel number 3 is returned.

Parameter: int IP_Baudrate IP_Baudrate

Direction: Up

Valid Interface:

IF2004

Valid values:

[SENSOR_ILD1302](#), [SENSOR_ILD1402](#), [SENSOR_ILD1700](#), [SENSOR_ODC2500](#)
and [SENSOR_ODC2600](#): 115200
[SENSOR_ODC2500](#) and [SENSOR_ODC2600](#): 691200
[SENSOR_ILD2200](#): 691200 or 1250000
[SENSOR_ILD1320](#), [SENSOR_ILD1420](#), [SENSOR_ILD2300](#), [SENSOR_IFD2445](#),
[SENSOR_IFD2451](#), [SENSOR_IFD2461](#), [SENSOR_IFD2471](#), [SENSOR_ODC2520](#),
[SENSOR_ACS7000](#), [CONTROLLER_CSP2008](#), [CONTROLLER_CBOX](#):
115200 or 691200

Unit: Baud

Description: Speed of the RS422 serial connection.

Parameter: int IP_DeviceInstance	IP_DeviceInstance
Direction: Up	
Valid Interface:	
IF2004_USB	
Valid values:	
Minimum: 0	
Maximum: 255	
Description: Instance number of the USB device.	
Parameter: String IP_SerialNumber	IP_SerialNumber
Direction: Up	
Valid Interface:	
IF2004_USB	
Valid values:	
Minimum: "0000001"	
Maximum: "9999999"	
Description: Serial number of USB adapter IF2004.	
Parameter: String IP_Port	IP_Port
Direction: Up	
Valid Interface:	
IF2004_USB	
Valid values:	
"COM1"	
"COM2"	
...	
Description: Name of the serial interface.".	
Parameter: int IP_ChannelNumber	IP_ChannelNumber
Direction: Up	
Valid Interface:	
IF2004_USB	
Valid values:	
Minimum: 0	
Maximum: 11	
Description: Channel number on USB adapter IF2004. If an sensor is is searched, the channel number is from 0 to 3, if USB_ADAPTER_IF2004 is searched, channel 4 is returned.	
Parameter: int IP_Baudrate	IP_Baudrate
Direction: Up	
Valid Interface:	
IF2004_USB	
Valid values:	
Minimum: 0	
Unit: Baud	
Description: Baudrate of the RS422 serial connection.	
Parameter: int IP_Parity	IP_Parity
Direction: Up	
Valid Interface:	
IF2004_USB	
Valid values:	
0=NOPARITY	
2=EVENPARITY	
Description: Parity of the RS422 serial connection.	

Parameter: String SA_UUID	SA_UUID
Direction: Up	
Valid Interface:	
TCP/IP	
Description: Universal identifier of the sensor, if supported by sensor.	
Parameter: String SA_Sensor	SA_Sensor
Direction: Up	
Valid Interface:	
TCP/IP	
Description: Name of the sensor, if supported by sensor.	
Parameter: String SA_Manufacturer	SA_Manufacturer
Direction: Up	
Valid Interface:	
TCP/IP	
Description: Manufacturer of the sensor, if supported by sensor.	
Parameter: String SA_MACAddress	SA_MACAddress
Direction: Up	
Valid Interface:	
TCP/IP	
Description: MAC address of the sensor, if known.	
Parameter: String IP_RemoteAddr	IP_RemoteAddr
Direction: Up	
Valid Interface:	
TCP/IP	
Description: IP address of the sensor, extracted from answer packet.	
Parameter: String SA_Address	SA_Address
Direction: Up	
Valid Interface:	
TCP/IP	
Description: IP address of the sensor, returned by sensor. This parameter is output only if it differs to IP_RemoteAddr.	
Parameter: String SA_SubnetMask	SA_SubnetMask
Direction: Up	
Valid Interface:	
TCP/IP	
Description: Netmask of the sensor. 0.0.0.0 means standard netmask for class A, B, C is used.	
Parameter: String SA_Gateway	SA_Gateway
Direction: Up	
Valid Interface:	
TCP/IP	
Description: Default gateway of the sensor.	

Parameter: int SA_StaticIP SA_StaticIP

Direction: Up

Valid Interface:

TCP/IP

Valid values:

0= FALSE
1= TRUE

Description: Sensor has a static IP address.

Parameter: int SA_AutoIPEnabled SA_AutoIPEnabled

Direction: Up

Valid Interface:

TCP/IP

Valid values:

0= FALSE
1= TRUE

Description: AutoIP mode of the sensor.

Parameter: int SA_DHCPEnabled SA_DHCPEnabled

Direction: Up

Valid Interface:

TCP/IP

Valid values:

-1= Not supported by sensor
0= FALSE
1= TRUE

Description: DHCP mode of the sensor.

Parameter: int SA_BootPEnabled SA_BootPEnabled

Direction: Up

Valid Interface:

TCP/IP

Valid values:

0= FALSE
1= TRUE

Description: BootP mode of the sensor.

Parameter: String SA_DHCPName SA_DHCPName

Direction: Up

Valid Interface:

TCP/IP

Description: Name which is used by sensor for registering at DHCP server.

Parameter: String SA_DNSServer SA_DNSServer

Direction: Up

Valid Interface:

TCP/IP

Description: DNS server address of sensor.

Parameter: String SA_Softwareversion SA_Softwareversion

Direction: Up

Valid Interface:

TCP/IP

Description: Software version of sensor.

Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid Interface:	
TCP/IP	
Description: Serial number of sensor.	
Parameter: String SA_LocalIPAddress	SA_LocalIPAddress
Direction: Up	
Valid Interface:	
TCP/IP	
Description: IP address of local computer, on which the sensor answer was received. If the answer was received on multiple addresses, all addresses are set, separated by newline.	
Parameter: String SA_LocalSubnetMask	SA_LocalSubnetMask
Direction: Up	
Valid Interface:	
TCP/IP	
Description: Subnet mask of local computer, on which the sensor answer was received. The number of subnet masks (separated by newline) is equal to the number at SA_LocalIPAddress.	
Parameter: String SA_LocalDefaultGateway	SA_LocalDefaultGateway
Direction: Up	
Valid Interface:	
TCP/IP	
Description: Default gateway of local computer, on which the sensor answer was received. The number of default gateways (separated by newline) is equal to the number at SA_LocalIPAddress.	
Parameter: int SA_LocalDHCPEnabled	SA_LocalDHCPEnabled
Direction: Up	
Valid Interface:	
TCP/IP	
Valid values:	
Specific bit is 0= not enabled	
Specific bit is 1= enabled	
Description: Shows if DHCP (or alternatively AutoIP) is enabled for the local IP address. The value is a bit field, bit 0 corresponds to the first local IP address, and so on.	
Parameter: String SA_DHCPServer	SA_DHCPServer
Direction: Up	
Valid Interface:	
TCP/IP	
Description: IP address of DHCP server, which was leasing the local IP address. The number of DHCP servers (separated by newline) is equal to the number at SA_LocalIPAddress.	
Parameter: String SA_LocalAdapterDescription	SA_LocalAdapterDescrip- tion
Direction: Up	
Valid Interface:	
TCP/IP	
Description: The name of the local network adapter which owns the local IP address. The number of adapter descriptions (separated by newline) is equal to the number at SA_LocalIPAddress.	

Parameter: int SA_SensorAccessible	SA_SensorAccessible
Direction: Up	
Valid Interface:	
TCP/IP	
Valid values:	
Specific bit is 0= not accessible	
Specific bit is 1= accessible	
Description: Sensor and local IP addresses and subnet masks are checked to determine if sensor is accessible by TCP/IP. The value is a bit field, bit 0 corresponds to the first local IP address, and so on.	
Parameter: int IP_DeviceInstance	IP_DeviceInstance
Direction: Up	
Valid Interface:	
WinUSB	
Valid values:	
Minimum: 0	
Maximum: 255	
Description: Instance number of the USB device.	
Parameter: int IP_CardInstance	IP_CardInstance
Direction: Up	
Valid Interface:	
IF2008	
Valid values:	
Minimum: 0	
Maximum: 15	
Description: Instance number of the IF2008 Interface card.	
Parameter: int IP_ChannelNumber	IP_ChannelNumber
Direction: Up	
Valid Interface:	
IF2008	
Valid values:	
Minimum: 0	
Maximum: 11	
Description: Channel number on IF2008 Interface card. If a sensor is searched, the channel number is from 0 to 5, if PCI_CARD_IF2008 is searched, entries for all valid channels from 6 to 11 are returned.	
Parameter: int IP_Baudrate	IP_Baudrate
Direction: Up	
Valid Interface:	
IF2008	
Valid values:	
Minimum: 0	
Unit: Baud	
Description: Baudrate of the RS422 serial connection.	
Parameter: int IP_Parity	IP_Parity
Direction: Up	
Valid Interface:	
IF2008	
Valid values:	
0= NOPARITY	
2= EVENPARITY	
Description: Parity of the RS422 serial connection.	

Additionally a sensor command to get more sensor information is called. It depends on sensor and hardware interface, which command is called. Depending on the command, please refer to the specific command later in this documentation. Following table shows sensor, interface and command:

SENSOR_ILR110x_115x	any interface	Get_Version
SENSOR_ILR118x	any interface	Get_Info
SENSOR_ILR1191	any interface	Get_Info
SENSOR_ODC1202	any interface	Get_Version
SENSOR_ILD1302	any interface	Get_Info
SENSOR_ILD1320	any interface	Get_Info
SENSOR_ILD1401	any interface	Get_Info
SENSOR_ILD1402	any interface	Get_Info
SENSOR_ILD1420	any interface	Get_Info
SENSOR_ILD1700	any interface	Get_Info
SENSOR_ILD2200	any interface	Get_Info
SENSOR_ILD2300	any but TCP/IP	Get_Info
SENSOR_IFD2401	any interface	Get_Version
SENSOR_IFD2431	any interface	Get_Version
SENSOR_IFD2445	any but TCP/IP	Get_Info
SENSOR_IFD2451	any but TCP/IP	Get_Info
SENSOR_IFD2461	any but TCP/IP	Get_Info
SENSOR_IFD2471	any but TCP/IP	Get_Info
SENSOR_ACS7000	any but TCP/IP	Get_Info
SENSOR_ODC2500	any interface	Get_Info
SENSOR_ODC2520	any but TCP/IP	Get_Info
SENSOR_ODC2600	any interface	Get_Info
CONTROLLER_CBOX	any but TCP/IP	Get_Info

8.3.4 Abort_FindSensor

Abort an active scan.

8.3.5 Set_IPConfig

Set IP configuration of sensors with ethernet interface.

Parameter: int IP_SensorType

IP_SensorType

Direction: Down

Valid values:

- [SENSOR_ILD2300](#) (29)
- [SENSOR_IFD2445](#) (39)
- [SENSOR_IFD2451](#) (30)
- [SENSOR_IFD2461](#) (44)
- [SENSOR_IFD2471](#) (26)

SENSOR_ODC2520 (37)
 SENSOR_DT3100 (28)
 SENSOR_DT6100 (16)
 CONTROLLER_DT6200 (33)
 CONTROLLER_DT6500 (15)
 CONTROLLER_KSS6380 (18)
 SENSOR_LLT27xx (31)
 CONTROLLER_CSP2008 (32)
 CONTROLLER_CBOX (38)
 ETH_IF1032 (34)
 SENSOR_ACS7000 (35)

Description: Type of sensor to set IP configuration.

Parameter: String IP_MACAddress

IP_MACAddress

Direction: Down

Valid for sensor:

SENSOR_DT3100 (28)
 SENSOR_DT6100 (16)
 CONTROLLER_DT6500 (15), not DT6530
 CONTROLLER_KSS6380 (18)
 SENSOR_LLT27xx (31)

Description: Mac address used to identify the sensor.

Parameter: String IP_OldIPAddress

IP_OldIPAddress

Direction: Down

Valid for sensor:

SENSOR_DT3100 (28)
 SENSOR_DT6100 (16)
 CONTROLLER_DT6500 (15), not DT6530
 CONTROLLER_KSS6380 (18)
 SENSOR_LLT27xx (31)

Description: Old IP address of sensor to be able to communicate with.

Parameter: String IP_UUID

IP_UUID

Direction: Down

Valid for sensor:

SENSOR_ILD2300 (29)
 SENSOR_IFD2445 (39)
 SENSOR_IFD2451 (30)
 SENSOR_IFD2461 (44)
 SENSOR_IFD2471 (26)
 SENSOR_ODC2520 (37)
 CONTROLLER_CSP2008 (32)
 CONTROLLER_CBOX (38)
 CONTROLLER_DT6200 (33)
 CONTROLLER_DT6500 (15), only DT6530
 ETH_IF1032 (34)
 SENSOR_ACS7000 (35)

Description: UUID used to identify the sensor.

Parameter: String IP_Password IP_Password

Direction: Down

Valid for sensor:

- SENSOR_ILD2300 (29)
- SENSOR_IFD2445 (39)
- SENSOR_IFD2451 (30)
- SENSOR_IFD2461 (44)
- SENSOR_IFD2471 (26)
- SENSOR_ODC2520 (37)
- CONTROLLER_CSP2008 (32)
- CONTROLLER_DT6200 (33)
- CONTROLLER_DT6500 (15), only DT6530
- SENSOR_ACS7000 (35)
- ETH_IF1032 (34)

Description: Password which allow changing IP configuration.

Parameter: String IP_NewIPAddress IP_NewIPAddress

Direction: Down

Valid for sensor: all

Description: New IP address to set at sensor.

Parameter: int IP_AllowReserved IP_AllowReserved

Direction: Down

Valid for sensor: all

Valid values:

- 0 = FALSE
- 1 = TRUE

Default: 1 (will become obligatory at MEDAQLib V4.0)

Description: Sepcifies if new IP address may be in the reserverd range starting from 224.0.0.0 (Multicast, Class D, Class E). Not recommended, because sensor may no longer be accessible!

Parameter: String IP_Gateway IP_Gateway

Direction: Down

Valid for sensor: all

Description: New default gateway to set at sensor.

Parameter: String IP_SubnetMask IP_SubnetMask

Direction: Down

Valid for sensor: all

Description: New subnet mask to set at sensor.

Parameter: int IP_StaticIP IP_StaticIP

Direction: Down

Valid for sensor:

- SENSOR_LL27xx (31)

Valid values:

- 0 = Static IP address should be set
- 1 = Do not set static IP address

Description: Specify if sensor should use static IP address.

Parameter: int IP_DHCPEnabled IP_DHCPEnabled

Direction: Down

Valid for sensor: all

Valid values:

0= DHCP is disabled

1= DHCP is enabled

Description: Specify if sensor should use DHCP.

Parameter: int IP_AutoIPEnabled IP_AutoIPEnabled

Direction: Down

Valid for sensor:

[SENSOR_DT3100](#) (28)

[SENSOR_DT6100](#) (16)

[CONTROLLER_DT6500](#) (15), not DT6530

[CONTROLLER_KSS6380](#) (18)

Valid values:

0= AutoIP is disabled

1= AutoIP is enabled

Description: Specify if sensor should use AutoIP.

Parameter: int IP_BootPEnabled IP_BootPEnabled

Direction: Down

Valid for sensor:

[SENSOR_DT3100](#) (28)

[SENSOR_DT6100](#) (16)

[CONTROLLER_DT6500](#) (15), not DT6530

[CONTROLLER_KSS6380](#) (18)

Valid values:

0= BootP is disabled

1= BootP is enabled

Description: Specify if sensor should use BootP.

Parameter: int IA_Success IA_Success

Direction: Up

Valid for sensor: all

Valid values:

0= Operation failed

1= Operation successful

Description: Returns result of Set_IPConfig operation. Some sensors may not return an acknowledge, in this case, success means that the command was send to sensor.

9 Commands for ILR sensors

9.1 Commands for ILR110x_115x

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [RS232](#) (native)
- [TCP/IP](#) (additional, e.g. RS232/RS422 to TCP/IP comm server and RS232 high level interface)
- [IF2004_USB](#) (native)
- [IF2008](#) (native)

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_Parameters](#) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib timeout checking.

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are already scaled at sensor.
- Scaled values are identical to raw values.

9.1.1 General commands

9.1.1.1 General

9.1.1.1.1 Get_Parameters (GAP)

Retrieve all parameters from the sensor.

Parameter: String SA_Version

SA_Version

Direction: Up

Description: Firmware version.

Parameter: int SA_PilotLaser

SA_PilotLaser

Direction: Up

Valid values:

0= off

1= on

Description: Pilot laser behaviour.

Parameter: int SA_SensorBaudrate

SA_SensorBaudrate

Direction: Up

Valid values:

4800

9600

19200

38400

57600

Description: Sensor baudrate.

9.1. Commands for ILR110x_115x

Parameter: int SA_SensorDatabits	SA_SensorDatabits
Direction: Up	
Valid values:	
7	
8	
Description: Sensor data bits.	
Parameter: int SA_SensorStopbits	SA_SensorStopbits
Direction: Up	
Valid values:	
1	
2	
Description: Sensor stop bits.	
Parameter: int SA_ContinuousMode	SA_ContinuousMode
Direction: Up	
Valid values:	
0= continuous	
1= single	
Description: Sensor is sending data continuous or only single values.	
Parameter: int SA_Q1Value	SA_Q1Value
Direction: Up	
Valid values:	
0= low	
1= high	
Description: The actual state of output Q1.	
Parameter: int SA_ModeQ1	SA_ModeQ1
Direction: Up	
Valid values:	
0= not active	
1= switching point	
2= switching points	
Description: The mode of output Q1.	
Parameter: int SA_LimitQ1-1	SA_LimitQ1-1
Direction: Up	
Valid values:	
Minimum: -48000	
Maximum: 96000	
Description: The limit 1 of output Q1.	
Parameter: int SA_LimitQ1-2	SA_LimitQ1-2
Direction: Up	
Valid values:	
Minimum: -48000	
Maximum: 96000	
Description: The limit 2 of output Q1.	
Parameter: int SA_HysteresisQ1	SA_HysteresisQ1
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 999	
Description: The hysteresis of output Q1.	

9.1. Commands for ILR110x_115x

Parameter: int SA_NormQ1	SA_NormQ1
Direction: Up	
Valid values:	
0= normal	
1= inverted	
Description: Specifies if output Q1 is normal or inverted.	
Parameter: int SA_Q2Value	SA_Q2Value
Direction: Up	
Valid values:	
0= low	
1= high	
Description: The actual state of output Q2.	
Parameter: int SA_ModeQ2	SA_ModeQ2
Direction: Up	
Valid values:	
0= not active	
1= switching point	
2= switching points	
Description: The mode of output Q2.	
Parameter: int SA_LimitQ2-1	SA_LimitQ2-1
Direction: Up	
Valid values:	
Minimum: -48000	
Maximum: 96000	
Description: The limit 1 of output Q2.	
Parameter: int SA_LimitQ2-2	SA_LimitQ2-2
Direction: Up	
Valid values:	
Minimum: -48000	
Maximum: 96000	
Description: The limit 2 of output Q2.	
Parameter: int SA_HysteresisQ2	SA_HysteresisQ2
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 999	
Description: The hysteresis of output Q2.	
Parameter: int SA_NormQ2	SA_NormQ2
Direction: Up	
Valid values:	
0= normal	
1= inverted	
Description: Specifies if output Q2 is normal or inverted.	
Parameter: int SA_AnalogValue	SA_AnalogValue
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 4095	
Description: The actual value of analog output.	

9.1. Commands for ILR110x_115x

Parameter: int SA_LimitQA-1	SA_LimitQA-1
Direction: Up	
Valid values:	
Minimum: -48000	
Maximum: 96000	
Description: The limit 1 of analog output.	
Parameter: int SA_LimitQA-2	SA_LimitQA-2
Direction: Up	
Valid values:	
Minimum: -48000	
Maximum: 96000	
Description: The limit 2 of analog output.	
Parameter: int SA_NormQA	SA_NormQA
Direction: Up	
Valid values:	
0= normal	
1= inverted	
Description: Specifies if analog output is normal or inverted.	
Parameter: int SA_OutputFormat	SA_OutputFormat
Direction: Up	
Valid values:	
0= mm	
1= inch*100	
Description: Output format of measured values.	
Parameter: int SA_Offset	SA_Offset
Direction: Up	
Valid values:	
Minimum: -48000	
Maximum: 48000	
Description: Offset value.	
Parameter: int SA_Password	SA_Password
Direction: Up	
Valid values:	
0= disabled	
1= enabled	
Description: Password protection for sensor menu.	
Parameter: int SA_ErrorStatus	SA_ErrorStatus
Direction: Up	
Valid values:	
Minimum: 0x00	
Maximum: 0xff	
Description: Error status as 8 bit field.	
00000000: no error.	
00000010: PLL UNLOCKED - Counter error.	
00000100: LOW VOLT - Error in supply voltage: voltage too low (or error in measurement of supply voltage).	
00101000: OVERTEMP - Temperature too high (above 85 °C inside); Measurement switched off.	
00010000: Dist (mm) >Maximum - No target in range or sensor badly aligne.	
00100000: Temperature warning (below -10 °C or above 70 °C).	
01000000: BLINDING - External light too strong or internal error.	
10000000: LAS.ERR. - Measurement laser faulty.	

9.1. Commands for ILR110x_115x

9.1.1.1.2 Get_Energy (GDB)

Get the amount of receiving by sensor.

Parameter: int SA_Energy SA_Energy
Direction: Up
Unit: dB
Valid values:
 Minimum: -120
 Maximum: 0
Description: Energy value.

9.1.1.1.3 Get_SerialNbr (GNR)

Get the serial number of sensor.

Parameter: String SA_SerialNumber SA_SerialNumber
Direction: Up
Description: Serial number.

9.1.1.1.4 Get_ErrorStatus (GSI)

Get error status from the sensor.

Parameter: int SA_ErrorStatus SA_ErrorStatus
Direction: Up
Valid values:
 Minimum: 0x00
 Maximum: 0xff
Description: Error status as 8 bit field.
 00000000: no error.
 00000010: PLL UNLOCKED - Counter error.
 00000100: LOW VOLT - Error in supply voltage: voltage too low (or error in measurement of supply voltage).
 00101000: OVERTEMP - Temperature too high (above 85 °C inside); Measurement switched off.
 00010000: Dist (mm) >Maximum - No target in range or sensor badly aligne.
 00100000: Temperature warning (below -10 °C or above 70 °C).
 01000000: BLINDING - External light too strong or internal error.
 10000000: LAS.ERR. - Measurement laser faulty.

9.1.1.1.5 Get_Temperature (GTE)

Retrieve the temperature inside of the sensor.

Parameter: int SA_Temperature SA_Temperature
Direction: Up
Unit: °C
Description: Sensor temperature.

9.1. Commands for ILR110x_115x

9.1.1.1.6 Get_Version (GVE)

Get the version of sensor firmware.

Parameter: String SA_Version

SA_Version

Direction: Up

Description: Firmware version.

9.1.1.1.7 Set_Stand-by (ISB)

Set the sensor in stand-by mode or reactivates it.

Parameter: int SP_Stand-by

SP_Stand-by

Direction: Down

Valid values:

0= operation

1= stand-by

Description: Sensor stand-by mode.

9.1.1.1.8 Set_VisibleLaser (IVL)

Set the behaviour of the pilot laser of the sensor.

Parameter: int SP_PilotLaser

SP_PilotLaser

Direction: Down

Valid values:

0= off

1= on

Description: Pilot laser behaviour.

9.1.1.2 Triggering

9.1.1.2.1 Set_ContinousMode (ICM)

Set the measurement mode.

Parameter: int SP_ContinuousMode

SP_ContinuousMode

Direction: Down

Valid values:

0= continuous

1= single

Description: Sensor is sending data continuous or only single values.

9.1.1.2.2 Exec_ContMeasure (ECM)

Continuous measurement output ist set and triggered by the next request for measured values (ESM).

9.1. Commands for ILR110x_115x

9.1.1.2.3 Trg_SingleMeasure (ESM)

Request for measured value with single measurement output.

9.1.1.3 Parameter management

9.1.1.3.1 Save_Parameters (EPW)

Store all actual parameters in sensor memory.

9.1.2 Measurement

9.1.2.1 General

9.1.2.1.1 Set_Offset (IDO)

Set the offset which is added by the sensor to distance values.

Parameter: int SP_Offset

SP_Offset

Direction: Down

Valid values:

Minimum: -12000 [mm] or -48000 [100*inch]

Maximum: 12000 [mm] or 48000 [100*inch]

Description: Offset value.

9.1.3 Data output

9.1.3.1 Switching outputs

9.1.3.1.1 Set_HysteresisQ1 (IH1)

Hysteresis setting around the switching point Q1 in [mm] or [100*inch].

Parameter: int SP_HysteresisQ1

SP_HysteresisQ1

Direction: Down

Valid values:

Minimum: 0 [mm] / [100*inch]

Maximum: 254 [mm] or 999 [100*inch]

Description: Hysteresis Q1.

9.1.3.1.2 Set_HysteresisQ2 (IH2)

Hysteresis setting around the switching point Q2 in [mm] or [100*inch].

Parameter: int SP_HysteresisQ2

SP_HysteresisQ2

Direction: Down

Valid values:

Minimum: 0 [mm] / [100*inch]

Maximum: 254 [mm] or 999 [100*inch]

Description: Hysteresis Q2.

9.1. Commands for ILR110x_115x

9.1.3.1.3 Set_LimitQ1-1 (IL1)

Setting of the first switch point of Q1 in [mm] or [100*inch].

Parameter: int SP_LimitQ1-1

SP_LimitQ1-1

Direction: Down

Valid values:

Minimum: Offset [mm] / [100*inch]

Maximum: 12000+Offset [mm] or 48000+Offset [100*inch]

Description: Limit Q1-1.

9.1.3.1.4 Set_LimitQ2-1 (IL2)

Setting of the first switch point of Q2 in [mm] or [100*inch].

Parameter: int SP_LimitQ2-1

SP_LimitQ2-1

Direction: Down

Valid values:

Minimum: Offset [mm] / [100*inch]

Maximum: 12000+Offset [mm] or 48000+Offset [100*inch]

Description: Limit Q2-1.

9.1.3.1.5 Set_LimitQ1-2 (IL4)

Setting of the second switch point of Q1 in [mm] or [100*inch].

Parameter: int SP_LimitQ1-2

SP_LimitQ1-2

Direction: Down

Valid values:

Minimum: Offset [mm] / [100*inch]

Maximum: 12000+Offset [mm] or 48000+Offset [100*inch]

Description: Limit Q1-2.

9.1.3.1.6 Set_LimitQ2-2 (IL5)

Setting of the second switch point of Q2 in [mm] or [100*inch].

Parameter: int SP_LimitQ2-2

SP_LimitQ2-2

Direction: Down

Valid values:

Minimum: Offset [mm] / [100*inch]

Maximum: 12000+Offset [mm] or 48000+Offset [100*inch]

Description: Limit Q2-2.

9.1.3.1.7 Set_ModeQ1 (IM1)

Set the mode of output Q1.

Parameter: int SP_ModeQ1

SP_ModeQ1

Direction: Down

Valid values:

0= not active

1= switching point

2= switching points

Description: The mode of output Q1.

9.1. Commands for ILR110x_115x

9.1.3.1.8 Set_ModeQ2 (IM2)

Set the mode of output Q2.

Parameter: int SP_ModeQ2 SP_ModeQ2
Direction: Down
Valid values:
 0= not active
 1= switching point
 2= switching points
Description: The mode of output Q2.

9.1.3.1.9 Set_NormQ1 (IN1)

Set the norm of output Q1.

Parameter: int SP_NormQ1 SP_NormQ1
Direction: Down
Valid values:
 0= normal
 1= inverted
Description: Specifies if output Q1 is normal or inverted.

9.1.3.1.10 Set_NormQ2 (IN2)

Set the norm of output Q2.

Parameter: int SP_NormQ2 SP_NormQ2
Direction: Down
Valid values:
 0= normal
 1= inverted
Description: Specifies if output Q2 is normal or inverted.

9.1.3.2 Analog output

9.1.3.2.1 Set_LimitQA-1 (IL3)

Setting of the 0% point of the analog characteristic. Only valid for sensors ILR1100 and ILR1150 (with analog output).

Parameter: int SP_LimitQA-1 SP_LimitQA-1
Direction: Down
Valid values:
Minimum: Offset
Maximum: 12000+Offset
Description: Limit QA-1.

9.1.3.2.2 Set_LimitQA-2 (IL6)

Setting of the 100% point of the analog characteristic. Only valid for sensors ILR1100 and ILR1150 (with analog output).

Parameter: int SP_LimitQA-2 SP_LimitQA-2
Direction: Down
Valid values:
 Minimum: Offset
 Maximum: 12000+Offset
Description: Limit QA-2.

9.1.3.2.3 Set_NormQA (INA)

Set the norm of analog output. Only valid for sensors ILR1100 and ILR1150 (with analog output).

Parameter: int SP_NormQA SP_NormQA
Direction: Down
Valid values:
 0= normal
 1= inverted
Description: Specifies if analog output is normal or inverted.

9.2 Commands for ILR118x

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [RS232](#) (native).
- [TCP/IP](#) (additional, e.g. RS232/RS422 to TCP/IP comm server and RS232 high level interface).
- [IF2004_USB](#) (native).
- [IF2008](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_AllParameters](#) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib timeout checking and to calculate datarate.

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are already scaled at sensor.
- Scaled values are identical to raw values, except error values are scaled depending of [IP_ScaleErrorValues](#).

9.2.1 General commands

9.2.1.1 General

9.2.1.1.1 Set_Autostart (AS)

Set which function will be carried out when power becomes available to the sensor.

Parameter: int SP_AutostartCommand

SP_AutostartCommand

Direction: Down

Valid values:

- 0= DT
- 1= DS
- 2= DW
- 3= DX
- 4= DF
- 5= DM
- 6= TP
- 7= ID
- 8= LO

Description: Autostart command.

Parameter: int SA_AutostartCommand

SA_AutostartCommand

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= DT
- 1= DS
- 2= DW
- 3= DX
- 4= DF
- 5= DM
- 6= TP
- 7= ID
- 8= LO

Description: Adapted value from sensor if parameter to set was invalid.

9.2.1.1.2 Get_Autostart (AS)

Get which function will be carried out when power becomes available to the sensor.

Parameter: int SA_AutostartCommand

SA_AutostartCommand

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= DT
- 1= DS
- 2= DW
- 3= DX
- 4= DF
- 5= DM
- 6= TP
- 7= ID
- 8= LO

Description: Autostart command.

9.2.1.1.3 Get_Info (ID)

Retrieve information (like serial number) of the sensor.

Parameter: String SA_Version SA_Version
Direction: Up
Description: Sensor name, range, serial number and version

9.2.1.1.4 Get_AllParameters (PA)

Retrieve all parameters from the sensor.

Parameter: int SA_Average SA_Average
Direction: Up
Valid values:
Minimum: 1
Maximum: 20
Description: Average value.

Parameter: int SA_OutputFormat SA_OutputFormat
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= decimal distance
 1= hex distance
 2= binary distance
 3= decimal (distance + signal strength)
Description: Output format of values.

Parameter: int SA_MeasureTime SA_MeasureTime
Direction: Up
Valid values:
Minimum: 0
Maximum: 25
Description: Measure time index.

Parameter: double SA_ScaleFactor SA_ScaleFactor
Direction: Up
Valid values:
Minimum: -3.40282e+38 (-FLT_MAX)
Maximum: 3.40282e+38 (FLT_MAX)
Description: Scaling factor.

Parameter: int SA_ErrorMode SA_ErrorMode
Direction: Up
Valid values:
 0= last valid value
 1= switch to bounds
 2= switch to negated bounds
Description: Error mode.

9.2. Commands for ILR118x

Parameter: double SA_AlarmStart	SA_AlarmStart
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Alarm start.	
Parameter: double SA_AlarmHysteresis	SA_AlarmHysteresis
Direction: Up	
Valid values:	
Minimum: -100000	
Maximum: 100000	
Description: Alarm hysteresis.	
Parameter: double SA_AlarmWidth	SA_AlarmWidth
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Alarm width.	
Parameter: double SA_RangeBegin	SA_RangeBegin
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Range begin.	
Parameter: double SA_RangeEnd	SA_RangeEnd
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Range end.	
Parameter: int SA_PrecedingValues	SA_PrecedingValues
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 10	
Description: Designates the number of preceding measuring values that will be evaluated in the case of non-conforming measurement.	
Parameter: double SA_ValidRange	SA_ValidRange
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Defines the range of permissible values. If this range is exceeded in negative or positive direction, the respective measuring value will be corrected accordingly.	

9.2. Commands for ILR118x

Parameter: int SA_InvalidValues	SA_InvalidValues
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 100	
Description:	Stands for the number of values that are out of the permissible value range; in the event of out-of-tolerance values arriving in succession, the most recently corrected value will be included in the correction process for the next out-of-tolerance value.
Parameter: int SA_TriggerDelay	SA_TriggerDelay
Direction: Up	
Unit: ms	
Valid values:	
Minimum: 0	
Maximum: 9999	
Description:	Trigger delay.
Parameter: int SA_TriggerEdge	SA_TriggerEdge
Direction: Up	
Valid values:	
0= falling	
1= rising	
Description:	Trigger edge.
Parameter: int SA_AutostartTrigger	SA_AutostartTrigger
Direction: Up	
Valid values:	
0= off	
1= on	
Description:	Autostart trigger.
Parameter: int SA_AutostartEdge	SA_AutostartEdge
Direction: Up	
Valid values:	
0= falling	
1= rising	
Description:	Autostart trigger edge.
Parameter: int SA_SensorBaudrate	SA_SensorBaudrate
Direction: Up	
Valid values:	
2400	
4800	
9600	
19200	
38400	
Description:	Sensor baudrate.

9.2. Commands for ILR118x

Parameter: int SA_AutostartCommand	SA_AutostartCommand
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= DT	
1= DS	
2= DW	
3= DX	
4= DF	
5= DM	
6= TP	
7= ID	
8= LO	
Description: Autostart command.	
Parameter: double SA_Offset	SA_Offset
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Offset value which is set.	

9.2.1.1.5 Get_Temperature (TP)

Retrieve the temperature inside of the sensor.

Parameter: double SA_Temperature	SA_Temperature
Direction: Up	
Unit: °C	
Description: Sensor temperature.	

9.2.1.1.6 Laser_Off (LF)

Switch the laser off.

9.2.1.1.7 Laser_On (LO)

Switch the laser on.

9.2.1.2 Tracking

9.2.1.2.1 DistanceTracking (DT)

Start distance tracking mode.

9.2.1.2.2 DistanceTracking7m (DS)

Start distance tracking (7 m) mode.

9.2.1.2.3 DistanceTracking10Hz (DW)

Start distance tracking (10 Hz) mode. Only valid for ILR1181.

9.2.1.2.4 DistanceTracking50Hz (DX)

Start distance tracking (10 Hz) mode. Only valid for ILR1182.

9.2.1.2.5 StopTracking (<ESC>)

Stop any tracking mode.

9.2.1.3 Triggering

9.2.1.3.1 DistanceTriggered (DF)

Start distance tracking (with external trigger) mode.

9.2.1.3.2 DistanceMeasure (DM)

Measure one distance value.

9.2.1.3.3 Set_MeasureTime (ST)

Set an index for measure time of one distance value.

Parameter: int SP_MeasureTime

SP_MeasureTime

Direction: Down

Valid values:

Minimum: 0

Maximum: 25

Description: Measure time index.

Parameter: int SA_MeasureTime

SA_MeasureTime

Direction: Up

Valid values:

Minimum: 0

Maximum: 25

Description: Adapted value from sensor if parameter to set was invalid.

9.2.1.3.4 Get_MeasureTime (ST)

Get an index for measure time of one distance value.

Parameter: int SA_MeasureTime

SA_MeasureTime

Direction: Up

Valid values:

Minimum: 0

Maximum: 25

Description: Measure time index.

9.2.1.3.5 Set_TriggerDelay (TD)

Set the behaviour of the trigger input.

Parameter: int SP_TriggerDelay SP_TriggerDelay

Direction: Down

Unit: ms

Valid values:

Minimum: 0

Maximum: 9999

Description: Trigger delay.

Parameter: int SP_TriggerEdge SP_TriggerEdge

Direction: Down

Valid values:

0= falling

1= rising

Description: Trigger edge.

Parameter: int SA_TriggerDelay SA_TriggerDelay

Direction: Up

Unit: ms

Valid values:

Minimum: 0

Maximum: 9999

Description: Adapted value from sensor if parameter to set was invalid.

Parameter: int SA_TriggerEdge SA_TriggerEdge

Direction: Up

Valid values:

0= falling

1= rising

Description: Adapted value from sensor if parameter to set was invalid.

9.2.1.3.6 Get_TriggerDelay (TD)

Get the behaviour of the trigger input.

Parameter: int SA_TriggerDelay SA_TriggerDelay

Direction: Up

Unit: ms

Valid values:

Minimum: 0

Maximum: 9999

Description: Trigger delay.

Parameter: int SA_TriggerEdge SA_TriggerEdge

Direction: Up

Valid values:

0= falling

1= rising

Description: Trigger edge.

9.2.1.3.7 Set_TriggerMode (TM)

Set parameters for the auto-start trigger function which allows external triggering of the auto-start command that was set via parameter AS.

Parameter: int SP_AutostartTrigger	SP_AutostartTrigger
Direction: Down	
Valid values:	
0= off	
1= on	
Description:	Autostart trigger.
Parameter: int SP_AutostartEdge	SP_AutostartEdge
Direction: Down	
Valid values:	
0= falling	
1= rising	
Description:	Autostart trigger edge.
Parameter: int SA_AutostartTrigger	SA_AutostartTrigger
Direction: Up	
Valid values:	
0= off	
1= on	
Description:	Adapted value from sensor if parameter to set was invalid.
Parameter: int SA_AutostartEdge	SA_AutostartEdge
Direction: Up	
Valid values:	
0= falling	
1= rising	
Description:	Adapted value from sensor if parameter to set was invalid.

9.2.1.3.8 Get_TriggerMode (TM)

Get parameters for the auto-start trigger function which allows external triggering of the auto-start command that was set via parameter AS.

Parameter: int SA_AutostartTrigger	SA_AutostartTrigger
Direction: Up	
Valid values:	
0= off	
1= on	
Description:	Autostart trigger.
Parameter: int SA_AutostartEdge	SA_AutostartEdge
Direction: Up	
Valid values:	
0= falling	
1= rising	
Description:	Autostart trigger edge.

9.2.1.4 Interfaces

9.2.1.4.1 Set_Baudrate (BR)

Set the baudrate of the sensors serial interface. After receiving the answer of the command, the baudrate of the local serial interface is changed, too.

Attention! To avoid synchronisation problems, sending data by sensor should be disabled first (sensor command StopTracking).

Parameter: int SP_SensorBaudrate

SP_SensorBaudrate

Direction: Down

Valid values:

- 2400
- 4800
- 9600
- 19200
- 38400

Description: Sensor baudrate.

Parameter: int SA_SensorBaudrate

SA_SensorBaudrate

Direction: Up

Valid values:

- 2400
- 4800
- 9600
- 19200
- 38400

Description: Adapted value from sensor if parameter to set was invalid.

9.2.1.4.2 Get_Baudrate (BR)

Get the baudrate of the sensors serial interface.

Parameter: int SA_SensorBaudrate

SA_SensorBaudrate

Direction: Up

Valid values:

- 2400
- 4800
- 9600
- 19200
- 38400

Description: Sensor baudrate.

9.2.1.5 Parameter management

9.2.1.5.1 Reset_Parameters (PR)

Reset all parameters of sensor to factory defaults and return new parameters.

Parameter: int SA_Average

SA_Average

Direction: Up

Valid values:

- Minimum: 1
- Maximum: 20

Description: Average value.

9.2. Commands for ILR118x

Parameter: int SA_OutputFormat	SA_OutputFormat
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= decimal distance	
1= hex distance	
2= binary distance	
3= decimal (distance + signal strength)	
Description: Output format of values.	
Parameter: int SA_MeasureTime	SA_MeasureTime
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 25	
Description: Measure time index.	
Parameter: double SA_ScaleFactor	SA_ScaleFactor
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Scaling factor.	
Parameter: int SA_ErrorMode	SA_ErrorMode
Direction: Up	
Valid values:	
0= last valid value	
1= switch to bounds	
2= switch to negated bounds	
Description: Error mode.	
Parameter: double SA_AlarmStart	SA_AlarmStart
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Alarm start.	
Parameter: double SA_AlarmHysteresis	SA_AlarmHysteresis
Direction: Up	
Valid values:	
Minimum: -100000	
Maximum: 100000	
Description: Alarm hysteresis.	
Parameter: double SA_AlarmWidth	SA_AlarmWidth
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Alarm width.	

Parameter: double SA_RangeBegin	SA_RangeBegin
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Range begin.	
Parameter: double SA_RangeEnd	SA_RangeEnd
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Range end.	
Parameter: int SA_PrecedingValues	SA_PrecedingValues
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 10	
Description: Designates the number of preceding measuring values that will be evaluated in the case of non-conforming measurement.	
Parameter: double SA_ValidRange	SA_ValidRange
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Defines the range of permissible values. If this range is exceeded in negative or positive direction, the respective measuring value will be corrected accordingly.	
Parameter: int SA_InvalidValues	SA_InvalidValues
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 100	
Description: Stands for the number of values that are out of the permissible value range; in the event of out-of-tolerance values arriving in succession, the most recently corrected value will be included in the correction process for the next out-of-tolerance value.	
Parameter: int SA_TriggerDelay	SA_TriggerDelay
Direction: Up	
Unit: ms	
Valid values:	
Minimum: 0	
Maximum: 9999	
Description: Trigger delay.	
Parameter: int SA_TriggerEdge	SA_TriggerEdge
Direction: Up	
Valid values:	
0= falling	
1= rising	
Description: Trigger edge.	

9.2. Commands for ILR118x

Parameter: int SA_AutostartTrigger	SA_AutostartTrigger
Direction: Up	
Valid values:	
0= off	
1= on	
Description: Autostart trigger.	
Parameter: int SA_AutostartEdge	SA_AutostartEdge
Direction: Up	
Valid values:	
0= falling	
1= rising	
Description: Autostart trigger edge.	
Parameter: int SA_SensorBaudrate	SA_SensorBaudrate
Direction: Up	
Valid values:	
2400	
4800	
9600	
19200	
38400	
Description: Sensor baudrate.	
Parameter: int SA_AutostartCommand	SA_AutostartCommand
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= DT	
1= DS	
2= DW	
3= DX	
4= DF	
5= DM	
6= TP	
7= ID	
8= LO	
Description: Autostart command.	
Parameter: double SA_Offset	SA_Offset
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Offset value which is set.	

9.2.2 Measurement

9.2.2.1 Measurement value processing

9.2.2.1.1 Set_ScaleFactor (SF)

Set the scaling factor how the sensor scale distance values.

9.2. Commands for ILR118x

Parameter: double SP_ScaleFactor	SP_ScaleFactor
Direction: Down	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Scaling factor.	
Parameter: double SA_ScaleFactor	SA_ScaleFactor
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Adapted value from sensor if parameter to set was invalid.	

9.2.2.1.2 Get_ScaleFactor (SF)

Get the scaling factor how the sensor scale distance values.

Parameter: double SA_ScaleFactor	SA_ScaleFactor
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Scaling factor.	

9.2.2.1.3 Set_Offset (OF)

Set the offset which is added by the sensor to distance values.

Parameter: double SP_Offset	SP_Offset
Direction: Down	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Offset value.	
Parameter: double SA_Offset	SA_Offset
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Adapted value from sensor if parameter to set was invalid.	

9.2.2.1.4 Get_Offset (OF)

Get the offset which is added by the sensor to distance values.

Parameter: double SA_Offset	SA_Offset
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Offset value.	

9.2.2.1.5 CurrentDistAsOffset (SO)

Set the current distance value as offset.

Parameter: double SA_Offset

SA_Offset

Direction: Up

Valid values:

Minimum: -3.40282e+38 (-FLT_MAX)

Maximum: 3.40282e+38 (FLT_MAX)

Description: Offset value which is set.

9.2.2.1.6 Set_AverageValue (SA)

Set the average value for floating averaging.

Parameter: int SP_Average

SP_Average

Direction: Down

Valid values:

Minimum: 1

Maximum: 20

Description: Average value.

Parameter: int SA_Average

SA_Average

Direction: Up

Valid values:

Minimum: 1

Maximum: 20

Description: Adapted value from sensor if parameter to set was invalid.

9.2.2.1.7 Get_AverageValue (SA)

Get the average value for floating averaging.

Parameter: int SA_Average

SA_Average

Direction: Up

Valid values:

Minimum: 1

Maximum: 20

Description: Average value.

9.2.3 Data output

9.2.3.1 General

9.2.3.1.1 Set_OutputFormat (SD)

Set the output format how values are sent from sensor.

Parameter: int SP_OutputFormat

SP_OutputFormat

Direction: Down

Valid values:

0= decimal distance

1= hex distance

2= binary distance

3= decimal (distance + signal strength)

Description: Output format of values.

Mode binary is currently not supported by MEDAQLib. So if it is selected, no values can be read.

Parameter: int SA_OutputFormat SA_OutputFormat

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = decimal distance
- 1 = hex distance
- 2 = binary distance
- 3 = decimal (distance + signal strength)

Description: Adapted value from sensor if parameter to set was invalid.

9.2.3.1.2 Get_OutputFormat (SD)

Get the output format how values are sent from sensor.

Parameter: int SA_OutputFormat SA_OutputFormat

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = decimal distance
- 1 = hex distance
- 2 = binary distance
- 3 = decimal (distance + signal strength)

Description: Output format of values.

9.2.3.1.3 Set_RemovalMeasVal (RM)

Set how invalid measure values should be treated by the sensor.

Parameter: int SP_PrecedingValues SP_PrecedingValues

Direction: Down

Valid values:

- Minimum:** 0
Maximum: 10

Description: Designates the number of preceding measuring values that will be evaluated in the case of non-conforming measurement.

Parameter: double SP_ValidRange SP_ValidRange

Direction: Down

Valid values:

- Minimum:** 0
Maximum: 3.40282e+38 (FLT_MAX)

Description: Defines the range of permissible values. If this range is exceeded in negative or positive direction, the respective measuring value will be corrected accordingly.

Parameter: int SP_InvalidValues SP_InvalidValues

Direction: Down

Valid values:

- Minimum:** 0
Maximum: 100

Description: Stands for the number of values that are out of the permissible value range; in the event of out-of-tolerance values arriving in succession, the most recently corrected value will be included in the correction process for the next out-of-tolerance value.

Parameter: int SA_PrecedingValues	SA_PrecedingValues
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 10	
Description: Adapted value from sensor if parameter to set was invalid.	
Parameter: double SA_ValidRange	SA_ValidRange
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Adapted value from sensor if parameter to set was invalid.	
Parameter: int SA_InvalidValues	SA_InvalidValues
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 100	
Description: Adapted value from sensor if parameter to set was invalid.	

9.2.3.1.4 Get_RemovalMeasVal (RM)

Get how invalid measure values should be treated by the sensor.

Parameter: int SA_PrecedingValues	SA_PrecedingValues
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 10	
Description: Designates the number of preceding measuring values that will be evaluated in the case of non-conforming measurement.	
Parameter: double SA_ValidRange	SA_ValidRange
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Defines the range of permissible values. If this range is exceeded in negative or positive direction, the respective measuring value will be corrected accordingly.	
Parameter: int SA_InvalidValues	SA_InvalidValues
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 100	
Description: Stands for the number of values that are out of the permissible value range; in the event of out-of-tolerance values arriving in succession, the most recently corrected value will be included in the correction process for the next out-of-tolerance value.	

9.2.3.1.5 Set_ErrorMode (SE)

Set the behaviour of digital and analog outputs in case of an error.

Parameter: int SP_ErrorMode SP_ErrorMode

Direction: Down

Valid values:

- 0= last valid value
- 1= switch to bounds
- 2= switch to negated bounds

Description: Error mode.

Parameter: int SA_ErrorMode SA_ErrorMode

Direction: Up

Valid values:

- 0= last valid value
- 1= switch to bounds
- 2= switch to negated bounds

Description: Adapted value from sensor if parameter to set was invalid.

9.2.3.1.6 Get_ErrorMode (SE)

Get the behaviour of digital and analog outputs in case of an error.

Parameter: int SA_ErrorMode SA_ErrorMode

Direction: Up

Valid values:

- 0= last valid value
- 1= switch to bounds
- 2= switch to negated bounds

Description: Error mode.

9.2.3.2 Switching outputs

9.2.3.2.1 Set_AlarmStart (AC)

Sets the beginning of the distance range, for which the switching output will be turned active.

Parameter: double SP_AlarmStart SP_AlarmStart

Direction: Down

Valid values:

- Minimum:** -3.40282e+38 (-FLT_MAX)
- Maximum:** 3.40282e+38 (FLT_MAX)

Description: Alarm start.

Parameter: double SA_AlarmStart SA_AlarmStart

Direction: Up

Valid values:

- Minimum:** -3.40282e+38 (-FLT_MAX)
- Maximum:** 3.40282e+38 (FLT_MAX)

Description: Adapted value from sensor if parameter to set was invalid.

9.2.3.2.2 Get_AlarmStart (AC)

Gets the beginning of the distance range, for which the switching output will be turned active.

Parameter: double SA_AlarmStart

SA_AlarmStart

Direction: Up

Valid values:

Minimum: -3.40282e+38 (-FLT_MAX)

Maximum: 3.40282e+38 (FLT_MAX)

Description: Alarm start.

9.2.3.2.3 Set_AlarmHysteresis (AH)

Set the switching hysteresis at the beginning and the end point of the active range of the switching output.

Parameter: double SP_AlarmHysteresis

SP_AlarmHysteresis

Direction: Down

Valid values:

Minimum: -100000

Maximum: 100000

Description: Alarm hysteresis.

Parameter: double SA_AlarmHysteresis

SA_AlarmHysteresis

Direction: Up

Valid values:

Minimum: -100000

Maximum: 100000

Description: Adapted value from sensor if parameter to set was invalid.

9.2.3.2.4 Get_AlarmHysteresis (AH)

Get the switching hysteresis at the beginning and the end point of the active range of the switching output.

Parameter: double SA_AlarmHysteresis

SA_AlarmHysteresis

Direction: Up

Valid values:

Minimum: -100000

Maximum: 100000

Description: Alarm hysteresis.

9.2.3.2.5 Set_AlarmWidth (AW)

Set the length of the active range for the switching output.

Parameter: double SP_AlarmWidth

SP_AlarmWidth

Direction: Down

Valid values:

Minimum: 0

Maximum: 3.40282e+38 (FLT_MAX)

Description: Alarm width.

Parameter: double SA_AlarmWidth SA_AlarmWidth
Direction: Up
Valid values:
Minimum: 0
Maximum: 3.40282e+38 (FLT_MAX)
Description: Adapted value from sensor if parameter to set was invalid.

9.2.3.2.6 Get_AlarmWidth (AW)

Get the length of the active range for the switching output.

Parameter: double SA_AlarmWidth SA_AlarmWidth
Direction: Up
Valid values:
Minimum: 0
Maximum: 3.40282e+38 (FLT_MAX)
Description: Alarm width.

9.2.3.3 Analog output

9.2.3.3.1 Set_RangeBegin4mA (RB)

Set the starting point of the distance range that is provided at the analog output.

Parameter: double SP_RangeBegin SP_RangeBegin
Direction: Down
Valid values:
Minimum: -3.40282e+38 (-FLT_MAX)
Maximum: 3.40282e+38 (FLT_MAX)
Description: Range begin.

Parameter: double SA_RangeBegin SA_RangeBegin
Direction: Up
Valid values:
Minimum: -3.40282e+38 (-FLT_MAX)
Maximum: 3.40282e+38 (FLT_MAX)
Description: Adapted value from sensor if parameter to set was invalid.

9.2.3.3.2 Get_RangeBegin4mA (RB)

Get the starting point of the distance range that is provided at the analog output.

Parameter: double SA_RangeBegin SA_RangeBegin
Direction: Up
Valid values:
Minimum: -3.40282e+38 (-FLT_MAX)
Maximum: 3.40282e+38 (FLT_MAX)
Description: Range begin.

9.2.3.3.3 Set_RangeEnd20mA (RE)

Set the end point of the distance range that is provided at the analog output.

Parameter: double SP_RangeEnd

SP_RangeEnd

Direction: Down

Valid values:

Minimum: -3.40282e+38 (-FLT_MAX)

Maximum: 3.40282e+38 (FLT_MAX)

Description: Range end.

Parameter: double SA_RangeEnd

SA_RangeEnd

Direction: Up

Valid values:

Minimum: -3.40282e+38 (-FLT_MAX)

Maximum: 3.40282e+38 (FLT_MAX)

Description: Adapted value from sensor if parameter to set was invalid.

9.2.3.3.4 Get_RangeEnd20mA (RE)

Get the end point of the distance range that is provided at the analog output.

Parameter: double SA_RangeEnd

SA_RangeEnd

Direction: Up

Valid values:

Minimum: -3.40282e+38 (-FLT_MAX)

Maximum: 3.40282e+38 (FLT_MAX)

Description: Range end.

9.3 Commands for ILR1191

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [RS232](#) (native).
- [TCP/IP](#) (additional, e.g. RS232/RS422 to TCP/IP comm server and RS232 high level interface).
- [IF2004_USB](#) (native).
- [IF2008](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_AllParameters](#) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib to calculate datarate and to assign values.

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are already scaled at sensor.
- Scaled values are identical to raw values, except error values are scaled depending of [IP_ScaleErrorValues](#).

9.3.1 General commands

9.3.1.1 General

9.3.1.1.1 Set_Autostart (AS)

Set which function will be carried out when power becomes available to the sensor.

Parameter: int SP_AutostartCommand

SP_AutostartCommand

Direction: Down

Valid values:

- 0= ID
- 1= DM
- 2= DT
- 3= DF
- 4= VM
- 5= VT
- 6= TP
- 7= HW
- 8= PA
- 9= MF
- 10= TD
- 11= SA
- 12= SF
- 13= MW
- 14= OF
- 15= SE
- 16= Q1
- 17= Q2
- 18= QA
- 19= BR
- 20= SD
- 21= TE
- 22= BB
- 23= AB
- 24= SC
- 25= PL
- 26= AS

Description: Autostart command.

Parameter: int SA_AutostartCommand

SA_AutostartCommand

Direction: Up

Valid values:

- 0= ID
- 1= DM
- 2= DT
- 3= DF
- 4= VM
- 5= VT
- 6= TP
- 7= HW
- 8= PA
- 9= MF

```

 10= TD
 11= SA
 12= SF
 13= MW
 14= OF
 15= SE
 16= Q1
 17= Q2
 18= QA
 19= BR
 20= SD
 21= TE
 22= BB
 23= AB
 24= SC
 25= PL
 26= AS
 -1= unknown
  
```

Description: Autostart command.

9.3.1.1.2 Get_Autostart (AS)

Get which function will be carried out when power becomes available to the sensor.

Parameter: int SA_AutostartCommand

SA_AutostartCommand

Direction: Up

Valid values:

```

 0= ID
 1= DM
 2= DT
 3= DF
 4= VM
 5= VT
 6= TP
 7= HW
 8= PA
 9= MF
 10= TD
 11= SA
 12= SF
 13= MW
 14= OF
 15= SE
 16= Q1
 17= Q2
 18= QA
 19= BR
 20= SD
 21= TE
 22= BB
 23= AB
 24= SC
  
```

9.3. Commands for ILR1191

25= PL

26= AS

-1= unknown

Description: Autostart command.

9.3.1.1.3 Set_PilotLaser (PL)

Set the behaviour of the pilot laser of the sensor.

Parameter: int SP_PilotLaser

SP_PilotLaser

Direction: Down

Valid values:

0= off

1= on

2= flashing (2 Hz)

3= flashing (5 Hz)

Description: Pilot laser behaviour.

Parameter: int SA_PilotLaser

SA_PilotLaser

Direction: Up

Valid values:

0= off

1= on

2= flashing (2 Hz)

3= flashing (5 Hz)

Description: Pilot laser behaviour.

9.3.1.1.4 Get_PilotLaser (PL)

Get the behaviour of the pilot laser of the sensor.

Parameter: int SA_PilotLaser

SA_PilotLaser

Direction: Up

Valid values:

0= off

1= on

2= flashing (2 Hz)

3= flashing (5 Hz)

Description: Pilot laser behaviour.

9.3.1.1.5 Get_Info (ID)

Retrieve information (like serial number) of the sensor.

Parameter: String SA_Version

SA_Version

Direction: Up

Description: Firmware version.

9.3.1.1.6 Get_AllParameters (PA)

Retrieve all parameters from the sensor.

Parameter: int SA_MeasFrequency	SA_MeasFrequency
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 2000	
Description: Measure frequency.	
Parameter: double SA_TriggerDelay	SA_TriggerDelay
Direction: Up	
Unit: ms	
Valid values:	
Minimum: 0.0	
Maximum: 314.15	
Description: Trigger delay.	
Parameter: int SA_TriggerEdge	SA_TriggerEdge
Direction: Up	
Valid values:	
0= falling	
1= rising	
Description: Trigger edge.	
Parameter: int SA_TriggerMode	SA_TriggerMode
Direction: Up	
Valid values:	
0= single	
1= measure	
Description: Specifies if one laser shot should be done or single measuring should be started.	
Parameter: int SA_Average	SA_Average
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 30000	
Description: Average value.	
Parameter: double SA_ScaleFactor	SA_ScaleFactor
Direction: Up	
Valid values:	
Minimum: -10.0	
Maximum: 10.0	
Description: Scaling factor.	
Parameter: double SA_WindowMin	SA_WindowMin
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Window minimum value.	

Parameter: double SA_WindowMax	SA_WindowMax
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Window maximum value.	
Parameter: double SA_Offset	SA_Offset
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Offset value which is set.	
Parameter: int SA_ErrorMode	SA_ErrorMode
Direction: Up	
Valid values:	
0= last valid value	
1= switch to bounds	
2= switch to negated bounds	
Description: Error mode.	
Parameter: double SA_Q1Start	SA_Q1Start
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Start value.	
Parameter: double SA_Q1Width	SA_Q1Width
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Width.	
Parameter: double SA_Q1Hysteresis	SA_Q1Hysteresis
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Hysteresis.	
Parameter: int SA_Q1Negation	SA_Q1Negation
Direction: Up	
Valid values:	
0= false	
1= true	
Description: Negation.	
Parameter: double SA_Q2Start	SA_Q2Start
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Start value.	

9.3. Commands for ILR1191

Parameter: double SA_Q2Width	SA_Q2Width
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Width.	
Parameter: double SA_Q2Hysteresis	SA_Q2Hysteresis
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Hysteresis.	
Parameter: int SA_Q2Negation	SA_Q2Negation
Direction: Up	
Valid values:	
0= false	
1= true	
Description: Negation.	
Parameter: double SA_RangeBegin	SA_RangeBegin
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Lower limit value.	
Parameter: double SA_RangeEnd	SA_RangeEnd
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Upper limit value.	
Parameter: int SA_SensorBaudrate	SA_SensorBaudrate
Direction: Up	
Valid values:	
9600	
19200	
38400	
57600	
115200	
230400	
460800	
Description: Sensor baudrate.	
Parameter: int SA_OutputFormat	SA_OutputFormat
Direction: Up	
Valid values:	
0= decimal	
1= hex	
2= binary	
Description: Output format of values.	

9.3. Commands for ILR1191

Parameter: int SA_OutputContent	SA_OutputContent
Direction: Up	
Valid values:	
0= Measuring value	
1= Measuring value, signal strength	
2= Measuring value, sensor temperature	
3= Measuring value, signal strength, sensor temperature	
Description: Get which data is transmitted by sensor.	
Parameter: int SA_TerminationChar	SA_TerminationChar
Direction: Up	
Valid values:	
0= <CRLF>	
1= <CR>	
2= <LF>	
3= <STX>	
4= <ETX>	
5= Tabulator	
6= Space	
7= Comma	
8= Colon	
9= Semicolon	
Description: Termination character.	
Parameter: int SA_SSIFormat	SA_SSIFormat
Direction: Up	
Valid values:	
0= binary	
1= grey code	
Description: SSI transmission format.	
Parameter: int SA_PilotLaser	SA_PilotLaser
Direction: Up	
Valid values:	
0= off	
1= on	
2= flashing (2 Hz)	
3= flashing (5 Hz)	
Description: Pilot laser behaviour.	
Parameter: int SA_AutostartCommand	SA_AutostartCommand
Direction: Up	
Valid values:	
0= ID	
1= DM	
2= DT	
3= DF	
4= VM	
5= VT	
6= TP	
7= HW	
8= PA	
9= MF	
10= TD	

9.3. Commands for ILR1191

```

11= SA
12= SF
13= MW
14= OF
15= SE
16= Q1
17= Q2
18= QA
19= BR
20= SD
21= TE
22= BB
23= AB
24= SC
25= PL
26= AS
-1= unknown

```

Description: Autostart command.

9.3.1.1.7 Get_Temperature (TP)

Retrieve the temperature inside of the sensor.

Parameter: double SA_Temperature

SA_Temperature

Direction: Up

Unit: °C

Description: Sensor temperature.

9.3.1.1.8 Get_HWDiagnosis (HW)

Retrieve internal sensor diagnostic information.

Parameter: String SA_Diagnosis

SA_Diagnosis

Direction: Up

Description: Sensor diagnostic information.

9.3.1.1.9 Trigger_ColdStart (DR)

Reboots the sensor and executes the autostart command.

9.3.1.2 Tracking

9.3.1.2.1 DistanceTracking (DT)

Start distance tracking mode.

9.3.1.2.2 SpeedTracking (VT)

Start speed (velocity) tracking mode.

9.3.1.2.3 StopTracking (<ESC>)

Stop any tracking mode.

9.3.1.3 Triggering

9.3.1.3.1 DistanceTriggered (DF)

Start distance tracking (with external trigger) mode.

9.3.1.3.2 DistanceMeasure (DM)

Measure one distance value.

9.3.1.3.3 SpeedMeasure (VM)

Measure one speed (velocity) value.

Attention! To leave that mode with StopTracking, there is a delay of average value (SA) / measure frequency (MF) seconds.

For example, if average value is 20 and measure frequency is 10 Hz, it takes 2 seconds to leave this mode. In extreme case, this could be 30000/1 seconds, more than 8 hours.

9.3.1.3.4 Set_TriggerDelay (TD)

Set the behaviour of the trigger input.

Parameter: double SP_TriggerDelay

SP_TriggerDelay

Direction: Down

Unit: ms

Valid values:

Minimum: 0.0

Maximum: 314.15

Description: Trigger delay.

Parameter: int SP_TriggerEdge

SP_TriggerEdge

Direction: Down

Valid values:

0= falling

1= rising

Description: Trigger edge.

Parameter: int SP_TriggerMode

SP_TriggerMode

Direction: Down

Valid values:

0= single

1= measure

Description: Specifies if one laser shot should be done or single measuring should be started.

Parameter: double SA_TriggerDelay	SA_TriggerDelay
Direction: Up	
Unit: ms	
Valid values:	
Minimum: 0.0	
Maximum: 314.15	
Description: Trigger delay.	
Parameter: int SA_TriggerEdge	SA_TriggerEdge
Direction: Up	
Valid values:	
0= falling	
1= rising	
Description: Trigger edge.	
Parameter: int SA_TriggerMode	SA_TriggerMode
Direction: Up	
Valid values:	
0= single	
1= measure	
Description: Specifies if one laser shot should be done or single measuring should be started.	

9.3.1.3.5 Get_TriggerDelay (TD)

Get the behaviour of the trigger input.

Parameter: double SA_TriggerDelay	SA_TriggerDelay
Direction: Up	
Unit: ms	
Valid values:	
Minimum: 0.0	
Maximum: 314.15	
Description: Trigger delay.	
Parameter: int SA_TriggerEdge	SA_TriggerEdge
Direction: Up	
Valid values:	
0= falling	
1= rising	
Description: Trigger edge.	
Parameter: int SA_TriggerMode	SA_TriggerMode
Direction: Up	
Valid values:	
0= single	
1= measure	
Description: Specifies if one laser shot should be done or single measuring should be started.	

9.3.1.4 Interfaces

9.3.1.4.1 Set_Baudrate (BR)

Set the baudrate of the sensors serial interface. After receiving the answer of the command, the baudrate of the local serial interface is changed, too.

Attention! If baudrate is set to 9600, the command ID? (currently set to supported) will timeout.

If Set_Autostart is set to ID?, the sensor reboots cyclic and must be send back to manufacturer.

Attention! If baudrate is set to a baudrate, which the computer does not support (e.g. 230400 or 460800), it cannot be changed back and must be send back to manufacturer.

Attention! To avoid synchronisation problems, sending data by sensor should be disabled first (sensor command StopTracking).

Parameter: int SP_SensorBaudrate

SP_SensorBaudrate

Direction: Down

Valid values:

- 9600
- 19200
- 38400
- 57600
- 115200
- 230400
- 460800

Description: Sensor baudrate.

Parameter: int SA_SensorBaudrate

SA_SensorBaudrate

Direction: Up

Valid values:

- 9600
- 19200
- 38400
- 57600
- 115200
- 230400
- 460800

Description: Sensor baudrate.

9.3.1.4.2 Get_Baudrate (BR)

Get the baudrate of the sensors serial interface.

Parameter: int SA_SensorBaudrate

SA_SensorBaudrate

Direction: Up

Valid values:

- 9600
- 19200
- 38400
- 57600
- 115200
- 230400
- 460800

Description: Sensor baudrate.

9.3.1.5 Parameter management

9.3.1.5.1 Reset Parameters (PR)

Reset all parameters of sensor to factory defaults and return new parameters.

Parameter: int SA_MeasFrequency	SA_MeasFrequency
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 2000	
Description: Measure frequency.	
Parameter: double SA_TriggerDelay	SA_TriggerDelay
Direction: Up	
Unit: ms	
Valid values:	
Minimum: 0.0	
Maximum: 314.15	
Description: Trigger delay.	
Parameter: int SA_TriggerEdge	SA_TriggerEdge
Direction: Up	
Valid values:	
0= falling	
1= rising	
Description: Trigger edge.	
Parameter: int SA_Average	SA_Average
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 30000	
Description: Average value.	
Parameter: double SA_ScaleFactor	SA_ScaleFactor
Direction: Up	
Valid values:	
Minimum: -10.0	
Maximum: 10.0	
Description: Scaling factor.	
Parameter: double SA_WindowMin	SA_WindowMin
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Window minimum value.	
Parameter: double SA_WindowMax	SA_WindowMax
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Window maximum value.	

Parameter: double SA_Offset	SA_Offset
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Offset value which is set.	
Parameter: int SA_ErrorMode	SA_ErrorMode
Direction: Up	
Valid values:	
0= last valid value	
1= switch to bounds	
2= switch to negated bounds	
Description: Error mode.	
Parameter: double SA_Q1Start	SA_Q1Start
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Start value.	
Parameter: double SA_Q1Width	SA_Q1Width
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Width.	
Parameter: double SA_Q1Hysteresis	SA_Q1Hysteresis
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Hysteresis.	
Parameter: int SA_Q1Negation	SA_Q1Negation
Direction: Up	
Valid values:	
0= false	
1= true	
Description: Negation.	
Parameter: double SA_Q2Start	SA_Q2Start
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Start value.	
Parameter: double SA_Q2Width	SA_Q2Width
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Width.	

Parameter: double SA_Q2Hysteresis	SA_Q2Hysteresis
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Hysteresis.	
Parameter: int SA_Q2Negation	SA_Q2Negation
Direction: Up	
Valid values:	
0= false	
1= true	
Description: Negation.	
Parameter: double SA_RangeBegin	SA_RangeBegin
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Lower limit value.	
Parameter: double SA_RangeEnd	SA_RangeEnd
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Upper limit value.	
Parameter: int SA_SensorBaudrate	SA_SensorBaudrate
Direction: Up	
Valid values:	
9600	
19200	
38400	
57600	
115200	
230400	
460800	
Description: Sensor baudrate.	
Parameter: int SA_OutputFormat	SA_OutputFormat
Direction: Up	
Valid values:	
0= decimal	
1= hex	
2= binary	
Description: Output format of values.	
Parameter: int SA_OutputContent	SA_OutputContent
Direction: Up	
Valid values:	
0= Measuring value	
1= Measuring value, signal strength	
2= Measuring value, sensor temperature	
3= Measuring value, signal strength, sensor temperature	
Description: Get which data is transmitted by sensor.	

Parameter: int SA_TerminationChar	SA_TerminationChar
Direction: Up	
Valid values:	
0= <CRLF>	
1= <CR>	
2= <LF>	
3= <STX>	
4= <ETX>	
5= Tabulator	
6= Space	
7= Comma	
8= Colon	
9= Semicolon	
Description: Termination character.	
Parameter: int SA_SSIFormat	SA_SSIFormat
Direction: Up	
Valid values:	
0= binary	
1= grey code	
Description: SSI transmission format.	
Parameter: int SA_PilotLaser	SA_PilotLaser
Direction: Up	
Valid values:	
0= off	
1= on	
2= flashing (2 Hz)	
3= flashing (5 Hz)	
Description: Pilot laser behaviour.	
Parameter: int SA_AutostartCommand	SA_AutostartCommand
Direction: Up	
Valid values:	
0= ID	
1= DM	
2= DT	
3= DF	
4= VM	
5= VT	
6= TP	
7= HW	
8= PA	
9= MF	
10= TD	
11= SA	
12= SF	
13= MW	
14= OF	
15= SE	
16= Q1	
17= Q2	
18= QA	
19= BR	

20= SD
 21= TE
 22= BB
 23= AB
 24= SC
 25= PL
 26= AS
 -1= unknown

Description: Autostart command.

9.3.2 Measurement

9.3.2.1 General

9.3.2.1.1 Set_MeasFreq (MF)

Set the measure frequency of the sensor.

Parameter: int SP_MeasFrequency

SP_MeasFrequency

Direction: Down

Valid values:

Minimum: 1

Maximum: 2000

Description: Measure frequency.

Parameter: int SA_MeasFrequency

SA_MeasFrequency

Direction: Up

Valid values:

Minimum: 1

Maximum: 2000

Description: Measure frequency.

9.3.2.1.2 Get_MeasFreq (MF)

Get the measure frequency of the sensor.

Parameter: int SA_MeasFrequency

SA_MeasFrequency

Direction: Up

Valid values:

Minimum: 1

Maximum: 2000

Description: Measure frequency.

9.3.2.2 Measurement value processing

9.3.2.2.1 Set_ScaleFactor (SF)

Set the scaling factor how the sensor scale distance values.

Parameter: double SP_ScaleFactor

SP_ScaleFactor

Direction: Down

Valid values:

Minimum: -10.0

Maximum: 10.0

Description: Scaling factor.

9.3. Commands for ILR1191

Parameter: double SA_ScaleFactor SA_ScaleFactor
Direction: Up
Valid values:
Minimum: -10.0
Maximum: 10.0
Description: Scaling factor.

9.3.2.2.2 Get_ScaleFactor (SF)

Get the scaling factor how the sensor scale distance values.

Parameter: double SA_ScaleFactor SA_ScaleFactor
Direction: Up
Valid values:
Minimum: -10.0
Maximum: 10.0
Description: Scaling factor.

9.3.2.2.3 Set_Offset (OF)

Set the offset which is added by the sensor to distance values.

Parameter: double SP_Offset SP_Offset
Direction: Down
Valid values:
Minimum: -3.40282e+38 (-FLT_MAX)
Maximum: 3.40282e+38 (FLT_MAX)
Description: Offset value.

Parameter: double SA_Offset SA_Offset
Direction: Up
Valid values:
Minimum: -3.40282e+38 (-FLT_MAX)
Maximum: 3.40282e+38 (FLT_MAX)
Description: Offset value.

9.3.2.2.4 Get_Offset (OF)

Get the offset which is added by the sensor to distance values.

Parameter: double SA_Offset SA_Offset
Direction: Up
Valid values:
Minimum: -3.40282e+38 (-FLT_MAX)
Maximum: 3.40282e+38 (FLT_MAX)
Description: Offset value.

9.3.2.2.5 CurrentDistAsOffset (SO)

Set the current distance value as offset.

Parameter: double SA_Offset

SA_Offset

Direction: Up

Valid values:

Minimum: -3.40282e+38 (-FLT_MAX)

Maximum: 3.40282e+38 (FLT_MAX)

Description: Offset value which is set.

9.3.3 Data output

9.3.3.1 General

9.3.3.1.1 Set_OutputFormat (SD)

Set the output format how values are sent from sensor.

Parameter: int SP_OutputFormat

SP_OutputFormat

Direction: Down

Valid values:

0= decimal

1= hex

2= binary

Description: Output format of values.

Parameter: int SP_OutputContent

SP_OutputContent

Direction: Down

Valid values:

0= Measuring value

1= Measuring value, signal strength

2= Measuring value, sensor temperature

3= Measuring value, signal strength, sensor temperature

Description: Set which data is transmitted by sensor.

Parameter: int SA_OutputFormat

SA_OutputFormat

Direction: Up

Valid values:

0= decimal

1= hex

2= binary

Description: Output format of values.

Parameter: int SA_OutputContent

SA_OutputContent

Direction: Up

Valid values:

0= Measuring value

1= Measuring value, signal strength

2= Measuring value, sensor temperature

3= Measuring value, signal strength, sensor temperature

Description: Get which data is transmitted by sensor.

9.3.3.1.2 Get_OutputFormat (SD)

Get the output format how values are sent from sensor.

Parameter: int SA_OutputFormat SA_OutputFormat

Direction: Up

Valid values:

- 0= decimal
- 1= hex
- 2= binary

Description: Output format of values.

Parameter: int SA_OutputContent SA_OutputContent

Direction: Up

Valid values:

- 0= Measuring value
- 1= Measuring value, signal strength
- 2= Measuring value, sensor temperature
- 3= Measuring value, signal strength, sensor temperature

Description: Get which data is transmitted by sensor.

9.3.3.1.3 Set_TerminatingChar (TE)

Set the termination character of each measurement.

Parameter: int SP_TerminationChar SP_TerminationChar

Direction: Down

Valid values:

- 0= <CRLF>
- 1= <CR>
- 2= <LF>
- 3= <STX>
- 4= <ETX>
- 5= Tabulator
- 6= Space
- 7= Comma
- 8= Colon
- 9= Semicolon

Description: Termination character.

Parameter: int SA_TerminationChar SA_TerminationChar

Direction: Up

Valid values:

- 0= <CRLF>
- 1= <CR>
- 2= <LF>
- 3= <STX>
- 4= <ETX>
- 5= Tabulator
- 6= Space
- 7= Comma
- 8= Colon
- 9= Semicolon

Description: Termination character.

9.3.3.1.4 Get_TerminatingChar (TE)

Get the termination character of each measurement.

Parameter: int SA_TerminationChar

SA_TerminationChar

Direction: Up

Valid values:

- 0= <CRLF>
- 1= <CR>
- 2= <LF>
- 3= <STX>
- 4= <ETX>
- 5= Tabulator
- 6= Space
- 7= Comma
- 8= Colon
- 9= Semicolon

Description: Termination character.

9.3.3.1.5 Set_ErrorMode (SE)

Set the behaviour of digital and analog outputs in case of an error.

Parameter: int SP_ErrorMode

SP_ErrorMode

Direction: Down

Valid values:

- 0= last valid value
- 1= switch to bounds
- 2= switch to negated bounds

Description: Error mode.

Parameter: int SA_ErrorMode

SA_ErrorMode

Direction: Up

Valid values:

- 0= last valid value
- 1= switch to bounds
- 2= switch to negated bounds

Description: Error mode.

9.3.3.1.6 Get_ErrorMode (SE)

Get the behaviour of digital and analog outputs in case of an error.

Parameter: int SA_ErrorMode

SA_ErrorMode

Direction: Up

Valid values:

- 0= last valid value
- 1= switch to bounds
- 2= switch to negated bounds

Description: Error mode.

9.3.3.1.7 Set_MeasureWindow (MW)

Set a metrological range by definition of a starting point x and an end point as limits for output of measured values.

Parameter: double SP_WindowMin SP_WindowMin

Direction: Down

Valid values:

Minimum: -3.40282e+38 (-FLT_MAX)

Maximum: 3.40282e+38 (FLT_MAX)

Description: Window minimum value.

Parameter: double SP_WindowMax SP_WindowMax

Direction: Down

Valid values:

Minimum: -3.40282e+38 (-FLT_MAX)

Maximum: 3.40282e+38 (FLT_MAX)

Description: Window maximum value.

Parameter: double SA_WindowMin SA_WindowMin

Direction: Up

Valid values:

Minimum: -3.40282e+38 (-FLT_MAX)

Maximum: 3.40282e+38 (FLT_MAX)

Description: Window minimum value.

Parameter: double SA_WindowMax SA_WindowMax

Direction: Up

Valid values:

Minimum: -3.40282e+38 (-FLT_MAX)

Maximum: 3.40282e+38 (FLT_MAX)

Description: Window maximum value.

9.3.3.1.8 Get_MeasureWindow (MW)

Get a metrological range by definition of a starting point x and an end point as limits for output of measured values.

Parameter: double SA_WindowMin SA_WindowMin

Direction: Up

Valid values:

Minimum: -3.40282e+38 (-FLT_MAX)

Maximum: 3.40282e+38 (FLT_MAX)

Description: Window minimum value.

Parameter: double SA_WindowMax SA_WindowMax

Direction: Up

Valid values:

Minimum: -3.40282e+38 (-FLT_MAX)

Maximum: 3.40282e+38 (FLT_MAX)

Description: Window maximum value.

9.3.3.1.9 Set_AverageValue (SA)

Set the average value for block wise averaging.

Parameter: int SP_Average SP_Average

Direction: Down

Valid values:

Minimum: 1

Maximum: 30000

Description: Average value.

Parameter: int SA_Average SA_Average

Direction: Up

Valid values:

Minimum: 1

Maximum: 30000

Description: Average value.

9.3.3.1.10 Get_AverageValue (SA)

Get the average value for block wise averaging.

Parameter: int SA_Average SA_Average

Direction: Up

Valid values:

Minimum: 1

Maximum: 30000

Description: Average value.

9.3.3.2 Switching outputs

9.3.3.2.1 Set_Out1Parameters (Q1)

Set parameters of the switching outputs Q1.

Parameter: double SP_Q1Start SP_Q1Start

Direction: Down

Valid values:

Minimum: -3.40282e+38 (-FLT_MAX)

Maximum: 3.40282e+38 (FLT_MAX)

Description: Start value.

Parameter: double SP_Q1Width SP_Q1Width

Direction: Down

Valid values:

Minimum: -3.40282e+38 (-FLT_MAX)

Maximum: 3.40282e+38 (FLT_MAX)

Description: Width.

Parameter: double SP_Q1Hysteresis	SP_Q1Hysteresis
Direction: Down	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Hysteresis.	
Parameter: int SP_Q1Negation	SP_Q1Negation
Direction: Down	
Valid values:	
0= false	
1= true	
Description: Negation.	
Parameter: double SA_Q1Start	SA_Q1Start
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Start value.	
Parameter: double SA_Q1Width	SA_Q1Width
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Width.	
Parameter: double SA_Q1Hysteresis	SA_Q1Hysteresis
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Hysteresis.	
Parameter: int SA_Q1Negation	SA_Q1Negation
Direction: Up	
Valid values:	
0= false	
1= true	
Description: Negation.	

9.3.3.2.2 Get_Out1Parameters (Q1)

Get parameters of the switching outputs Q1.

Parameter: double SA_Q1Start	SA_Q1Start
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Start value.	

Parameter: double SA_Q1Width	SA_Q1Width
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Width.	
Parameter: double SA_Q1Hysteresis	SA_Q1Hysteresis
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Hysteresis.	
Parameter: int SA_Q1Negation	SA_Q1Negation
Direction: Up	
Valid values:	
0= false	
1= true	
Description: Negation.	

9.3.3.2.3 Set_Out2Parameters (Q2)

Set parameters of the switching outputs Q2.

Parameter: double SP_Q2Start	SP_Q2Start
Direction: Down	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Start value.	
Parameter: double SP_Q2Width	SP_Q2Width
Direction: Down	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Width.	
Parameter: double SP_Q2Hysteresis	SP_Q2Hysteresis
Direction: Down	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Hysteresis.	
Parameter: int SP_Q2Negation	SP_Q2Negation
Direction: Down	
Valid values:	
0= false	
1= true	
Description: Negation.	

9.3. Commands for ILR1191

Parameter: double SA_Q2Start	SA_Q2Start
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Start value.	
Parameter: double SA_Q2Width	SA_Q2Width
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Width.	
Parameter: double SA_Q2Hysteresis	SA_Q2Hysteresis
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Hysteresis.	
Parameter: int SA_Q2Negation	SA_Q2Negation
Direction: Up	
Valid values:	
0= false	
1= true	
Description: Negation.	

9.3.3.2.4 Get_Out2Parameters (Q2)

Get parameters of the switching outputs Q2.

Parameter: double SA_Q2Start	SA_Q2Start
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Start value.	
Parameter: double SA_Q2Width	SA_Q2Width
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Width.	
Parameter: double SA_Q2Hysteresis	SA_Q2Hysteresis
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Description: Hysteresis.	

Parameter: int SA_Q2Negation SA_Q2Negation
Direction: Up
Valid values:
 0= false
 1= true
Description: Negation.

9.3.3.3 Analog output

9.3.3.3.1 Set_AnalogOutLimits (QA)

Set parameters of the analog output QA .

Parameter: double SP_RangeBegin SP_RangeBegin
Direction: Down
Valid values:
Minimum: -3.40282e+38 (-FLT_MAX)
Maximum: 3.40282e+38 (FLT_MAX)
Description: Lower limit value.

Parameter: double SP_RangeEnd SP_RangeEnd
Direction: Down
Valid values:
Minimum: -3.40282e+38 (-FLT_MAX)
Maximum: 3.40282e+38 (FLT_MAX)
Description: Upper limit value.

Parameter: double SA_RangeBegin SA_RangeBegin
Direction: Up
Valid values:
Minimum: -3.40282e+38 (-FLT_MAX)
Maximum: 3.40282e+38 (FLT_MAX)
Description: Lower limit value.

Parameter: double SA_RangeEnd SA_RangeEnd
Direction: Up
Valid values:
Minimum: -3.40282e+38 (-FLT_MAX)
Maximum: 3.40282e+38 (FLT_MAX)
Description: Upper limit value.

9.3.3.3.2 Get_AnalogOutLimits (QA)

Get parameters of the analog output QA.

Parameter: double SA_RangeBegin SA_RangeBegin
Direction: Up
Valid values:
Minimum: -3.40282e+38 (-FLT_MAX)
Maximum: 3.40282e+38 (FLT_MAX)
Description: Lower limit value.

Parameter: double SA_RangeEnd SA_RangeEnd
Direction: Up
Valid values:
Minimum: -3.40282e+38 (-FLT_MAX)
Maximum: 3.40282e+38 (FLT_MAX)
Description: Upper limit value.

9.3.3.4 SSI

9.3.3.4.1 Set_FormatSSI (SC)

Set the transmission format of SSI output.

Parameter: int SP_SSIFormat SP_SSIFormat
Direction: Down
Valid values:
 0= binary
 1= grey code
Description: SSI transmission format.

Parameter: int SA_SSIFormat SA_SSIFormat
Direction: Up
Valid values:
 0= binary
 1= grey code
Description: SSI transmission format.

9.3.3.4.2 Get_FormatSSI (SC)

Get the transmission format of SSI output.

Parameter: int SA_SSIFormat SA_SSIFormat
Direction: Up
Valid values:
 0= binary
 1= grey code
Description: SSI transmission format.

10 Commands for ILD sensors

ILD1401 demo case, ILD1800 and ILD2000 are no longer supported.

10.1 Commands for ILD1401

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [RS232](#) (native).
- [TCP/IP](#) (additional, e.g. RS232/RS422 to TCP/IP comm server and RS232 high level interface).
- [IF2004_USB](#) (for sensor ILD1402 in compatibility mode).
- [IF2008](#) (for sensor ILD1402 in compatibility mode).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_Info](#) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib to scale data.

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor, from 0 to 4095.
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command [Get_Info](#)), error values are scaled depending of [IP_ScaleErrorValues](#).

10.1.1 General commands

10.1.1.1 General

10.1.1.1.1 Get_Info (INFO)

Retrieve some information about the sensor.

Parameter: String SA_ArticleNumber

SA_ArticleNumber

Direction: Up

Valid values:

Numeric value

Description: Article number of the sensor

Parameter: String SA_Option

SA_Option

Direction: Up

Valid values:

Numeric value

Description: Option of the sensor

Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the sensor	
Parameter: double SA_Range	SA_Range
Direction: Up	
Valid values:	
Minimum: 0.0	
Unit: mm	
Description: Range of the sensor	
Parameter: String SA_Softwareversion	SA_Softwareversion
Direction: Up	
Description: Software version of the sensor	
Parameter: String SA_Date	SA_Date
Direction: Up	
Description: Software release date of the sensor	
Parameter: int SA_OutputType	SA_OutputType
Direction: Up	
Valid values:	
0= analog	
1= RS232	
Description: Data output (only values, not answer) interface of the sensor	
Parameter: int SA_ErrorHandler	SA_ErrorHandler
Direction: Up	
Valid values:	
0= hold last value	
1= error values	
Description: If the sensor cannot measure values, it can output the last valid value or it can output an error value (only at analog output).	
Parameter: int SA_Median_OnOff	SA_Median_OnOff
Direction: Up	
Valid values:	
0= none	
1= Median 3	
Description: The sensor can perform averaging (Median over 3 values).	

10.1.1.2 Get_Version (VERSION)

Retrieve the sensor software version.

Parameter: String SA_Softwareversion	SA_Softwareversion
Direction: Up	
Description: Software version of the sensor.	

10.1.1.1.3 Reset_Boot (BOOT)

Resets the sensor. This command has no parameters.

10.1.2 Measurement

10.1.2.1 Set_Median (MEDIAN)

Set the internal averaging mode of the sensor.

Parameter: int SP_Median_OnOff

SP_Median_OnOff

Direction: Down

Valid values:

0= none

1= Median 3

Description: The sensor can perform averaging (Median over 3 values).

10.1.3 Data output

10.1.3.1 General

10.1.3.1.1 SaveLastMV (SAVELASTMV)

Specifies the error handling of sensor if it cannot measure values.

Parameter: int SP_ErrorHandler

SP_ErrorHandler

Direction: Down

Valid values:

0= hold last value

1= error values

Description: If the sensor cannot measure values, it can output the last valid value or it can output an error value (only at analog output).

10.1.3.1.2 Set_OutputChannel (OUTPUTCHANNEL)

Set the output channel of the sensor.

Parameter: int SP_OutputType

SP_OutputType

Direction: Down

Valid values:

0= analog

1= RS232

Description: Specifies data output (only values, not answer) interface of the sensor.

10.2 Commands for ILD1302/ILD1402

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [RS232](#) (native) for ILD1402 and (additional, e.g. [IF2001_USB](#) (RS422) and RS232 high level interface) for ILD1302.
- [IF2004](#) (native).
- [TCP/IP](#) (additional, e.g. RS232/RS422 to TCP/IP comm server and RS232 high level interface).
- [IF2004_USB](#) (native).
- [IF2008](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_CIMode](#) [Set_CIMode1x02](#) (if not active) and [Get_Settings](#) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib timeout checking, to calculate datarate and to scale data.

If second bit of [IP_AutomaticMode](#) is set (2), MEDAQLib calls sensor command [Dat_Out_On](#) and [Laser_On](#) at [OpenSensor](#).

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor, from 0 to 16383.
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command [Get_Settings](#), error values are scaled depending of [IP_ScaleErrorValues](#).

10.2.1 General commands

10.2.1.1 General

10.2.1.1.1 Get_Info (GET_INFO)

Retrieve some information about the sensor.

Parameter: String SA_Sensor SA_Sensor
Direction: Up
Description: Name of the sensor.

Parameter: String SA_SensorType SA_SensorType
Direction: Up
Description: Type of the sensor.

Parameter: String SA_ArticleNumber	SA_ArticleNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Article number of the sensor.	
Parameter: String SA_Option	SA_Option
Direction: Up	
Valid values:	
Numeric value	
Description: Option of the sensor.	
Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the sensor.	
Parameter: double SA_Range	SA_Range
Direction: Up	
Valid values:	
Minimum: 0.0	
Unit: mm	
Description: Range of the sensor.	
Parameter: String SA_Softwareversion	SA_Softwareversion
Direction: Up	
Description: Software version of the sensor.	
Parameter: String SA_BootLoaderVer	SA_BootLoaderVer
Direction: Up	
Description: Boot loader version of the sensor.	
Parameter: String SA_Date	SA_Date
Direction: Up	
Description: Software release date of the sensor.	
Parameter: int SA_OutputType	SA_OutputType
Direction: Up	
Valid values:	
0= current (4..20mA)	
1= RS422	
Description: Data output (only values, not answer) interface of the sensor.	
Parameter: int SA_ErrorHandler	SA_ErrorHandler
Direction: Up	
Valid values:	
0= hold last value	
1= error values	
2..99= hold last valid for n values	
Description: If the sensor cannot measure values, it can output the last valid value or it can output error values at analog interface.	

Parameter: int SA_AvType	SA_AvType
Direction: Up	
Valid values:	
0= moving	
1= Median	
Description: The averaging type.	
Parameter: int SA_MovingCount	SA_MovingCount
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 128	
Description: The moving averaging value, if AvType is moving.	
Parameter: int SA_MedianIndex	SA_MedianIndex
Direction: Up	
Valid values:	
3	
5	
7	
9	
Description: The Median value, if AvType is Median.	
Parameter: int SA_Speed	SA_Speed
Direction: Up	
Valid values:	
0= 1.5kHz	
1= 1.0kHz	
2= 750Hz	
3= 375Hz	
4= 50Hz	
Valid for sensor:	
ILD1402	
Description: The output speed of the sensor.	
Parameter: int SA_ASCII	SA_ASCII
Direction: Up	
Valid values:	
0= off (binary 2 bytes/value)	
1= on (ASCII 6 bytes/value)	
Description: Returns the mode the sensor is sending data (only values).	
Parameter: int SA_OutputMode	SA_OutputMode
Direction: Up	
Valid values:	
0= continuous	
1= timed	
2= triggered	
Description: The output mode of the sensor.	
Parameter: int SA_Keylock	SA_Keylock
Direction: Up	
Valid values:	
0= off (keys enabled)	
1= on (keys locked)	
2= auto (locked after 5 minutes)	
Description: The keypad state at the sensor.	

Parameter: int SA_SaveSettingsMode	SA_SaveSettingsMode
Direction: Up	
Valid values:	
0= temporary in RAM	
1= persistant in Flash	
Description: The mode if parameters should be temporay or stored persistant.	
Parameter: int SA_ExtInputMode	SA_ExtInputMode
Direction: Up	
Valid values:	
0= used for teaching	
1= used as trigger	
Description: Specifies if the external input is used for teaching or as trigger input.	
Parameter: int SA_PeakSearching	SA_PeakSearching
Direction: Up	
Valid values:	
0= global maximum	
1= first peak	
2= last peak	
Description: Specifies how the peak searching algorithm does work.	
Parameter: int SA_Threshold	SA_Threshold
Direction: Up	
Valid values:	
0= lower	
1= normal	
2= higher	
3= highest	
Description: Specifies the spectral threshold.	
Parameter: double SA_TeachValue1	SA_TeachValue1
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 16368.0	
Description: The lower teach limit.	
Parameter: double SA_TeachValue2	SA_TeachValue2
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 16368.0	
Description: The higher teach limit.	

10.2.1.1.2 Get_Settings (GET_SETTINGS)

Retrieve detailed information about the sensor.

Parameter: int SA_OutputType	SA_OutputType
Direction: Up	
Valid values:	
0= current (4..20mA)	
1= RS422	
Description: Data output (only values, not answer) interface of the sensor.	

Parameter: double SA_TeachValue1	SA_TeachValue1
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 16368.0	
Description: The lower teach limit.	
Parameter: double SA_TeachValue2	SA_TeachValue2
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 16368.0	
Description: The higher teach limit.	
Parameter: int SA_ErrorHandler	SA_ErrorHandler
Direction: Up	
Valid values:	
0= hold last value	
1= error values	
2..99= hold last valid for n values	
Description: If the sensor cannot measure values, it can output the last valid value or it can output error values at analog interface.	
Parameter: int SA_AvType	SA_AvType
Direction: Up	
Valid values:	
0= moving	
1= Median	
Description: The averaging type.	
Parameter: int SA_MovingCount	SA_MovingCount
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 128	
Description: The moving averaging value, if AvType is moving.	
Parameter: int SA_MedianIndex	SA_MedianIndex
Direction: Up	
Valid values:	
3	
5	
7	
9	
Description: The Median value, if AvType is Median.	
Parameter: int SA_Speed	SA_Speed
Direction: Up	
Valid values:	
0= 1.5kHz	
1= 1.0kHz	
2= 750Hz	
3= 375Hz	
4= 50Hz	
Valid for sensor:	
ILD1402	
Description: The output speed of the sensor.	

Parameter: int SA_SensorBaudrate	SA_SensorBaudrate
Direction: Up	
Valid values:	
0= 115200 Baud	
1= 57600 Baud	
2= 38400 Baud	
3= 19200 Baud	
4= 9600 Baud	
Description: The serial connection baudrate of the sensor.	
Parameter: int SA_ASCII	SA_ASCII
Direction: Up	
Valid values:	
0= off (binary 2 bytes/value)	
1= on (ASCII 6 bytes/value)	
Description: Returns the mode the sensor is sending data (only values).	
Parameter: int SA_OutputMode	SA_OutputMode
Direction: Up	
Valid values:	
0= continuous	
1= timed	
2= triggered	
Description: The output mode of the sensor.	
Parameter: int SA_OutputTime	SA_OutputTime
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 65535	
Unit: ms	
Description: Data output time of the sensor. It is used for timeout check.	
Parameter: int SA_Keylock	SA_Keylock
Direction: Up	
Valid values:	
0= off (keys enabled)	
1= on (keys locked)	
2= auto (locked after 5 minutes)	
Description: The keypad state at the sensor.	
Parameter: int SA_SaveSettingsMode	SA_SaveSettingsMode
Direction: Up	
Valid values:	
0= temporary in RAM	
1= persistant in Flash	
Description: The mode if parameters should be temporay or stored persistant.	
Parameter: int SA_ExtInputMode	SA_ExtInputMode
Direction: Up	
Valid values:	
0= used for teaching	
1= used as trigger	
Description: Specifies if the external input is used for teaching or as trigger input.	

Parameter: int SA_PeakSearching	SA_PeakSearching
Direction: Up	
Valid values:	
0= global maximum	
1= first peak	
2= last peak	
Description: Specifies how the peak searching algorithm does work.	
Parameter: int SA_Threshold	SA_Threshold
Direction: Up	
Valid values:	
0= lower	
1= normal	
2= higher	
3= highest	
Description: Specifies the spectral threshold.	
Parameter: double SA_Range	SA_Range
Direction: Up	
Valid values:	
Minimum: 0.0	
Unit: mm	
Description: Range of the sensor.	
Parameter: int SA_Reserved1	SA_Reserved1
Direction: Up	
Description: Reserved for further use.	
Parameter: int SA_Reserved2	SA_Reserved2
Direction: Up	
Description: Reserved for further use.	
Parameter: int SA_Reserved3	SA_Reserved3
Direction: Up	
Description: Reserved for further use.	
Parameter: int SA_Reserved4	SA_Reserved4
Direction: Up	
Description: Reserved for further use.	

10.2.1.3 Reset_Boot (RESET_BOOT)

Resets the sensor.

If first bit of [IP_AutomaticMode](#) is set (1), [Get_Settings](#) is called automatically after this command. Otherwise, you have to call it manually.

10.2.1.2 User level

10.2.1.2.1 Set_KeyLock (SET_KEYLOCK)

Locks/Unlocks the keypad of sensor.

Parameter: int SP_Keylock	SP_Keylock
Direction: Down	
Valid values:	
0= off (keys enabled)	
1= on (keys locked)	
2= auto (locked after 5 minutes)	
Description: The keypad state at the sensor.	

10.2.1.3 Interfaces

10.2.1.3.1 Set_Baudrate (SET_BAUDRATE)

Set the baudrate of the serial interface of sensor. After receiving the answer of the command, the baudrate of the local serial interface is changed, too.

Attention! To avoid synchronisation problems, sending data by sensor should be disabled first (sensor command [Dat_Out_Off](#)).

Parameter: int SP_SensorBaudrate

SP_SensorBaudrate

Direction: Down

Valid values:

- 0= 115200 Baud
- 1= 57600 Baud
- 2= 38400 Baud
- 3= 19200 Baud
- 4= 9600 Baud

Description: The serial connection baudrate of the sensor.

Parameter: int CP_InterruptDataTransfer

CP_InterruptDataTransfer

Direction: Down

Valid values:

- 0= false
- 1= true

Default: 0

Description: After switch baudrate, synchronisation may be impossible if serial line has high load. Enabling this parameter resolves the problem by disabling data transfer first ([Dat_Out_Off](#)) and enabling it again ([Dat_Out_On](#)) at end.

10.2.1.4 Parameter management

10.2.1.4.1 Set_Default (SET_DEFAULT)

Resets the sensor to factory settings.

If first bit of [IP_AutomaticMode](#) is set (1), [Get_Settings](#) is called automatically after this command. Otherwise, you have to call it manually.

10.2.1.4.2 Set_SaveSettingsMode (SET_SAVE_SETTINGS_MODE)

Set the save settings mode of sensor.

Parameter: int SP_SaveSettingsMode

SP_SaveSettingsMode

Direction: Down

Valid values:

- 0= temporary in RAM
- 1= persistant in Flash

Description: The mode if parameters should be temporary or stored persistant.

10.2.1.5 Internal controller commands

10.2.1.5.1 Laser_Off (LASER_OFF)

Switch the laser off.

10.2.1.5.2 Laser_On (LASER_ON)

Switch the laser on.

10.2.1.5.3 Set_ExtInputMode (SET_EXT_INPUT_MODE)

Set the mode of external input at sensor.

Parameter: int SP_ExtInputMode

SP_ExtInputMode

Direction: Down

Valid values:

- 0= used for teaching
- 1= used as trigger

Description: Specifies if the external input is used for teaching or as trigger input.

10.2.1.5.4 Get_CI_Mode (GET_CI_MODE)

Retrieve the sensor mode.

Only available at ILD1402.

Parameter: int SA_CI_Mode

SA_CI_Mode

Direction: Up

Valid values:

- 0= ILD1401 compatibility mode
- 1= ILD1402 mode

Description: Sensor mode.

10.2.1.5.5 Set_CI_Mode_1401 (SET_CI_MODE_1401)

Set the sensor in compatibility mode for ILD1401.

Only available at ILD1402.

No other commands expect Set_CI_Mode_1x02 will work now.

Attention! If the interface is [IF2004](#), the sensor cannot be accessed any longer, because IF2004 does not support 38400 Baud (Baudrate of ILD1401).

10.2.1.5.6 Set_CI_Mode_1402 (SET_CI_MODE_1402)

Set the sensor back to ILD1402 mode.

Only available at ILD1402.

If first bit of [IP_AutomaticMode](#) is set (1), [Get_Settings](#) is called automatically after this command. Otherwise, you have to call it manually.

10.2.1.5.7 Set_OperationMode (SET_MODE)

Set the operation mode at sensor.
Only available at ILD1402 Option 207.

Parameter: int SP_OperationMode SP_OperationMode
Direction: Down
Valid values:
 1 = Measure mode
 2 = User mode
Description: The operation mode.

10.2.1.5.8 Set_LaserDiode1 (SET_LD1)

Set the power of laser diode 1 at sensor.
Only available at ILD1402 Option 207.

Parameter: int SP_LaserDiode1 SP_LaserDiode1
Direction: Down
Valid values:
Minimum: 0
Maximum: 25
Description: The power of laser diode 1.

10.2.1.5.9 Set_LaserDiode2 (SET_LD2)

Set the power of laser diode 2 at sensor.
Only available at ILD1402 Option 207.

Parameter: int SP_LaserDiode2 SP_LaserDiode2
Direction: Down
Valid values:
Minimum: 0
Maximum: 25
Description: The power of laser diode 2.

10.2.1.5.10 Get_LaserDiodeError (GET_LDERROR)

Get the error state of laser diode at sensor.
Only available at ILD1402 Option 207.

Parameter: int SA_LaserDiodeError SA_LaserDiodeError
Direction: Up
Valid values:
 Bit combination of eight bits
Description: The error state of laser diode.

10.2.1.5.11 Get_CurrentOfMonitor (GET_IMON)

Get the current at monitor diode.

Only available at ILD1402 Option 207.

Parameter: int SA_CurrentOfMonitor

SA_CurrentOfMonitor

Direction: Up

Valid values:

Minimum: 0

Maximum: 2147483647 (INT_MAX)

Unit: nA

Description: Current at monitor diode.

10.2.2 Measurement

10.2.2.1 Set_Speed (SET_SCANRATE)

Set the data acquisition speed of the sensor.

Only available at ILD1402.

Parameter: int SP_Speed

SP_Speed

Direction: Down

Valid values:

0= 1.5kHz

1= 1.0kHz

2= 750Hz

3= 375Hz

4= 50Hz

Description: The output speed of the sensor.

10.2.2.2 Set_PeakSearching (SET_PEAKSEARCHING)

Set the mode of external input at sensor.

Parameter: int SP_PeakSearching

SP_PeakSearching

Direction: Down

Valid values:

0= global maximum

1= first peak

2= last peak

Description: Specifies how the peak searching algorithm does work.

10.2.2.3 Get_Video

Get recent video signal from sensor.

Parameter: Binary data SA_VideoSignal

SA_VideoSignal

Direction: Up

Valid values:

256 bytes, convertible to 128 words.

Description: Raw video signal

10.2.2.4 Set_Threshold (SET_THRESHOLD)

Set the spectral threshold of sensor.

Parameter: int SP_Threshold SP_Threshold

Direction: Down

Valid values:

- 0= lower
- 1= normal
- 2= higher
- 3= highest

Description: Specifies the spectral threshold.

10.2.2.5 Set_Av (SET_AV)

Set averaging type and value of sensor.

Parameter: int SP_AvType SP_AvType

Direction: Down

Valid values:

- 0= moving
- 1= Median

Description: The averaging type.

Parameter: int SP_MovingCount SP_MovingCount

Direction: Down

Valid values:

- Minimum: 1
- Maximum: 128

Description: The moving averaging value, if AvType is moving.

Parameter: int SP_MedianIndex SP_MedianIndex

Direction: Down

Valid values:

- 3
- 5
- 7
- 9

Description: The Median value, if AvType is Median.

10.2.3 Data output

10.2.3.1 General

10.2.3.1.1 Dat_Out_Off (DAT_OUT_OFF)

Switch off data output from sensor.

10.2.3.1.2 Dat_Out_On (DAT_OUT_ON)

Switch on data output from sensor.

10.2.3.1.3 Set_ErrorHandler (SET_ANALOG_ERROR_HANDLER)

Set the behaviour on invalid values at analog interface sensor.

Parameter: int SP_ErrorHandler

SP_ErrorHandler

Direction: Down

Valid values:

0= hold last value

1= error values

2..99= hold last valid for n values

Description: If the sensor cannot measure values, it can output the last valid value or it can output error values at analog interface.

10.2.3.1.4 ASCII_Output (ASCII_OUTPUT)

Set digital data transfer (only values, no sensor answer) to ASCII or binary.

Parameter: int SP_ASCII

SP_ASCII

Direction: Down

Valid values:

0= off (binary 2 bytes/value)

1= on (ASCII 6 bytes/value)

Description: Returns the mode the sensor is sending data (only values).

10.2.3.1.5 Set_OutputType (SET_OUTPUT_CHANNEL)

Set the output type of sensor.

Parameter: int SP_OutputType

SP_OutputType

Direction: Down

Valid values:

0= current (4..20mA)

1= RS422

Description: Data output (only values, not answer) interface of the sensor.

10.2.3.1.6 Set_OutputMode (SET_OUTPUTMODE)

Set the output mode of sensor.

Parameter: int SP_OutputMode

SP_OutputMode

Direction: Down

Valid values:

0= continuous

1= timed

2= triggered

Description: Data output mode of the sensor.

10.2.3.1.7 Set_OutputTime (SET_OUTPUTTIME_MS)

Set the output time of sensor.

Parameter: int SP_OutputTime SP_OutputTime
Direction: Down
Valid values:
Minimum: 1
Maximum: 65535
Unit: ms
Description: Data output time of the sensor.

10.2.3.2 Analog output

10.2.3.2.1 Set_TeachValue (SET_TEACH_VALUE)

Set the teaching values at sensor.

Parameter: double SP_TeachValue1 SP_TeachValue1
Direction: Down
Valid values:
Minimum: 0.0
Maximum: 16368.0
Description: The lower teach limit.

Parameter: double SP_TeachValue2 SP_TeachValue2
Direction: Down
Valid values:
Minimum: 0.0
Maximum: 16368.0
Description: The higher teach limit.

10.2.3.2.2 Reset_TeachValue (RESET_TEACH_VALUE)

Reset the teaching values at sensor.

10.3 Commands for ILD1320/ILD1420

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [RS232](#) (additional, e.g. IF2001_USB (RS422) and RS232 high level interface).
- [IF2004](#) (native).
- [IF2004_USB](#) (native).
- [IF2008](#) (native).

If first bit of `IP_AutomaticMode` is set (1), MEDAQLib calls automatically sensor command `Get_AllParameters` (SP_Additional= 1) after `OpenSensor`.

Otherwise, you have to call it manually to allow MEDAQLib timeout checking, to calculate datarate, to interpret and scale data and to assign values. If second bit of `IP_AutomaticMode` is set (2), MEDAQLib calls optionally sensor command `Set_DataOutInterface`, `Get_LaserPower` and optionally `Set_LaserPower` at `OpenSensor`.

Meaning of raw and scaled values (function `Poll` and `TransferData`):

- Raw values are as it comes directly from sensor.
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command `Get_AllParameters` (SP_Additional= 1)).

The values of one data frame are filled in the arrays one after another. Each array always starts with a new data frame.

10.3.1 General commands

10.3.1.1 General

10.3.1.1.1 Get_Help (HELP)

Retrieve a help text from sensor for a specific command.

Parameter: String SP_Command SP_Command
Direction: Down
Valid values:
 "" (empty string, means general help)
 or any command name
Description: Name of the command.

Parameter: String SA_HelpText SA_HelpText
Direction: Up
Description: Help text to the command.

10.3.1.1.2 Get_Info (GETINFO)

Retrieve information about the sensor.

Parameter: String SA_Sensor SA_Sensor
Direction: Up
Description: Name of the sensor.

Parameter: String SA_SerialNumber SA_SerialNumber
Direction: Up
Valid values:
 Numeric value
Description: Serial number of the sensor.

Parameter: String SA_Option	SA_Option
Direction: Up	
Valid values:	
Numeric value	
Description: Option of the sensor.	
Parameter: String SA_ArticleNumber	SA_ArticleNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Article number of the sensor.	
Parameter: String SA_CableHead	SA_CableHead
Direction: Up	
Description: Sensor cable type.	
Parameter: double SA_Range	SA_Range
Direction: Up	
Valid values:	
Minimum: 0.0	
Unit: mm	
Description: Range of the sensor.	
Parameter: String SA_Softwareversion	SA_Softwareversion
Direction: Up	
Description: Software version of firmware in the sensor.	
Parameter: String SA_HardwareRevision	SA_HardwareRevision
Direction: Up	
Valid values:	
Numeric value	
Description: Revision index of the controller board.	
Parameter: String SA_BuildID	SA_BuildID
Direction: Up	
Description: Build ID	
Parameter: String SA_BuildTimestamp	SA_BuildTimestamp
Direction: Up	
Description: Build timestamp	
Parameter: String SA_BootVersion	SA_BootVersion
Direction: Up	
Description: Boot version	

10.3.1.1.3 Get_OutputInfo_RS422 (GETOUTINFO_RS422)

Retrieve information which data is output at RS422 interface.

Parameter: int SA_OutputVideoRaw_RS422	SA_OutputVideoRaw_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if raw video signal is transmitted.	
This parameter is only available at ILD1420, not ILD1320.	

Parameter: int SA_OutputAdditionalShutterTime_RS422	SA_OutputAdditionalShutterTime_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	
Parameter: int SA_OutputAdditionalCounter_RS422	SA_OutputAdditionalCounter_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SA_OutputAdditionalTimestampLo_RS422	SA_OutputAdditionalTimestampLo_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp (lower 16 bit) is transmitted.	
Parameter: int SA_OutputAdditionalTimestampHi_RS422	SA_OutputAdditionalTimestampHi_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp (upper 16 bit) is transmitted.	
Parameter: int SA_OutputAdditionalIntensity1_RS422	SA_OutputAdditionalIntensity1_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity 1 is transmitted.	
Parameter: int SA_OutputAdditionalDistanceRaw1_RS422	SA_OutputAdditionalDistanceRaw1_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if uncalibrated distance 1 is transmitted.	
Parameter: int SA_OutputDistance1_RS422	SA_OutputDistance1_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if distance 1 is transmitted.	
Parameter: int SA_OutputAdditionalState_RS422	SA_OutputAdditionalState_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	

10.3.1.1.4 Set_Unit (UNIT)

Set the unit for configuration and display in the web diagram.

Parameter: int SP_DisplayUnit SP_DisplayUnit
Direction: Down
Valid values:
 0= mm
 1= Inch
Description: Unit.

10.3.1.1.5 Get_Unit (UNIT)

Get the unit for configuration and display in the web diagram.

Parameter: int SA_DisplayUnit SA_DisplayUnit
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= mm
 1= Inch
Description: Unit.

10.3.1.1.6 Reset_Boot (RESET)

Resets the sensor.

10.3.1.1.7 Reset_Counter (RESETCNT)

Resets sensor counter values.

Parameter: int SP_ResetTimestamp SP_ResetTimestamp
Direction: Down
Valid values:
 0= No
 1= Yes
Description: Reset timestamp value.

Parameter: int SP_ResetMeasCounter SP_ResetMeasCounter
Direction: Down
Valid values:
 0= No
 1= Yes
Description: Reset counter value.

10.3.1.1.8 Set_Keylock (KEYLOCK)

Set key lock for sensor.

Parameter: int SP_Keylock SP_Keylock

Direction: Down

Valid values:

0= Inactive (NONE)

1= Active

2= Automatic (AUTO)

Description: Keylock.

Parameter: int SP_KeylockTime SP_KeylockTime

Direction: Down

Valid values:

Minimum: 1

Maximum: 60

Unit: Minutes

Description: Keylock time (only used at automatic keylock).

10.3.1.1.9 Get_Keylock (KEYLOCK)

Get key lock for sensor.

Parameter: int SA_Keylock SA_Keylock

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= Inactive (NONE)

1= Active

2= Automatic (AUTO)

Description: Keylock.

Parameter: int SA_KeylockTime SA_KeylockTime

Direction: Up

Valid values:

Minimum: 0

Maximum: 60

Unit: Minutes

Description: Keylock time (only available at automatic keylock).

Parameter: int SA_KeylockState SA_KeylockState

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= Inactive (IS_INACTIVE)

1= Active (IS_ACTIVE)

Description: Actual keylock state (only available at automatic keylock).

10.3.1.1.10 Set_KeyFunction (KEYFUNC)

Set key function for sensor.

Parameter: int SP_KeyFunction SP_KeyFunction
Direction: Down
Valid values:
 0= Key has no function (NONE)
 1= Use key to master the measuring value (MASTER)
 2= Use key to teach the analog output (TEACH)
Description: Key function

10.3.1.1.11 Get_KeyFunction (KEYFUNC)

Get key function for sensor.

Parameter: int SA_KeyFunction SA_KeyFunction
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= Key has no function (NONE)
 1= Use key to master the measuring value (MASTER)
 2= Use key to teach the analog output (TEACH)
Description: Key function

10.3.1.1.12 Set_Echo (ECHO)

Set echo for sensor commands.

Parameter: int SP_Echo SP_Echo
Direction: Down
Valid values:
 0= Off
 1= On
Description: Echo mode.

10.3.1.1.13 Get_Echo (ECHO)

Get the echo mode.

Parameter: int SA_Echo SA_Echo
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= Off
 1= On
Description: Echo mode.

10.3.1.1.14 Get_AllParameters (PRINT)

Get all parameters from sensor.

Parameter: int SP_Additional	SP_Additional
Direction: Down	
Valid values:	
0= No	
1= Yes	
Description:	If set, additional information about sensor is output.
Parameter: int SA_UserLevel	SA_UserLevel
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= User	
1= Professional	
Description:	Actual user level.
Parameter: int SA_DefaultUser	SA_DefaultUser
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= User	
1= Professional	
Description:	Default user level.
Parameter: int SA_Echo	SA_Echo
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Off	
1= On	
Description:	Echo mode.
Parameter: int SA_DisplayUnit	SA_DisplayUnit
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= mm	
1= Inch	
Description:	Unit.
Parameter: int SA_Keylock	SA_Keylock
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Inactive (NONE)	
1= Active	
2= Automatic (AUTO)	
Description:	Keylock.

Parameter: int SA_KeylockTime	SA_KeylockTime
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 60	
Unit: Minutes	
Description: Keylock time (only available at automatic keylock).	
Parameter: int SA_KeylockState	SA_KeylockState
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Inactive (IS_INACTIVE)	
1= Active (IS_ACTIVE)	
Description: Actual keylock state (only available at automatic keylock).	
Parameter: int SA_KeyFunction	SA_KeyFunction
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Key has no function (NONE)	
1= Use key to master the measuring value (MASTER)	
2= Use key to teach the analog output (TEACH)	
Description: Key function	
Parameter: int SA_TriggerMode	SA_TriggerMode
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Edge	
2= Level (PULSE)	
3= Software	
Description: Trigger mode.	
Parameter: int SA_TriggerMoment	SA_TriggerMoment
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Input	
1= Output	
Description: Trigger moment.	
This parameter is only available at ILD1420, not ILD1320.	
Parameter: int SA_TriggerCount	SA_TriggerCount
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 16383	
Description: Number of values to measure. 0 means no trigger (NONE),	
16383 means endless measurement (INFINITE).	

Parameter: int SA_SensorBaudrate	SA_SensorBaudrate
Direction: Up	
Valid values:	
1000000	
921600	
691200	
460800	
256000	
230400	
128000	
115200	
56000	
19200	
9600	
Unit: Baud	
Description: Baudrate of sensor.	
Parameter: int SA_ApplicationLanguage	SA_ApplicationLanguage
Direction: Up	
Valid values:	
0 = English (EN)	
1 = German (DE)	
Description: Language of web interface.	
Parameter: int SA_MultiFunctionInputMode	SA_MultiFunctionInputMode
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = MFI has no function (NONE)	
1 = Use MFI to master the measuring value (MASTER)	
2 = Use MFI to teach the analog output (TEACH)	
3 = Use MFI to trigger the measuring process (TRIGGER)	
Description: Specifies how to use the multi function input.	
Parameter: int SA_MultiFunctionInputLevel	SA_MultiFunctionInput- Level
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = High (HTL_HIGH)	
1 = Low (HTL_LOW)	
Description: Multi function input level.	
Parameter: int SA_InitialChartType	SA_InitialChartType
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Measure (MEAS)	
1 = Video signal (VIDEO)	
2 = Automatic (AUTO)	
Description: Initial type of the chart at web interface.	

Parameter: int SA_ChartType SA_ChartType

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = Measure (MEAS)
- 1 = Video signal (VIDEO)

Description: Type of the chart at web interface.

Parameter: int SA_TargetMode SA_TargetMode

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = Standard Target (STANDARD)
- 1 = Multi-Surface Target (MULTISURFACE)
- 2 = Light Penetration Target (PENETRATION)

Description: Target mode.

This parameter is only available at ILD1420, not ILD1320.

Parameter: int SA_MeasurePeak SA_MeasurePeak

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = Greatest Amplitude (DISTA)
- 1 = First Peak (DIST1)
- 2 = Last Peak (DISTL)

Description: Peak to measure.

Parameter: double SA_Measrate SA_Measrate

Direction: Up

Valid values:

- 0.25
- 0.5
- 1.0
- 2.0
- 4.0

Unit: kHz

Description: Samplerate of measurement.

4.0 kHz is only available at ILD1420, not ILD1320.

Parameter: int SA_LaserPower SA_LaserPower

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = Full
- 1 = Off

Description: Laser power.

Parameter: int SA_ROIStart SA_ROIStart

Direction: Up

Valid values:

Minimum: 0
Maximum: 511

Unit: Pixel

Description: First position on CCD.

This parameter is only available at ILD1420, not ILD1320.

Parameter: int SA_ROIEnd SA_ROIEnd

Direction: Up

Valid values:

Minimum: 0

Maximum: 511

Unit: Pixel

Description: Last position on CCD.

This parameter is only available at ILD1420, not ILD1320.

Parameter: int SA_AveragingType SA_AveragingType

Direction: Up

Valid values:

-1 = Unknown parameter value from sensor

0 = None

1 = Moving average (MOVING)

2 = Recursive averaging (RECURSIVE)

3 = Median

Description: Averaging type.

This parameter is only available at ILD1420, not ILD1320.

Parameter: int SA_MovingCount SA_MovingCount

Direction: Up

Valid values:

2

4

8

16

32

64

128

Description: Number of value for the averaging window.

This parameter is only available at moving average.

This parameter is only available at ILD1420, not ILD1320.

Parameter: int SA_RecursiveCount SA_RecursiveCount

Direction: Up

Valid values:

Minimum: 2

Maximum: 32768

Description: Number of values for recursive averaging.

This parameter is only available at recursive average.

This parameter is only available at ILD1420, not ILD1320.

Parameter: int SA_MedianCount SA_MedianCount

Direction: Up

Valid values:

3

5

7

9

Description: Number of values to build median.

This parameter is only available at median.

This parameter is only available at ILD1420, not ILD1320.

Parameter: int SA_Master	SA_Master
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = no (NONE)	
1 = yes (MASTER=	
Description: Specifies if mastering is active.	
Parameter: double SA_MasterValue	SA_MasterValue
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 2 * measuring range	
Unit: mm	
Description: Master value	
Parameter: int SA_DataOutInterface	SA_DataOutInterface
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = None	
1 = RS422	
6 = Analog	
Description: Active interface for data output.	
Parameter: int SA_Resampling	SA_Resampling
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 3000000	
Description: Resampling value.	
This parameter is only available at ILD1420, not ILD1320.	
Parameter: int SA_ResampleRS422	SA_ResampleRS422
Direction: Up	
Valid values:	
0 = no	
1 = yes	
Description: RS422 output is resampled.	
This parameter is only available at ILD1420, not ILD1320.	
Parameter: int SA_ResampleAnalog	SA_ResampleAnalog
Direction: Up	
Valid values:	
0 = no	
1 = yes	
Description: Analog output is resampled.	
This parameter is only available at ILD1420, not ILD1320.	
Parameter: int SA_HoldLastValid	SA_HoldLastValid
Direction: Up	
Valid values:	
Minimum: -1	
Maximum: 1024	
Description: Values to replace by last valid value. -1 means no value to hold,	
0 means never output an error value (always hold last valid value).	

Parameter: int SA_OutputDistance1_RS422	SA_OutputDistance1_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if distance 1 is transmitted.	
Parameter: int SA_OutputAdditionalIntensity1_RS422	SA_OutputAdditionalIntensity1_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity 1 is transmitted.	
Parameter: int SA_OutputAdditionalDistanceRaw1_RS422	SA_OutputAdditionalDistanceRaw1_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if uncalibrated distance 1 is transmitted.	
Parameter: int SA_OutputAdditionalShutterTime_RS422	SA_OutputAdditionalShutterTime_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	
Parameter: int SA_OutputAdditionalCounter_RS422	SA_OutputAdditionalCounter_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SA_OutputAdditionalTimestamp_RS422	SA_OutputAdditionalTimestamp_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SA_OutputAdditionalIntensity_RS422	SA_OutputAdditionalIntensity_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity is transmitted.	
Parameter: int SA_OutputAdditionalState_RS422	SA_OutputAdditionalState_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	

Parameter: int SA_OutputAdditionalDistanceRaw_RS422	SA_OutputAdditionalDistanceRaw_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if uncalibrated distance is transmitted.	
Parameter: int SA_OutputVideoRaw_RS422	SA_OutputVideoRaw_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if raw video signal is transmitted.	
This parameter is only available at ILD1420, not ILD1320.	
Parameter: int SA_ErrorOutput1	SA_ErrorOutput1
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Output trigger disabled (NONE)	
1= No valid distance (no peak found, out of range) (DIST)	
2= Distance is out of scaled analog range (TEACH)	
3= Distance is above set threshold (L1)	
Description: Condition for error output.	
Value 2 and 3 are only available at special option 200	
Parameter: int SA_DataSource	SA_DataSource
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Distance 1	
Description: Data source to be checked.	
Only available at option 200.	
Parameter: double SA_UpperLimit	SA_UpperLimit
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 2* measuring range	
Unit: mm	
Description: Upper limit.	
Only available at option 200.	
Parameter: double SA_ErrorHysteresis	SA_ErrorHysteresis
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 2* measuring range	
Unit: mm	
Description: Error hysteresis.	

Parameter: int SA_ErrorOutHoldTime	SA_ErrorOutHoldTime
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 1000	
Unit: ms	
Description: Minimum hold period.	
Parameter: int SA_ErrorLevelOut1	SA_ErrorLevelOut1
Direction: Up	
Valid values:	
0= NPN	
1= PNP	
2= Push-Pull (PUSHPULL)	
3= Push-Pull negated (PUSHPULLNEG)	
Description: Error level for out 1.	
Parameter: int SA_AnalogScaleMode	SA_AnalogScaleMode
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Standard	
1= Two point (TWOPOINT)	
Description: Analog scale mode.	
Parameter: double SA_MinValue	SA_MinValue
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: +2* measuring range	
Unit: mm	
Description: Value which represents lowest voltage/current (at two point scaling).	
Parameter: double SA_MaxValue	SA_MaxValue
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: +2* measuring range	
Unit: mm	
Description: Value which represents highest voltage/current (at two point scaling).	
Parameter: String SA_Sensor	SA_Sensor
Direction: Up	
Description: Name of the sensor.	
Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the sensor.	

Parameter: String SA_Option	SA_Option
Direction: Up	
Valid values:	
Numeric value	
Description: Option of the sensor.	
Parameter: String SA_ArticleNumber	SA_ArticleNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Article number of the sensor.	
Parameter: String SA_CableHead	SA_CableHead
Direction: Up	
Description: Sensor cable type.	
Parameter: double SA_Range	SA_Range
Direction: Up	
Valid values:	
Minimum: 0.0	
Unit: mm	
Description: Range of the sensor.	
Parameter: String SA_Softwareversion	SA_Softwareversion
Direction: Up	
Description: Software version of firmware in the sensor.	
Parameter: String SA_HardwareRevision	SA_HardwareRevision
Direction: Up	
Valid values:	
Numeric value	
Description: Revision index of the controller board.	
Parameter: String SA_BuildID	SA_BuildID
Direction: Up	
Description: Build ID	
Parameter: String SA_BuildTimestamp	SA_BuildTimestamp
Direction: Up	
Description: Build timestamp	
Parameter: String SA_BootVersion	SA_BootVersion
Direction: Up	
Description: Boot version	
Parameter: String SA_CurrentName	SA_CurrentName
Direction: Up	
Description: Name of current (active) setting.	
Parameter: String SA_SettingNames	SA_SettingNames
Direction: Up	
Description: List (separated by new line).	
Parameter: String SA_PresetNames	SA_PresetNames
Direction: Up	
Description: List (separated by new line).	

Parameter: int SA_Automatic SA_Automatic

Direction: Up

Valid values:

0= no

1= yes

Description: Automatic selection.

Parameter: String SA_InitialName SA_InitialName

Direction: Up

Description: Name of setting.

Parameter: int SA_PresetMode SA_PresetMode

Direction: Up

Valid values:

0= Static

1= Balanced

2= Dynamic

3= None (if no preset setting is active)

Description: Preset mode.

10.3.1.2 User level

10.3.1.2.1 Logout (LOGOUT)

Change user level to user.

10.3.1.2.2 Login (LOGIN)

Change user level to professional.

Parameter: String SP_Password SP_Password

Direction: Down

Description: Valid password to login.

10.3.1.2.3 Get_UserLevel (GETUSERLEVEL)

Retrieve actual user level.

Parameter: int SA_UserLevel SA_UserLevel

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= User

1= Professional

Description: Actual user level.

10.3.1.2.4 Set_DefaultUser (STDUSER)

Set the default user level after booting the system.

Parameter: int SP_DefaultUser SP_DefaultUser
Direction: Down
Valid values:
 0= User
 1= Professional
Description: Default user level.

10.3.1.2.5 Get_DefaultUser (STDUSER)

Get the default user level after booting the system.

Parameter: int SA_DefaultUser SA_DefaultUser
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= User
 1= Professional
Description: Default user level.

10.3.1.2.6 Set_Password (PASSWD)

Change the password for login.

Parameter: String SP_OldPassword SP_OldPassword
Direction: Down
Description: Old password.

Parameter: String SP_NewPassword SP_NewPassword
Direction: Down
Description: New password.

10.3.1.3 Triggering

10.3.1.3.1 Set_TriggerMode (TRIGGER)

Set the trigger mode.

Parameter: int SP_TriggerMode SP_TriggerMode
Direction: Down
Valid values:
 0= None
 1= Edge
 2= Level (PULSE)
 3= Software
Description: Trigger mode.

10.3.1.3.2 Get_TriggerMode (TRIGGER)

Get the active trigger mode.

Parameter: int SA_TriggerMode

SA_TriggerMode

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= Edge
- 2= Level (PULSE)
- 3= Software

Description: Trigger mode.

10.3.1.3.3 Set_TriggerMoment (TRIGGERAT)

Set the trigger time.

This command is only available at ILD1420, not ILD1320.

Parameter: int SP_TriggerMoment

SP_TriggerMoment

Direction: Down

Valid values:

- 0= Input
- 1= Output

Description: Trigger moment.

10.3.1.3.4 Get_TriggerMoment (TRIGGERAT)

Get the active trigger time.

This command is only available at ILD1420, not ILD1320.

Parameter: int SA_TriggerMoment

SA_TriggerMoment

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Input
- 1= Output

Description: Trigger moment.

10.3.1.3.5 Set_TriggerCount (TRIGGERCOUNT)

Set the number of values to measure at trigger.

Parameter: int SP_TriggerCount

SP_TriggerCount

Direction: Down

Valid values:

- Minimum:** 0
- Maximum:** 16383

Description: Number of values to measure. 0 means no trigger (NONE), 16383 means endless measurement (INFINITE).

10.3.1.3.6 Get_TriggerCount (TRIGGERCOUNT)

Get the number of values to measure at trigger.

Parameter: int SA_TriggerCount

SA_TriggerCount

Direction: Up

Valid values:

Minimum: 0

Maximum: 16383

Description: Number of values to measure. 0 means no trigger (NONE), 16383 means endless measurement (INFINITE).

10.3.1.3.7 Software_Trigger (TRIGGERSW)

Execute a software trigger.

10.3.1.4 Interfaces

10.3.1.4.1 Set_Baudrate (BAUDRATE)

Set baudrate of sensor for serial RS422 communication. After receiving the answer of the command, the baudrate of the local serial interface is changed, too.

Parameter: int SP_SensorBaudrate

SP_SensorBaudrate

Direction: Down

Valid values:

1000000

921600

691200

460800

256000

230400

128000

115200

56000

19200

9600

Unit: Baud

Description: Baudrate of sensor.

10.3.1.4.2 Get_Baudrate (BAUDRATE)

Get baudrate of sensor for serial RS422 communication.

Parameter: int SA_SensorBaudrate

SA_SensorBaudrate

Direction: Up

Valid values:

1000000

921600

691200

460800
256000
230400
128000
115200
56000
19200
9600

Unit: Baud

Description: Baudrate of sensor.

10.3.1.4.3 Set_AppLanguage (LANGUAGE)

Set language of web interface.

Parameter: int SP_ApplicationLanguage

SP_ApplicationLanguage

Direction: Down

Valid values:

0= English (EN)
1= German (DE)

Description: Language of web interface.

10.3.1.4.4 Get_AppLanguage (LANGUAGE)

Get language of web interface.

Parameter: int SA_ApplicationLanguage

SA_ApplicationLanguage

Direction: Up

Valid values:

0= English (EN)
1= German (DE)

Description: Language of web interface.

10.3.1.5 Parameter management

10.3.1.5.1 Save_InterfaceParameters (BASICSETTINGS STORE)

Save actual interface parameters at controller.

10.3.1.5.2 Load_InterfaceParameters (BASICSETTINGS READ)

Load actual interface parameters into controller RAM.

10.3.1.5.3 Save_MeasureParameters (MEASSETTINGS STORE)

Save actual measurement parameters at controller.

Parameter: String SP_SettingName

SP_SettingName

Direction: Down

Description: Name of setting. If name is empty, current setting will be overwritten.

10.3.1.5.4 Load_MeasureParameters (MEASSETTINGS READ)

Load stored measurement parameters into controller RAM.

Parameter: String SP_SettingName SP_SettingName
Direction: Down
Description: Name of setting.

10.3.1.5.5 Rename_MeasureParameters (MEASSETTINGS RENAME)

Rename stored measurement parameters at controller.

Parameter: String SP_OldName SP_OldName
Direction: Down
Description: Actual name of setting.

Parameter: String SP_NewName SP_NewName
Direction: Down
Description: New name of setting.

Parameter: int SP_Overwrite SP_Overwrite
Direction: Down
Valid values:
 0= no
 1= yes
Description: Force overwriting existing setting.

10.3.1.5.6 Get_CurrentMeasureSetting (MEASSETTINGS CURRENT)

Get name of current (active) measurement parameters.

Parameter: String SA_CurrentName SA_CurrentName
Direction: Up
Description: Name of current (active) setting.

10.3.1.5.7 Get_MeasureSettingsList (MEASSETTINGS LIST)

Get list of all user settings.

Parameter: String SA_SettingNames SA_SettingNames
Direction: Up
Description: List (separated by new line).

10.3.1.5.8 Delete_MeasureParameters (MEASSETTINGS DELETE)

Delete stored measurement parameters at controller.

Parameter: String SP_SettingName SP_SettingName
Direction: Down
Description: Name of setting.

10.3.1.5.9 Get_MeasurePresetList (MEASSETTINGS PRESETLIST)

Get list of all preset settings.

Parameter: String SA_PresetNames SA_PresetNames
Direction: Up
Description: List (separated by new line).

10.3.1.5.10 Set_InitialMeasureSetting (MEASSETTINGS INITIAL)

Set initial (boot time) measure setting.

Parameter: int SP_Automatic SP_Automatic
Direction: Down
Valid values:
 0= no
 1= yes
Description: Automatic selection.

Parameter: String SP_InitialName SP_InitialName
Direction: Down
Description: Name of setting.

10.3.1.5.11 Get_InitialMeasureSetting (MEASSETTINGS INITIAL)

Get initial (boot time) measure setting.

Parameter: int SA_Automatic SA_Automatic
Direction: Up
Valid values:
 0= no
 1= yes
Description: Automatic selection.

Parameter: String SA_InitialName SA_InitialName
Direction: Up
Description: Name of setting.

10.3.1.5.12 Set_MeasurePresetMode (MEASSETTINGS PRESETMODE)

Set mode of current preset settings.

Parameter: int SP_PresetMode SP_PresetMode
Direction: Down
Valid values:
 0= Static
 1= Balanced
 2= Dynamic
 3= None (if no preset setting is active)
Description: Preset mode.

10.3.1.5.13 Get_MeasurePresetMode (MEASSETTINGS PRESETMODE)

Get mode of current preset settings.

Parameter: int SA_PresetMode

SA_PresetMode

Direction: Up

Valid values:

- 0= Static
- 1= Balanced
- 2= Dynamic
- 3= None (if no preset setting is active)

Description: Preset mode.

10.3.1.5.14 Set_Default (SETDEFAULT)

Reset the sensor to default settings.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_DefaultType

SP_DefaultType

Direction: Down

Valid values:

- 0= Delete all settings and load the factory settings (ALL)
- 1= Delete all measuring settings (MEASSETTINGS)
- 2= Delete all basic settings (BASICSETTINGS)

Description: Specifies which settings should be reset.

10.3.1.5.15 Export_Parameters (EXPORT)

Exports the settings of the sensor..

Parameter: int SP_ExportType

SP_ExportType

Direction: Down

Valid values:

- 0= Exports only the measuring settings with name (MEASSETTINGS)
- 1= Exports only the basic settings (BASICSETTINGS)
- 2= Exports all measuring settings (MEASSETTINGS_ALL)
- 3= Exports basic settings and all measuring settings (ALL)

Description: Export type.

Parameter: String SP_SettingName

SP_SettingName

Direction: Down

Description: Name of setting to be exported.

This parameter is only used at export type 0.

Parameter: String SA_ExportData

SA_ExportData

Direction: Up

Description: Exported data in ASCII format.

10.3.1.5.16 Import_Parameters (IMPORT)

Imports the settings of the sensor..

Parameter: int SP_ForceOverwrite	SP_ForceOverwrite
Direction: Down	
Valid values:	
0= Do not overwrite existing setting	
1= Allow to overwrites existing settings (FORCE)	
Description: Specify if existing settings can be overwritten.	
Parameter: int SP_ApplyImmediately	SP_ApplyImmediately
Direction: Down	
Valid values:	
0= Just store imported settings.	
1= Apply the imported settings (APPLY)	
Description: Specify if settings only should be stored or additionally applied.	
Parameter: String SP_ImportData	SP_ImportData
Direction: Down	
Description: Data to be imported (from a former call to Export_Parameters).	

10.3.1.6 Internal controller commands

10.3.1.6.1 Set_MultiFunctionInputMode (MFIFUNC)

Set the mode of multi function input at sensor.

Parameter: int SP_MultiFunctionInputMode	SP_MultiFunctionInputMode
Direction: Down	
Valid values:	
0= MFI has no function (NONE)	
1= Use MFI to master the measuring value (MASTER)	
2= Use MFI to teach the analog output (TEACH)	
3= Use MFI to trigger the measuring process (TRIGGER)	
Description: Specifies how to use the multi function input.	

10.3.1.6.2 Get_MultiFunctionInputMode (MFIFUNC)

Get the mode of multi function input at sensor.

Parameter: int SA_MultiFunctionInputMode	SA_MultiFunctionInputMode
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= MFI has no function (NONE)	
1= Use MFI to master the measuring value (MASTER)	
2= Use MFI to teach the analog output (TEACH)	
3= Use MFI to trigger the measuring process (TRIGGER)	
Description: Specifies how to use the multi function input.	

10.3.1.6.3 Set_MultiFunctionInputLevel (MFILEVEL)

Set level of multi function input.

Parameter: int SP_MultiFunctionInputLevel

Direction: Down

Valid values:

0= High (HTL_HIGH)

1= Low (HTL_LOW)

Description: Multi function input level.

SP_MultiFunctionInput-
Level

10.3.1.6.4 Get_MultiFunctionInputLevel (MFILEVEL)

Get level of multi function input.

Parameter: int SA_MultiFunctionInputLevel

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= High (HTL_HIGH)

1= Low (HTL_LOW)

Description: Multi function input level.

SA_MultiFunctionInput-
Level

10.3.1.6.5 Set_InitialChartType (CHARTTYPE_INITIAL)

Set initial type of the chart at web interface.

Parameter: int SP_InitialChartType

SP_InitialChartType

Direction: Down

Valid values:

0= Measure (MEAS)

1= Video signal (VIDEO)

2= Automatic (AUTO)

Description: Initial type of the chart at web interface.

10.3.1.6.6 Get_InitialChartType (CHARTTYPE_INITIAL)

Get initial type of the chart at web interface.

Parameter: int SA_InitialChartType

SA_InitialChartType

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= Measure (MEAS)

1= Video signal (VIDEO)

2= Automatic (AUTO)

Description: Initial type of the chart at web interface.

10.3.1.6.7 Set_ChartType (CHARTTYPE)

Set current type of the chart at web interface.

Parameter: int SP_ChartType SP_ChartType
Direction: Down
Valid values:
 0= Measure (MEAS)
 1= Video signal (VIDEO)
Description: Type of the chart at web interface.

10.3.1.6.8 Get_ChartType (CHARTTYPE)

Get current type of the chart at web interface.

Parameter: int SA_ChartType SA_ChartType
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= Measure (MEAS)
 1= Video signal (VIDEO)
Description: Type of the chart at web interface.

10.3.1.6.9 Set_TargetMode (TARGETMODE)

Selecting a target loads a predefined sensor configuration which achieves the best results for the selected material.

This command is only available at ILD1420, not ILD1320.

Parameter: int SP_TargetMode SP_TargetMode
Direction: Down
Valid values:
 0= Standard Target (STANDARD)
 1= Multi-Surface Target (MULTISURFACE)
 2= Light Penetration Target (PENETRATION)
Description: Target mode.

10.3.1.6.10 Get_TargetMode (TARGETMODE)

Target for a predefined sensor configuration which achieves the best results for the selected material.

This command is only available at ILD1420, not ILD1320.

Parameter: int SA_TargetMode SA_TargetMode
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= Standard Target (STANDARD)
 1= Multi-Surface Target (MULTISURFACE)
 2= Light Penetration Target (PENETRATION)
Description: Target mode.

10.3.1.6.11 Get_HTTP (GETHTTP)

Retrieve HTTP data from sensor.

Parameter: String SP_HTPPPath SP_HTPPPath
Direction: Down
Description: HTTP file name.

Parameter: int SP_HTTPMaxLength SP_HTTPMaxLength
Direction: Down
Valid values:
Minimum: 1
Maximum: 65535
Unit: Bytes
Description: Maximum transfer length.

Parameter: String SP_HTTPeTag SP_HTTPeTag
Direction: Down
Description: HTTP eTag (data is not transmitted if eTag does match).

Parameter: int CP_HTTPRetryCount CP_HTTPRetryCount
Direction: Down
Valid values:
Minimum: 0
Maximum: 2147483647 (INT_MAX)
Default: 5
Description: Maximum retry count on error (0 means do not retry).

Parameter: String SA_HTTPHeader SA_HTTPHeader
Direction: Up
Description: HTTP header.

Parameter: String SA_HTTPBody SA_HTTPBody
Direction: Up
Description: HTTP body.

10.3.2 Measurement

10.3.2.1 General

10.3.2.1.1 Set_MeasurePeak (MEASPEAK)

Select the peak to measure.

Parameter: int SP_MeasurePeak SP_MeasurePeak
Direction: Down
Valid values:
 0= Greatest Amplitude (DISTA)
 1= First Peak (DIST1)
 2= Last Peak (DISTL)
Description: Peak to measure.

10.3.2.1.2 Get_MeasurePeak (MEASPEAK)

Get the selected peak to measure.

Parameter: int SA_MeasurePeak

SA_MeasurePeak

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = Greatest Amplitude (DISTA)
- 1 = First Peak (DIST1)
- 2 = Last Peak (DISTL)

Description: Peak to measure.

10.3.2.1.3 Set_Samplerate (MEASRATE)

Set the samplerate.

Parameter: double SP_Measrate

SP_Measrate

Direction: Down

Valid values:

- 0.25
- 0.5
- 1.0
- 2.0
- 4.0

Unit: kHz

Description: Samplerate of measurement.

4.0 kHz is only available at ILD1420, not ILD1320.

10.3.2.1.4 Get_Samplerate (MEASRATE)

Get the samplerate.

Parameter: double SA_Measrate

SA_Measrate

Direction: Up

Valid values:

- 0.25
- 0.5
- 1.0
- 2.0
- 4.0

Unit: kHz

Description: Samplerate of measurement.

4.0 kHz is only available at ILD1420, not ILD1320.

10.3.2.1.5 Set_LaserPower (LASERPOW)

Specify the laser power at sensor.

Parameter: int SP_LaserPower

SP_LaserPower

Direction: Down

Valid values:

- 0 = Full
- 1 = Off

Description: Laser power.

10.3.2.1.6 Get_LaserPower (LASERPOW)

Get the laser power from sensor.

Parameter: int SA_LaserPower SA_LaserPower
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= Full
 1= Off
Description: Laser power.

10.3.2.1.7 Get_VideoStreamSignal

Read one video signal from video stream.

Parameter: int SP_ReadMode SP_ReadMode
Direction: Down
Valid values:
 0= Each video signal
 1= Only newest video signal
 2= Automatic
Description: This mode specifies if each video signal should be read or only the latest one. If set to automatic each video signal is read until the buffer does not overflow. If the buffer becomes full one or more video signals are discarded.

Parameter: int SP_WaitVideoTimeout SP_WaitVideoTimeout
Direction: Down
Unit: ms
Valid values:
Minimum: 0
Maximum: 2147483647 (INT_MAX)
Description: Timeout to wait for a video signal.

Parameter: Binary data SA_VideoRaw SA_VideoRaw
Direction: Up
Valid values:
 768 words (each 2 byte), each word is an intensity value.
Description: Raw video signal

Parameter: double SA_VideoTimestamp SA_VideoTimestamp
Direction: Up
Valid values:
Minimum: 0
Maximum: 1.79769e+308 (DBL_MAX)
Unit: ms
Description: Timestamp of the video signal. It starts from 1970 Jan 01 at 01:00. It is generated when the video has arrived at TCP/IP socket.

Parameter: int SA_SkippedVideo SA_SkippedVideo
Direction: Up
Valid values:
Minimum: 0
Maximum: 2147483647 (INT_MAX)
Description: Number of skipped video signals, if SP_ReadMode is not 0.

10.3.2.2 Video signal

10.3.2.2.1 Set_ROI (ROI)

Set the region of intererst for processing video signal.
 This command is only available at ILD1420, not ILD1320.

Parameter: int SP_ROIStart SP_ROIStart
Direction: Down
Valid values:
Minimum: 0
Maximum: 511
Unit: Pixel
Description: First pixel.

Parameter: int SP_ROIEnd SP_ROIEnd
Direction: Down
Valid values:
Minimum: 0
Maximum: 511
Unit: Pixel
Description: Last pixel.

10.3.2.2.2 Get_ROI (ROI)

Get the region of intererst for processing video signal.
 This command is only available at ILD1420, not ILD1320.

Parameter: int SA_ROIStart SA_ROIStart
Direction: Up
Valid values:
Minimum: 0
Maximum: 511
Unit: Pixel
Description: First pixel.

Parameter: int SA_ROIEnd SA_ROIEnd
Direction: Up
Valid values:
Minimum: 0
Maximum: 511
Unit: Pixel
Description: Last pixel.

10.3.2.3 Measurement value processing

10.3.2.3.1 Set_Averaging (AVERAGE)

Set data averaging at sensor.

This command is only available at ILD1420, not ILD1320.

Parameter: int SP_AveragingType

SP_AveragingType

Direction: Down

Valid values:

- 0= None
- 1= Moving average (MOVING)
- 2= Recursive averaging (RECURSIVE)
- 3= Median

Description: Averaging type.

Parameter: int SP_MovingCount

SP_MovingCount

Direction: Down

Valid values:

- 2
- 4
- 8
- 16
- 32
- 64
- 128

Description: Number of value for the averaging window. This parameter is only used at moving average.

Parameter: int SP_RecursiveCount

SP_RecursiveCount

Direction: Down

Valid values:

- Minimum: 2
- Maximum: 32768

Description: Number of values for recursive averaging. This parameter is only used at recursive average.

Parameter: int SP_MedianCount

SP_MedianCount

Direction: Down

Valid values:

- 3
- 5
- 7
- 9

Description: Number of values to build median. This parameter is only used at median.

10.3.2.3.2 Get_Averaging (AVERAGE)

Get data averaging at sensor.

This command is only available at ILD1420, not ILD1320.

Parameter: int SA_AveragingType

SA_AveragingType

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= Moving average (MOVING)
- 2= Recursive averaging (RECURSIVE)
- 3= Median

Description: Averaging type.

Parameter: int SA_MovingCount

SA_MovingCount

Direction: Up

Valid values:

- 2
- 4
- 8
- 16
- 32
- 64
- 128

Description: Number of value for the averaging window.

This parameter is only available at moving average.

Parameter: int SA_RecursiveCount

SA_RecursiveCount

Direction: Up

Valid values:

- Minimum: 2
- Maximum: 32768

Description: Number of values for recursive averaging.

This parameter is only available at recursive average.

Parameter: int SA_MedianCount

SA_MedianCount

Direction: Up

Valid values:

- 3
- 5
- 7
- 9

Description: Number of values to build median.

This parameter is only available at median.

10.3.2.3.3 Set_MasterValue (MASTERMV)

Set the master value.

Parameter: int SP_Master

SP_Master

Direction: Down

Valid values:

- 0= no (NONE)
- 1= yes (MASTER=

Description: Specifies if mastering should be done or resetted.

Parameter: double SP_MasterValue SP_MasterValue
Direction: Down
Valid values:
 Minimum: 0
 Maximum: 2 * measuring range
Unit: mm
Description: Master value

10.3.2.3.4 Get_MasterValue (MASTERMV)

Get the master value.

Parameter: int SA_Master SA_Master
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = no (NONE)
 1 = yes (MASTER=)
Description: Specifies if mastering is active.

Parameter: double SA_MasterValue SA_MasterValue
Direction: Up
Valid values:
 Minimum: 0
 Maximum: 2 * measuring range
Unit: mm
Description: Master value

10.3.3 Data output

10.3.3.1 General

10.3.3.1.1 Set_DataOutInterface (OUTPUT)

Set the active interface for data output.

If first bit of [IP_AutomaticMode](#) is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_DataOutInterface SP_DataOutInterface
Direction: Down
Valid values:
 0 = None
 1 = RS422
 6 = Analog
Description: Active interface for data output.

10.3.3.1.2 Get_DataOutInterface (OUTPUT)

Get the active interface for data output.

Parameter: int SA_DataOutInterface

SA_DataOutInterface

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= RS422
- 6= Analog

Description: Active interface for data output.

10.3.3.1.3 Set_ResamplingDevice (OUTREDUCEDEVICE)

Set the devices for which resampling is active.

This command is only available at ILD1420, not ILD1320.

Parameter: int SP_ResampleRS422

SP_ResampleRS422

Direction: Down

Valid values:

- 0= no
- 1= yes

Description: Specify if RS422 output should be resampled.

Parameter: int SP_ResampleAnalog

SP_ResampleAnalog

Direction: Down

Valid values:

- 0= no
- 1= yes

Description: Specify if analog output should be resampled.

10.3.3.1.4 Get_ResamplingDevice (OUTREDUCEDEVICE)

Get the devices for which resampling is active.

This command is only available at ILD1420, not ILD1320.

Parameter: int SA_ResampleRS422

SA_ResampleRS422

Direction: Up

Valid values:

- 0= no
- 1= yes

Description: RS422 output is resampled.

Parameter: int SA_ResampleAnalog

SA_ResampleAnalog

Direction: Up

Valid values:

- 0= no
- 1= yes

Description: Analog output is resampled.

10.3.3.1.5 Set_ResamplingCount (OUTREDUCECOUNT)

Set reduce count for resampling.

This command is only available at ILD1420, not ILD1320.

Parameter: int SP_Resampling SP_Resampling
Direction: Down
Valid values:
Minimum: 1
Maximum: 3000000
Description: Resampling value.

10.3.3.1.6 Get_ResamplingCount (OUTREDUCECOUNT)

Get reduce count for resampling.

This command is only available at ILD1420, not ILD1320.

Parameter: int SA_Resampling SA_Resampling
Direction: Up
Valid values:
Minimum: 1
Maximum: 3000000
Description: Resampling value.

10.3.3.1.7 Set_HoldLastValid (OUTHOLD)

Set the number of values to be replaced by last valid value instead of error values.

Parameter: int SP_HoldLastValid SP_HoldLastValid
Direction: Down
Valid values:
Minimum: -1
Maximum: 1024
Description: Values to replace by last valid value. -1 means no value to hold,
 0 means never output an error value (always hold last valid value).

10.3.3.1.8 Get_HoldLastValid (OUTHOLD)

Get the number of values to be replaced by last valid value instead of error values.

Parameter: int SA_HoldLastValid SA_HoldLastValid
Direction: Up
Valid values:
Minimum: -1
Maximum: 1024
Description: Values to replace by last valid value. -1 means no value to hold,
 0 means never output an error value (always hold last valid value).

10.3.3.2 Selected measurement values

10.3.3.2.1 Set_OutputAdditional_RS422 (OUTADD_RS422)

Set the additional data to be output at RS422 interface.

Parameter: int SP_OutputAdditionalShutterTime_RS422

Direction: Down

SP_OutputAdditionalShutterTime_RS422

Valid values:

0= no

1= yes

Description: Specify if shutter time is transmitted.

Parameter: int SP_OutputAdditionalCounter_RS422

Direction: Down

SP_OutputAdditionalCounter_RS422

Valid values:

0= no

1= yes

Description: Specify if counter is transmitted.

Parameter: int SP_OutputAdditionalTimestamp_RS422

Direction: Down

SP_OutputAdditionalTimestamp_RS422

Valid values:

0= no

1= yes

Description: Specify if timestamp is transmitted.

Parameter: int SP_OutputAdditionalIntensity_RS422

Direction: Down

SP_OutputAdditionalIntensity_RS422

Valid values:

0= no

1= yes

Description: Specify if intensity is transmitted.

Parameter: int SP_OutputAdditionalState_RS422

Direction: Down

SP_OutputAdditionalState_RS422

Valid values:

0= no

1= yes

Description: Specify if state is transmitted.

Parameter: int SP_OutputAdditionalDistanceRaw_RS422

Direction: Down

SP_OutputAdditionalDistanceRaw_RS422

Valid values:

0= no

1= yes

Description: Specify if uncalibrated distance is transmitted.

10.3.3.2.2 Get_OutputAdditional_RS422 (OUTADD_RS422)

Get the additional data which is output at RS422 interface.

Parameter: int SA_OutputAdditionalShutterTime_RS422

SA_OutputAdditionalShutterTime_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if shutter time is transmitted.

Parameter: int SA_OutputAdditionalCounter_RS422

SA_OutputAdditionalCounter_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if counter is transmitted.

Parameter: int SA_OutputAdditionalTimestamp_RS422

SA_OutputAdditionalTimestamp_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if timestamp is transmitted.

Parameter: int SA_OutputAdditionalIntensity_RS422

SA_OutputAdditionalIntensity_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if intensity is transmitted.

Parameter: int SA_OutputAdditionalState_RS422

SA_OutputAdditionalState_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if state is transmitted.

Parameter: int SA_OutputAdditionalDistanceRaw_RS422

SA_OutputAdditionalDistanceRaw_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if uncalibrated distance is transmitted.

10.3.3.2.3 Set_Output_RS422 (OUT_RS422)

Set the data to be output at RS422 interface.

Parameter: int SP_OutputDistance1_RS422

SP_OutputDistance1_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if distance is transmitted.

Parameter: int SP_OutputAdditionalShutterTime_RS422	SP_OutputAdditionalShutterTime_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	
Parameter: int SP_OutputAdditionalCounter_RS422	SP_OutputAdditionalCounter_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SP_OutputAdditionalTimestamp_RS422	SP_OutputAdditionalTimestamp_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SP_OutputAdditionalIntensity_RS422	SP_OutputAdditionalIntensity_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity is transmitted.	
Parameter: int SP_OutputAdditionalState_RS422	SP_OutputAdditionalState_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	
Parameter: int SP_OutputAdditionalDistanceRaw_RS422	SP_OutputAdditionalDistanceRaw_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if uncalibrated distance is transmitted.	
Parameter: int SP_OutputVideoRaw_RS422	SP_OutputVideoRaw_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if raw video signal is transmitted.	
This parameter is only available at ILD1420, not ILD1320.	

10.3.3.2.4 Get_Output_RS422 (OUT_RS422)

Get the data which is output at RS422 interface.

Parameter: int SA_OutputDistance1_RS422

SA_OutputDistance1_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if distance is transmitted.

Parameter: int SA_OutputAdditionalShutterTime_RS422

SA_OutputAdditionalShutterTime_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if shutter time is transmitted.

Parameter: int SA_OutputAdditionalCounter_RS422

SA_OutputAdditionalCounter_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if counter is transmitted.

Parameter: int SA_OutputAdditionalTimestamp_RS422

SA_OutputAdditionalTimestamp_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if timestamp is transmitted.

Parameter: int SA_OutputAdditionalIntensity_RS422

SA_OutputAdditionalIntensity_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if intensity is transmitted.

Parameter: int SA_OutputAdditionalState_RS422

SA_OutputAdditionalState_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if state is transmitted.

Parameter: int SA_OutputAdditionalDistanceRaw_RS422

SA_OutputAdditionalDistanceRaw_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if uncalibrated distance is transmitted.

10.3. Commands for ILD1320/ILD1420

Parameter: int SA_OutputVideoRaw_RS422 SA_OutputVideoRaw_RS422
Direction: Up
Valid values:
 0= no
 1= yes
Description: Specify if raw video signal is transmitted.
 This parameter is only available at ILD1420, not ILD1320.

10.3.3.2.5 Set_OutputVideo_RS422 (OUTVIDEO_RS422)

Set the video signal to be output at RS422 interface.
 This command is only available at ILD1420, not ILD1320.

Parameter: int SP_OutputVideoRaw_RS422 SP_OutputVideoRaw_RS422
Direction: Down
Valid values:
 0= no
 1= yes
Description: Specify if raw video signal is transmitted.

10.3.3.2.6 Get_OutputVideo_RS422 (OUTVIDEO_RS422)

Get the video signal which is output at RS422 interface.
 This command is only available at ILD1420, not ILD1320.

Parameter: int SA_OutputVideoRaw_RS422 SA_OutputVideoRaw_RS422
Direction: Up
Valid values:
 0= no
 1= yes
Description: Specify if raw video signal is transmitted.

10.3.3.3 Switching outputs

10.3.3.3.1 Set_ErrorOutput1 (ERROROUT1)

Set condition to be used to set error output 1.

Parameter: int SP_ErrorOutput1 SP_ErrorOutput1
Direction: Down
Valid values:
 0= Output trigger disabled (NONE)
 1= No valid distance (no peak found, out of range) (DIST)
 2= Distance is out of scaled analog range (TEACH)
 3= Distance is above set threshold (L1)
Description: Condition for error output.
 Value 2 and 3 are only available at special option 200

10.3.3.3.2 Get_ErrorOutput1 (ERRORROUT1)

Get condition to be used to set error output 1.

Parameter: int SA_ErrorOutput1

SA_ErrorOutput1

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = Output trigger disabled (NONE)
- 1 = No valid distance (no peak found, out of range) (DIST)
- 2 = Distance is out of scaled analog range (TEACH)
- 3 = Distance is above set threshold (L1)

Description: Condition for error output.

Value 2 and 3 are only available at special option 200

10.3.3.3.3 Set_ErrorLimit (ERRORLIMIT)

Set the error limits.

Only available at option 200.

Parameter: int SP_DataSource

SP_DataSource

Direction: Down

Valid values:

- 0 = Distance 1

Description: Data source to be checked.

Parameter: double SP_UpperLimit

SP_UpperLimit

Direction: Down

Valid values:

- Minimum:** 0
- Maximum:** 2* measuring range

Unit: mm

Description: Upper limit.

10.3.3.3.4 Get_ErrorLimit (ERRORLIMIT)

Get the error limits.

Only available at option 200.

Parameter: int SA_DataSource

SA_DataSource

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = Distance 1

Description: Data source to be checked.

Parameter: double SA_UpperLimit

SA_UpperLimit

Direction: Up

Valid values:

- Minimum:** 0
- Maximum:** 2* measuring range

Unit: mm

Description: Upper limit.

10.3.3.3.5 Set_ErrorHysteresis (ERRORHYSTERESIS)

Set the hysteresis of the threshold function (Set_ErrorLimit).
Only available at option 200.

Parameter: double SP_ErrorHysteresis SP_ErrorHysteresis
Direction: Down
Valid values:
Minimum: 0
Maximum: 2* measuring range
Unit: mm
Description: Error hysteresis.

10.3.3.6 Get_ErrorHysteresis (ERRORHYSTERESIS)

Get the hysteresis of the threshold function (Set_ErrorLimit).
Only available at option 200.

Parameter: double SA_ErrorHysteresis SA_ErrorHysteresis
Direction: Up
Valid values:
Minimum: 0
Maximum: 2* measuring range
Unit: mm
Description: Error hysteresis.

10.3.3.7 Set_ErrorOutHoldTime (ERROROUTHOLD)

Set the minimum hold period of the threshold function (Set_ErrorLimit)
Only available at option 200.

Parameter: int SP_ErrorOutHoldTime SP_ErrorOutHoldTime
Direction: Down
Valid values:
Minimum: 0
Maximum: 1000
Unit: ms
Description: Minimum hold period.

10.3.3.8 Get_ErrorOutHoldTime (ERROROUTHOLD)

Get the minimum hold period of the threshold function (Set_ErrorLimit)
Only available at option 200.

Parameter: int SA_ErrorOutHoldTime SA_ErrorOutHoldTime
Direction: Up
Valid values:
Minimum: 0
Maximum: 1000
Unit: ms
Description: Minimum hold period.

10.3.3.3.9 Set_ErrorLevelOut1 (ERRORLEVELOUT1)

Set level of error output 1 on error.

Parameter: int SP_ErrorLevelOut1 SP_ErrorLevelOut1
Direction: Down
Valid values:
 0= NPN
 1= PNP
 2= Push-Pull (PUSHPULL)
 3= Push-Pull negated (PUSHPULLNEG)
Description: Error level for out 1.

10.3.3.3.10 Get_ErrorLevelOut1 (ERRORLEVELOUT1)

Get level of error output 1 on error.

Parameter: int SA_ErrorLevelOut1 SA_ErrorLevelOut1
Direction: Up
Valid values:
 0= NPN
 1= PNP
 2= Push-Pull (PUSHPULL)
 3= Push-Pull negated (PUSHPULLNEG)
Description: Error level for out 1.

10.3.3.4 Analog output

10.3.3.4.1 Set_AnalogScale (ANALOGSCALE)

Set the scaling factor for analog output.

Parameter: int SP_AnalogScaleMode SP_AnalogScaleMode
Direction: Down
Valid values:
 0= Standard
 1= Two point (TWOPOINT)
Description: Analog scale mode.

Parameter: double SP_MinValue SP_MinValue
Direction: Down
Valid values:
Minimum: 0
Maximum: +2* measuring range
Unit: mm
Description: Value which represents lowest voltage/current (at two point scaling).

Parameter: double SP_MaxValue SP_MaxValue

Direction: Down

Valid values:

Minimum: 0

Maximum: +2* measuring range

Unit: mm

Description: Value which represents highest voltage/current (at two point scaling).

10.3.3.4.2 Get_AnalogScale (ANALOGSCALE)

Get the scaling factor for analog output.

Parameter: int SA_AnalogScaleMode SA_AnalogScaleMode

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= Standard

1= Two point (TWOPOINT)

Description: Analog scale mode.

Parameter: double SA_MinValue SA_MinValue

Direction: Up

Valid values:

Minimum: 0

Maximum: +2* measuring range

Unit: mm

Description: Value which represents lowest voltage/current (at two point scaling).

Parameter: double SA_MaxValue SA_MaxValue

Direction: Up

Valid values:

Minimum: 0

Maximum: +2* measuring range

Unit: mm

Description: Value which represents highest voltage/current (at two point scaling).

10.4 Commands for ILD1700

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [RS232](#) (additional, e.g. IF2001_USB (RS422) and RS232 high level interface).
- [IF2004](#) (native).
- [TCP/IP](#) (additional, e.g. RS232/RS422 to TCP/IP comm server and RS232 high level interface).
- [IF2004_USB](#) (native).

- [IF2008](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_Settings](#) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib timeout checking, to calculate datarate and to scale data.

If second bit of [IP_AutomaticMode](#) is set (2), MEDAQLib calls sensor command [Dat_Out_On](#), [Laser_On](#) and optionally [Set_OutputType](#) at [OpenSensor](#).

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor, from 0 to 16383.
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command [Get_Settings](#)), error values are scaled depending of [IP_ScaleErrorValues](#).

10.4.1 General commands

10.4.1.1 General

10.4.1.1.1 Get_Info (GET_INFO)

Retrieve some information about the sensor.

Parameter: String SA_Sensor

SA_Sensor

Direction: Up

Description: Name of the sensor.

Parameter: String SA_SensorType

SA_SensorType

Direction: Up

Description: Type of the sensor.

Parameter: String SA_Softwareversion

SA_Softwareversion

Direction: Up

Description: Software version of the sensor.

Parameter: int SA_OutputType

SA_OutputType

Direction: Up

Valid values:

0= current (4..20mA)

1= voltage (0..10V)

2= RS422

Description: Data output (only values, not answer) interface of the sensor.

Parameter: int SA_ErrorOutput

SA_ErrorOutput

Direction: Up

Valid values:

0= sync error mode

1= sync switch mode

2= trigger error mode

3= trigger switch mode

Description: Sync/trigger and error/switch mode respectively of the sensor.

Parameter: int SA_Speed	SA_Speed
Direction: Up	
Valid values:	
0= 2.5kHz	
1= 1.25kHz	
2= 625Hz	
3= 312.5Hz	
Description: The output speed of the sensor.	
Parameter: double SA_Samplerate	SA_Samplerate
Direction: Up	
Valid values:	
2500	
1250	
625	
312.5 (or other on specific sensor settings)	
Unit: Hz	
Description: The output speed of the sensor.	
Parameter: int SA_AvType	SA_AvType
Direction: Up	
Valid values:	
0= recursive	
1= moving	
2= Median	
Description: The averaging type.	
Parameter: int SA_AvIndex	SA_AvIndex
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 15	
Description: The averaging index. Averaging $N = 2^{AvIndex}$.	
Parameter: int SA_ErrorHandler	SA_ErrorHandler
Direction: Up	
Valid values:	
0= error values	
1= hold last value	
Description: If the sensor cannot measure values, it can output the last valid value or it can output an error values.	
Parameter: int SA_Sync_TrMode	SA_Sync_TrMode
Direction: Up	
Valid values:	
Condition: at sync mode (ErrorOutput is 0 or 1):	
0= synchronous master off	
1= synchronous master on	
2= slave	
3= alternating synchronous master	
Valid values:	
Condition: at trigger mode (ErrorOutput is 2 or 3):	
0= edge L/H	
1= edge H/L	
2= level H	
3= level L	
Description: The sync mode or the trigger mode of the sensor.	

Parameter: int SA_ASCII	SA_ASCII
Direction: Up	
Valid values:	
0= off (binary 2 bytes/value)	
1= on (ASCII 6 bytes/value)	
Description: Returns the mode the sensor is sending data (only values).	
Parameter: int SA_Keylock	SA_Keylock
Direction: Up	
Valid values:	
0= off (keys enabled)	
1= on (keys locked)	
Description: The keypad state at the sensor.	
Parameter: double SA_Range	SA_Range
Direction: Up	
Valid values:	
Minimum: 0.0	
Unit: mm	
Description: Range of the sensor.	
Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the sensor.	
Parameter: String SA_Option	SA_Option
Direction: Up	
Valid values:	
Numeric value	
Description: Option of the sensor.	
Parameter: String SA_ArticleNumber	SA_ArticleNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Article number of the sensor.	
Parameter: String SA_Date	SA_Date
Direction: Up	
Description: Software release date of the sensor.	
Parameter: String SA_BootLoaderVer	SA_BootLoaderVer
Direction: Up	
Description: Boot loader version of the sensor.	
Parameter: String SA_SWType	SA_SWType
Direction: Up	
Description: Software type of the sensor.	
Parameter: int SA_EnableFlash	SA_EnableFlash
Direction: Up	
Valid values:	
0= locked	
1= enabled	
Description: The flash is locked or enabled for writing.	

10.4.1.1.2 Get_Settings (GET_SETTINGS)

Retrieve detailed information about the sensor.

Parameter: int SA_OutputType SA_OutputType

Direction: Up

Valid values:

- 0= current (4..20mA)
- 1= voltage (0..10V)
- 2= RS422

Description: Data output (only values, not answer) interface of the sensor.

Parameter: int SA_Speed SA_Speed

Direction: Up

Valid values:

- 0= 2.5kHz
- 1= 1.25kHz
- 2= 625Hz
- 3= 312.5Hz

Description: The output speed of the sensor.

Parameter: int SA_AvIndex SA_AvIndex

Direction: Up

Valid values:

- Minimum:** 0
- Maximum:** 15

Description: The averaging index. Averaging $N = 2^{AvIndex}$.

Parameter: int SA_ErrorHandler SA_ErrorHandler

Direction: Up

Valid values:

- 0= error values
- 1= hold last value

Description: If the sensor cannot measure values, it can output the last valid value or it can output an error values.

Parameter: int SA_Sync_TrgMode SA_Sync_TrgMode

Direction: Up

Valid values:

Condition: at sync mode (ErrorOutput is 0 or 1):

- 0= synchronous master off
- 1= synchronous master on
- 2= slave
- 3= alternating synchronous master

Valid values:

Condition: at trigger mode (ErrorOutput is 2 or 3):

- 0= edge L/H
- 1= edge H/L
- 2= level H
- 3= level L

Description: The sync mode or the trigger mode of the sensor.

Parameter: int SA_AvType SA_AvType

Direction: Up

Valid values:

- 0= recursive
- 1= moving
- 2= Median

Description: The averaging type.

Parameter: int SA_ErrorOutput SA_ErrorOutput

Direction: Up

Valid values:

- 0= sync error mode
- 1= sync switch mode
- 2= trigger error mode
- 3= trigger switch mode

Description: Sync/trigger and error/switch mode respectively of the sensor.

Parameter: int SA_SensorBaudrate SA_SensorBaudrate

Direction: Up

Valid values:

- 0= 115200 Baud
- 1= 57600 Baud
- 2= 19200 Baud
- 3= 9600 Baud

Description: The serial connection baudrate of the sensor.

Parameter: int SA_ASCII SA_ASCII

Direction: Up

Valid values:

- 0= off (binary 2 bytes/value)
- 1= on (ASCII 6 bytes/value)

Description: Returns the mode the sensor is sending data (only values).

Parameter: int SA_Upper_limit SA_Upper_limit

Direction: Up

Valid values:

- Minimum:** 0
- Maximum:** 16368

Description: The upper limit of the sensor.

Parameter: int SA_Lower_limit SA_Lower_limit

Direction: Up

Valid values:

- Minimum:** 0
- Maximum:** 16368

Description: The lower limit of the sensor.

Parameter: int SA_Upper_hysteresis SA_Upper_hysteresis

Direction: Up

Valid values:

- Minimum:** 0
- Maximum:** 16368

Description: The upper hysteresis of the sensor.

10.4. Commands for ILD1700

Parameter: int SA_Lower_hysteresis	SA_Lower_hysteresis
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 16368	
Description: The lower hysteresis of the sensor.	
Parameter: int SA_Master_value	SA_Master_value
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 16368	
Description: The master value of the sensor.	
Parameter: int SA_Master_MidPoint_Setup	SA_Master_MidPoint_Setup
Direction: Up	
Valid values:	
Condition: switch mode: 0/2= Not mastered 1/3= Mastered	
Valid values:	
Condition: error mode: 0/1= mid-point value not set 2/3= mid-point value set	
Description: The master and midpoint of the sensor.	
Parameter: double SA_Range	SA_Range
Direction: Up	
Valid values:	
Minimum: 0.0	
Unit: mm	
Description: Range of the sensor.	
Parameter: int SA_AssignLimits_ErrorOutput	SA_AssignLimits_ErrorOutput
Direction: Up	
Valid values:	
0= Set_LowerLimit_F1 1= Set_UpperLimit_F1	
Description: The assignment of the error outputs of the sensor.	
Parameter: int SA_Keylock	SA_Keylock
Direction: Up	
Valid values:	
0= off (keys enabled) 1= on (keys locked)	
Description: The keypad state at the sensor.	
Parameter: int SA_DatOut	SA_DatOut
Direction: Up	
Valid values:	
0= Dat_Out_Off 1= Dat_Out_On	
Description: Data output from sensor.	

Parameter: int SA_LaserState SA_LaserState

Direction: Up

Valid values:

- 0= Laser_Off
- 1= Laser_On

Description: Laser state of sensor.

Parameter: int SA_EnableFlash SA_EnableFlash

Direction: Up

Valid values:

- 0= locked
- 1= enabled

Description: The flash is locked or enabled for writing.

10.4.1.1.3 Reset_Boot (RESET_BOOT)

Resets the sensor.

10.4.1.2 User level

10.4.1.2.1 Set_KeyLock (SET_KEYLOCK)

Locks/Unlocks the keypad of sensor.

Parameter: int SP_Keylock SP_Keylock

Direction: Down

Valid values:

- 0= off (keys enabled)
- 1= on (keys locked)

Description: The keypad state at the sensor.

10.4.1.3 Interfaces

10.4.1.3.1 Set_Baudrate (SET_BAUDRATE)

Set the baudrate of the serial interface of sensor. After receiving the answer of the command, the baudrate of the local serial interface is changed, too.

Attention! To avoid synchronisation problems, sending data by sensor should be disabled first (sensor command Dat_Out_Off).

Parameter: int SP_SensorBaudrate SP_SensorBaudrate

Direction: Down

Valid values:

- 0= 115200 Baud
- 1= 57600 Baud
- 2= 19200 Baud
- 3= 9600 Baud

Description: The serial connection baudrate of the sensor.

Parameter: int CP_InterruptDataTransfer CP_InterruptDataTransfer
Direction: Down
Valid values:
 0= false
 1= true
Default: 0
Description: After switch baudrate, synchronisation may be impossible if serial line has high load. Enabling this parameter resolves the problem by disabling data transfer first ([Dat_Out_Off](#)) and enabling it again ([Dat_Out_On](#)) at end.

10.4.1.4 Parameter management

10.4.1.4.1 Set_Default (SET_DEFAULT)

Resets the sensor to factory settings.
 If first bit of [IP_AutomaticMode](#) is set (1), [Get_Settings](#) is called automatically after this command. Otherwise, you have to call it manually.

10.4.1.4.2 WriteFlashZero (WriteFlashZero)

Locks/enables the flash of sensor for writing.

Parameter: int SP_EnableFlash SP_EnableFlash
Direction: Down
Valid values:
 0= locked
 1= enabled
Description: The flash is locked or enabled for writing.

10.4.1.5 Internal controller commands

10.4.1.5.1 Laser_Off (LASER_OFF)

Switch the laser off.

10.4.1.5.2 Laser_On (LASER_ON)

Switch the laser on.

10.4.1.5.3 Set_Sync_TrgMode (SET_SYNCMODE/TRIGGERMODE)

Set the synchronisation and trigger mode respectively.

Parameter: int SP_Sync_TrgMode

SP_Sync_TrgMode

Direction: Down

Valid values:

Condition: at sync mode (ErrorOutput is 0 or 1):

0= synchronous master off

1= synchronous master on

2= slave

3= alternating synchronous master

Valid values:

Condition: at trigger mode (ErrorOutput is 2 or 3):

0= edge L/H

1= edge H/L

2= level H

3= level L

Description: The sync mode or the trigger mode of the sensor.

10.4.1.5.4 Set_ErrorOutput (SET_ERROROUTPUT)

Set the synchronisation or trigger mode and set the error or switch mode.

Parameter: int SP_ErrorOutput

SP_ErrorOutput

Direction: Down

Valid values:

0= sync error mode

1= sync switch mode

2= trigger error mode

3= trigger switch mode

Description: Sync/trigger and error/switch mode respectively of the sensor.

10.4.2 Measurement

10.4.2.1 Set_MeasureMode

Set the measure mode of sensor. This command is only available for sensors with option 12

Parameter: int SP_MeasureMode

SP_MeasureMode

Direction: Down

Valid values:

0= Diffuse reflexion

1= Direct reflexion

Description: The measure mode of the sensor.

10.4.2.2 Set_Speed (SET_SPEED)

Set the data acquisition speed of the sensor.

Parameter: int SP_Speed

SP_Speed

Direction: Down

Valid values:

0= 2.5kHz

1= 1.25kHz

2= 625Hz

3= 312.5Hz

Description: The output speed of the sensor.

10.4.2.3 Set_VideoMode

Enter/Leave the video mode of the sensor.

Parameter: int SP_VideoMode

SP_VideoMode

Direction: Down

Valid values:

0= off

1= on

Description: Switch video mode on or off.

10.4.2.4 Get_Video

Get recent video signal from sensor.

Parameter: Binary data SA_VideoSignal

SA_VideoSignal

Direction: Up

Valid values:

512 bytes, each byte is an intensity value.

Description: Raw video signal

10.4.2.5 Set_Av0 (SET_AV0)

Set averaging index AvIndex= 0, Averaging N= 1.

10.4.2.6 Set_Av1 (SET_AV1)

Set averaging index AvIndex= 2, Averaging N= 4.

10.4.2.7 Set_Av2 (SET_AV2)

Set averaging index AvIndex= 5, Averaging N= 32.

10.4.2.8 Set_Av3 (SET_AV3)

Set averaging index AvIndex= 7, Averaging N= 128.

10.4.2.9 Set_AvX (SET_AVX)

Set averaging index of sensor.

Parameter: int SP_AvIndex SP_AvIndex
Direction: Down
Valid values:
Minimum: 0
Maximum: 15
Description: The averaging index. Averaging $N = 2^{AvIndex}$.

10.4.2.10 Set_Av_T (SET_AV_T)

Set averaging type of sensor.

Parameter: int SP_AvType SP_AvType
Direction: Down
Valid values:
 0= recursive
 1= moving
 2= Median
Description: The averaging type.

10.4.2.11 Get_MeasValue (GET_MEASVALUE)

Is only useful in trigger mode (ErrorOutput is 2 or 3).

Tells the sensor to measure values without hardware trigger condition (software trigger).

Parameter: int SP_MeasValue SP_MeasValue
Direction: Down
Valid values:
Minimum: 1
Maximum: 2147483647 (INT_MAX)
Description: The number of values to measure.

10.4.3 Data output

10.4.3.1 General

10.4.3.1.1 Dat_Out_Off (DAT_OUT_OFF)

Switch off data output from sensor.

10.4.3.1.2 Dat_Out_On (DAT_OUT_ON)

Switch on data output from sensor.

10.4.3.1.3 Set_ErrorHandler (SET_ERRORHANDLER)

Set the behaviour on invalid values at sensor.

Parameter: int SP_ErrorHandler

SP_ErrorHandler

Direction: Down

Valid values:

0= error values

1= hold last value

Description: If the sensor cannot measure values, it can output the last valid value or it can output an error values.

10.4.3.1.4 ASCII_Output (ASCII_OUTPUT)

Set digital data transfer (only values, no sensor answer) to ASCII or binary.

Parameter: int SP_ASCII

SP_ASCII

Direction: Down

Valid values:

0= off (binary 2 bytes/value)

1= on (ASCII 6 bytes/value)

Description: Returns the mode the sensor is sending data (only values).

10.4.3.1.5 Set_OutputType (SET_OUTPUTTYP)

Set the output type of sensor.

Parameter: int SP_OutputType

SP_OutputType

Direction: Down

Valid values:

0= current (4..20mA)

1= voltage (0..10V)

2= RS422

Description: Data output (only values, not answer) interface of the sensor.

10.4.3.2 Switching outputs

10.4.3.2.1 Set_Limits (SET_LIMITS)

Set sensor limits.

Parameter: int SP_Upper_limit

SP_Upper_limit

Direction: Down

Valid values:

Minimum: 0

Maximum: 16368

Description: Upper limit.

Parameter: int SP_Lower_limit	SP_Lower_limit
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 16368	
Description: Lower limit.	
Parameter: int SP_Upper_hysteresis	SP_Upper_hysteresis
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 16368	
Description: Upper hysteresis.	
Parameter: int SP_Lower_hysteresis	SP_Lower_hysteresis
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 16368	
Description: Lower hysteresis.	
Parameter: int SP_Master_value	SP_Master_value
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 16368	
Description: Master value.	

10.4.3.2.2 Set_UpperLimit_F1 (SET_UPPERLIMIT_F1)

Assign the upper limit of sensor to error output 1 and lower limit to error output 2.

10.4.3.2.3 Set_LowerLimit_F1 (SET_LOWERLIMIT_F1)

Assign the lower limit of sensor to error output 1 and upper limit to error output 2.

10.4.3.3 Analog output

10.4.3.3.1 Zero (SET_ZERO)

Autozero the analog output value.

Parameter: int SP_Zero	SP_Zero
Direction: Down	
Valid values:	
0= reset	
1= set	
Description: Set or reset autozero mode.	

10.5 Commands for ILD2200

See sensor manual for detailed description of sensor commands. Driver for ILD2200 also includes ILD2220 support.

This sensor supports following interfaces:

- [RS232](#) (additional, e.g. IF2001_USB (RS422) and RS232 high level interface).
- [IF2004](#) (native).
- [TCP/IP](#) (additional, e.g. RS232/RS422 to TCP/IP comm server and RS232 high level interface).
- [IF2004_USB](#) (native).
- [IF2008](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_Settings](#) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib to calculate datarate and to scale data.

If second bit of [IP_AutomaticMode](#) is set (2), MEDAQLib calls sensor command [Dat_Out_On](#) and [Laser_On](#) at [OpenSensor](#).

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor (including two unused bits in high byte, mask it out by raw&0xffff), from 0 to 65535 (without unused bits).
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command [Get_Settings](#)), error values are scaled depending of [IP_ScaleErrorValues](#).

If [Transmit_Intensity](#) is enabled, distance and intensity values are filled in the arrays alternating. Each array always starts with an distance value.

10.5.1 General commands

10.5.1.1 General

10.5.1.1.1 Get_Info (INFO)

Retrieve some information about the sensor.

Parameter: String SA_Sensor SA_Sensor
Direction: Up
Description: Name of the sensor.

Parameter: String SA_SensorType SA_SensorType
Direction: Up
Description: Type of the sensor.

Parameter: double SA_Samplerate	SA_Samplerate
Direction: Up	
Valid values:	
20000	
10000	
5000	
2500	
Unit: Hz	
Description: The output speed of the sensor.	
Parameter: double SA_Range	SA_Range
Direction: Up	
Valid values:	
Minimum: 0.0	
Unit: mm	
Description: Range of the sensor.	
Parameter: String SA_Option	SA_Option
Direction: Up	
Valid values:	
Numeric value	
Description: Option of the sensor.	
Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the sensor.	
Parameter: int SA_AvIndex	SA_AvIndex
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 15	
Description: The averaging index. Averaging $N = 2^{AvIndex}$.	
Parameter: int SA_Output_Analog	SA_Output_Analog
Direction: Up	
Valid values:	
0= not avail	
1= voltage (+-5V)	
Description: The analog output type.	
Parameter: int SA_Output_Digital	SA_Output_Digital
Direction: Up	
Valid values:	
0= not avail	
2= RS422	
Description: The digital output type.	

10.5.1.1.2 Get_Settings (Get_Settings)

Retrieve detailed information about the sensor.

Parameter: int SA_Speed SA_Speed

Direction: Up

Valid values:

0= 10kHz

1= 5kHz

2= 2.5kHz

3= 20kHz

Description: The output speed of the sensor.

Parameter: int SA_AvIndex SA_AvIndex

Direction: Up

Valid values:

Minimum: 0

Maximum: 15

Description: The averaging index. Averaging $N = 2^{AvIndex}$.

Parameter: int SA_ErrorHandler SA_ErrorHandler

Direction: Up

Valid values:

0= error values

1= hold last value

Description: If the sensor cannot measure values, it can output the last valid value or it can output an error values.

Parameter: int SA_AvType SA_AvType

Direction: Up

Valid values:

0= recursive

1= moving

2= Median

Description: The averaging type.

Parameter: double SA_OffsetValue SA_OffsetValue

Direction: Up

Valid values:

Minimum: -50.0

Maximum: +50.0

Unit: %

Description: The offset value of the sensor.

Parameter: int SA_ZeroPoint SA_ZeroPoint

Direction: Up

Valid values:

0= absolute

1= relative (zero is set)

Description: Autozero off/on.

Parameter: double SA_Range	SA_Range
Direction: Up	
Valid values:	
Minimum: 0.0	
Unit: mm	
Description: Range of the sensor.	
Parameter: int SA_Keylock	SA_Keylock
Direction: Up	
Valid values:	
0= off (keys enabled)	
1= on (keys locked)	
Description: The keypad state at the sensor.	
Parameter: int SA_DatOut	SA_DatOut
Direction: Up	
Valid values:	
0= Dat_Out_Off	
1= Dat_Out_On	
Description: Data output from sensor.	
Parameter: int SA_LaserState	SA_LaserState
Direction: Up	
Valid values:	
0= Laser_Off	
1= Laser_On	
Description: Laser state of sensor.	

10.5.1.1.3 Get_Version

Retrieve the sensor software version.

Parameter: String SA_Softwareversion	SA_Softwareversion
Direction: Up	
Description: Software version of the sensor.	

10.5.1.1.4 Reset_Boot (RESET)

Resets the sensor.

10.5.1.2 User level

10.5.1.2.1 Set_KeyLock (SET_TASTENSPERRE)

Locks/Unlocks the keypad of sensor.

Parameter: int SP_Keylock	SP_Keylock
Direction: Down	
Valid values:	
0= off (keys enabled)	
1= on (keys locked)	
Description: The keypad state at the sensor.	

10.5.1.3 Internal controller commands

10.5.1.3.1 Laser_Off (LASER_OFF)

Switch off the laser.

10.5.1.3.2 Laser_On (LASER_ON)

Switch on the laser.

10.5.2 Measurement

10.5.2.1 Set_Av0 (AVG 0)

Set averaging index AvIndex= 0, Averaging N= 1.

10.5.2.2 Set_Av1 (AVG 1)

Set averaging index AvIndex= 2, Averaging N= 4.

10.5.2.3 Set_Av2 (AVG 2)

Set averaging index AvIndex= 5, Averaging N= 32.

10.5.2.4 Set_Av3 (AVG 3)

Set averaging index AvIndex= 7, Averaging N= 128.

10.5.2.5 Set_AvX (AVG n)

Set averaging index of sensor.

Parameter: int SP_AvIndex SP_AvIndex

Direction: Down

Valid values:

Minimum: 0

Maximum: 15

Description: The averaging index. Averaging $N = 2^{AvIndex}$.

10.5.2.6 Set_Av_T (AVGTYP)

Set averaging type of sensor.

Parameter: int SP_AvType SP_AvType

Direction: Down

Valid values:

0= recursive

1= moving

2= Median

Description: The averaging type.

10.5.3 Data output

10.5.3.1 General

10.5.3.1.1 Dat_Out_Off (STOP)

Switch off data output from sensor.

10.5.3.1.2 Dat_Out_On (START)

Switch on data output from sensor.

10.5.3.2 Switching outputs

10.5.3.2.1 Transmit_Intensity

Tells the sensor to transmit the intesity value after each distance value.

This command is only available with sensor option 204 and 243.

Option 204: This setting will not be stored persistently in the sensor so it has to be set after each reset or power on.

Option 243: This setting is stored persistently in the sensor. But to tell MEDAQLib about this you have to set it after connecting to sensor.

Parameter: int SP_TransmitIntensity

SP_TransmitIntensity

Direction: Down

Valid values:

0 = no

1 = yes

Description: Transmit intensitiy value.

10.5.3.3 Analog output

10.5.3.3.1 Zero (ZERO)

Autozero the analog output value.

10.6 Commands for ILD2300

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [RS232](#) (additional, e.g. IF2001_USB (RS422) and RS232 high level interface).
- [IF2004](#) (native).
- [TCP/IP](#) (native).
- [IF2004_USB](#) (native).

- [IF2008](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_AllParameters](#) (SP_Additional= 1) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib timeout checking, to calculate datarate, to interpret and scale data and to assign values. If second bit of [IP_AutomaticMode](#) is set (2), MEDAQLib calls optionally sensor command [Set_IPDataTransferMode](#), [Set_DataOutInterface](#), [Get_LaserPower](#) and optionally [Set_LaserPower](#) at [OpenSensor](#).

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor.
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command [Get_AllParameters](#) (SP_Additional= 1)).

The values of one data frame are filled in the arrays one after another. Each array always starts with a new data frame.

10.6.1 General commands

10.6.1.1 General

10.6.1.1.1 Get_Help (HELP)

Retrieve a help text from sensor for a specific command.

Parameter: String SP_Command	SP_Command
Direction: Down	
Valid values:	
"" (empty string, means general help) or any command name	
Description: Name of the command.	

Parameter: String SA_HelpText	SA_HelpText
Direction: Up	
Description: Help text to the command.	

10.6.1.1.2 Get_Info (GETINFO)

Retrieve information about the sensor.

Parameter: String SA_Sensor	SA_Sensor
Direction: Up	
Description: Name of the sensor.	

Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the sensor.	
Parameter: String SA_Option	SA_Option
Direction: Up	
Valid values:	
Numeric value	
Description: Option of the sensor.	
Parameter: String SA_ArticleNumber	SA_ArticleNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Article number of the sensor.	
Parameter: String SA_MacAddress	SA_MacAddress
Direction: Up	
Valid values:	
Valid MAC address in form of xx-xx-xx-xx-xx-xx	
Description: MAC address (low level ethernet address) of the sensor.	
Parameter: String SA_Softwareversion	SA_Softwareversion
Direction: Up	
Description: Software version of firmware in the sensor.	
Parameter: String SA_ImageType	SA_ImageType
Direction: Up	
Description: Firmware image type used by the sensor.	
Parameter: double SA_Range	SA_Range
Direction: Up	
Valid values:	
Minimum: 0.0	
Unit: mm	
Description: Range of the sensor.	
Parameter: String SA_CalibrationTable	SA_CalibrationTable
Direction: Up	
Description: Name of calibration table which is actually used.	

10.6.1.1.3 Get_OutputInfo_RS422 (GETOUTINFO_RS422)

Retrieve information which data is output at RS422 interface.

Parameter: int SA_OutputAdditionalShutterTime_RS422	SA_OutputAdditionalShutterTime_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	

Parameter: int SA_OutputAdditionalCounter_RS422	SA_OutputAdditionalCounter_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SA_OutputAdditionalTimestamp_RS422	SA_OutputAdditionalTimestamp_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SA_OutputAdditionalTemperature_RS422	SA_OutputAdditionalTemperature_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if temperature is transmitted.	
Parameter: int SA_OutputAdditionalIntensity1_RS422	SA_OutputAdditionalIntensity1_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity 1 is transmitted.	
Parameter: int SA_OutputAdditionalIntensity2_RS422	SA_OutputAdditionalIntensity2_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity 2 is transmitted.	
Parameter: int SA_OutputDistance1_RS422	SA_OutputDistance1_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if distance 1 is transmitted.	
Parameter: int SA_OutputDistance2_RS422	SA_OutputDistance2_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if distance 2 is transmitted.	
Parameter: int SA_OutputAdditionalState_RS422	SA_OutputAdditionalState_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	

Parameter: int SA_OutputThickness12_RS422	SA_OutputThickness12_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if thickness between first and second peak is transmitted.	
Parameter: int SA_OutputStatisticMin_RS422	SA_OutputStatisticMin_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if min value is transmitted.	
Parameter: int SA_OutputStatisticMax_RS422	SA_OutputStatisticMax_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if max value is transmitted.	
Parameter: int SA_OutputStatisticPeak2Peak_RS422	SA_OutputStatistic- Peak2Peak_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if peak to peak value is transmitted.	

10.6.1.1.4 Get_OutputInfo_ETH (GETOUTINFO_ETH)

Retrieve information which data is output at ETH interface.

Parameter: int SA_OutputAdditionalShutterTime_ETH	SA_OutputAdditionalShut- terTime_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	
Parameter: int SA_OutputAdditionalCounter_ETH	SA_OutputAdditional- Counter_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SA_OutputAdditionalTimestamp_ETH	SA_OutputAdditionalTimes- tamp_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	

Parameter: int SA_OutputAdditionalTemperature_ETH	SA_OutputAdditionalTemperature_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if temperature is transmitted.	
Parameter: int SA_OutputAdditionalIntensity1_ETH	SA_OutputAdditionalIntensity1_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity 1 is transmitted.	
Parameter: int SA_OutputAdditionalIntensity2_ETH	SA_OutputAdditionalIntensity2_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity 2 is transmitted.	
Parameter: int SA_OutputDistance1_ETH	SA_OutputDistance1_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if distance 1 is transmitted.	
Parameter: int SA_OutputDistance2_ETH	SA_OutputDistance2_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if distance 2 is transmitted.	
Parameter: int SA_OutputAdditionalState_ETH	SA_OutputAdditionalState_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	
Parameter: int SA_OutputAdditionalTrgCounter_ETH	SA_OutputAdditionalTrgCounter_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if trigger counter is transmitted.	
Parameter: int SA_OutputThickness12_ETH	SA_OutputThickness12_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if thickness between first and second peak is transmitted.	

Parameter: int SA_OutputStatisticMin_ETH	SA_OutputStatisticMin_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if min value is transmitted.	
Parameter: int SA_OutputStatisticMax_ETH	SA_OutputStatisticMax_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if max value is transmitted.	
Parameter: int SA_OutputStatisticPeak2Peak_ETH	SA_OutputStatisticPeak2Peak_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if peak to peak value is transmitted.	

10.6.1.1.5 Set_SyncMode (SYNC)

Set the synchronisation mode.

Parameter: int SP_SyncMode	SP_SyncMode
Direction: Down	
Valid values:	
0= None	
1= Slave	
2= Master	
3= Master alternating (MASTER_ALT)	
Description: Synchronisation mode.	
Parameter: int SP_SyncTermination	SP_SyncTermination
Direction: Down	
Valid values:	
0= Off (TERMOFF)	
1= On 120 Ohm (TERMON)	
Description: Termination of external input.	

10.6.1.1.6 Get_SyncMode (SYNC)

Get the synchronisation mode.

Parameter: int SA_SyncMode	SA_SyncMode
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Slave	
2= Master	
3= Master alternating (MASTER_ALT)	
Description: Synchronisation mode.	

Parameter: int SA_SyncTermination SA_SyncTermination
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = Off (TERMOFF)
 1 = On 120 Ohm (TERMON)
Description: Termination of external input.

10.6.1.1.7 Set_Unit (UNIT)

Set the unit for configuration and display in the web diagram.

Parameter: int SP_DisplayUnit SP_DisplayUnit
Direction: Down
Valid values:
 0 = mm
 1 = Inch
Description: Unit.

10.6.1.1.8 Get_Unit (UNIT)

Get the unit for configuration and display in the web diagram.

Parameter: int SA_DisplayUnit SA_DisplayUnit
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = mm
 1 = Inch
Description: Unit.

10.6.1.1.9 Reset_Boot (RESET)

Resets the sensor.

10.6.1.1.10 Reset_Counter (RESETCNT)

Resets sensor counter values.

Parameter: int SP_ResetTimestamp SP_ResetTimestamp
Direction: Down
Valid values:
 0 = No
 1 = Yes
Description: Reset timestamp value.

Parameter: int SP_ResetMeasCounter	SP_ResetMeasCounter
Direction: Down	
Valid values:	
0= No	
1= Yes	
Description: Reset counter value.	
Parameter: int SP_ResetTriggerCounter	SP_ResetTriggerCounter
Direction: Down	
Valid values:	
0= No	
1= Yes	
Description: Reset trigger counter value.	

10.6.1.1.11 Set_Echo (ECHO)

Set echo for sensor commands.

Parameter: int SP_Echo	SP_Echo
Direction: Down	
Valid values:	
0= Off	
1= On	
Description: Echo mode.	

10.6.1.1.12 Get_Echo (ECHO)

Get the echo mode.

Parameter: int SA_Echo	SA_Echo
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Off	
1= On	
Description: Echo mode.	

10.6.1.1.13 Get_AllParameters (PRINT)

Get all parameters from sensor.

Parameter: int SP_Additional	SP_Additional
Direction: Down	
Valid values:	
0= No	
1= Yes	
Description: If set, additional information about sensor, sensor and material is output.	

Parameter: int SA_SyncMode SA_SyncMode

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = None
- 1 = Slave
- 2 = Master
- 3 = Master alternating (MASTER_ALT)

Description: Synchronisation mode.

Parameter: int SA_SyncTermination SA_SyncTermination

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = Off (TERMOFF)
- 1 = On 120 Ohm (TERMON)

Description: Specifies if termination should be activated.

Parameter: int SA_UserLevel SA_UserLevel

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = User
- 1 = Professional

Description: Actual user level.

Parameter: int SA_DefaultUser SA_DefaultUser

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = User
- 1 = Professional

Description: Default user level.

Parameter: int SA_Echo SA_Echo

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = Off
- 1 = On

Description: Echo mode.

Parameter: int SA_DisplayUnit SA_DisplayUnit

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = mm
- 1 = Inch

Description: Unit.

Parameter: int SA_TriggerMode	SA_TriggerMode
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Edge	
2= Level (PULSE)	
3= Software	
Description: Trigger mode.	
Parameter: int SA_TriggerTermination	SA_TriggerTermination
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Off (TERMOFF)	
1= On 120 Ohm (TERMON)	
Description: Specifies if termination should be activated.	
Parameter: int SA_TriggerMoment	SA_TriggerMoment
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Input	
1= Output	
Description: Trigger moment.	
Parameter: int SA_TriggerLevel	SA_TriggerLevel
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= High	
1= Low	
Description: Trigger level.	
Parameter: int SA_TriggerCount	SA_TriggerCount
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 16383	
Description: Number of values to measure. 0 means no trigger, 16383 means endless measurement.	
Parameter: int SA_TriggerOutput	SA_TriggerOutput
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Triggered values (TRIGGERED)	
1= All values (ALL)	
Description: Output only triggered values or all values.	
Parameter: int SA_EthernetMode	SA_EthernetMode
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Ethernet	
1= Ethercat	
Description: Ethernet mode.	

Parameter: int SA_DHCPEnabled	SA_DHCPEnabled
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = FALSE	
1 = TRUE	
Description: Get settings if sensor should use a static IP address ask for IP at DHCP server (dynamic IP address).	
Parameter: String SA_Address	SA_Address
Direction: Up	
Valid values:	
Valid IP address in form of xxx.xxx.xxx.xxx	
Description: IP address of the sensor. If DHCP is enabled it returns the currently assigned IP address.	
Parameter: String SA_SubnetMask	SA_SubnetMask
Direction: Up	
Valid values:	
Valid network mask (e.g. 255.255.255.0 for a Class C network)	
Description: Network mask of the sensor. If DHCP is enabled it returns the currently assigned network mask.	
Parameter: String SA_Gateway	SA_Gateway
Direction: Up	
Valid values:	
Valid IP address of default gateway in form of xxx.xxx.xxx.xxx	
Description: Address of the default gateway. If DHCP is enabled it returns the currently assigned default gateway.	
Parameter: int SA_Protocol	SA_Protocol
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = TCP server (SERVER/TCP)	
1 = TCP client (CLIENT/TCP)	
2 = UDP sender (CLIENT/UDP)	
3 = None	
Description: Specifies if data should be send using TCP or UDP.	
Parameter: String SA_RemoteAddress	SA_RemoteAddress
Direction: Up	
Valid values:	
Valid IP address of receiver of data	
Description: Address of remote computer to send data to.	
Parameter: int SA_Port	SA_Port
Direction: Up	
Valid values:	
Minimum: 1024	
Maximum: 65535	
Description: Port to send data to or to listen for incoming requests.	

Parameter: int SA_SensorBaudrate	SA_SensorBaudrate
Direction: Up	
Valid values:	
4000000	
3500000	
3000000	
2500000	
2000000	
1500000	
921600	
691200	
460800	
230400	
115200	
9600	
Unit: Baud	
Description: Baudrate of sensor.	
Parameter: int SA_ActiveParameterSet	SA_ActiveParameterSet
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 8	
Description: Number of the currently active parameter set in RAM.	
Parameter: int SA_MeasurePeak	SA_MeasurePeak
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Greatest Amplitude (DISTA)	
1= Greatest Area (DISTW)	
2= First Peak (DIST1)	
Description: Peak to measure.	
Parameter: int SA_MeasureMode	SA_MeasureMode
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Distance diffuse (DIST_DIFFUSE)	
1= Distance direct (DIST_DIRECT)	
2= Thickness	
3= Video	
Description: Measure mode.	
Parameter: int SA_ShutterMode	SA_ShutterMode
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Exposure time is adapted automatically (MEAS)	
1= Exposure time is set manually (MANUAL)	
2= Use two fixed exposure times alternately (2TIMEALT)	
3= Use adapted time and factor * adapted time (see Set_ShutterFactor) alternately (AUTOTIMEALT)	
Description: Shutter mode.	

Parameter: double SA_Measrate SA_Measrate

Direction: Up

Valid values:

- 1.5
- 2.5
- 5.0
- 10.0
- 20.0
- 30.0
- 49.0

Unit: kHz

Description: Samplerate of measurement.

Parameter: double SA_ShutterTime1 SA_ShutterTime1

Direction: Up

Valid values:

- Minimum:** 0.1
- Maximum:** 512.0

Unit: μ s

Description: First shutter time.

Parameter: double SA_ShutterTime2 SA_ShutterTime2

Direction: Up

Valid values:

- Minimum:** 0.1
- Maximum:** 512.0

Unit: μ s

Description: Second shutter time.

Parameter: int SA_LaserPower SA_LaserPower

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = Full
- 1 = Reduced
- 2 = Off

Description: Laser power.

Parameter: int SA_ROIStart SA_ROIStart

Direction: Up

Valid values:

- Minimum:** 0
- Maximum:** 511

Unit: Pixel

Description: First position on CCD.

Parameter: int SA_ROIEnd SA_ROIEnd

Direction: Up

Valid values:

- Minimum:** 0
- Maximum:** 511

Unit: Pixel

Description: Last position on CCD.

Parameter: int SA_VideoAverage	SA_VideoAverage
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Recursive over 2 lines (REC2)	
2= Recursive over 4 lines (REC4)	
3= Recursive over 8 lines (REC8)	
4= Moving over 2 lines (MOV2)	
5= Moving over 3 lines (MOV3)	
6= Moving over 4 lines (MOV4)	
7= Median over 3 lines (MED3)	
Description: Averaging mode.	
Parameter: double SA_Threshold	SA_Threshold
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 99.0	
Unit: %	
Description: Video threshold.	
Parameter: String SA_ActiveMaterial	SA_ActiveMaterial
Direction: Up	
Description: Name of material.	
Parameter: int SA_AveragingType	SA_AveragingType
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Moving average (MOVING)	
2= Recursive averaging (RECURSIVE)	
3= Median	
Description: Averaging type.	
Parameter: int SA_MovingCount	SA_MovingCount
Direction: Up	
Valid values:	
2	
4	
8	
16	
32	
64	
128	
Description: Number of value for the averaging window. This parameter is only available at moving average.	
Parameter: int SA_RecursiveCount	SA_RecursiveCount
Direction: Up	
Valid values:	
Minimum: 2	
Maximum: 32768	
Description: Number of values for recursive averaging. This parameter is only available at recursive average.	

Parameter: int SA_MedianCount	SA_MedianCount
Direction: Up	
Valid values:	
3	
5	
7	
9	
Description: Number of values to build median. This parameter is only available at median.	
Parameter: int SA_SpikeCorrection	SA_SpikeCorrection
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = off	
1 = on	
Description: Spike correction.	
Parameter: int SA_NbrEvaluatedValues	SA_NbrEvaluatedValues
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 10	
Description: Number of values to evaluate for spike correction.	
Parameter: double SA_ToleranceRange	SA_ToleranceRange
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 100.0	
Description: Tolerance range for spike correction.	
Parameter: int SA_NbrCorrectedValues	SA_NbrCorrectedValues
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 100	
Description: Number of values to correct at spike correction.	
Parameter: int SA_StatisticDepth	SA_StatisticDepth
Direction: Up	
Valid values:	
Minimum: 2	
Maximum: 2147483647 (INT_MAX)	
Description: Window size for floating statistic calculation. The value must be power of two (2, 4, 8, ..., 16384). Value greater as 16384 calculates statistic over all values.	
Parameter: int SA_Master	SA_Master
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = no (NONE)	
1 = yes (MASTER=	
Description: Specifies if mastering is active.	

Parameter: double SA_MasterValue	SA_MasterValue
Direction: Up	
Valid values:	
Minimum: -2* measuring range	
Maximum: +2* measuring range	
Unit: mm	
Description: Master value	
Parameter: int SA_DataOutInterface	SA_DataOutInterface
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= RS422	
2= Ethernet	
3= HTTP	
Description: Active interface for data output.	
Parameter: int SA_Resampling	SA_Resampling
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 3000000	
Description: Resampling value.	
Parameter: int SA_ResampleRS422	SA_ResampleRS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: RS422 output is resampled.	
Parameter: int SA_ResampleEthernet	SA_ResampleEthernet
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Output over ethernet is resampled.	
Parameter: int SA_HoldLastValid	SA_HoldLastValid
Direction: Up	
Valid values:	
Minimum: -1	
Maximum: 1024	
Description: Values to replace by last valid value. -1 means no value to hold, 0 means never output an error value (always hold last valid value).	
Parameter: double SA_MeasureValueCnt	SA_MeasureValueCnt
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 4294967295.0 (UINT_MAX)	
Description: Number of values to be output. 0 means no output, 4294967295 means continuous output.	

Parameter: int SA_OutputAdditionalIntensity1_RS422	SA_OutputAdditionalIntensity1_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity 1 is transmitted.	
Parameter: int SA_OutputAdditionalIntensity2_RS422	SA_OutputAdditionalIntensity2_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity 2 is transmitted.	
Parameter: int SA_OutputDistance1_RS422	SA_OutputDistance1_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if distance 1 is transmitted.	
Parameter: int SA_OutputDistance2_RS422	SA_OutputDistance2_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if distance 2 is transmitted.	
Parameter: int SA_OutputThickness12_RS422	SA_OutputThickness12_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if thickness between first and second peak is transmitted.	
Parameter: int SA_OutputStatisticMin_RS422	SA_OutputStatisticMin_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if min value is transmitted.	
Parameter: int SA_OutputStatisticMax_RS422	SA_OutputStatisticMax_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if max value is transmitted.	
Parameter: int SA_OutputStatisticPeak2Peak_RS422	SA_OutputStatisticPeak2Peak_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if peak to peak value is transmitted.	

Parameter: int SA_OutputThickness12_ETH	SA_OutputThickness12_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if thickness between first and second peak is transmitted.	
Parameter: int SA_OutputStatisticMin_ETH	SA_OutputStatisticMin_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if min value is transmitted.	
Parameter: int SA_OutputStatisticMax_ETH	SA_OutputStatisticMax_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if max value is transmitted.	
Parameter: int SA_OutputStatisticPeak2Peak_ETH	SA_OutputStatisticPeak2Peak_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if peak to peak value is transmitted.	
Parameter: int SA_OutputAdditionalShutterTime_RS422	SA_OutputAdditionalShutterTime_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	
Parameter: int SA_OutputAdditionalCounter_RS422	SA_OutputAdditionalCounter_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SA_OutputAdditionalTimestamp_RS422	SA_OutputAdditionalTimestamp_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SA_OutputAdditionalIntensity_RS422	SA_OutputAdditionalIntensity_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity is transmitted.	

Parameter: int SA_OutputAdditionalState_RS422	SA_OutputAdditionalState_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	
Parameter: int SA_OutputAdditionalTemperature_RS422	SA_OutputAdditionalTemperature_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if temperature is transmitted.	
Parameter: int SA_OutputAdditionalIntensity1_ETHERNET	SA_OutputAdditionalIntensity1_ETHERNET
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity 1 is transmitted.	
Parameter: int SA_OutputAdditionalIntensity2_ETHERNET	SA_OutputAdditionalIntensity2_ETHERNET
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity 2 is transmitted.	
Parameter: int SA_OutputDistance1_ETHERNET	SA_OutputDistance1_ETHERNET
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if distance 1 is transmitted.	
Parameter: int SA_OutputDistance2_ETHERNET	SA_OutputDistance2_ETHERNET
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if distance 2 is transmitted.	
Parameter: int SA_OutputAdditionalShutterTime_ETHERNET	SA_OutputAdditionalShutterTime_ETHERNET
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	
Parameter: int SA_OutputAdditionalCounter_ETHERNET	SA_OutputAdditionalCounter_ETHERNET
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	

Parameter: int SA_OutputAdditionalTimestamp_ETH	SA_OutputAdditionalTimestamp_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SA_OutputAdditionalIntensity_ETH	SA_OutputAdditionalIntensity_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity is transmitted.	
Parameter: int SA_OutputAdditionalState_ETH	SA_OutputAdditionalState_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	
Parameter: int SA_OutputAdditionalTrgCounter_ETH	SA_OutputAdditionalTrgCounter_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if trigger counter is transmitted.	
Parameter: int SA_OutputAdditionalTemperature_ETH	SA_OutputAdditionalTemperature_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if temperature is transmitted.	
Parameter: int SA_OutputVideoRaw_ETH	SA_OutputVideoRaw_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if raw video signal is transmitted.	
Parameter: int SA_OutputVideoCorr_ETH	SA_OutputVideoCorr_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if corrected video signal is transmitted.	
Parameter: String SA_Sensor	SA_Sensor
Direction: Up	
Direction: Up	
Description: Name of the sensor.	

Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the sensor.	
Parameter: String SA_Option	SA_Option
Direction: Up	
Valid values:	
Numeric value	
Description: Option of the sensor.	
Parameter: String SA_ArticleNumber	SA_ArticleNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Article number of the sensor.	
Parameter: String SA_MacAddress	SA_MacAddress
Direction: Up	
Valid values:	
Valid MAC address in form of xx-xx-xx-xx-xx-xx	
Description: MAC address (low level ethernet address) of the sensor.	
Parameter: String SA_Softwareversion	SA_Softwareversion
Direction: Up	
Description: Software version of firmware in the sensor.	
Parameter: String SA_ImageType	SA_ImageType
Direction: Up	
Description: Firmware image type used by the sensor.	
Parameter: double SA_Range	SA_Range
Direction: Up	
Valid values:	
Minimum: 0.0	
Unit: mm	
Description: Range of the sensor.	
Parameter: String SA_CalibrationTable	SA_CalibrationTable
Direction: Up	
Description: Name of calibration table which is actually used.	
Parameter: String SA_MaterialName	SA_MaterialName
Direction: Up	
Description: Name of the active material.	
Parameter: String SA_Description	SA_Description
Direction: Up	
Description: Description of the active material.	
Parameter: double SA_RefractiveIndex_nF	SA_RefractiveIndex_nF
Direction: Up	
Valid values:	
Minimum: 1.0	
Maximum: 4.0	
Description: Refractive index of the active material.	

Parameter: double SA_WaveLength SA_WaveLength
Direction: Up
Unit: nm
Description: The wave length where the refractive index taken from.

10.6.1.2 User level

10.6.1.2.1 Logout (LOGOUT)

Change user level to user.

10.6.1.2.2 Login (LOGIN)

Change user level to professional.

Parameter: String SP_Password SP_Password
Direction: Down
Description: Valid password to login.

10.6.1.2.3 Get_UserLevel (GETUSERLEVEL)

Retrieve actual user level.

Parameter: int SA_UserLevel SA_UserLevel
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = User
 1 = Professional
Description: Actual user level.

10.6.1.2.4 Set_DefaultUser (STDUSER)

Set the default user level after booting the system.

Parameter: int SP_DefaultUser SP_DefaultUser
Direction: Down
Valid values:
 0 = User
 1 = Professional
Description: Default user level.

10.6.1.2.5 Get_DefaultUser (STDUSER)

Get the default user level after booting the system.

Parameter: int SA_DefaultUser SA_DefaultUser
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = User
 1 = Professional
Description: Default user level.

10.6.1.2.6 Set_Password (PASSWD)

Change the password for login.

Parameter: String SP_OldPassword	SP_OldPassword
Direction: Down	
Description: Old password.	
Parameter: String SP_NewPassword	SP_NewPassword
Direction: Down	
Description: New password.	

10.6.1.3 Triggering

10.6.1.3.1 Set_TriggerMode (TRIGGER)

Set the trigger mode.

Parameter: int SP_TriggerMode	SP_TriggerMode
Direction: Down	
Valid values:	
0= None	
1= Edge	
2= Level (PULSE)	
3= Software	
Description: Trigger mode.	
Parameter: int SP_TriggerTermination	SP_TriggerTermination
Direction: Down	
Valid values:	
0= Off (TERMOFF)	
1= On 120 Ohm (TERMON)	
Description: Termination of external input.	

10.6.1.3.2 Get_TriggerMode (TRIGGER)

Get the active trigger mode.

Parameter: int SA_TriggerMode	SA_TriggerMode
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Edge	
2= Level (PULSE)	
3= Software	
Description: Trigger mode.	

Parameter: int SA_TriggerTermination SA_TriggerTermination
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = Off (TERMOFF)
 1 = On 120 Ohm (TERMON)
Description: Termination of external input.

10.6.1.3.3 Set_TriggerMoment (TRIGGERAT)

Set the trigger time.

Parameter: int SP_TriggerMoment SP_TriggerMoment
Direction: Down
Valid values:
 0 = Input
 1 = Output
Description: Trigger moment.

10.6.1.3.4 Get_TriggerMoment (TRIGGERAT)

Get the active trigger time.

Parameter: int SA_TriggerMoment SA_TriggerMoment
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = Input
 1 = Output
Description: Trigger moment.

10.6.1.3.5 Set_TriggerLevel (TRIGGERLEVEL)

Set the trigger level.

Parameter: int SP_TriggerLevel SP_TriggerLevel
Direction: Down
Valid values:
 0 = High
 1 = Low
Description: Trigger level.

10.6.1.3.6 Get_TriggerLevel (TRIGGERLEVEL)

Get the active trigger level.

Parameter: int SA_TriggerLevel SA_TriggerLevel
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = High
 1 = Low
Description: Trigger level.

10.6.1.3.7 Set_TriggerCount (TRIGGERCOUNT)

Set the number of values to measure at trigger.

Parameter: int SP_TriggerCount SP_TriggerCount
Direction: Down
Valid values:
Minimum: 0
Maximum: 16383
Description: Number of values to measure. 0 means no trigger, 16383 means endless measurement.

10.6.1.3.8 Get_TriggerCount (TRIGGERCOUNT)

Get the number of values to measure at trigger.

Parameter: int SA_TriggerCount SA_TriggerCount
Direction: Up
Valid values:
Minimum: 0
Maximum: 16383
Description: Number of values to measure. 0 means no trigger, 16383 means endless measurement.

10.6.1.3.9 Software_Trigger (TRIGGERSW)

Execute a software trigger.

10.6.1.3.10 Set_TriggerOutput (TRIGGEROUT)

Specifies which values should be output at trigger mode.

Parameter: int SP_TriggerOutput SP_TriggerOutput
Direction: Down
Valid values:
 0= Triggered values (TRIGGERED)
 1= All values (ALL)
Description: Output only triggered values or all values.

10.6.1.3.11 Get_TriggerOutput (TRIGGEROUT)

Specifies which values should be output at trigger mode.

Parameter: int SA_TriggerOutput SA_TriggerOutput
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= Triggered values (TRIGGERED)
 1= All values (ALL)
Description: Output only triggered values or all values.

10.6.1.4 Interfaces

10.6.1.4.1 Set_IPConfiguration (IPCONFIG)

Set the IP configuration at sensor.

Parameter: int SP_DHCPEnabled SP_DHCPEnabled

Direction: Down

Valid values:

0= FALSE

1= TRUE

Description: Specify if sensor should use a static IP address or ask for IP at DHCP server (dynamic IP address).

Parameter: String SP_Address SP_Address

Direction: Down

Valid values:

Valid IP address in form of xxx.xxx.xxx.xxx

Description: IP address of the sensor. This parameter is only evaluated on static IP assignment.

Parameter: String SP_SubnetMask SP_SubnetMask

Direction: Down

Valid values:

Valid network mask (e.g. 255.255.255.0 for a Class C network)

Description: Network mask of the sensor. This parameter is only evaluated on static IP assignment.

Parameter: String SP_Gateway SP_Gateway

Direction: Down

Valid values:

Valid IP address of default gateway in form of xxx.xxx.xxx.xxx

Description: The default gateway must be specified if the sensor should communicate with peers in foreign subnets. This parameter is only evaluated on static IP assignment.

10.6.1.4.2 Get_IPConfiguration (IPCONFIG)

Get the IP configuration at sensor.

Parameter: int SA_DHCPEnabled SA_DHCPEnabled

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= FALSE

1= TRUE

Description: Get settings if sensor should use a static IP address ask for IP at DHCP server (dynamic IP address).

Parameter: String SA_Address	SA_Address
Direction: Up	
Valid values:	Valid IP address in form of xxx.xxx.xxx.xxx
Description:	IP address of the sensor. If DHCP is enabled it returns the currently assigned IP address.
Parameter: String SA_SubnetMask	SA_SubnetMask
Direction: Up	
Valid values:	Valid network mask (e.g. 255.255.255.0 for a Class C network)
Description:	Network mask of the sensor. If DHCP is enabled it returns the currently assigned network mask.
Parameter: String SA_Gateway	SA_Gateway
Direction: Up	
Valid values:	Valid IP address of default gateway in form of xxx.xxx.xxx.xxx
Description:	Address of the default gateway. If DHCP is enabled it returns the currently assigned default gateway.

10.6.1.4.3 Set_IPDataTransferMode (MEATRANSFER)

Set IP protocol at sensor.

Parameter: int SP_Protocol	SP_Protocol
Direction: Down	
Valid values:	0= TCP server (SERVER/TCP) 1= TCP client (CLIENT/TCP) 2= UDP sender (CLIENT/UDP) 3= None
Description:	Specifies if data should be send using TCP or UDP.
Parameter: String SP_RemoteAddress	SP_RemoteAddress
Direction: Down	
Valid values:	Valid IP address of receiver of data
Description:	Address of remote computer to send data to. On TCP server this parameter is ignored.
Parameter: int SP_Port	SP_Port
Direction: Down	
Valid values:	Minimum: 1024 Maximum: 65535
Description:	Port to send data to or to listen for incoming requests.

10.6.1.4.4 Get_IPDataTransferMode (MEASTRANSFER)

Get IP protocol at sensor.

Parameter: int SA_Protocol

SA_Protocol

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= TCP server (SERVER/TCP)
- 1= TCP client (CLIENT/TCP)
- 2= UDP sender (CLIENT/UDP)
- 3= None

Description: Specifies if data should be send using TCP or UDP.

Parameter: String SA_RemoteAddress

SA_RemoteAddress

Direction: Up

Valid values:

Valid IP address of receiver of data

Description: Address of remote computer to send data to.

Parameter: int SA_Port

SA_Port

Direction: Up

Valid values:

- Minimum:** 1024
Maximum: 65535

Description: Port to send data to or to listen for incoming requests.

10.6.1.4.5 Set_EthernetMode (ETHERMODE)

Switches ethernet mode between Ethernet and Ethercat.

Parameter: int SP_EthernetMode

SP_EthernetMode

Direction: Down

Valid values:

- 0= Ethernet
- 1= Ethercat

Description: Ethernet mode.

10.6.1.4.6 Get_EthernetMode (ETHERMODE)

Get ethernet mode of sensor.

Parameter: int SA_EthernetMode

SA_EthernetMode

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Ethernet
- 1= Ethercat

Description: Ethernet mode.

10.6.1.4.7 Set_Baudrate (BAUDRATE)

Set baudrate of sensor for serial RS422 communication. After receiving the answer of the command, the baudrate of the local serial interface is changed, too.

Parameter: int SP_SensorBaudrate SP_SensorBaudrate
Direction: Down
Valid values:
 4000000
 3500000
 3000000
 2500000
 2000000
 1500000
 921600
 691200
 460800
 230400
 115200
 9600
Unit: Baud
Description: Baudrate of sensor.

10.6.1.4.8 Get_Baudrate (BAUDRATE)

Get baudrate of sensor for serial RS422 communication.

Parameter: int SA_SensorBaudrate SA_SensorBaudrate
Direction: Up
Valid values:
 4000000
 3500000
 3000000
 2500000
 2000000
 1500000
 921600
 691200
 460800
 230400
 115200
 9600
Unit: Baud
Description: Baudrate of sensor.

10.6.1.5 Parameter management

10.6.1.5.1 Save_Parameters (STORE)

Save actual parameters at sensor. There can be saved several settings on different locations. So it is easy to switch to another setting.

Parameter: int SP_ParameterSet SP_ParameterSet
Direction: Down
Valid values:
Minimum: 1
Maximum: 8
Description: Location to save the settings.

10.6.1.5.2 Load_Parameters (READ)

Load actual parameters into sensor RAM. There can be loaded several settings from different locations. So it is easy to switch to another setting.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_ParameterType SP_ParameterType
Direction: Down
Valid values:
 0= All settings (ALL)
 1= Device settings (DEVICE)
 2= Measurement settings (MEAS)
Description: Specifies which settings should be loaded.

Parameter: int SP_ParameterSet SP_ParameterSet
Direction: Down
Valid values:
Minimum: 1
Maximum: 8
Description: Location from where the settings should be loaded.

10.6.1.5.3 Get_ActiveParameterSet (READ)

Get the number of the currently active parameter set in RAM.

Parameter: int SA_ActiveParameterSet SA_ActiveParameterSet
Direction: Up
Valid values:
Minimum: 1
Maximum: 8
Description: Number of the currently active parameter set in RAM.

10.6.1.5.4 Set_Default (SETDEFAULT)

Reset the sensor to default settings.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_DefaultType SP_DefaultType
Direction: Down
Valid values:
 0= All settings (ALL)
 1= Keep device settings temporary (NODEVICE)
 2= Only material table (MATERIAL)
Description: Specifies which settings should be resetted.

10.6.2 Measurement

10.6.2.1 General

10.6.2.1.1 Set_MeasureMode (MEASMODE)

Set the measure mode.

If first bit of **IP_AutomaticMode** is set (1), **Get_AllParameters** (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_MeasureMode

SP_MeasureMode

Direction: Down

Valid values:

- 0= Distance diffuse (DIST_DIFFUSE)
- 1= Distance direct (DIST_DIRECT)
- 2= Thickness
- 3= Video

Description: Measure mode.

10.6.2.1.2 Get_MeasureMode (MEASMODE)

Get the measure mode.

Parameter: int SA_MeasureMode

SA_MeasureMode

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Distance diffuse (DIST_DIFFUSE)
- 1= Distance direct (DIST_DIRECT)
- 2= Thickness
- 3= Video

Description: Measure mode.

10.6.2.1.3 Set_MeasurePeak (MEASPEAK)

Select the peak to measure.

Parameter: int SP_MeasurePeak

SP_MeasurePeak

Direction: Down

Valid values:

- 0= Greatest Amplitude (DISTA)
- 1= Greatest Area (DISTW)
- 2= First Peak (DIST1)

Description: Peak to measure.

10.6.2.1.4 Get_MeasurePeak (MEASPEAK)

Get the selected peak to measure.

Parameter: int SA_MeasurePeak

SA_MeasurePeak

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Greatest Amplitude (DISTA)
- 1= Greatest Area (DISTW)
- 2= First Peak (DIST1)

Description: Peak to measure.

10.6.2.1.5 Get_Video (GETVIDEO)

Get recent video signals from sensor.

Parameter: Binary data SA_VideoRaw

SA_VideoRaw

Direction: Up

Valid values:

- 512 words (each 2 byte), each word is an intensity value.

Description: Raw video signal

Parameter: Binary data SA_VideoCorr

SA_VideoCorr

Direction: Up

Valid values:

- 512 words (each 2 byte), each word is an intensity value.

Description: Filtered video signal

Parameter: double SA_VideoTimestamp

SA_VideoTimestamp

Direction: Up

Valid values:

Minimum: 0

Maximum: 1.79769e+308 (DBL_MAX)

Unit: ms

Description: Timestamp of the video signal. It starts from 1970 Jan 01 at 01:00. It is generated when the video has arrived at TCP/IP socket.

Example how to read a video signal from sensor:

```

/* Do not forget to handle potential error after each call to MEDAQLib! */
/* Create sensor instance, open sensor via TCP/IP, set output to ethernet */
/* and than switch to video mode: */
err= SetIntExecSCmd (instance, "Set_MeasureMode", "SP_MeasureMode", 3 /*Video*/);

/* Select the desired video signal: */
err= SetParameterInt (instance, "SP_OutputVideoRaw_ETH", 1);
err= SetParameterInt (instance, "SP_OutputVideoCorr_ETH", 1);
err= ExecSCmd (instance, "Set_OutputVideo_ETH");

/* Aquire video signals: */
err= ExecSCmd (instance, "Get_Video");
WORD videoRaw[512], videoCorr[512];
DWORD maxlen= sizeof (videoRaw);
err= GetParameterBinary (instance, "SA_VideoRaw", (char *)videoRaw, &maxLen);
assert (maxlen==sizeof (videoRaw)); // additinal validity check
maxLen= sizeof (videoCorr);
err= GetParameterBinary (instance, "SA_VideoCorr", (char *)videoCorr, &maxLen);
assert (maxlen==sizeof (videoCorr)); // additinal validity check

/* Do anything with the received video signals */

```

10.6.2.1.6 Set_ShutterMode (SHUTTERMODE)

Set the shutter mode. This is an internal command. It should not be used by the customer.

Parameter: int SP_ShutterMode SP_ShutterMode

Direction: Down

Valid values:

- 0= Exposure time is adapted automatically (MEAS)
- 1= Exposure time is set manually (MANUAL)
- 2= Use two fixed exposure times alternately (2TIMEALT)
- 3= Use adapted time and factor * adapted time (see [Set_ShutterFactor](#)) alternately (AUTOTIMEALT)

Description: Shutter mode.

10.6.2.1.7 Get_ShutterMode (SHUTTERMODE)

Get the shutter mode. This is an internal command. It should not be used by the customer.

Parameter: int SA_ShutterMode SA_ShutterMode

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Exposure time is adapted automatically (MEAS)
- 1= Exposure time is set manually (MANUAL)
- 2= Use two fixed exposure times alternately (2TIMEALT)
- 3= Use adapted time and factor * adapted time (see [Set_ShutterFactor](#)) alternately (AUTOTIMEALT)

Description: Shutter mode.

10.6.2.1.8 Set_Samplerate (MEASRATE)

Set the samplerate.

Parameter: double SP_Measrate SP_Measrate

Direction: Down

Valid values:

- 1.5
- 2.5
- 5.0
- 10.0
- 20.0
- 30.0
- 49.0

Unit: kHz

Description: Samplerate of measurement.

10.6.2.1.9 Get_Samplerate (MEASRATE)

Get the samplerate.

Parameter: double SA_Measrate

SA_Measrate

Direction: Up

Valid values:

1.5

2.5

5.0

10.0

20.0

30.0

49.0

Unit: kHz

Description: Samplerate of measurement.

10.6.2.1.10 Set_ShutterTime (SHUTTER)

Set the fixed exposure times. This is an internal command. It should not be used by the customer.

Parameter: double SP_ShutterTime1

SP_ShutterTime1

Direction: Down

Valid values:

Minimum: 0.1

Maximum: 1638.0

Unit: μ s

Description: First exposure time.

Parameter: double SP_ShutterTime2

SP_ShutterTime2

Direction: Down

Valid values:

Minimum: 0.1

Maximum: 1638.0

Unit: μ s

Description: Second exposure time.

10.6.2.1.11 Get_ShutterTime (SHUTTER)

Get the fixed exposure times. This is an internal command. It should not be used by the customer.

Parameter: double SA_ShutterTime1

SA_ShutterTime1

Direction: Up

Valid values:

Minimum: 0.1

Maximum: 512.0

Unit: μ s

Description: First exposure time.

Parameter: double SA_ShutterTime2 SA_ShutterTime2
Direction: Up
Valid values:
 Minimum: 0.1
 Maximum: 512.0
Unit: μ s
Description: Second exposure time.

10.6.2.1.12 Set_ShutterFactor (SHUTTERFACTOR)

Set the factor for [Set_ShutterMode](#) when SP_ShutterMode is 3. This is an internal command. It should not be used by the customer.

Parameter: double SP_ShutterFactor SP_ShutterFactor
Direction: Down
Valid values:
 Minimum: 1.0
 Maximum: 100.0
Description: Factor for the second (alternating) exposure time.

10.6.2.1.13 Get_ShutterFactor (SHUTTERFACTOR)

Get the factor for [Set_ShutterMode](#) when SP_ShutterMode is 3. This is an internal command. It should not be used by the customer.

Parameter: double SA_ShutterFactor SA_ShutterFactor
Direction: Up
Valid values:
 Minimum: 1.0
 Maximum: 100.0
Description: Factor for the second (alternating) exposure time.

10.6.2.1.14 Set_LaserPower (LASERPOW)

Specify the laser power at sensor.

Parameter: int SP_LaserPower SP_LaserPower
Direction: Down
Valid values:
 0= Full
 1= Reduced
 2= Off
Description: Laser power.

10.6.2.1.15 Get_LaserPower (LASERPOW)

Get the laser power from sensor.

Parameter: int SA_LaserPower SA_LaserPower
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= Full
 1= Reduced
 2= Off
Description: Laser power.

10.6.2.2 Video signal

10.6.2.2.1 Set_ROI (ROI)

Set the region of interest for processing video signal.

Parameter: int SP_ROIStart

SP_ROIStart

Direction: Down

Valid values:

Minimum: 0

Maximum: 511

Unit: Pixel

Description: First pixel.

Parameter: int SP_ROIEnd

SP_ROIEnd

Direction: Down

Valid values:

Minimum: 0

Maximum: 511

Unit: Pixel

Description: Last pixel.

10.6.2.2.2 Get_ROI (ROI)

Get the region of interest for processing video signal.

Parameter: int SA_ROIStart

SA_ROIStart

Direction: Up

Valid values:

Minimum: 0

Maximum: 511

Unit: Pixel

Description: First pixel.

Parameter: int SA_ROIEnd

SA_ROIEnd

Direction: Up

Valid values:

Minimum: 0

Maximum: 511

Unit: Pixel

Description: Last pixel.

10.6.2.2.3 Set_VideoAverage (VSAVERAGE)

Set video averaging (before processing).

Parameter: int SP_VideoAverage

SP_VideoAverage

Direction: Down

Valid values:

0= None

1= Recursive over 2 lines (REC2)

2= Recursive over 4 lines (REC4)

3= Recursive over 8 lines (REC8)

4= Moving over 2 lines (MOV2)

5= Moving over 3 lines (MOV3)

6= Moving over 4 lines (MOV4)

7= Median over 3 lines (MED3)

Description: Averaging mode.

10.6.2.2.4 Get_VideoAverage (VSAVERAGE)

Get video averaging (before processing).

Parameter: int SA_VideoAverage

SA_VideoAverage

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = None
- 1 = Recursive over 2 lines (REC2)
- 2 = Recursive over 4 lines (REC4)
- 3 = Recursive over 8 lines (REC8)
- 4 = Moving over 2 lines (MOV2)
- 5 = Moving over 3 lines (MOV3)
- 6 = Moving over 4 lines (MOV4)
- 7 = Median over 3 lines (MED3)

Description: Averaging mode.

10.6.2.2.5 Set_Threshold (THRESHOLD)

Set threshold for video processing. This is an internal command. It should not be used by the customer.

Parameter: double SP_Threshold

SP_Threshold

Direction: Down

Valid values:

- Minimum:** 0.0
- Maximum:** 100.0

Unit: %

Description: Video threshold.

10.6.2.2.6 Get_Threshold (THRESHOLD)

Get threshold for video processing. This is an internal command. It should not be used by the customer.

Parameter: double SA_Threshold

SA_Threshold

Direction: Up

Valid values:

- Minimum:** 0.0
- Maximum:** 100.0

Unit: %

Description: Video threshold.

10.6.2.3 Material database

10.6.2.3.1 Get_MaterialTable (MATERIALTABLE)

Get a list of all materials for thickness calculation.

Parameter: String SA_MaterialTable

SA_MaterialTable

Direction: Up

Description: Whole table in one string, separated by new lines and commas.

Parameter: int SA_MaterialTableCount	SA_MaterialTableCount
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 20	
Description: Number of entries in the table. All following parameters exists from 1 to this number, e.g. SA_Pos1, SA_Pos2, ...	
Parameter: double SA_WaveLength	SA_WaveLength
Direction: Up	
Unit: nm	
Description: The wave length where the refractive index taken from.	
Parameter: int SA_Pos1..x	SA_Pos1..x
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 19	
Description: Index of the material in the table.	
Parameter: String SA_MaterialName1..x	SA_MaterialName1..x
Direction: Up	
Description: Name of the material in the table.	
Parameter: double SA_RefRACTIVEINDEX_nF1..x	SA_RefRACTIVEINDEX_nF1..x
Direction: Up	
Valid values:	
Minimum: 1.0	
Maximum: 4.0	
Description: Refractive index of the material.	
Parameter: String SA_Description1..x	SA_Description1..x
Direction: Up	
Description: Description of the material in the table.	

10.6.2.3.2 Set_ActiveMaterial (MATERIAL)

Set the active material for thickness calculation.

Parameter: String SP_ActiveMaterial	SP_ActiveMaterial
Direction: Down	
Description: Name of material.	

10.6.2.3.3 Get_ActiveMaterial (MATERIAL)

Get the active material for thickness calculation.

Parameter: String SA_ActiveMaterial	SA_ActiveMaterial
Direction: Up	
Description: Name of material.	

10.6.2.3.4 Get_MaterialInfo (MATERIALINFO)

Get information of active material.

Parameter: String SA_MaterialName	SA_MaterialName
Direction: Up	
Description: Name of the active material.	
Parameter: String SA_Description	SA_Description
Direction: Up	
Description: Description of the active material.	
Parameter: double SA_RefRACTIVEINDEX_nF	SA_RefRACTIVEINDEX_nF
Direction: Up	
Valid values:	
Minimum: 1.0	
Maximum: 4.0	
Description: Refractive index of the active material.	
Parameter: double SA_WaveLength	SA_WaveLength
Direction: Up	
Unit: nm	
Description: The wave length where the refractive index taken from.	

10.6.2.3.5 Edit_Material (MATERIALEDIT)

Edit or add new material by using refractive index.

Parameter: String SP_MaterialName	SP_MaterialName
Direction: Down	
Description: Name of the material.	
Parameter: String SP_Description	SP_Description
Direction: Down	
Description: Description of the material.	
Parameter: double SP_RefRACTIVEINDEX_nF	SP_RefRACTIVEINDEX_nF
Direction: Down	
Valid values:	
Minimum: 1.0	
Maximum: 4.0	
Description: Refractive index of the material.	

10.6.2.3.6 Delete_Material (MATERIALDELETE)

Deletes an existing material.

Parameter: String SP_MaterialName	SP_MaterialName
Direction: Down	
Description: Name of the material to delete.	

10.6.2.3.7 Clear_MaterialTable

Clear the whole material table.

10.6.2.4 Measurement value processing

10.6.2.4.1 Set_Averaging (AVERAGE)

Set data averaging at sensor.

Parameter: int SP_AveragingType

SP_AveragingType

Direction: Down

Valid values:

- 0= None
- 1= Moving average (MOVING)
- 2= Recursive averaging (RECURSIVE)
- 3= Median

Description: Averaging type.

Parameter: int SP_MovingCount

SP_MovingCount

Direction: Down

Valid values:

- 2
- 4
- 8
- 16
- 32
- 64
- 128

Description: Number of value for the averaging window. This parameter is only used at moving average.

Parameter: int SP_RecursiveCount

SP_RecursiveCount

Direction: Down

Valid values:

- Minimum: 2
- Maximum: 32768

Description: Number of values for recursive averaging. This parameter is only used at recursive average.

Parameter: int SP_MedianCount

SP_MedianCount

Direction: Down

Valid values:

- 3
- 5
- 7
- 9

Description: Number of values to build median. This parameter is only used at median.

10.6.2.4.2 Get_Averaging (AVERAGE)

Get data averaging at sensor.

Parameter: int SA_AveragingType SA_AveragingType

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = None
- 1 = Moving average (MOVING)
- 2 = Recursive averaging (RECURSIVE)
- 3 = Median

Description: Averaging type.

Parameter: int SA_MovingCount SA_MovingCount

Direction: Up

Valid values:

- 2
- 4
- 8
- 16
- 32
- 64
- 128

Description: Number of value for the averaging window. This parameter is only available at moving average.

Parameter: int SA_RecursiveCount SA_RecursiveCount

Direction: Up

Valid values:

- Minimum: 2
- Maximum: 32768

Description: Number of values for recursive averaging. This parameter is only available at recursive average.

Parameter: int SA_MedianCount SA_MedianCount

Direction: Up

Valid values:

- 3
- 5
- 7
- 9

Description: Number of values to build median. This parameter is only available at median.

10.6.2.4.3 Set_SpikeCorrection (SPIKECORR)

Set spike correction at sensor.

Parameter: int SP_SpikeCorrection SP_SpikeCorrection

Direction: Down

Valid values:

- 0 = off
- 1 = on

Description: Spike correction.

Parameter: int SP_NbrEvaluatedValues	SP_NbrEvaluatedValues
Direction: Down	
Valid values:	
Minimum: 1	
Maximum: 10	
Description: Number of values to evaluate for spike correction.	
Parameter: double SP_ToleranceRange	SP_ToleranceRange
Direction: Down	
Valid values:	
Minimum: 0.0	
Maximum: 100.0	
Description: Tolerance range for spike correction.	
Parameter: int SP_NbrCorrectedValues	SP_NbrCorrectedValues
Direction: Down	
Valid values:	
Minimum: 1	
Maximum: 100	
Description: Number of values to correct at spike correction.	

10.6.2.4.4 Get_SpikeCorrection (SPIKECORR)

Get spike correction at sensor.

Parameter: int SA_SpikeCorrection	SA_SpikeCorrection
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = off	
1 = on	
Description: Spike correction.	
Parameter: int SA_NbrEvaluatedValues	SA_NbrEvaluatedValues
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 10	
Description: Number of values to evaluate for spike correction.	
Parameter: double SA_ToleranceRange	SA_ToleranceRange
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 100.0	
Description: Tolerance range for spike correction.	
Parameter: int SA_NbrCorrectedValues	SA_NbrCorrectedValues
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 100	
Description: Number of values to correct at spike correction.	

10.6.2.4.5 Set_StatisticDepth (STATISTICDEPTH)

Set the window size for floating statistic calculation.

Parameter: int SP_StatisticDepth SP_StatisticDepth
Direction: Down
Valid values:
Minimum: 2
Maximum: 2147483647 (INT_MAX)
Description: Window size for floating statistic calculation. The value must be power of two (2, 4, 8, ..., 16384). Value greater as 16384 calculates statistic over all values.

10.6.2.4.6 Get_StatisticDepth (STATISTICDEPTH)

Get the window size for floating statistic calculation.

Parameter: int SA_StatisticDepth SA_StatisticDepth
Direction: Up
Valid values:
Minimum: 2
Maximum: 2147483647 (INT_MAX)
Description: Window size for floating statistic calculation. The value must be power of two (2, 4, 8, ..., 16384). Value greater as 16384 calculates statistic over all values.

10.6.2.4.7 Reset_Statistic (RESETSTATISTIC)

Reset the statistic (min and max values).

10.6.2.4.8 Set_MasterValue (MASTERMV)

Set the master value.

Parameter: int SP_Master SP_Master
Direction: Down
Valid values:
 0= no (NONE)
 1= yes (MASTER)=
Description: Specifies if mastering should be done or resetted.

Parameter: double SP_MasterValue SP_MasterValue
Direction: Down
Valid values:
Minimum: -2* measuring range
Maximum: +2* measuring range
Unit: mm
Description: Master value

10.6.2.4.9 Get_MasterValue (MASTERMV)

Get the master value.

Parameter: int SA_Master

SA_Master

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = no (NONE)
- 1 = yes (MASTER=

Description: Specifies if mastering is active.

Parameter: double SA_MasterValue

SA_MasterValue

Direction: Up

Valid values:

- Minimum: -2* measuring range
- Maximum: +2* measuring range

Unit: mm

Description: Master value

10.6.3 Data output

10.6.3.1 General

10.6.3.1.1 Set_DataOutInterface (OUTPUT)

Set the active interface for data output.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_DataOutInterface

SP_DataOutInterface

Direction: Down

Valid values:

- 0 = None
- 1 = RS422
- 2 = Ethernet
- 3 = HTTP

Description: Active interface for data output.

10.6.3.1.2 Get_DataOutInterface (OUTPUT)

Get the active interface for data output.

Parameter: int SA_DataOutInterface

SA_DataOutInterface

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = None
- 1 = RS422
- 2 = Ethernet
- 3 = HTTP

Description: Active interface for data output.

10.6.3.1.3 Set_Resampling (OUTREDUCE)

Set resampling to reduce output data.

Parameter: int SP_Resampling	SP_Resampling
Direction: Down	
Valid values:	
Minimum: 1	
Maximum: 3000000	
Description: Resampling value.	
Parameter: int SP_ResampleRS422	SP_ResampleRS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if RS422 output should be resampled.	
Parameter: int SP_ResampleEthernet	SP_ResampleEthernet
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if output over ethernet should be resampled.	

10.6.3.1.4 Get_Resampling (OUTREDUCE)

Get resampling for reducing output data.

Parameter: int SA_Resampling	SA_Resampling
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 3000000	
Description: Resampling value.	
Parameter: int SA_ResampleRS422	SA_ResampleRS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: RS422 output is resampled.	
Parameter: int SA_ResampleEthernet	SA_ResampleEthernet
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Output over ethernet is resampled.	

10.6.3.1.5 Set_HoldLastValid (OUTHOLD)

Set the number of values to be replaced by last valid value instead of error values.

Parameter: int SP_HoldLastValid SP_HoldLastValid
Direction: Down
Valid values:
Minimum: -1
Maximum: 1024
Description: Values to replace by last valid value. -1 means no value to hold,
 0 means never output an error value (always hold last valid value).

10.6.3.1.6 Get_HoldLastValid (OUTHOLD)

Get the number of values to be replaced by last valid value instead of error values.

Parameter: int SA_HoldLastValid SA_HoldLastValid
Direction: Up
Valid values:
Minimum: -1
Maximum: 1024
Description: Values to replace by last valid value. -1 means no value to hold,
 0 means never output an error value (always hold last valid value).

10.6.3.1.7 Set_MeasureValueCnt (GETVALUE)

Set the number of values to be output. If reached, the output stops.

Parameter: double SP_MeasureValueCnt SP_MeasureValueCnt
Direction: Down
Valid values:
Minimum: 0.0
Maximum: 4294967295.0 (UINT_MAX)
Description: Number of values to be output. 0 means no output, 4294967295
 means continuous output.

10.6.3.1.8 Get_MeasureValueCnt (GETVALUE)

Get the number of values to be output.

Parameter: double SA_MeasureValueCnt SA_MeasureValueCnt
Direction: Up
Valid values:
Minimum: 0.0
Maximum: 4294967295.0 (UINT_MAX)
Description: Number of values to be output. 0 means no output, 4294967295
 means continuous output.

10.6.3.2 Selected measurement values

10.6.3.2.1 Set_OutputDistance_RS422 (OUTDIST_RS422)

Set the distance data to be output at RS422 interface.

Parameter: int SP_OutputDistance1_RS422

SP_OutputDistance1_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if distance 1 is transmitted.

Parameter: int SP_OutputDistance2_RS422

SP_OutputDistance2_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if distance 2 is transmitted.

10.6.3.2.2 Get_OutputDistance_RS422 (OUTDIST_RS422)

Get the distance data to be output at RS422 interface.

Parameter: int SA_OutputDistance1_RS422

SA_OutputDistance1_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if distance 1 is transmitted.

Parameter: int SA_OutputDistance2_RS422

SA_OutputDistance2_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if distance 2 is transmitted.

10.6.3.2.3 Set_OutputThickness_RS422 (OUTTHICK_RS422)

Set the thickness data to be output at RS422 interface.

Parameter: int SP_OutputThickness12_RS422

SP_OutputThickness12_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if thickness between first and second peak is transmitted.

10.6.3.2.4 Get_OutputThickness_RS422 (OUTTHICK_RS422)

Get the thickness data to be output at RS422 interface.

Parameter: int SA_OutputThickness12_RS422

SA_OutputThickness12_-
RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if thickness between first and second peak is transmitted.

10.6.3.2.5 Set_OutputStatistic_RS422 (OUTSTATISTIC_RS422)

Set the statistic data to be output at RS422 interface.

Parameter: int SP_OutputStatisticMin_RS422

SP_OutputStatisticMin_-
RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if min value is transmitted.

Parameter: int SP_OutputStatisticMax_RS422

SP_OutputStatisticMax_-
RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if max value is transmitted.

Parameter: int SP_OutputStatisticPeak2Peak_RS422

SP_OutputStatistic-
Peak2Peak_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if peak to peak value is transmitted.

10.6.3.2.6 Get_OutputStatistic_RS422 (OUTSTATISTIC_RS422)

Get the statistic data to be output at RS422 interface.

Parameter: int SA_OutputStatisticMin_RS422

SA_OutputStatisticMin_-
RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if min value is transmitted.

Parameter: int SA_OutputStatisticMax_RS422

SA_OutputStatisticMax_-
RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if max value is transmitted.

Parameter: int SA_OutputStatisticPeak2Peak_ETH
Direction: Up
Valid values:
 0= no
 1= yes
Description: Specify if peak to peak value is transmitted.

SA_OutputStatistic-Peak2Peak_ETH

10.6.3.2.9 Set_OutputAdditional_RS422 (OUTADD_RS422)

Set the additional data to be output at RS422 interface.

Parameter: int SP_OutputAdditionalShutterTime_RS422
Direction: Down
Valid values:
 0= no
 1= yes
Description: Specify if shutter time is transmitted.

SP_OutputAdditionalShutterTime_RS422

Parameter: int SP_OutputAdditionalCounter_RS422
Direction: Down
Valid values:
 0= no
 1= yes
Description: Specify if counter is transmitted.

SP_OutputAdditionalCounter_RS422

Parameter: int SP_OutputAdditionalTimestamp_RS422
Direction: Down
Valid values:
 0= no
 1= yes
Description: Specify if timestamp is transmitted.

SP_OutputAdditionalTimestamp_RS422

Parameter: int SP_OutputAdditionalIntensity_RS422
Direction: Down
Valid values:
 0= no
 1= yes
Description: Specify if intensity is transmitted.

SP_OutputAdditionalIntensity_RS422

Parameter: int SP_OutputAdditionalState_RS422
Direction: Down
Valid values:
 0= no
 1= yes
Description: Specify if state is transmitted.

SP_OutputAdditionalState_RS422

Parameter: int SP_OutputAdditionalTemperature_RS422
Direction: Down
Valid values:
 0= no
 1= yes
Description: Specify if temperature is transmitted.

SP_OutputAdditionalTemperature_RS422

10.6.3.2.10 Get_OutputAdditional_RS422 (OUTADD_RS422)

Get the additional data to be output at RS422 interface.

Parameter: int SA_OutputAdditionalShutterTime_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if shutter time is transmitted.

SA_OutputAdditionalShutterTime_RS422

Parameter: int SA_OutputAdditionalCounter_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if counter is transmitted.

SA_OutputAdditionalCounter_RS422

Parameter: int SA_OutputAdditionalTimestamp_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if timestamp is transmitted.

SA_OutputAdditionalTimestamp_RS422

Parameter: int SA_OutputAdditionalIntensity_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if intensity is transmitted.

SA_OutputAdditionalIntensity_RS422

Parameter: int SA_OutputAdditionalState_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if state is transmitted.

SA_OutputAdditionalState_RS422

Parameter: int SA_OutputAdditionalTemperature_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if temperature is transmitted.

SA_OutputAdditionalTemperature_RS422

10.6.3.2.11 Set_OutputAdditional_ETH (OUTADD_ETH)

Set the additional data to be output at ethernet interface.

Parameter: int SP_OutputAdditionalShutterTime_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if shutter time is transmitted.

SP_OutputAdditionalShutterTime_ETH

Parameter: int SP_OutputAdditionalCounter_ETH	SP_OutputAdditionalCounter_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SP_OutputAdditionalTimestamp_ETH	SP_OutputAdditionalTimestamp_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SP_OutputAdditionalIntensity_ETH	SP_OutputAdditionalIntensity_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity is transmitted.	
Parameter: int SP_OutputAdditionalState_ETH	SP_OutputAdditionalState_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	
Parameter: int SP_OutputAdditionalTrgCounter_ETH	SP_OutputAdditionalTrgCounter_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if trigger counter is transmitted.	
Parameter: int SP_OutputAdditionalTemperature_ETH	SP_OutputAdditionalTemperature_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if temperature is transmitted.	

10.6.3.2.12 Get_OutputAdditional_ETH (OUTADD_ETH)

Get the additional data to be output at ethernet interface.

Parameter: int SA_OutputAdditionalShutterTime_ETH	SA_OutputAdditionalShutterTime_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	

Parameter: int SA_OutputAdditionalCounter_ETH	SA_OutputAdditionalCounter_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SA_OutputAdditionalTimestamp_ETH	SA_OutputAdditionalTimestamp_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SA_OutputAdditionalIntensity_ETH	SA_OutputAdditionalIntensity_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity is transmitted.	
Parameter: int SA_OutputAdditionalState_ETH	SA_OutputAdditionalState_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	
Parameter: int SA_OutputAdditionalTrgCounter_ETH	SA_OutputAdditionalTrgCounter_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if trigger counter is transmitted.	
Parameter: int SA_OutputAdditionalTemperature_ETH	SA_OutputAdditionalTemperature_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if temperature is transmitted.	

10.6.3.2.13 Set_OutputVideo_ETH (OUTVIDEO)

Set the video signal to be output at ethernet interface.

Parameter: int SP_OutputVideoRaw_ETH	SP_OutputVideoRaw_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if raw video signal is transmitted.	

Parameter: int SP_OutputVideoCorr_ETH SP_OutputVideoCorr_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if corrected video signal is transmitted.

10.6.3.2.14 Get_OutputVideo_ETH (OUTVIDEO)

Get the video signal to be output at ethernet interface.

Parameter: int SA_OutputVideoRaw_ETH SA_OutputVideoRaw_ETH

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if raw video signal is transmitted.

Parameter: int SA_OutputVideoCorr_ETH SA_OutputVideoCorr_ETH

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if corrected video signal is transmitted.

11 Commands for Confocal (IFD) sensors

IFD2400 and IFD2430 are no longer supported.

11.1 Commands for IFD2401/IFD2431

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [RS232](#) (native).
- [TCP/IP](#) (additional, e.g. RS232/RS422 to TCP/IP comm server and RS232 high level interface).
- [WinUSB](#) (native).
- [IF2004_USB](#) (native).
- [IF2008](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_Status](#), [Get_SpectralAv](#), [Get_FreeSR](#), [Get_DoubleFreq](#) and [Get_Frequencies](#) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib to calculate datarate, to interpret and scale data and to assign values.

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor, from 0 to 32767.
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command [Get_Status](#)).

The values of one data frame are filled in the arrays one after another. Each array always starts with a new data frame.

11.1.1 General commands

11.1.1.1 General

11.1.1.1.1 Get_Status (STS)

Retrieve detailed information about the controller and sensor.

Parameter: int SA_SRIndex

SA_SRIndex

Direction: Up

Valid values:

- 0= Free samplerate
- 1= 100 Hz (IFD2401) or 500 Hz (IFD2431)
- 2= 200 Hz (IFD2401) or 1000 Hz (IFD2431)
- 3= 400 Hz (IFD2401) or 2000 Hz (IFD2431)
- 4= 1000 Hz (IFD2401) or 5000 Hz (IFD2431)
- 5= 2000 Hz (IFD2401) or 10000 Hz (IFD2431)
- 6= 15625 Hz (only IFD2431)
- 7= 20000 Hz (only IFD2431)
- 8= 25000 Hz (only IFD2431)
- 9= 31250 Hz (only IFD2431)

Description: Samplerate index.

Parameter: int SA_MeasureMode	SA_MeasureMode
Direction: Up	
Valid values:	
0= Distance	
1= Thickness	
Description: Measure mode of the sensor.	
Parameter: int SA_Sensor	SA_Sensor
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 19	
Description: Number of active sensor.	
Parameter: int SA_ASCII	SA_ASCII
Direction: Up	
Valid values:	
0= Binary	
1= ASCII	
Description: Data Transfer mode (values) of sensor.	
Parameter: int SA_Averaging	SA_Averaging
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 9999	
Description: Data averaging factor.	
Parameter: int SA_X1..16	SA_X1..16
Direction: Up	
Valid values:	
0= off	
1= RS232/RS422	
9= USB	
Description: Selection of data of the sensor. This is necessary for data conversion and scaling.	
Depending on measure mode (distance / thickness) the selected value have different meaning. The following list shows the meaning for X1..16.	
X1 means (Distance / Thickness).	
X2 means (not used / Distance1).	
X3 means (Auto adaptive mode data / Distance2).	
X4 means (Intensity / Auto adaptive mode data).	
X5 means (not used / Intensity1).	
X6 means (not used / Intensity2).	
X7 means (Barycenter / Barycenter1).	
X8 means (not used / Barycenter2).	
X9 means (State Flags / State Flags).	
X10 means (Counter / Counter).	
X11 means (Encoder 1 LSB / Encoder 1 LSB).	
X12 means (Encoder 1 MSB / Encoder 1 MSB).	
X13 means (Encoder 2 LSB / Encoder 2 LSB).	
X14 means (Encoder 2 MSB / Encoder 2 MSB).	
X15 means (Encoder 3 LSB / Encoder 3 LSB).	
X16 means (Encoder 3 MSB / Encoder 3 MSB).	

Parameter: int SA_0_SOD	SA_0_SOD
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 7	
Description: Analog out BNC1: 0= X1, 1= X2, ..., 15= X16.	
Parameter: int SA_0_OV	SA_0_OV
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 99999	
Description: Analog out BNC1: Specifies the value which shoud output 0 V.	
Parameter: int SA_0_10V	SA_0_10V
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 99999	
Description: Analog out BNC1: Specifies the value which shoud output 10 V.	
Parameter: int SA_1_SOD	SA_1_SOD
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 7	
Description: Analog out BNC2: 0= X1, 1= X2, ..., 15= X16.	
Parameter: int SA_1_OV	SA_1_OV
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 99999	
Description: Analog out BNC2: Specifies the value which shoud output 0 V.	
Parameter: int SA_1_10V	SA_1_10V
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 99999	
Description: Analog out BNC2: Specifies the value which shoud output 10 V.	
Parameter: double SA_Range	SA_Range
Direction: Up	
Unit: μm	
Valid values:	
Minimum: 0.0	
Maximum: 2147483647.0 (INT_MAX)	
Description: Range of active sensor.	

11.1.1.1.2 Get_Version (VER)

Get the software version of the controller.

Parameter: String SA_Version	SA_Version
Direction: Up	
Description: Controller name, serial number and version	

11.1.1.1.3 Reset (RST)

Reset the controller and set the default parameter values.

Only valid for very early firmware versions (any before V1.2.54), removed at later versions.

11.1.1.2 Sensor

11.1.1.2.1 Set_ActiveSensor (SEN)

Set the active sensor (up to 20 sensors can be stored in controller) for measurement. If first bit of [IP_AutomaticMode](#) is set (1), [Get_Range](#) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_Sensor

SP_Sensor

Direction: Down

Valid values:

Minimum: 0

Maximum: 19

Description: Number of active sensor.

11.1.1.2.2 Get_ActiveSensor (SEN?)

Get the active sensor.

Parameter: int SA_Sensor

SA_Sensor

Direction: Up

Valid values:

Minimum: 0

Maximum: 19

Description: Number of active sensor.

11.1.1.2.3 Get_Range (SCA)

Get the range of active sensor.

Parameter: double SA_Range

SA_Range

Direction: Up

Unit: μm

Valid values:

Minimum: 0.0

Maximum: 2147483647.0 (INT_MAX)

Description: Range of active sensor.

11.1.1.2.4 Get_AllRanges (LUL)

Get the ranges of all calibrated sensors.

Parameter: String SA_Ranges

SA_Ranges

Direction: Up

Unit: μm (after conversion from string to int or double)

Valid values:

0.0 .. 2147483647.0 (INT_MAX)

Description: Ranges of all 20 sensors in a string, separated by commas.

11.1.1.2.5 Acquire_DarkSig (DRK)

Aquire dark signal.

Parameter: int SA_MinSRIIndex

SA_MinSRIIndex

Direction: Up

Valid values:

- 1 = 100 Hz (IFD2401) or 500 Hz (IFD2431)
- 2 = 200 Hz (IFD2401) or 1000 Hz (IFD2431)
- 3 = 400 Hz (IFD2401) or 2000 Hz (IFD2431)
- 4 = 1000 Hz (IFD2401) or 5000 Hz (IFD2431)
- 5 = 2000 Hz (IFD2401) or 10000 Hz (IFD2431)
- 6 = 15625 Hz (only IFD2431)
- 7 = 20000 Hz (only IFD2431)
- 8 = 25000 Hz (only IFD2431)
- 9 = 31250 Hz (only IFD2431)

Description: Minimal samplerate index.

11.1.1.2.6 FastDark (FDK)

Aquire dark signal for active sensor and samplerate.

Parameter: int SP_AveragingForDark

SP_AveragingForDark

Direction: Down

Valid values:

- Minimum:** 1
- Maximum:** 99

Description: Averaging factor for dark.

Parameter: int SP_Weighting

SP_Weighting

Direction: Down

Valid values:

- Minimum:** 1
- Maximum:** 100

Description: Weighting factor.

11.1.1.2.7 Set_AutoDark (ADK)

Enables/Disables the "Automatic Fast Dark" mode.

Parameter: int SP_AutoDark

SP_AutoDark

Direction: Down

Valid values:

- 0= off
- 1= on

Description: Automatic Fast Dark.

11.1.1.2.8 Get_AutoDark (ADK?)

Get the state of the "Automatic Fast Dark" mode.

Parameter: int SA_AutoDark

SA_AutoDark

Direction: Up

Valid values:

0 = off

1 = on

Description: Automatic Fast Dark.

11.1.1.3 Triggering

11.1.1.3.1 Continue (CTN)

Continues data acquisition (after trigger commands SingleShot_Trg, TriggerMode_Edge, TriggerMode_State or an error).

11.1.1.3.2 Start_Trigger (TRG)

Starts the trigger mode at controller.

11.1.1.3.3 End_Trigger

Stops the trigger mode at controller (after Start_Trigger).

11.1.1.3.4 SingleShot_Trg (TRE)

Starts the single shot trigger mode at controller.

Parameter: int SP_NumberOfPoints

SP_NumberOfPoints

Direction: Down

Valid values:

Minimum: 0

Maximum: 9999

Description: Number of data frames (each frame can consist of different values) to read.

11.1.1.3.5 Set_TrgMode_Edge (TRS)

Enable/Disable the "Start/Stop on Edge Trigger" mode.

Parameter: int SP_TriggerMode_Edge

SP_TriggerMode_Edge

Direction: Down

Valid values:

Minimum: 0

Maximum: 1

Description: Start/Stop on Edge Trigger state.

11.1.1.3.6 Set_TrgMode_State (TRN)

Enable/Disable the "Start/Stop on State Trigger" mode.

Parameter: int SP_TriggerMode_State SP_TriggerMode_State
Direction: Down
Valid values:
Minimum: 0
Maximum: 1
Description: Start/Stop on State Trigger state.

11.1.1.3.7 Set_ActiveEdge (TRF)

Set the active edge or state for Start_Trigger, TriggerMode_Edge or TriggerMode_State.

Parameter: int SP_ActiveEdge SP_ActiveEdge
Direction: Down
Valid values:
 0= rising edge or high state
 1= falling edge or low state
Description: Active state or edge.

11.1.1.3.8 Get_ActiveEdge (TRF?)

Get the active edge or state for Start_Trigger, TriggerMode_Edge or TriggerMode_State.

Parameter: int SA_ActiveEdge SA_ActiveEdge
Direction: Up
Valid values:
 0= rising edge or high state
 1= falling edge or low state
Description: Active state or edge.

11.1.1.3.9 Software_Trigger (STR)

Simulates an hardware trigger (SYNC IN) for trigger commands SingleShot_Trg, TriggerMode_Edge or TriggerMode_State.

Only available from firmware version V1.2.56

11.1.1.4 Encoder

11.1.1.4.1 RecenterEncoder (RCD)

Recenter encoder position.

Parameter: int SP_Encoder1 SP_Encoder1
Direction: Down
Valid values:
Minimum: 0
Maximum: 1
Description: Recenter Encoder 1.

Parameter: int SP_Encoder2 SP_Encoder2

Direction: Down

Valid values:

Minimum: 0

Maximum: 1

Description: Recenter Encoder 2.

Parameter: int SP_Encoder3 SP_Encoder3

Direction: Down

Valid values:

Minimum: 0

Maximum: 1

Description: Recenter Encoder 3.

11.1.1.5 Interfaces

11.1.1.5.1 Set_Baudrate (BAU)

Set the baudrate of the serial interface of controller. After receiving the answer of the command, the baudrate of the local serial interface is changed, too.

Attention! To avoid synchronisation problems, sending data by sensor should be disabled after changing baudrate for a short time (sensor command Start_Trigger).

Parameter: int SP_SensorBaudrate SP_SensorBaudrate

Direction: Down

Valid values:

9600

19200

38400

57600

115200

230400

460800

Description: Baudrate of controller.

11.1.1.5.2 Get_Baudrate (BAU?)

Get the baudrate of the serial interface of controller.

Parameter: int SA_SensorBaudrate SA_SensorBaudrate

Direction: Up

Valid values:

9600

19200

38400

57600

115200

230400

460800

Description: Baudrate of controller.

11.1.1.6 Parameter management

11.1.1.6.1 Save_Setup (SSU)

Save the current setup of controller to flash.

11.1.1.7 Internal controller commands

11.1.1.7.1 Set_LampTest (SLP)

Enable/Disable the "Lamp Test" mode.
Only available at IFD2401.

Parameter: int SP_LampTest

SP_LampTest

Direction: Down

Valid values:

Minimum: 0

Maximum: 1

Description: Lamp Test state.

11.1.1.7.2 Get_LampTest (SLP?)

Get the current "Lamp Test" mode.
Only available at IFD2401.

Parameter: int SA_LampTest

SA_LampTest

Direction: Up

Valid values:

Minimum: 0

Maximum: 1

Description: Lamp Test state.

11.1.1.7.3 Set_LampTestThr (CSL)

Set the threshold for "Lamp Test" mode.
Only available at IFD2401.

Parameter: int SP_LampTestThr

SP_LampTestThr

Direction: Down

Valid values:

Minimum: 0

Maximum: 998

Description: Lamp Test threshold.

11.1.1.7.4 Get_LampTestThr (CSL?)

Get the threshold for "Lamp Test" mode.
Only available at IFD2401.

Parameter: int SA_LampTestThr SA_LampTestThr
Direction: Up
Valid values:
 Minimum: 0
 Maximum: 998
Description: Lamp Test threshold.

11.1.1.7.5 Set_Watchdog (WDE)

Enable/Disable the Watchdog.

Parameter: int SP_Watchdog SP_Watchdog
Direction: Down
Valid values:
 Minimum: 0
 Maximum: 1
Description: Watchdog state.

11.1.1.7.6 Get_Watchdog (WDE?)

Get the Watchdog state.

Parameter: int SA_Watchdog SA_Watchdog
Direction: Up
Valid values:
 Minimum: 0
 Maximum: 1
Description: Watchdog state.

11.1.1.7.7 Set_WatchdogPrd (WDP)

Set the Watchdog period.

Parameter: int SP_WatchdogPeriod SP_WatchdogPeriod
Direction: Down
Unit: s
Valid values:
 Minimum: 10
 Maximum: 255
Description: Watchdog period.

11.1.1.7.8 Get_WatchdogPrd (WDP?)

Get the Watchdog period.

Parameter: int SA_WatchdogPeriod SA_WatchdogPeriod
Direction: Up
Unit: s
Valid values:
 Minimum: 10
 Maximum: 255
Description: Watchdog period.

11.1.2 Measurement

11.1.2.1 General

11.1.2.1.1 Set_MeasureMode (MOD)

Set the measure mode.

Parameter: int SP_MeasureMode SP_MeasureMode
Direction: Down
Valid values:
 0= Distance
 1= Thickness
Description: Measure mode of the sensor.

11.1.2.1.2 Get_MeasureMode (MOD?)

Get the current measure mode.

Parameter: int SA_MeasureMode SA_MeasureMode
Direction: Up
Valid values:
 0= Distance
 1= Thickness
Description: Measure mode of the sensor.

11.1.2.1.3 Set_FirstPeakMode (MSP)

Enable/Disable the "First peak" mode.

Parameter: int SP_FirstPeak SP_FirstPeak
Direction: Down
Valid values:
 Minimum: 0
 Maximum: 1
Description: First peak state.

11.1.2.1.4 Get_FirstPeakMode (MSP?)

Get the current "First peak" mode.

Parameter: int SA_FirstPeak SA_FirstPeak

Direction: Up

Valid values:

Minimum: 0

Maximum: 1

Description: First peak state.

11.1.2.1.5 Set_SRIndex (SRA)

Set the samplerate for data acquisition.

Parameter: int SP_SRIndex SP_SRIndex

Direction: Down

Valid values:

0= Free samplerate

1= 100 Hz (IFD2401) or 500 Hz (IFD2431)

2= 200 Hz (IFD2401) or 1000 Hz (IFD2431)

3= 400 Hz (IFD2401) or 2000 Hz (IFD2431)

4= 1000 Hz (IFD2401) or 5000 Hz (IFD2431)

5= 2000 Hz (IFD2401) or 10000 Hz (IFD2431)

6= 15625 Hz (only IFD2431)

7= 20000 Hz (only IFD2431)

8= 25000 Hz (only IFD2431)

9= 31250 Hz (only IFD2431)

Description: Samplerate index.

11.1.2.1.6 Get_SRIndex (SRA?)

Get the current samplerate.

Parameter: int SA_SRIndex SA_SRIndex

Direction: Up

Valid values:

0= Free samplerate

1= 100 Hz (IFD2401) or 500 Hz (IFD2431)

2= 200 Hz (IFD2401) or 1000 Hz (IFD2431)

3= 400 Hz (IFD2401) or 2000 Hz (IFD2431)

4= 1000 Hz (IFD2401) or 5000 Hz (IFD2431)

5= 2000 Hz (IFD2401) or 10000 Hz (IFD2431)

6= 15625 Hz (only IFD2431)

7= 20000 Hz (only IFD2431)

8= 25000 Hz (only IFD2431)

9= 31250 Hz (only IFD2431)

Description: Samplerate index.

11.1. Commands for IFD2401/IFD2431

11.1.2.1.7 Set_FreeSR (FRQ)

Set free samplerate value for data acquisition.

Parameter: int SP_FreeSR SP_FreeSR

Direction: Down

Valid values:

Minimum: 100

Maximum: 2000 (IFD2401) or 31250 (IFD2431)

Unit: Hz

Description: Free Frequency.

Parameter: int SA_FreeSR SA_FreeSR

Direction: Up

Valid values:

Minimum: 100

Maximum: 2000 (IFD2401) or 31250 (IFD2431)

Unit: Hz

Description: Free Samplerate.

11.1.2.1.8 Get_FreeSR (FRQ?)

Get free samplerate value.

Parameter: int SA_FreeSR SA_FreeSR

Direction: Up

Valid values:

Minimum: 100

Maximum: 2000 (IFD2401) or 31250 (IFD2431)

Unit: Hz

Description: Free Samplerate.

11.1.2.1.9 Set_Exposure (TEX)

Set free exposure time for data acquisition.

Parameter: int SP_Exposure SP_Exposure

Direction: Down

Valid values:

Minimum: 500 (IFD2401) or 32 (IFD2431)

Maximum: 10000 (IFD2401) or 2000 (IFD2431)

Unit: μ s

Description: Free Exposure time.

Parameter: int SA_Exposure SA_Exposure

Direction: Up

Valid values:

Minimum: 500 (IFD2401) or 32 (IFD2431)

Maximum: 10000 (IFD2401) or 2000 (IFD2431)

Unit: μ s

Description: Free Exposure time.

11.1.2.1.10 Get_Exposure (TEX?)

Get free exposure time.

Parameter: int SA_Exposure SA_Exposure
Direction: Up
Valid values:
 Minimum: 500 (IFD2401) or 32 (IFD2431)
 Maximum: 10000 (IFD2401) or 2000 (IFD2431)
Unit: μ s
Description: Free Exposure time.

11.1.2.1.11 Get_MinSR (FRM)

Get the minimum authorized samplerate.

Parameter: int SA_MinSR SA_MinSR
Direction: Up
Valid values:
 Minimum: 100
 Maximum: 2000 (IFD2401) or 31250 (IFD2431)
Unit: Hz
Description: Minimum samplerate (determined by dark signal).

11.1.2.1.12 Set_DoubleFreq (DFA)

Enables or disables the double frequency mode.

Only available at IFD2401 from firmware version V1.2.56.

Parameter: int SP_DoubleFrequency SP_DoubleFrequency
Direction: Down
Valid values:
 0= Disable
 1= Enable
Description: Double frequency mode.

11.1.2.1.13 Get_DoubleFreq (DFA?)

Get the double frequency mode.

Only available at IFD2401 from firmware version V1.2.56.

Parameter: int SA_DoubleFrequency SA_DoubleFrequency
Direction: Up
Valid values:
 0= Disable
 1= Enable
Description: Double frequency mode.

11.1.2.1.14 Set_Frequencies (DFF)

Set the frequencies for double frequency mode.
 SP_HighFrequency must be higher than SP_LowFrequency.
 Only available at IFD2401 from firmware version V1.2.56.

Parameter: int SP_LowFrequency SP_LowFrequency

Direction: Down

Valid values:

Minimum: 100

Maximum: 1850

Unit: Hz

Description: Low frequency.

Parameter: int SP_HighFrequency SP_HighFrequency

Direction: Down

Valid values:

Minimum: 100

Maximum: 1850

Unit: Hz

Description: High frequency.

11.1.2.1.15 Get_Frequencies (DFF?)

Get the frequencies of double frequency mode.
 Only available at IFD2401 from firmware version V1.2.56.

Parameter: int SA_LowFrequency SA_LowFrequency

Direction: Up

Valid values:

Minimum: 100

Maximum: 1850

Unit: Hz

Description: Low frequency.

Parameter: int SA_HighFrequency SA_HighFrequency

Direction: Up

Valid values:

Minimum: 100

Maximum: 1850

Unit: Hz

Description: High frequency.

11.1.2.1.16 Get_CCD (CCD)

Returns current CCD data.

Parameter: Binary data SA_CCD SA_CCD

Direction: Up

Valid values:

4096 (IFD2401) or 2048 (IFD2431) bytes, convertible to 2048 (IFD2401)
 or 1024 (IFD2431) words.

Description: Raw CCD line.

11.1. Commands for IFD2401/IFD2431

Parameter: Binary data SA_PreTreated SA_PreTreated
Direction: Up
Valid values:
 4096 (IFD2401) or 2048 (IFD2431) bytes, convertible to 2048 (IFD2401)
 or 1024 (IFD2431) words.
Description: Raw CCD line.

11.1.2.1.17 Get_DarkSig (SGD)

Get the dark signal table of controller.

Parameter: Binary data SA_DarkSig SA_DarkSig
Direction: Up
Valid values:
 4096 (IFD2401) or 2048 (IFD2431) bytes, convertible to 2048 (IFD2401)
 or 1024 (IFD2431) words.
Description: Dark signal table.

11.1.2.1.18 Start_Spectrum

Start the spectrum mode in controller. Only work via USB connection.
 On firmware versions below 1.2.36 the spectrum is shifted 8 bytes to left.
 This mode does work up to 4.7 kHz only. For higher samplerates data will be lost
 and the spectrums are corrupted. To avoid this, increase spectral averaging.

Parameter: int IP_UsbReadBufSize IP_UsbReadBufSize
Direction: Down
Unit: Bytes
Valid values:
Minimum: 2
Maximum: 1048576
Default: 131072
Description: Buffer size for read operations on USB, while spectrum mode is active. The value is always ceiled to the next power of two (512, 1024, 2048, ..., 32768, 65536).

Parameter: int IP_TimerResolution IP_TimerResolution
Direction: Down
Unit: ms
Valid values:
 -1 = Do not set timer resolution.
 0 = Use greatest possible accuracy.
 1..2147483647 (INT_MAX) = Resolution in milliseconds.
Unit: ms
Default: 0
Description: Timer resolution (for Windows scheduler, set by timeBeginPeriod).
 It is automatically reset at End_Spectrum.

11.1.2.1.19 Get_Spectrum

Read one Spectrum (PreTreated Signal). It is returned with a frequency of Samplerate devided by Spectral Averaging.

Parameter: int SP_WaitSpectrumTimeout	SP_WaitSpectrumTimeout
Direction: Down	
Unit: ms	
Valid values:	
Minimum: 0	
Maximum: 2147483647 (INT_MAX)	
Description: Timeout to wait for a spectrum.	
Parameter: int SP_ReadMode	SP_ReadMode
Direction: Down	
Valid values:	
0= Each spectrum	
1= Only newest spectrum	
2= Automatic	
Description: This mode specifies if each spectrum should be read or only the latest one. If set to automatic each spectrum is read until the buffer does not overflow. If the buffer becomes full one or more spectra are discarded.	
Parameter: Binary data SA_Spectrum	SA_Spectrum
Direction: Up	
Valid values:	
4096 (IFD2401) or 2048/1024 (IFD2431, depends on binning) bytes, convertible to 2048 (IFD2401) or 1024/512 (IFD2431, depends on binning) words.	
Description: PreTreated Signal.	
Parameter: double SA_Timestamp	SA_Timestamp
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 1.79769e+308 (DBL_MAX)	
Unit: ms	
Description: Timestamp of the signal. It starts from 1970 Jan 01 at 01:00. It is generated when the spectrum is read from USB	
Parameter: int SA_SkippedSpectra	SA_SkippedSpectra
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 2147483647 (INT_MAX)	
Description: Number of skipped spectra, if SP_NewestSpectrum is set to 1.	

11.1.2.1.20 End_Spectrum

End the spectrum mode in controller.

Parameter: int IP_UsbReadBufSize IP_UsbReadBufSize
Direction: Down
Unit: Bytes
Valid values:
 Minimum: 2
 Maximum: 1048576
Default: 512
Description: Buffer size for read operations on USB, after spectrum mode was finished. The value is always ceiled to the next power of two (512, 1024, 2048, ..., 32768, 65536).

11.1.2.2 Video signal

11.1.2.2.1 Set_Threshold (MNP)

Set the detection threshold for distance mode.

Parameter: double SP_Threshold SP_Threshold
Direction: Down
Valid values:
 Minimum: 0.000001
 Maximum: 1.0
Description: Threshold of the sensor.

11.1.2.2.2 Get_Threshold (MNP?)

Get the current detection threshold for distance mode.

Parameter: double SA_Threshold SA_Threshold
Direction: Up
Valid values:
 Minimum: 0.000001
 Maximum: 1.0
Description: Threshold of the sensor.

11.1.2.2.3 Set_Threshold1 (SPP)

Set the detection threshold for the strongest peak in thickness mode.

Parameter: double SP_Threshold1 SP_Threshold1
Direction: Down
Valid values:
 Minimum: 0.000001
 Maximum: 1.0
Description: Threshold of the sensor.

11.1.2.2.4 Get_Threshold1 (SPP?)

Get the current detection threshold for the strongest peak in thickness mode.

Parameter: double SA_Threshold1 SA_Threshold1
Direction: Up
Valid values:
Minimum: 0.000001
Maximum: 1.0
Description: Threshold of the sensor.

11.1.2.2.5 Set_Threshold2 (SDP)

Set the detection threshold for the second peak in thickness mode.

Parameter: double SP_Threshold2 SP_Threshold2
Direction: Down
Valid values:
Minimum: 0.000001
Maximum: 1.0
Description: Threshold of the sensor.

11.1.2.2.6 Get_Threshold2 (SDP?)

Get the current detection threshold for the second peak in thickness mode.

Parameter: double SA_Threshold2 SA_Threshold2
Direction: Up
Valid values:
Minimum: 0.000001
Maximum: 1.0
Description: Threshold of the sensor.

11.1.2.2.7 Set_SpectralAv (AVS)

Set spectral averaging at controller.

Note! Acquisition speed will be reduced.

Parameter: int SP_SpectralAv SP_SpectralAv
Direction: Down
Valid values:
Minimum: 1
Maximum: 9999
Description: Data averaging factor.

11.1.2.2.8 Get_SpectralAv (AVS?)

Returns current spectral averaging at controller.

Parameter: int SA_SpectralAv SA_SpectralAv
Direction: Up
Valid values:
Minimum: 1
Maximum: 9999
Description: Data averaging factor.

11.1.2.2.9 Set_Binning (FBH)

Set the binning value.

This is an internal command. It should not be used by the customer.
 Only available at IFD2431.

Parameter: int SP_Binning SP_Binning
Direction: Down
Valid values:
Minimum: 1
Maximum: 2
Description: Binning value

11.1.2.2.10 Get_Binning (FBH?)

Get the binning value.

This is an internal command. It should not be used by the customer.
 Only available at IFD2431.

Parameter: int SA_Binning SA_Binning
Direction: Up
Valid values:
Minimum: 1
Maximum: 2
Description: Binning value

11.1.2.3 Material database

11.1.2.3.1 Set_RefractIndex (SRI)

Set the refraction index for thickness measurement.

Parameter: double SP_RefraetIndex SP_RefraetIndex
Direction: Down
Valid values:
Minimum: 0.0001
Maximum: 9.9999
Description: Refractive index used by the sensor for thickness measurement.

11.1.2.3.2 Get_RefactIndex (SRI?)

Get the current refraction index.

Parameter: double SA_RefactIndex	SA_RefactIndex
Direction: Up	
Valid values:	
Minimum: 0.0001	
Maximum: 9.9999	
Description: Refractive index used by the sensor for thickness measurement.	
Parameter: int SP_RefactIndexFileIdx	SP_RefactIndexFileIdx
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 8	
Description: Refractive index file index.	
Parameter: int SA_RefactIndexFileIdx	SA_RefactIndexFileIdx
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 8	
Description: Refractive index file index.	
Parameter: String SA_RefactIndexFileName	SA_RefactIndexFileName
Direction: Up	
Description: Refractive index file name.	
Parameter: double SA_MinRefactIndex	SA_MinRefactIndex
Direction: Up	
Valid values:	
Minimum: 0.0000	
Maximum: 9.9999	
Description: Minimum refractive index.	
Parameter: double SA_MaxRefactIndex	SA_MaxRefactIndex
Direction: Up	
Valid values:	
Minimum: 0.0001	
Maximum: 9.9999	
Description: Maximum refractive index.	
Parameter: double SA_AvgRefactIndex	SA_AvgRefactIndex
Direction: Up	
Valid values:	
Minimum: 0.0001	
Maximum: 9.9999	
Description: Averaging refractive index.	

11.1.2.3.4 Get_RefIdxFile (INF?)

Request the selected refractive index file.

Parameter: int SA_RefraetIndexFileIdx	SA_RefraetIndexFileIdx
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 8	
Description: Refractive index file index.	
Parameter: String SA_RefraetIndexFileName	SA_RefraetIndexFileName
Direction: Up	
Description: Refractive index file name.	
Parameter: double SA_MinRefractIndex	SA_MinRefractIndex
Direction: Up	
Valid values:	
Minimum: 0.0000	
Maximum: 9.9999	
Description: Minimum refractive index.	
Parameter: double SA_MaxRefractIndex	SA_MaxRefractIndex
Direction: Up	
Valid values:	
Minimum: 0.0001	
Maximum: 9.9999	
Description: Maximum refractive index.	
Parameter: double SA_AvgRefractIndex	SA_AvgRefractIndex
Direction: Up	
Valid values:	
Minimum: 0.0001	
Maximum: 9.9999	
Description: Averaging refractive index.	

11.1.2.4 Light source

11.1.2.4.1 Set_LEDIntensity (LED)

Set the LED intensity level.

Only available at IFD2401.

Parameter: int SP_LEDIntensity	SP_LEDIntensity
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 100	
Description: LED intensity.	

11.1.2.4.2 Get_LEDIntensity (LED?)

Get the LED intensity level.
Only available at IFD2401.

Parameter: int SA_LEDIntensity SA_LEDIntensity
Direction: Up
Valid values:
Minimum: 0
Maximum: 100
Description: LED intensity.

11.1.2.4.3 Set_AutoAdaptLED (AAL)

Enable/Disable the "Auto-Adaptive LED" mode.
Only available at IFD2401.

Parameter: int SP_AutoAdaptLED SP_AutoAdaptLED
Direction: Down
Valid values:
Minimum: 0
Maximum: 1
Description: Auto-Adaptive LED state.

11.1.2.4.4 Get_AutoAdaptLED (AAL?)

Get the "Auto-Adaptive LED" mode.
Only available at IFD2401.

Parameter: int SA_AutoAdaptLED SA_AutoAdaptLED
Direction: Up
Valid values:
Minimum: 0
Maximum: 1
Description: Auto-Adaptive LED state.

11.1.2.4.5 Set_AdaptLEDThr (VTH)

Set the threshold value for the auto-adaptive LED mode.
Only available at IFD2401.

Parameter: int SP_AutoAdaptLEDThr SP_AutoAdaptLEDThr
Direction: Down
Valid values:
Minimum: 0
Maximum: 4095
Description: Auto-Adaptive LED threshold.

11.1.2.4.6 Get_AdaptLEDThr (VTH?)

Get the threshold value for the auto-adaptive LED mode.
Only available at IFD2401.

Parameter: int SA_AutoAdaptLEDThr SA_AutoAdaptLEDThr
Direction: Up
Valid values:
 Minimum: 0
 Maximum: 4095
Description: Auto-Adaptive LED threshold.

11.1.2.4.7 Set_IntLightSrc (CCL)

Set the state of the internal light source (LED).
Only available at IFD2401.

Parameter: int SP_LED_Off SP_LED_Off
Direction: Down
Valid values:
 Minimum: 0
 Maximum: 1
Description: LED on (0) or off (1).

11.1.2.4.8 Get_IntLightSrc (CCL?)

Get the state of the internal light source (LED).
Only available at IFD2401.

Parameter: int SA_LED_Off SA_LED_Off
Direction: Up
Valid values:
 Minimum: 0
 Maximum: 1
Description: LED on (0) or off (1).

11.1.2.5 Measurement value processing

11.1.2.5.1 Set_Averaging (AVR)

Set data averaging at controller. Note! Acquisition speed will be reduced.

Parameter: int SP_Averaging SP_Averaging
Direction: Down
Valid values:
 Minimum: 1
 Maximum: 9999
Description: Data averaging factor.

11.1.2.5.2 Get_Averaging (AVR?)

Returns current data averaging at controller.

Parameter: int SA_Averaging SA_Averaging
Direction: Up
Valid values:
 Minimum: 1
 Maximum: 9999
Description: Data averaging factor.

11.1.3 Data output

11.1.3.1 General

11.1.3.1.1 Set_IntensityMode (DFI)

Set normalized or raw intensity to transmit.
 Only available at IFD2401 from firmware version V1.2.56.

Parameter: int SP_TransmitIntensity SP_TransmitIntensity
Direction: Down
Valid values:
 0= Normalized
 1= Raw
Description: Intensity mode.

11.1.3.1.2 Get_IntensityMode (DFI?)

Get if normalized or raw intensity is transmitted.
 Only available at IFD2401 from firmware version V1.2.56.

Parameter: int SA_TransmitIntensity SA_TransmitIntensity
Direction: Up
Valid values:
 0= Normalized
 1= Raw
Description: Intensity mode.

11.1.3.1.3 Set_HoldLastValid (HLV)

Enable/Disable the "Hold Last Value" mode.

Parameter: int SP_LastValid SP_LastValid
Direction: Down
Valid values:
 Minimum: 0
 Maximum: 999
Description: Max number of points to hold.

11.1.3.1.4 Get_HoldLastValid (HLV?)

Get the current "Hold Last Value" mode.

Parameter: int SA_LastValid SA_LastValid
Direction: Up
Valid values:
Minimum: 0
Maximum: 999
Description: Max number of points to hold.

11.1.3.1.5 Set_MissingSignal (RSP)

Set behaviour for missing second peak in thickness mode.

Parameter: int SP_Option SP_Option
Direction: Down
Valid values:
Minimum: 0
Maximum: 1
Description: Mode for behaviour.

11.1.3.1.6 Get_MissingSignal (RSP?)

Returns behaviour for missing second peak in thickness mode.

Parameter: int SA_Option SA_Option
Direction: Up
Valid values:
Minimum: 0
Maximum: 1
Description: Mode for behaviour.

11.1.3.1.7 Set_Reverse (RVS)

Reverse the distance signal.

Parameter: int SP_Reverse SP_Reverse
Direction: Down
Valid values:
 0= Normal direction
 1= Reverse direction
Description: In reverse mode, measure range - distance is transmitted.

11.1.3.1.8 Get_Reverse (RVS?)

Returns setting for reverse mode.

Parameter: int SA_Reverse SA_Reverse
Direction: Up
Valid values:
 0= Normal direction
 1= Reverse direction
Description: Reverse mode

11.1.3.1.9 Set_Ascii (ASC)

Set digital data transfer (only values, no sensor answer) to ASCII mode.

11.1.3.1.10 Set_Binary (BIN)

Set digital data transfer (only values, no sensor answer) to binary mode.

11.1.3.2 Selected measurement values

11.1.3.2.1 Set_OutputData (SOD)

Select the data to be output from controller.

Parameter: int SP_X1..16

SP_X1..16

Direction: Down

Valid values:

- 0= off
- 1= RS232/RS422
- 9= USB

Description: Selection of data of the sensor. This is necessary for data conversion and scaling.

Depending on measure mode (distance / thickness) the selected value have different meaning. The following list shows the meaning for X1..16.

X1 means (Distance / Thickness).

X2 means (not used / Distance1).

X3 means (Auto adaptive mode data / Distance2).

X4 means (Intensity / Auto adaptive mode data).

X5 means (not used / Intensity1).

X6 means (not used / Intensity2).

X7 means (Barycenter / Barycenter1).

X8 means (not used / Barycenter2).

X9 means (State Flags / State Flags).

X10 means (Counter / Counter).

X11 means (Encoder 1 LSB / Encoder 1 LSB).

X12 means (Encoder 1 MSB / Encoder 1 MSB).

X13 means (Encoder 2 LSB / Encoder 2 LSB).

X14 means (Encoder 2 MSB / Encoder 2 MSB).

X15 means (Encoder 3 LSB / Encoder 3 LSB).

X16 means (Encoder 3 MSB / Encoder 3 MSB).

11.1.3.2.2 Get_OutputData (SOD?)

Get the data output from controller.

Parameter: int SA_X1..16

SA_X1..16

Direction: Up

Valid values:

- 0= off

1 = RS232/RS422

9 = USB

Description: Selection of data of the sensor. This is necessary for data conversion and scaling.

Depending on measure mode (distance / thickness) the selected value have different meaning. The following list shows the meaning for X1..16.

X1 means (Distance / Thickness).

X2 means (not used / Distance1).

X3 means (Auto adaptive mode data / Distance2).

X4 means (Intensity / Auto adaptive mode data).

X5 means (not used / Intensity1).

X6 means (not used / Intensity2).

X7 means (Barycenter / Barycenter1).

X8 means (not used / Barycenter2).

X9 means (State Flags / State Flags).

X10 means (Counter / Counter).

X11 means (Encoder 1 LSB / Encoder 1 LSB).

X12 means (Encoder 1 MSB / Encoder 1 MSB).

X13 means (Encoder 2 LSB / Encoder 2 LSB).

X14 means (Encoder 2 MSB / Encoder 2 MSB).

X15 means (Encoder 3 LSB / Encoder 3 LSB).

X16 means (Encoder 3 MSB / Encoder 3 MSB).

11.1.3.3 Analog output

11.1.3.3.1 Set_AnalogOut (ANA)

Setup the analog outputs.

Parameter: int SP_OutNr

SP_OutNr

Direction: Down

Valid values:

0 = BNC1

1 = BNC2

Description: Number of analog output.

Parameter: int SP_SOD

SP_SOD

Direction: Down

Valid values:

Minimum: 0

Maximum: 7

Description: Specifies the number (X1..X16) to output at analog out.

Parameter: int SP_0V

SP_0V

Direction: Down

Valid values:

Minimum: 0

Maximum: 99999

Description: Specifies the value which shoud output 0 V.

Parameter: int SP_10V SP_10V

Direction: Down

Valid values:

Minimum: 0

Maximum: 99999

Description: Specifies the value which shoud output 10 V.

11.1.3.3.2 Get_AnalogOut (ANA?)

Get settings of analog outputs.

Parameter: int SA_0_SOD SA_0_SOD

Direction: Up

Valid values:

Minimum: 0

Maximum: 7

Description: Analog out BNC1: 0= X1, 1= X2, ..., 15= X16.

Parameter: int SA_0_0V SA_0_0V

Direction: Up

Valid values:

Minimum: 0

Maximum: 99999

Description: Analog out BNC1: Specifies the value which shoud output 0 V.

Parameter: int SA_0_10V SA_0_10V

Direction: Up

Valid values:

Minimum: 0

Maximum: 99999

Description: Analog out BNC1: Specifies the value which shoud output 10 V.

Parameter: int SA_1_SOD SA_1_SOD

Direction: Up

Valid values:

Minimum: 0

Maximum: 7

Description: Analog out BNC2: 0= X1, 1= X2, ..., 15= X16.

Parameter: int SA_1_0V SA_1_0V

Direction: Up

Valid values:

Minimum: 0

Maximum: 99999

Description: Analog out BNC2: Specifies the value which shoud output 0 V.

Parameter: int SA_1_10V SA_1_10V

Direction: Up

Valid values:

Minimum: 0

Maximum: 99999

Description: Analog out BNC2: Specifies the value which shoud output 10 V.

11.1.3.3.3 Set_AnalogZero (SOF)

Set analog output to zero.

Parameter: int SP_Zero SP_Zero

Direction: Down

Valid values:

0 = normal

1 = set

Description: Set/reset the analog output 0V value.

11.1.4 Internal commands

11.1.4.1 Set_DataScale (CEE)

Set the scaling factor for X1, X2 and X3 in thickness mode.

This is an internal command. It should not be used by the customer.

Parameter: double SP_DataScale SP_DataScale

Direction: Down

Valid values:

Minimum: 0.0001 (from firmware V1.2.56), 1.0 (for older versions)

Maximum: 5.0

Description: Scaling factor of the sensor.

11.1.4.2 Get_DataScale (CEE?)

Get the scaling factor for X1, X2 and X3 in thickness mode.

This is an internal command. It should not be used by the customer.

Parameter: double SA_DataScale SA_DataScale

Direction: Up

Valid values:

Minimum: 0.0001 (from firmware V1.2.56), 1.0 (for older versions)

Maximum: 5.0

Description: Scaling factor of the sensor.

11.1.4.3 Set_BarycenterSca (CEB)

Set the scaling factor for barycenter values.

This is an internal command. It should not be used by the customer.

Parameter: double SP_BarycenterScale SP_BarycenterScale

Direction: Down

Valid values:

Minimum: 32.0

Maximum: 32.0

Description: Scaling factor for Barycenter.

11.1.4.4 Get_BarycenterSca (CEB?)

Get the scaling factor for barycenter values.

This is an internal command. It should not be used by the customer.

Parameter: double SA_BarycenterScale

SA_BarycenterScale

Direction: Up

Valid values:

Minimum: 32.0

Maximum: 32.0

Description: Scaling factor for Barycenter.

11.1.4.5 Set_BarycenterOff (CRB)

Set the offset values for barycenter values.

This is an internal command. It should not be used by the customer.

Parameter: double SP_BarycenterOffset

SP_BarycenterOffset

Direction: Down

Valid values:

Minimum: 0.0

Maximum: 1023.9

Description: Offset for Barycenter.

11.1.4.6 Get_BarycenterOff (CRB?)

Get the offset value for barycenter values.

This is an internal command. It should not be used by the customer.

Parameter: double SA_BarycenterOffset

SA_BarycenterOffset

Direction: Up

Valid values:

Minimum: 0.0

Maximum: 1023.9

Description: Offset for Barycenter.

11.1.4.7 Upload_RefIdxFile

Send a refractive index file to the controller.

This is an internal command. It should not be used by the customer.

Parameter: int SP_RefraIndexFileIdx

SP_RefraIndexFileIdx

Direction: Down

Valid values:

Minimum: 1

Maximum: 8

Description: Index on which position the file should be stored.

11.1. Commands for IFD2401/IFD2431

Parameter: String SP_RefraIndexFile SP_RefraIndexFile
Direction: Down
Valid values:
 2049 (IFD2401) or 1025 (IFD2431) lines
 First line is a name (limited to 19 characters), the other lines are refractive index values with precision four.
Description: Refractive index file.

11.1.4.8 Get_WhiteRef (SGW)

Get the white reference table table of controller.
 This is an internal command. It should not be used by the customer.

Parameter: Binary data SA_WhiteRef SA_WhiteRef
Direction: Up
Valid values:
 4096 (IFD2401) or 2048 (IFD2431) bytes, convertible to 2048 (IFD2401)
 or 1024 (IFD2431) words.
Description: White reference table.

11.1.4.9 Get_NormSig (SGN)

Get the norm signal table of controller.
 This is an internal command. It should not be used by the customer.

Parameter: Binary data SA_NormSig SA_NormSig
Direction: Up
Valid values:
 4096 (IFD2401) or 2048 (IFD2431) bytes, convertible to 2048 (IFD2401)
 or 1024 (IFD2431) words.
Description: Norm signal table.

11.1.4.10 Get_CalibTable (SGC)

Get the calibration table of controller.
 This is an internal command. It should not be used by the customer.

Parameter: Binary data SA_CalibTable SA_CalibTable
Direction: Up
Valid values:
 4096 (IFD2401) or 2048 (IFD2431) bytes, convertible to 2048 (IFD2401)
 or 1024 (IFD2431) words.
Description: Calibration table.

11.1.4.11 Upload_CalibTable

Send a calibration table for selected sensor to the controller.

This is an internal command. It should not be used by the customer.

Parameter: int SP_BlockSize

SP_BlockSize

Direction: Down

Unit: Values

Valid values:

Minimum: 1

Maximum: 2147483647 (INT_MAX)

Default: 64

Description: Size of blocks in which the calibration table is separated before sending.

Parameter: Binary data SP_CalibTable

SP_CalibTable

Direction: Down

Valid values:

2048 (IFD2401) or 1024 (IFD2431) values of datatype float (each 4 byte).

Description: Calibration table for selected sensor.

Parameter: double SP_FullRange

SP_FullRange

Direction: Down

Unit: μm

Valid values:

Minimum: 1.17549e-38 (FLT_MIN)

Maximum: 3.40282e+38 (FLT_MAX)

Description: Measure range of selected sensor.

11.2 Commands for IFD2445/IFD2451/IFD2461/IFD2471

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [RS232](#) (additional, e.g. IF2001_USB (RS422) and RS232 high level interface).
- [IF2004](#) (native).
- [TCP/IP](#) (native).
- [IF2004_USB](#) (native).
- [IF2008](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_AllParameters](#) (SP_Additional= 1) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib timeout checking, to calculate datarate, to interpret and scale data and to assign values. If second bit of [IP_AutomaticMode](#) is set (2), MEDAQLib calls optionally sensor command [Set_IPDataTransferMode](#) and [Set_DataOutInterface](#) at [OpenSensor](#).

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor.
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command [Get_AllParameters](#) (SP_Additional= 1)).

The values of one data frame are filled in the arrays one after another. Each array always starts with a new data frame.

11.2.1 General commands

11.2.1.1 General

11.2.1.1.1 Get_Help (HELP)

Retrieve a help text from controller for a specific command.

Parameter: String SP_Command

SP_Command

Direction: Down

Valid values:

"" (empty string, means general help)
or any command name

Description: Name of the command.

Parameter: String SA_HelpText

SA_HelpText

Direction: Up

Description: Help text to the command.

11.2.1.1.2 Get_Info (GETINFO)

Retrieve information about the controller.

Parameter: String SA_Sensor

SA_Sensor

Direction: Up

Valid values:

Numeric value

Description: Name of the controller.

Parameter: String SA_SerialNumber

SA_SerialNumber

Direction: Up

Valid values:

Numeric value

Description: Serial number of the controller.

Parameter: String SA_Option

SA_Option

Direction: Up

Valid values:

Numeric value

Description: Option of the controller.

Parameter: String SA_ArticleNumber

SA_ArticleNumber

Direction: Up

Valid values:

Numeric value

Description: Article number of the controller.

Parameter: String SA_MacAddress

SA_MacAddress

Direction: Up

Valid values:

Valid MAC address in form of xx-xx-xx-xx-xx-xx

Description: MAC address (low level ethernet address) of the controller.

Parameter: String SA_Softwareversion	SA_Softwareversion
Direction: Up	
Description: Software version of firmware in the controller.	
Parameter: String SA_ImageType	SA_ImageType
Direction: Up	
Description: Firmware image type used by the controller.	
Parameter: String SA_HardwareRevision	SA_HardwareRevision
Direction: Up	
Valid values:	
Numeric value	
Description: Revision index of the controller board.	

11.2.1.1.3 Get_Temperature (GETTEMP)

Get temperature of controller.

Parameter: double SA_Temperature	SA_Temperature
Direction: Up	
Unit: °C	
Description: Temperature.	

11.2.1.1.4 Set_Echo (ECHO)

Set echo for sensor commands.

Parameter: int SP_Echo	SP_Echo
Direction: Down	
Valid values:	
0= Off	
1= On	
Description: Echo mode.	

11.2.1.1.5 Get_Echo (ECHO)

Get the echo mode.

Parameter: int SA_Echo	SA_Echo
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Off	
1= On	
Description: Echo mode.	

11.2.1.1.6 Get_AllParameters (PRINT)

Get all parameters from controller.

Parameter: int SP_Additional	SP_Additional
Direction: Down	
Valid values:	
0= No	
1= Yes	
Description:	If set, additional information about controller, sensor and material is output.
Parameter: int SA_SyncMode	SA_SyncMode
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Master	
2= Slave	
3= Slave with external trigger (SLAVE_EXT)	
Description:	Synchronisation mode.
Parameter: int SA_UserLevel	SA_UserLevel
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= User	
1= Professional	
Description:	Actual user level.
Parameter: int SA_DefaultUser	SA_DefaultUser
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= User	
1= Professional	
Description:	Default user level.
Parameter: int SA_Echo	SA_Echo
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Off	
1= On	
Description:	Echo mode.
Parameter: int SA_ActiveSensor	SA_ActiveSensor
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 19	
Description:	Index of active sensor head.

Parameter: int SA_TriggerMode	SA_TriggerMode
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Edge	
2= Level (PULSE)	
3= Software	
4= Encoder	
Description: Trigger mode.	
Parameter: int SA_TriggerMoment	SA_TriggerMoment
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Input	
1= Output	
Description: Trigger moment.	
Parameter: int SA_TriggerLevel	SA_TriggerLevel
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= High	
1= Low	
Description: Trigger level.	
Parameter: int SA_TriggerCount	SA_TriggerCount
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 16383	
Description: Number of values to measure. 0 means no trigger, 16383 means endless measurement.	
Parameter: int SA_EncoderNumber	SA_EncoderNumber
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 3	
Description: Encoder number to trigger on.	
Parameter: double SA_EncoderIncrements	SA_EncoderIncrements
Direction: Up	
Valid values:	
Minimum: 1.0	
Maximum: 2147483647.0 (INT_MAX)	
Description: Number of encoder increments before trigger. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.	

Parameter: double SA_EncoderMinValue	SA_EncoderMinValue
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 4294967295.0 (UINT_MAX)	
Description: Lower encoder limit. Above this value it will be triggered. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.	
Parameter: double SA_Encoder.MaxValue	SA_Encoder.MaxValue
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 4294967295.0 (UINT_MAX)	
Description: Upper encoder limit. Below this value it will be triggered. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.	
Parameter: int SA_EncoderInterpolation1	SA_EncoderInterpolation1
Direction: Up	
Valid values:	
1	
2	
4	
Description: Encoder interpolation.	
Parameter: int SA_EncoderInterpolation2	SA_EncoderInterpolation2
Direction: Up	
Valid values:	
1	
2	
4	
Description: Encoder interpolation.	
Parameter: int SA_EncoderInterpolation3	SA_EncoderInterpolation3
Direction: Up	
Valid values:	
1	
2	
4	
Description: Encoder interpolation.	
Parameter: int SA_EncoderMode1	SA_EncoderMode1
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = No action (NONE)	
1 = Set encoder value to preset value only one time (ONE)	
2 = Set encoder value to preset value each time (EVER)	
Description: Mode of encoder when reference is reached.	

Parameter: int SA_EncoderMode2	SA_EncoderMode2
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= No action (NONE)	
1= Set encoder value to preset value only one time (ONE)	
2= Set encoder value to preset value each time (EVER)	
Description: Mode of encoder when reference is reached.	
Parameter: int SA_EncoderMode3	SA_EncoderMode3
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= No action (NONE)	
1= Set encoder value to preset value only one time (ONE)	
2= Set encoder value to preset value each time (EVER)	
Description: Mode of encoder when reference is reached.	
Parameter: double SA_EncoderPreload1	SA_EncoderPreload1
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 4294967295.0 (UINT_MAX)	
Description: Preload value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.	
Parameter: double SA_EncoderPreload2	SA_EncoderPreload2
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 4294967295.0 (UINT_MAX)	
Description: Preload value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.	
Parameter: double SA_EncoderPreload3	SA_EncoderPreload3
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 4294967295.0 (UINT_MAX)	
Description: Preload value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.	
Parameter: double SA_Encoder.MaxValue1	SA_Encoder.MaxValue1
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 4294967295.0 (UINT_MAX)	
Description: Maximum value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.	
Parameter: double SA_Encoder.MaxValue2	SA_Encoder.MaxValue2
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 4294967295.0 (UINT_MAX)	
Description: Maximum value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.	

Parameter: double SA_Encoder.MaxValue3	SA_Encoder.MaxValue3
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 4294967295.0 (UINT_MAX)	
Description: Maximum value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.	
Parameter: int SA_EthernetMode	SA_EthernetMode
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Ethernet	
1 = Ethercat	
Description: Ethernet mode.	
Parameter: int SA_DHCPEnabled	SA_DHCPEnabled
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = FALSE	
1 = TRUE	
Description: Get settings if controller should use a static IP address ask for IP at DHCP server (dynamic IP address).	
Parameter: String SA_Address	SA_Address
Direction: Up	
Valid values:	
Valid IP address in form of xxx.xxx.xxx.xxx	
Description: IP address of the controller. If DHCP is enabled it returns the currently assigned IP address.	
Parameter: String SA_SubnetMask	SA_SubnetMask
Direction: Up	
Valid values:	
Valid network mask (e.g. 255.255.255.0 for a Class C network)	
Description: Network mask of the controller. If DHCP is enabled it returns the currently assigned network mask.	
Parameter: String SA_Gateway	SA_Gateway
Direction: Up	
Valid values:	
Valid IP address of default gateway in form of xxx.xxx.xxx.xxx	
Description: Address of the default gateway. If DHCP is enabled it returns the currently assigned default gateway.	
Parameter: int SA_Protocol	SA_Protocol
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = TCP server (SERVER/TCP)	
1 = TCP client (CLIENT/TCP)	
2 = UDP sender (CLIENT/UDP)	
3 = None	
Description: Specifies if data should be send using TCP or UDP.	

Parameter: String SA_RemoteAddress	SA_RemoteAddress
Direction: Up	
Valid values:	
Valid IP address of receiver of data	
Description:	Address of remote computer to send data to.
Parameter: int SA_Port	SA_Port
Direction: Up	
Valid values:	
Minimum: 1024	
Maximum: 65535	
Description:	Port to send data to or to listen for incoming requests.
Parameter: int SA_SensorBaudrate	SA_SensorBaudrate
Direction: Up	
Valid values:	
4000000	
3500000	
3000000	
2500000	
2000000	
1500000	
921600	
691200	
460800	
230400	
115200	
9600	
Unit:	Baud
Description:	Baudrate of controller.
Parameter: int SA_MeasureMode	SA_MeasureMode
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Distance	
1 = Thickness	
2 = Video	
3 = MultiLayer (only IFD2451/IFD2461/IFD2471)	
4 = VideoStream	
Description:	Measure mode.
Parameter: int SA_RefractiveCorrection	SA_RefractiveCorrection
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Off	
1 = On	
Description:	Tells if refractive correction is enabled.
Parameter: int SA_NumberOfPeaks	SA_NumberOfPeaks
Direction: Up	
Valid values:	
Minimum: 2	
Maximum: 6	
Description:	Number of peaks to detect.

Parameter: int SA_ShutterMode	SA_ShutterMode
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Search the best exposure time and measuring rate automatically (SEARCH)	
1 = Control exposure time, measuring rate is set manually (MEAS)	
2 = Exposure time and measuring rate are set manually (MANUAL)	
3 = Use two fixed exposure times alternately (2TIMEALT)	
4 = Use the suitable exposure time of two fixed ones (2TIMES)	
Description: Shutter mode.	
Parameter: double SA_Measrate	SA_Measrate
Direction: Up	
Valid values:	
Minimum: 0.1 for IFD2445/IFD2451/IFD2461, 0.3 for IFD2471	
Maximum: 2.5 for IFD2445, 10.0 for IFD2451, 25.0 for IFD2461, 70.0 for IFD2471	
Unit: kHz	
Description: Samplerate of measurement. For older firmware versions (before V007.117.134.02), only discrete values are valid: 0.1, 0.2, 0.3, 1.0, 2.5, 5.0, 10.0, 25.0, 50.0, 70.0	
Parameter: double SA_ShutterTime1	SA_ShutterTime1
Direction: Up	
Valid values:	
Minimum: 0.075	
Maximum: 10000.0 (IFD2445/IFD2451/IFD2461) or 3333.325 (IFD2471)	
Unit: μ s	
Description: First shutter time.	
Parameter: double SA_ShutterTime2	SA_ShutterTime2
Direction: Up	
Valid values:	
Minimum: 0.075	
Maximum: 10000.0 (IFD2445/IFD2451/IFD2461) or 3333.325 (IFD2471)	
Unit: μ s	
Description: Second shutter time.	
Parameter: int SA_ROIStart	SA_ROIStart
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 511	
Unit: Pixel	
Description: First position on CCD.	
Parameter: int SA_ROIEnd	SA_ROIEnd
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 511	
Unit: Pixel	
Description: Last position on CCD.	

Parameter: int SA_VideoAverage	SA_VideoAverage
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Recursive over 2 lines (REC2)	
2= Recursive over 4 lines (REC4)	
3= Recursive over 8 lines (REC8)	
4= Moving over 2 lines (MOV2)	
5= Moving over 3 lines (MOV3)	
6= Moving over 4 lines (MOV4)	
7= Median over 3 lines (MED3)	
Description: Averaging mode.	
Parameter: double SA_Threshold	SA_Threshold
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 99.0	
Unit: %	
Description: Video threshold.	
Parameter: String SA_ActiveMaterial	SA_ActiveMaterial
Direction: Up	
Description: Name of material.	
Parameter: String SA_MaterialMultiPeak12	SA_MaterialMultiPeak12
Direction: Up	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Name of material between first and second peak.	
Parameter: String SA_MaterialMultiPeak23	SA_MaterialMultiPeak23
Direction: Up	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Name of material between second and third peak.	
Parameter: String SA_MaterialMultiPeak34	SA_MaterialMultiPeak34
Direction: Up	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Name of material between third and fourth peak.	
Parameter: String SA_MaterialMultiPeak45	SA_MaterialMultiPeak45
Direction: Up	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Name of material between fourth and fifth peak.	

Parameter: String SA_MaterialMultiPeak56	SA_MaterialMultiPeak56
Direction: Up	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Name of material between fifth and sixth peak.	
Parameter: int SA_AveragingType	SA_AveragingType
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Moving average (MOVING)	
2= Recursive averaging (RECURSIVE)	
3= Median	
Description: Averaging type.	
Parameter: int SA_MovingCount	SA_MovingCount
Direction: Up	
Valid values:	
2	
4	
8	
16	
32	
64	
128	
256	
512	
1024	
Description: Number of value for the averaging window. This parameter is only available at moving average.	
Parameter: int SA_RecursiveCount	SA_RecursiveCount
Direction: Up	
Valid values:	
Minimum: 2	
Maximum: 32768	
Description: Number of values for recursive averaging. This parameter is only available at recursive average.	
Parameter: int SA_MedianCount	SA_MedianCount
Direction: Up	
Valid values:	
3	
5	
7	
9	
Description: Number of values to build median. This parameter is only available at median.	

Parameter: int SA_SpikeCorrection	SA_SpikeCorrection
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= off	
1= on	
Description: Spike correction.	
Parameter: int SA_NbrEvaluatedValues	SA_NbrEvaluatedValues
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 10	
Description: Number of values to evaluate for spike correction.	
Parameter: double SA_ToleranceRange	SA_ToleranceRange
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 100.0	
Description: Tolerance range for spike correction.	
Parameter: int SA_NbrCorrectedValues	SA_NbrCorrectedValues
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 100	
Description: Number of values to correct at spike correction.	
Parameter: int SA_StatisticSignal	SA_StatisticSignal
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Distance 1	
1= Distance 2	
2= Distance 3 (only IFD2451/IFD2461/IFD2471)	
3= Distance 4 (only IFD2451/IFD2461/IFD2471)	
4= Distance 5 (only IFD2451/IFD2461/IFD2471)	
5= Distance 6 (only IFD2451/IFD2461/IFD2471)	
6= Thickness 12	
7= Thickness 13 (only IFD2451/IFD2461/IFD2471)	
8= Thickness 14 (only IFD2451/IFD2461/IFD2471)	
9= Thickness 15 (only IFD2451/IFD2461/IFD2471)	
10= Thickness 16 (only IFD2451/IFD2461/IFD2471)	
11= Thickness 23 (only IFD2451/IFD2461/IFD2471)	
12= Thickness 24 (only IFD2451/IFD2461/IFD2471)	
13= Thickness 25 (only IFD2451/IFD2461/IFD2471)	
14= Thickness 26 (only IFD2451/IFD2461/IFD2471)	
15= Thickness 34 (only IFD2451/IFD2461/IFD2471)	
16= Thickness 35 (only IFD2451/IFD2461/IFD2471)	
17= Thickness 36 (only IFD2451/IFD2461/IFD2471)	
18= Thickness 45 (only IFD2451/IFD2461/IFD2471)	
19= Thickness 46 (only IFD2451/IFD2461/IFD2471)	
20= Thickness 56 (only IFD2451/IFD2461/IFD2471)	
Description: Value which is used for statistic calculation.	

Parameter: int SA_MasterSignal SA_MasterSignal

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Distance 1
- 1= Distance 2
- 2= Distance 3 (only IFD2451/IFD2461/IFD2471)
- 3= Distance 4 (only IFD2451/IFD2461/IFD2471)
- 4= Distance 5 (only IFD2451/IFD2461/IFD2471)
- 5= Distance 6 (only IFD2451/IFD2461/IFD2471)
- 6= Thickness 12
- 7= Thickness 13 (only IFD2451/IFD2461/IFD2471)
- 8= Thickness 14 (only IFD2451/IFD2461/IFD2471)
- 9= Thickness 15 (only IFD2451/IFD2461/IFD2471)
- 10= Thickness 16 (only IFD2451/IFD2461/IFD2471)
- 11= Thickness 23 (only IFD2451/IFD2461/IFD2471)
- 12= Thickness 24 (only IFD2451/IFD2461/IFD2471)
- 13= Thickness 25 (only IFD2451/IFD2461/IFD2471)
- 14= Thickness 26 (only IFD2451/IFD2461/IFD2471)
- 15= Thickness 34 (only IFD2451/IFD2461/IFD2471)
- 16= Thickness 35 (only IFD2451/IFD2461/IFD2471)
- 17= Thickness 36 (only IFD2451/IFD2461/IFD2471)
- 18= Thickness 45 (only IFD2451/IFD2461/IFD2471)
- 19= Thickness 46 (only IFD2451/IFD2461/IFD2471)
- 20= Thickness 56 (only IFD2451/IFD2461/IFD2471)

Description: Value which is used for mastering.

Parameter: int SA_MeasurePeak SA_MeasurePeak

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= First peak at distance mode resp. first and last peak at thickness mode (F_L)
- 1= Last peak at distance mode resp. last and next to last peak at thickness mode (L_SL)
- 2= First peak at distance mode resp. first and second peak at thickness mode (F_S)
- 3= Highest peak at distance mode resp. highest and second highest peak at thickness mode (H_SH)

Description: Peaks to evalualte.

Parameter: int SA_StatisticDepth SA_StatisticDepth

Direction: Up

Valid values:

Minimum: 2

Maximum: 2147483647 (INT_MAX)

Description: Window size for floating statistic calculation. The value must be power of two (2, 4, 8, ..., 16384). Value greater as 16384 calculates statistic over all values.

Parameter: int SA_Master SA_Master

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor

0= no (NONE)

1= yes (MASTER=

Description: Specifies if mastering is active.

Parameter: double SA_MasterValue

SA_MasterValue

Direction: Up

Valid values:

Minimum: -2* measuring range

Maximum: +2* measuring range

Unit: mm

Description: Master value

Parameter: int SA_DataOutInterface

SA_DataOutInterface

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= None

1= RS422

2= Ethernet

3= HTTP

4= Ethercat

Description: Active interface for data output.

Parameter: int SA_Resampling

SA_Resampling

Direction: Up

Valid values:

Minimum: 1

Maximum: 4200000

Description: Resampling value.

Parameter: int SA_ResampleAnalog

SA_ResampleAnalog

Direction: Up

Valid values:

0= no

1= yes

Description: Analog output is resampled.

Parameter: int SA_ResampleRS422

SA_ResampleRS422

Direction: Up

Valid values:

0= no

1= yes

Description: RS422 output is resampled.

Parameter: int SA_ResampleEthernet

SA_ResampleEthernet

Direction: Up

Valid values:

0= no

1= yes

Description: Output over ethernet is resampled.

Parameter: int SA_HoldLastValid	SA_HoldLastValid
Direction: Up	
Valid values:	
Minimum: -1	
Maximum: 1024	
Description: Values to replace by last valid value. -1 means no value to hold, 0 means never output an error value (always hold last valid value).	
Parameter: int SA_OutputDistance1_RS422	SA_OutputDistance1_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if distance 1 is transmitted.	
Parameter: int SA_OutputDistance2_RS422	SA_OutputDistance2_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if distance 2 is transmitted.	
Parameter: int SA_OutputDistance3_RS422	SA_OutputDistance3_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if distance 3 is transmitted.	
Parameter: int SA_OutputDistance4_RS422	SA_OutputDistance4_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if distance 4 is transmitted.	
Parameter: int SA_OutputDistance5_RS422	SA_OutputDistance5_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if distance 5 is transmitted.	

Parameter: int SA_OutputDistance6_RS422	SA_OutputDistance6_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if distance 6 is transmitted.	
Parameter: int SA_OutputDistance1_ETH	SA_OutputDistance1_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if distance 1 is transmitted.	
Parameter: int SA_OutputDistance2_ETH	SA_OutputDistance2_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if distance 2 is transmitted.	
Parameter: int SA_OutputDistance3_ETH	SA_OutputDistance3_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if distance 3 is transmitted.	
Parameter: int SA_OutputDistance4_ETH	SA_OutputDistance4_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if distance 4 is transmitted.	

Parameter: int SA_OutputDistance5_ETH	SA_OutputDistance5_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if distance 5 is transmitted.	
Parameter: int SA_OutputDistance6_ETH	SA_OutputDistance6_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if distance 6 is transmitted.	
Parameter: int SA_OutputThickness12_RS422	SA_OutputThickness12_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if thickness between first and second peak is transmitted.	
Parameter: int SA_OutputThickness13_RS422	SA_OutputThickness13_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between first and third peak is transmitted.	
Parameter: int SA_OutputThickness14_RS422	SA_OutputThickness14_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between first and fourth peak is transmitted.	

Parameter: int SA_OutputThickness15_RS422	SA_OutputThickness15_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between first and fifth peak is transmitted.	
Parameter: int SA_OutputThickness16_RS422	SA_OutputThickness16_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between first and sixth peak is transmitted.	
Parameter: int SA_OutputThickness23_RS422	SA_OutputThickness23_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and third peak is transmitted.	
Parameter: int SA_OutputThickness24_RS422	SA_OutputThickness24_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and fourth peak is transmitted.	
Parameter: int SA_OutputThickness25_RS422	SA_OutputThickness25_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and fifth peak is transmitted.	

Parameter: int SA_OutputThickness26_RS422	SA_OutputThickness26_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and sixth peak is transmitted.	
Parameter: int SA_OutputThickness34_RS422	SA_OutputThickness34_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between third and fourth peak is transmitted.	
Parameter: int SA_OutputThickness35_RS422	SA_OutputThickness35_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between third and fifth peak is transmitted.	
Parameter: int SA_OutputThickness36_RS422	SA_OutputThickness36_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between third and sixth peak is transmitted.	
Parameter: int SA_OutputThickness45_RS422	SA_OutputThickness45_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between fourth and fifth peak is transmitted.	

Parameter: int SA_OutputThickness46_RS422	SA_OutputThickness46_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description:	Specify if thickness between fourth and sixth peak is transmitted.
Parameter: int SA_OutputThickness56_RS422	SA_OutputThickness56_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description:	Specify if thickness between fifth and sixth peak is transmitted.
Parameter: int SA_OutputThickness12_ETH	SA_OutputThickness12_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description:	Specify if thickness between first and second peak is transmitted.
Parameter: int SA_OutputThickness13_ETH	SA_OutputThickness13_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description:	Specify if thickness between first and third peak is transmitted.
Parameter: int SA_OutputThickness14_ETH	SA_OutputThickness14_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description:	Specify if thickness between first and fourth peak is transmitted.

Parameter: int SA_OutputThickness15_ETH	SA_OutputThickness15_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between first and fifth peak is transmitted.	
Parameter: int SA_OutputThickness16_ETH	SA_OutputThickness16_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between first and sixth peak is transmitted.	
Parameter: int SA_OutputThickness23_ETH	SA_OutputThickness23_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and third peak is transmitted.	
Parameter: int SA_OutputThickness24_ETH	SA_OutputThickness24_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and fourth peak is transmitted.	
Parameter: int SA_OutputThickness25_ETH	SA_OutputThickness25_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and fifth peak is transmitted.	

Parameter: int SA_OutputThickness26_ETH	SA_OutputThickness26_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and sixth peak is transmitted.	
Parameter: int SA_OutputThickness34_ETH	SA_OutputThickness34_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between third and fourth peak is transmitted.	
Parameter: int SA_OutputThickness35_ETH	SA_OutputThickness35_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between third and fifth peak is transmitted.	
Parameter: int SA_OutputThickness36_ETH	SA_OutputThickness36_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between third and sixth peak is transmitted.	
Parameter: int SA_OutputThickness45_ETH	SA_OutputThickness45_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between fourth and fifth peak is transmitted.	

Parameter: int SA_OutputThickness46_ETH	SA_OutputThickness46_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description:	Specify if thickness between fourth and sixth peak is transmitted.
Parameter: int SA_OutputThickness56_ETH	SA_OutputThickness56_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description:	Specify if thickness between fifth and sixth peak is transmitted.
Parameter: int SA_OutputStatisticMin_RS422	SA_OutputStatisticMin_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description:	Specify if min value is transmitted.
Parameter: int SA_OutputStatisticMax_RS422	SA_OutputStatisticMax_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description:	Specify if max value is transmitted.
Parameter: int SA_OutputStatisticPeak2Peak_RS422	SA_OutputStatisticPeak2Peak_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description:	Specify if peak to peak value is transmitted.
Parameter: int SA_OutputStatisticMin_ETH	SA_OutputStatisticMin_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description:	Specify if min value is transmitted.
Parameter: int SA_OutputStatisticMax_ETH	SA_OutputStatisticMax_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description:	Specify if max value is transmitted.

Parameter: int SA_OutputStatisticPeak2Peak_ETH	SA_OutputStatistic-Peak2Peak_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if peak to peak value is transmitted.	
Parameter: int SA_OutputAdditionalShutterTime_RS422	SA_OutputAdditionalShutterTime_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	
Parameter: int SA_OutputAdditionalEncoder1_RS422	SA_OutputAdditionalEncoder1_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if encoder 1 is transmitted.	
Parameter: int SA_OutputAdditionalEncoder2_RS422	SA_OutputAdditionalEncoder2_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if encoder 2 is transmitted.	
Parameter: int SA_OutputAdditionalEncoder3_RS422	SA_OutputAdditionalEncoder3_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if encoder 3 is transmitted.	
Parameter: int SA_OutputAdditionalCounter_RS422	SA_OutputAdditionalCounter_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SA_OutputAdditionalTimestamp_RS422	SA_OutputAdditionalTimestamp_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SA_OutputAdditionalIntensity_RS422	SA_OutputAdditionalIntensity_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity is transmitted.	

Parameter: int SA_OutputAdditionalState_RS422	SA_OutputAdditionalState_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	
Parameter: int SA_OutputAdditionalTrgTimeDiff_RS422	SA_OutputAdditionalTrgTimeDiff_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if trigger time difference is transmitted.	
Parameter: int SA_OutputAdditionalMeasrate_RS422	SA_OutputAdditionalMeasrate_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measrate is transmitted.	
Parameter: int SA_OutputAdditionalShutterTime_ETH	SA_OutputAdditionalShutterTime_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	
Parameter: int SA_OutputAdditionalEncoder1_ETH	SA_OutputAdditionalEncoder1_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if encoder 1 is transmitted.	
Parameter: int SA_OutputAdditionalEncoder2_ETH	SA_OutputAdditionalEncoder2_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if encoder 2 is transmitted.	
Parameter: int SA_OutputAdditionalEncoder3_ETH	SA_OutputAdditionalEncoder3_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if encoder 3 is transmitted.	
Parameter: int SA_OutputAdditionalCounter_ETH	SA_OutputAdditionalCounter_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	

Parameter: int SA_OutputAdditionalTimestamp_ETH	SA_OutputAdditionalTimestamp_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SA_OutputAdditionalIntensity_ETH	SA_OutputAdditionalIntensity_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity is transmitted.	
Parameter: int SA_OutputAdditionalState_ETH	SA_OutputAdditionalState_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	
Parameter: int SA_OutputAdditionalTrgTimeDiff_ETH	SA_OutputAdditionalTrgTimeDiff_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if trigger time difference is transmitted.	
Parameter: int SA_OutputAdditionalMeasrate_ETH	SA_OutputAdditionalMeasrate_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measrate is transmitted.	
Parameter: int SA_OutputVideoRaw_ETH	SA_OutputVideoRaw_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if raw video signal is transmitted.	
Parameter: int SA_OutputVideoDark_ETH	SA_OutputVideoDark_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if dark preprocessed video signal is transmitted.	
Parameter: int SA_OutputVideoLight_ETH	SA_OutputVideoLight_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if light preprocessed video signal is transmitted.	

Parameter: int SA_OutputVideoDarkTable_ETH	SA_OutputVideoDarkTable_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if dark table is transmitted.	
Parameter: int SA_OutputVideoLightTable_ETH	SA_OutputVideoLight- Table_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if light table is transmitted.	
Parameter: int SA_OutputVideoThreshold_ETH	SA_OutputVideoThreshold_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if video threshold table is transmitted.	
Parameter: int SA_ErrorOutput1	SA_ErrorOutput1
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Intensity error (ER1)	
2= Out of range (ER2)	
3= Intensity error or out of range (ER12)	
4= Below low limit (LI1)	
5= Above high limit (LI2)	
6= Out of limits (LI12)	
Description: Condition for error output.	
Parameter: int SA_ErrorOutput2	SA_ErrorOutput2
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Intensity error (ER1)	
2= Out of range (ER2)	
3= Intensity error or out of range (ER12)	
4= Below low limit (LI1)	
5= Above high limit (LI2)	
6= Out of limits (LI12)	
Description: Condition for error output.	
Parameter: int SA_DataSource	SA_DataSource
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Distance 1	
1= Distance 2	
3= Distance 3 (only IFD2451/IFD2461/IFD2471)	

4= Distance 4 (only IFD2451/IFD2461/IFD2471)
 5= Distance 5 (only IFD2451/IFD2461/IFD2471)
 6= Distance 6 (only IFD2451/IFD2461/IFD2471)
 2= Thickness 12
 7= Thickness 13 (only IFD2451/IFD2461/IFD2471)
 8= Thickness 14 (only IFD2451/IFD2461/IFD2471)
 9= Thickness 15 (only IFD2451/IFD2461/IFD2471)
 10= Thickness 16 (only IFD2451/IFD2461/IFD2471)
 11= Thickness 23 (only IFD2451/IFD2461/IFD2471)
 12= Thickness 24 (only IFD2451/IFD2461/IFD2471)
 13= Thickness 25 (only IFD2451/IFD2461/IFD2471)
 14= Thickness 26 (only IFD2451/IFD2461/IFD2471)
 15= Thickness 34 (only IFD2451/IFD2461/IFD2471)
 16= Thickness 35 (only IFD2451/IFD2461/IFD2471)
 17= Thickness 36 (only IFD2451/IFD2461/IFD2471)
 18= Thickness 45 (only IFD2451/IFD2461/IFD2471)
 19= Thickness 46 (only IFD2451/IFD2461/IFD2471)
 20= Thickness 56 (only IFD2451/IFD2461/IFD2471)

Description: Data source to be checked.

Parameter: double SA_LowerLimit SA_LowerLimit

Direction: Up

Valid values:

Minimum: -120.0

Maximum: 120.0

Unit: mm

Description: Lower limit.

Parameter: double SA_UpperLimit SA_UpperLimit

Direction: Up

Valid values:

Minimum: -120.0

Maximum: 120.0

Unit: mm

Description: Upper limit.

Parameter: int SA_ErrorLevel SA_ErrorLevel

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= High

1= Low

Description: Error level.

Parameter: int SA_AnalogOutput SA_AnalogOutput

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= Distance 1

1= Distance 2

3= Distance 3 (only IFD2451/IFD2461/IFD2471)

4= Distance 4 (only IFD2451/IFD2461/IFD2471)

5= Distance 5 (only IFD2451/IFD2461/IFD2471)

6= Distance 6 (only IFD2451/IFD2461/IFD2471)

2= Thickness 12
 7= Thickness 13 (only IFD2451/IFD2461/IFD2471)
 8= Thickness 14 (only IFD2451/IFD2461/IFD2471)
 9= Thickness 15 (only IFD2451/IFD2461/IFD2471)
 10= Thickness 16 (only IFD2451/IFD2461/IFD2471)
 11= Thickness 23 (only IFD2451/IFD2461/IFD2471)
 12= Thickness 24 (only IFD2451/IFD2461/IFD2471)
 13= Thickness 25 (only IFD2451/IFD2461/IFD2471)
 14= Thickness 26 (only IFD2451/IFD2461/IFD2471)
 15= Thickness 34 (only IFD2451/IFD2461/IFD2471)
 16= Thickness 35 (only IFD2451/IFD2461/IFD2471)
 17= Thickness 36 (only IFD2451/IFD2461/IFD2471)
 18= Thickness 45 (only IFD2451/IFD2461/IFD2471)
 19= Thickness 46 (only IFD2451/IFD2461/IFD2471)
 20= Thickness 56 (only IFD2451/IFD2461/IFD2471)

Description: Data to be used for analog output.

Parameter: int SA_AnalogRange SA_AnalogRange

Direction: Up

Valid values:

-1= Unknown parameter value from sensor
 0= None
 1= 0 - 5V
 2= 0 - 10V
 3= -5 - 5V
 4= -10 - 10V
 5= 4 - 20mA

Description: Analog output range.

Parameter: int SA_AnalogScaleMode SA_AnalogScaleMode

Direction: Up

Valid values:

-1= Unknown parameter value from sensor
 0= Standard
 1= Two point (TWOPOINT)

Description: Analog scale mode.

Parameter: double SA_MinValue SA_MinValue

Direction: Up

Valid values:

Minimum: -120.0
Maximum: 120.0

Unit: mm

Description: Value which represents lowest voltage/current (at two point scaling).

Parameter: double SA_MaxValue SA_MaxValue

Direction: Up

Valid values:

Minimum: -120.0
Maximum: 120.0

Unit: mm

Description: Value which represents highest voltage/current (at two point scaling).

Parameter: String SA_Sensor	SA_Sensor
Direction: Up	
Description: Name of the controller.	
Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the controller.	
Parameter: String SA_Option	SA_Option
Direction: Up	
Valid values:	
Numeric value	
Description: Option of the controller.	
Parameter: String SA_ArticleNumber	SA_ArticleNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Article number of the controller.	
Parameter: String SA_MacAddress	SA_MacAddress
Direction: Up	
Valid values:	
Valid MAC address in form of xx-xx-xx-xx-xx-xx	
Description: MAC address (low level ethernet address) of the controller.	
Parameter: String SA_Softwareversion	SA_Softwareversion
Direction: Up	
Description: Software version of firmware in the controller.	
Parameter: String SA_ImageType	SA_ImageType
Direction: Up	
Description: Firmware image type used by the controller.	
Parameter: String SA_HardwareRevision	SA_HardwareRevision
Direction: Up	
Valid values:	
Numeric value	
Description: Revision index of the controller board.	
Parameter: int SA_Pos	SA_Pos
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 19	
Description: Index of the active sensor head in the table.	
Parameter: String SA_Name	SA_Name
Direction: Up	
Description: Name of the active sensor head.	

Parameter: double SA_Range	SA_Range
Direction: Up	
Valid values:	
Minimum: 0.0	
Description: Measurement range of the active sensor head.	
Parameter: String SA_Unit	SA_Unit
Direction: Up	
Description: Unit of the measurement range.	
Parameter: String SA_Serial	SA_Serial
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the active sensor head.	
Parameter: String SA_SensorTable	SA_SensorTable
Direction: Up	
Description: Whole table in one string, separated by new lines and commas.	
Parameter: int SA_SensorTableCount	SA_SensorTableCount
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 20	
Description: Number of entries in the table. All following parameters exists from 1 to this number, e.g. SA_Pos1, SA_Pos2, ...	
Parameter: int SA_Pos1..x	SA_Pos1..x
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 19	
Description: Index of the sensor head in the table.	
Parameter: String SA_Name1..x	SA_Name1..x
Direction: Up	
Description: Name of the sensor head in the table.	
Parameter: double SA_Range1..x	SA_Range1..x
Direction: Up	
Valid values:	
Minimum: 0.0	
Description: Measurement range of the sensor head in the table.	
Parameter: String SA_Unit1..x	SA_Unit1..x
Direction: Up	
Description: Unit of the measurement range.	
Parameter: String SA_Serial1..x	SA_Serial1..x
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the sensor head in the table.	

Parameter: String SA_MaterialName	SA_MaterialName
Direction: Up	
Description: Name of the active material.	
Parameter: String SA_Description	SA_Description
Direction: Up	
Description: Description of the active material.	
Parameter: double SA_RefRACTIVEINDEX_nF	SA_RefRACTIVEINDEX_nF
Direction: Up	
Valid values:	
Minimum: 1.0	
Maximum: 4.0	
Description: Refractive index of the active material at 486 nm.	
Parameter: double SA_RefRACTIVEINDEX_nd	SA_RefRACTIVEINDEX_nd
Direction: Up	
Valid values:	
Minimum: 1.0	
Maximum: 4.0	
Description: Refractive index of the active material at 587 nm.	
Parameter: double SA_RefRACTIVEINDEX_nC	SA_RefRACTIVEINDEX_nC
Direction: Up	
Valid values:	
Minimum: 1.0	
Maximum: 4.0	
Description: Refractive index of the active material at 656 nm.	
Parameter: double SA_AbbeNumber_vd	SA_AbbeNumber_vd
Direction: Up	
Valid values:	
Minimum: 10.0	
Maximum: 100.0	
Description: Abbe number of the active material or 0.0 if not specified.	
Parameter: int SA_Described_by	SA_Described_by
Direction: Up	
Valid values:	
0= nF, nd and nC	
1= nd and abbe value	
Description: Tells if nF, nd and nC or if nd and abbe value is valid.	
Parameter: int SA_UnlinearizedMode	SA_UnlinearizedMode
Direction: Up	
Valid values:	
0= off	
1= on	
Description: Unlinearized data mode.	
Parameter: double SA_DarkCorrThreshold	SA_DarkCorrThreshold
Direction: Up	
Valid values:	
Minimum: 1.0	
Maximum: 99.0	
Unit: %	
Description: Darkcorr threshold.	

11.2.1.1.7 Set_SyncMode (SYNC)

Set the synchronisation mode.

Parameter: int SP_SyncMode SP_SyncMode
Direction: Down
Valid values:
 0= None
 1= Master
 2= Slave
 3= Slave with external trigger (SLAVE_EXT)
Description: Synchronisation mode.

11.2.1.1.8 Get_SyncMode (SYNC)

Get the synchronisation mode.

Parameter: int SA_SyncMode SA_SyncMode
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= None
 1= Master
 2= Slave
 3= Slave with external trigger (SLAVE_EXT)
Description: Synchronisation mode.

11.2.1.1.9 Reset_Boot (RESET)

Resets the sensor.

11.2.1.2 User level

11.2.1.2.1 Logout (LOGOUT)

Change user level to user.

11.2.1.2.2 Login (LOGIN)

Change user level to professional.

Parameter: String SP_Password SP_Password
Direction: Down
Description: Valid password to login.

11.2.1.2.3 Get_UserLevel (GETUSERLEVEL)

Retrieve actual user level.

Parameter: int SA_UserLevel SA_UserLevel
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = User
 1 = Professional
Description: Actual user level.

11.2.1.2.4 Set_DefaultUser (STDUSER)

Set the default user level after booting the system.

Parameter: int SP_DefaultUser SP_DefaultUser
Direction: Down
Valid values:
 0 = User
 1 = Professional
Description: Default user level.

11.2.1.2.5 Get_DefaultUser (STDUSER)

Get the default user level after booting the system.

Parameter: int SA_DefaultUser SA_DefaultUser
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = User
 1 = Professional
Description: Default user level.

11.2.1.2.6 Set_Password (PASSWD)

Change the password for login.

Parameter: String SP_OldPassword SP_OldPassword
Direction: Down
Description: Old password.

Parameter: String SP_NewPassword SP_NewPassword
Direction: Down
Description: New password.

11.2.1.3 Sensor

11.2.1.3.1 Get_SensorTable (SENSORTABLE)

Get a list of all teached sensor heads.

Parameter: String SA_SensorTable	SA_SensorTable
Direction: Up	
Description: Whole table in one string, separated by new lines and commas.	
Parameter: int SA_SensorTableCount	SA_SensorTableCount
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 20	
Description: Number of entries in the table. All following parameters exists from 1 to this number, e.g. SA_Pos1, SA_Pos2, ...	
Parameter: int SA_Pos1..x	SA_Pos1..x
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 19	
Description: Index of the sensor head in the table.	
Parameter: String SA_Name1..x	SA_Name1..x
Direction: Up	
Description: Name of the sensor head in the table.	
Parameter: double SA_Range1..x	SA_Range1..x
Direction: Up	
Valid values:	
Minimum: 0.0	
Description: Measurement range of the sensor head in the table.	
Parameter: String SA_Unit1..x	SA_Unit1..x
Direction: Up	
Description: Unit of the measurement range.	
Parameter: String SA_Serial1..x	SA_Serial1..x
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the sensor head in the table.	

11.2.1.3.2 Set_ActiveSensor (SENSORHEAD)

Change to another sensor head.

Parameter: int SP_ActiveSensor	SP_ActiveSensor
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 19	
Description: Index of new sensor head.	

11.2.1.3.3 Get_ActiveSensor (SENSORHEAD)

Get active sensor head.

Parameter: int SA_ActiveSensor SA_ActiveSensor
Direction: Up
Valid values:
Minimum: 0
Maximum: 19
Description: Index of active sensor head.

11.2.1.3.4 Get_SensorInfo (SENSORINFO)

Get information of active sensor head.

Parameter: int SA_Pos SA_Pos
Direction: Up
Valid values:
Minimum: 0
Maximum: 19
Description: Index of the active sensor head in the table.

Parameter: String SA_Name SA_Name
Direction: Up
Description: Name of the active sensor head.

Parameter: double SA_Range SA_Range
Direction: Up
Valid values:
Minimum: 0.0
Description: Measurement range of the active sensor head.

Parameter: String SA_Unit SA_Unit
Direction: Up
Description: Unit of the measurement range.

Parameter: String SA_Serial SA_Serial
Direction: Up
Valid values:
 Numeric value
Description: Serial number of the active sensor head.

11.2.1.3.5 DarkCorr (DARKCORR)

Make a dark correction.

11.2.1.3.6 Set_DarkCorrThreshold (DARKCORRTHRES)

Set the darkcorr threshold at the controller.

Parameter: double SP_DarkCorrThreshold

SP_DarkCorrThreshold

Direction: Down

Valid values:

Minimum: 1.0

Maximum: 99.0

Unit: %

Description: Darkcorr threshold.

11.2.1.3.7 Get_DarkCorrThreshold (DARKCORRTHRES)

Get the darkcorr threshold from controller.

Parameter: double SA_DarkCorrThreshold

SA_DarkCorrThreshold

Direction: Up

Valid values:

Minimum: 1.0

Maximum: 99.0

Unit: %

Description: Darkcorr threshold.

11.2.1.3.8 LightCorr (LIGHTCORR)

Make a light correction.

Only available at IFD2471.

11.2.1.4 Triggering

11.2.1.4.1 Set_TriggerMode (TRIGGER)

Set the trigger mode.

Parameter: int SP_TriggerMode

SP_TriggerMode

Direction: Down

Valid values:

0= None

1= Edge

2= Level (PULSE)

3= Software

4= Encoder

Description: Trigger mode.

11.2.1.4.2 Get_TriggerMode (TRIGGER)

Get the active trigger mode.

Parameter: int SA_TriggerMode SA_TriggerMode
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= None
 1= Edge
 2= Level (PULSE)
 3= Software
 4= Encoder
Description: Trigger mode.

11.2.1.4.3 Set_TriggerMoment (TRIGGERAT)

Set the trigger time.

Parameter: int SP_TriggerMoment SP_TriggerMoment
Direction: Down
Valid values:
 0= Input
 1= Output
Description: Trigger moment.

11.2.1.4.4 Get_TriggerMoment (TRIGGERAT)

Get the active trigger time.

Parameter: int SA_TriggerMoment SA_TriggerMoment
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= Input
 1= Output
Description: Trigger moment.

11.2.1.4.5 Set_TriggerLevel (TRIGGERLEVEL)

Set the trigger level.

Parameter: int SP_TriggerLevel SP_TriggerLevel
Direction: Down
Valid values:
 0= High
 1= Low
Description: Trigger level.

11.2.1.4.6 Get_TriggerLevel (TRIGGERLEVEL)

Get the active trigger level.

Parameter: int SA_TriggerLevel

SA_TriggerLevel

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = High
- 1 = Low

Description: Trigger level.

11.2.1.4.7 Set_TriggerCount (TRIGGERCOUNT)

Set the number of values to measure at trigger.

Parameter: int SP_TriggerCount

SP_TriggerCount

Direction: Down

Valid values:

- Minimum:** 0
- Maximum:** 16383

Description: Number of values to measure. 0 means no trigger, 16383 means endless measurement.

11.2.1.4.8 Get_TriggerCount (TRIGGERCOUNT)

Get the number of values to measure at trigger.

Parameter: int SA_TriggerCount

SA_TriggerCount

Direction: Up

Valid values:

- Minimum:** 0
- Maximum:** 16383

Description: Number of values to measure. 0 means no trigger, 16383 means endless measurement.

11.2.1.4.9 Software_Trigger (TRIGGERSW)

Execute a software trigger.

11.2.1.4.10 Set_TriggerOnEncoder (TRIGGERENC)

Set the trigger on encoder mode.

Parameter: int SP_EncoderNumber

SP_EncoderNumber

Direction: Down

Valid values:

- Minimum:** 1
- Maximum:** 3

Description: Encoder number to trigger on.

Parameter: double SP_EncoderIncrements	SP_EncoderIncrements
Direction: Down	
Valid values:	
Minimum: 1.0	
Maximum: 2147483647.0 (INT_MAX)	
Description: Number of encoder increments before trigger. 0 means output data continuous. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.	
Parameter: double SP_EncoderMinValue	SP_EncoderMinValue
Direction: Down	
Valid values:	
Minimum: 0.0	
Maximum: 4294967295.0 (UINT_MAX)	
Description: Lower encoder limit. Above this value it will be triggered. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.	
Parameter: double SP_EncoderMaxValue	SP_EncoderMaxValue
Direction: Down	
Valid values:	
Minimum: 0.0	
Maximum: 4294967295.0 (UINT_MAX)	
Description: Upper encoder limit. Below this value it will be triggered. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.	

11.2.1.4.11 Get_TriggerOnEncoder (TRIGGERENC)

Get the trigger on encoder mode.

Parameter: int SA_EncoderNumber	SA_EncoderNumber
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 3	
Description: Encoder number to trigger on.	
Parameter: double SA_EncoderIncrements	SA_EncoderIncrements
Direction: Up	
Valid values:	
Minimum: 1.0	
Maximum: 2147483647.0 (INT_MAX)	
Description: Number of encoder increments before trigger. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.	
Parameter: double SA_EncoderMinValue	SA_EncoderMinValue
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 4294967295.0 (UINT_MAX)	
Description: Lower encoder limit. Above this value it will be triggered. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.	

Parameter: double SA_Encoder.MaxValue SA_Encoder.MaxValue

Direction: Up

Valid values:

- Minimum:** 0.0
- Maximum:** 4294967295.0 (UINT_MAX)

Description: Upper encoder limit. Below this value it will be triggered. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.

11.2.1.5 Encoder

11.2.1.5.1 Set_EncoderInterpolation1 (ENCINTERPOL1)

Set the interpolation for encoder 1.

Parameter: int SP_EncoderInterpolation1 SP_EncoderInterpolation1

Direction: Down

Valid values:

- 1
- 2
- 4

Description: Encoder interpolation.

11.2.1.5.2 Get_EncoderInterpolation1 (ENCINTERPOL1)

Get the interpolation of encoder 1.

Parameter: int SA_EncoderInterpolation1 SA_EncoderInterpolation1

Direction: Up

Valid values:

- 1
- 2
- 4

Description: Encoder interpolation.

11.2.1.5.3 Set_EncoderInterpolation2 (ENCINTERPOL2)

Set the interpolation for encoder 2.

Parameter: int SP_EncoderInterpolation2 SP_EncoderInterpolation2

Direction: Down

Valid values:

- 1
- 2
- 4

Description: Encoder interpolation.

11.2.1.5.4 Get_EncoderInterpolation2 (ENCINTERPOL2)

Get the interpolation of encoder 2.

Parameter: int SA_EncoderInterpolation2

SA_EncoderInterpolation2

Direction: Up

Valid values:

1

2

4

Description: Encoder interpolation.

11.2.1.5.5 Set_EncoderInterpolation3 (ENCINTERPOL3)

Set the interpolation for encoder 3.

Parameter: int SP_EncoderInterpolation3

SP_EncoderInterpolation3

Direction: Down

Valid values:

1

2

4

Description: Encoder interpolation.

11.2.1.5.6 Get_EncoderInterpolation3 (ENCINTERPOL3)

Get the interpolation of encoder 3.

Parameter: int SA_EncoderInterpolation3

SA_EncoderInterpolation3

Direction: Up

Valid values:

1

2

4

Description: Encoder interpolation.

11.2.1.5.7 Set_EncoderMode1 (ENCREF1)

Set the behaviour of encoder 1 when reference is reached.

Parameter: int SP_EncoderMode1

SP_EncoderMode1

Direction: Down

Valid values:

0= No action (NONE)

1= Set encoder value to preset value only one time (ONE)

2= Set encoder value to preset value each time (EVER)

Description: Mode of encoder when reference is reached.

11.2.1.5.8 Get_EncoderMode1 (ENCREF1)

Get the behaviour of encoder 1 when reference is reached.

Parameter: int SA_EncoderMode1

SA_EncoderMode1

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= No action (NONE)
- 1= Set encoder value to preset value only one time (ONE)
- 2= Set encoder value to preset value each time (EVER)

Description: Mode of encoder when reference is reached.

11.2.1.5.9 Set_EncoderMode2 (ENCREF2)

Set the behaviour of encoder 2 when reference is reached.

Parameter: int SP_EncoderMode2

SP_EncoderMode2

Direction: Down

Valid values:

- 0= No action (NONE)
- 1= Set encoder value to preset value only one time (ONE)
- 2= Set encoder value to preset value each time (EVER)

Description: Mode of encoder when reference is reached.

11.2.1.5.10 Get_EncoderMode2 (ENCREF2)

Get the behaviour of encoder 2 when reference is reached.

Parameter: int SA_EncoderMode2

SA_EncoderMode2

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= No action (NONE)
- 1= Set encoder value to preset value only one time (ONE)
- 2= Set encoder value to preset value each time (EVER)

Description: Mode of encoder when reference is reached.

11.2.1.5.11 Set_EncoderMode3 (ENCREF3)

Set the behaviour of encoder 3 when reference is reached.

Parameter: int SP_EncoderMode3

SP_EncoderMode3

Direction: Down

Valid values:

- 0= No action (NONE)
- 1= Set encoder value to preset value only one time (ONE)
- 2= Set encoder value to preset value each time (EVER)

Description: Mode of encoder when reference is reached.

11.2.1.5.12 Get_EncoderMode3 (ENCREF3)

Get the behaviour of encoder 3 when reference is reached.

Parameter: int SA_EncoderMode3

SA_EncoderMode3

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = No action (NONE)
- 1 = Set encoder value to preset value only one time (ONE)
- 2 = Set encoder value to preset value each time (EVER)

Description: Mode of encoder when reference is reached.

11.2.1.5.13 Set_EncoderPreload1 (ENCVALUE1)

Set preload value for encoder 1.

Parameter: double SP_EncoderPreload1

SP_EncoderPreload1

Direction: Down

Valid values:

- Minimum:** 0.0
- Maximum:** 4294967295.0 (UINT_MAX)

Description: Preload value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.

11.2.1.5.14 Get_EncoderPreload1 (ENCVALUE1)

Get preload value for encoder 1.

Parameter: double SA_EncoderPreload1

SA_EncoderPreload1

Direction: Up

Valid values:

- Minimum:** 0.0
- Maximum:** 4294967295.0 (UINT_MAX)

Description: Preload value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.

11.2.1.5.15 Set_EncoderPreload2 (ENCVALUE2)

Set preload value for encoder 2.

Parameter: double SP_EncoderPreload2

SP_EncoderPreload2

Direction: Down

Valid values:

- Minimum:** 0.0
- Maximum:** 4294967295.0 (UINT_MAX)

Description: Preload value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.

11.2.1.5.16 Get_EncoderPreload2 (ENCVALUE2)

Get preload value for encoder 2.

Parameter: double SA_EncoderPreload2

SA_EncoderPreload2

Direction: Up

Valid values:

Minimum: 0.0

Maximum: 4294967295.0 (UINT_MAX)

Description: Preload value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.

11.2.1.5.17 Set_EncoderPreload3 (ENCVALUE3)

Set preload value for encoder 3.

Parameter: double SP_EncoderPreload3

SP_EncoderPreload3

Direction: Down

Valid values:

Minimum: 0.0

Maximum: 4294967295.0 (UINT_MAX)

Description: Preload value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.

11.2.1.5.18 Get_EncoderPreload3 (ENCVALUE3)

Get preload value for encoder 3.

Parameter: double SA_EncoderPreload3

SA_EncoderPreload3

Direction: Up

Valid values:

Minimum: 0.0

Maximum: 4294967295.0 (UINT_MAX)

Description: Preload value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.

11.2.1.5.19 Load_Encoder1 (ENCSET)

Load the encoder 1 with the preset value.

11.2.1.5.20 Load_Encoder2 (ENCSET)

Load the encoder 2 with the preset value.

11.2.1.5.21 Load_Encoder3 (ENCSET)

Load the encoder 3 with the preset value.

11.2.1.5.22 EnableRef_Encoder1 (ENCRESET)

Reset reference counter of encoder 1. If encoder mode is 2, at next reference encoder value will be set to preset value again

11.2.1.5.23 EnableRef_Encoder2 (ENCRESET)

Reset reference counter of encoder 2. If encoder mode is 2, at next reference encoder value will be set to preset value again

11.2.1.5.24 EnableRef_Encoder3 (ENCRESET)

Reset reference counter of encoder 3. If encoder mode is 2, at next reference encoder value will be set to preset value again

11.2.1.5.25 Set_Encoder.MaxValue1 (ENCMAX1)

Set maximum value for encoder 1 before it wraps around.

Parameter: double SP_Encoder.MaxValue1

SP_Encoder.MaxValue1

Direction: Down

Valid values:

Minimum: 0.0

Maximum: 4294967295.0 (UINT_MAX)

Description: Maximum value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.

11.2.1.5.26 Get_Encoder.MaxValue1 (ENCMAX1)

Get maximum value for encoder 1 before it wraps around.

Parameter: double SA_Encoder.MaxValue1

SA_Encoder.MaxValue1

Direction: Up

Valid values:

Minimum: 0.0

Maximum: 4294967295.0 (UINT_MAX)

Description: Maximum value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.

11.2.1.5.27 Set_Encoder.MaxValue2 (ENCMAX2)

Set maximum value for encoder 2 before it wraps around.

Parameter: double SP_Encoder.MaxValue2

SP_Encoder.MaxValue2

Direction: Down

Valid values:

Minimum: 0.0

Maximum: 4294967295.0 (UINT_MAX)

Description: Maximum value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.

11.2.1.5.28 Get_Encoder.MaxValue2 (ENCMAX2)

Get maximum value for encoder 2 before it wraps around.

Parameter: double SA_Encoder.MaxValue2

SA_Encoder.MaxValue2

Direction: Up

Valid values:

Minimum: 0.0

Maximum: 4294967295.0 (UINT_MAX)

Description: Maximum value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.

11.2.1.5.29 Set_Encoder.MaxValue3 (ENCMAX3)

Set maximum value for encoder 3 before it wraps around.

Parameter: double SP_Encoder.MaxValue3

SP_Encoder.MaxValue3

Direction: Down

Valid values:

Minimum: 0.0

Maximum: 4294967295.0 (UINT_MAX)

Description: Maximum value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.

11.2.1.5.30 Get_Encoder.MaxValue3 (ENCMAX3)

Get maximum value for encoder 3 before it wraps around.

Parameter: double SA_Encoder.MaxValue3

SA_Encoder.MaxValue3

Direction: Up

Valid values:

Minimum: 0.0

Maximum: 4294967295.0 (UINT_MAX)

Description: Maximum value. This value is in whole numbers. But to avoid (signed) integer overflow, it's type is double.

11.2.1.6 Interfaces

11.2.1.6.1 Set_IPConfiguration (IPCONFIG)

Set the IP configuration at controller.

Parameter: int SP_DHCPEnabled

SP_DHCPEnabled

Direction: Down

Valid values:

0= FALSE

1= TRUE

Description: Specify if controller should use a static IP address or ask for IP at DHCP server (dynamic IP address).

Parameter: String SP_Address	SP_Address
Direction: Down	
Valid values:	Valid IP address in form of xxx.xxx.xxx.xxx
Description:	IP address of the controller. This parameter is only evaluated on static IP assignment.
Parameter: String SP_SubnetMask	SP_SubnetMask
Direction: Down	
Valid values:	Valid network mask (e.g. 255.255.255.0 for a Class C network)
Description:	Network mask of the controller. This parameter is only evaluated on static IP assignment.
Parameter: String SP_Gateway	SP_Gateway
Direction: Down	
Valid values:	Valid IP address of default gateway in form of xxx.xxx.xxx.xxx
Description:	The default gateway must be specified if the controller should communicate with peers in foreign subnets. This parameter is only evaluated on static IP assignment.

11.2.1.6.2 Get_IPConfiguration (IPCONFIG)

Get the IP configuration at controller.

Parameter: int SA_DHCPEnabled	SA_DHCPEnabled
Direction: Up	
Valid values:	-1 = Unknown parameter value from sensor 0 = FALSE 1 = TRUE
Description:	Get settings if controller should use a static IP address ask for IP at DHCP server (dynamic IP address).
Parameter: String SA_Address	SA_Address
Direction: Up	
Valid values:	Valid IP address in form of xxx.xxx.xxx.xxx
Description:	IP address of the controller. If DHCP is enabled it returns the currently assigned IP address.
Parameter: String SA_SubnetMask	SA_SubnetMask
Direction: Up	
Valid values:	Valid network mask (e.g. 255.255.255.0 for a Class C network)
Description:	Network mask of the controller. If DHCP is enabled it returns the currently assigned network mask.
Parameter: String SA_Gateway	SA_Gateway
Direction: Up	
Valid values:	Valid IP address of default gateway in form of xxx.xxx.xxx.xxx
Description:	Address of the default gateway. If DHCP is enabled it returns the currently assigned default gateway.

11.2.1.6.3 Set_IPDataTransferMode (MEATRANSFER)

Set IP protocol at controller.

Parameter: int SP_Protocol

SP_Protocol

Direction: Down

Valid values:

- 0= TCP server (SERVER/TCP)
- 1= TCP client (CLIENT/TCP)
- 2= UDP sender (CLIENT/UDP)
- 3= None

Description: Specifies if data should be send using TCP or UDP.

Parameter: String SP_RemoteAddress

SP_RemoteAddress

Direction: Down

Valid values:

Valid IP address of receiver of data

Description: Address of remote computer to send data to. On TCP server this parameter is ignored.

Parameter: int SP_Port

SP_Port

Direction: Down

Valid values:

- Minimum: 1024
- Maximum: 65535

Description: Port to send data to or to listen for incoming requests.

11.2.1.6.4 Get_IPDataTransferMode (MEATRANSFER)

Get IP protocol at controller.

Parameter: int SA_Protocol

SA_Protocol

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= TCP server (SERVER/TCP)
- 1= TCP client (CLIENT/TCP)
- 2= UDP sender (CLIENT/UDP)
- 3= None

Description: Specifies if data should be send using TCP or UDP.

Parameter: String SA_RemoteAddress

SA_RemoteAddress

Direction: Up

Valid values:

Valid IP address of receiver of data

Description: Address of remote computer to send data to.

Parameter: int SA_Port

SA_Port

Direction: Up

Valid values:

- Minimum: 1024
- Maximum: 65535

Description: Port to send data to or to listen for incoming requests.

11.2.1.6.5 Set_EthernetMode (ETHERMODE)

Switches ethernet mode between Ethernet and Ethercat.

Parameter: int SP_EthernetMode

SP_EthernetMode

Direction: Down

Valid values:

0= Ethernet

1= Ethercat

Description: Ethernet mode.

11.2.1.6.6 Get_EthernetMode (ETHERMODE)

Get ethernet mode of controller.

Parameter: int SA_EthernetMode

SA_EthernetMode

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= Ethernet

1= Ethercat

Description: Ethernet mode.

11.2.1.6.7 Set_Baudrate (BAUDRATE)

Set baudrate of controller for serial RS422 communication. After receiving the answer of the command, the baudrate of the local serial interface is changed, too.

Parameter: int SP_SensorBaudrate

SP_SensorBaudrate

Direction: Down

Valid values:

4000000

3500000

3000000

2500000

2000000

1500000

921600

691200

460800

230400

115200

9600

Unit: Baud

Description: Baudrate of controller.

11.2.1.6.8 Get_Baudrate (BAUDRATE)

Get baudrate of controller for serial RS422 communication.

Parameter: int SA_SensorBaudrate SA_SensorBaudrate
Direction: Up
Valid values:
 4000000
 3500000
 3000000
 2500000
 2000000
 1500000
 921600
 691200
 460800
 230400
 115200
 9600
Unit: Baud
Description: Baudrate of controller.

11.2.1.7 Parameter management

11.2.1.7.1 Save_Parameters (STORE)

Save actual parameters at controller. There can be saved several settings on different locations. So it is easy to switch to another setting.

Parameter: int SP_ParameterSet SP_ParameterSet
Direction: Down
Valid values:
Minimum: 1
Maximum: 8
Description: Location to save the settings.

11.2.1.7.2 Load_Parameters (READ)

Load actual parameters into controller RAM. There can be loaded several settings from different locations. So it is easy to switch to another setting.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_ParameterType SP_ParameterType
Direction: Down
Valid values:
 0= All settings (ALL)
 1= Device settings (DEVICE)
 2= Measurement settings (MEAS)
Description: Specifies which settings should be loaded.

Parameter: int SP_ParameterSet SP_ParameterSet
Direction: Down
Valid values:
Minimum: 1
Maximum: 8
Description: Location from where the settings should be loaded.

11.2.1.7.3 Set_Default (SETDEFAULT)

Reset the controller to default settings.
 If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_DefaultType SP_DefaultType
Direction: Down
Valid values:
 0= All settings (ALL)
 1= Keep device settings temporary (NODEVICE)
 2= Only material table (MATERIAL)
Description: Specifies which settings should be reset.

11.2.2 Measurement

11.2.2.1 General

11.2.2.1.1 Set_MeasureMode (MEASMODE)

Set the measure mode.
 If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_MeasureMode SP_MeasureMode
Direction: Down
Valid values:
 0= Distance
 1= Thickness
 2= Video
 3= MultiLayer (only IFD2451/IFD2461/IFD2471 with MultiLayer option)
 4= VideoStream
Description: Measure mode.

Parameter: int IP_TimerResolution IP_TimerResolution
Direction: Down
Unit: ms
Valid values:
 -1= Do not set timer resolution.
 0= Use greatest possible accuracy.
 1..2147483647 (INT_MAX)= Resolution in milliseconds.
Unit: ms
Default: 0 if measure mode is VideoStream, otherwise -1

Description: This parameter is necessary at video stream mode. Unless MEDAQLib waits for new video signals (using Sleep API function) and the default resolution for this function is 15 milli seconds there may be a time jitter at processing video signals (e.g. 15, 0, 0, 0 ..., 0, 15, 0, 0, ...). So this parameter changes the Windows timer resolution (for Windows scheduler, set by timeBeginPeriod). If it is not reseted manually when switching off video stream mode (set to -1), it is automatically reseted at CloseSensor.

11.2.2.1.2 Get_MeasureMode (MEASMODE)

Get the measure mode.

Parameter: int SA_MeasureMode

SA_MeasureMode

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = Distance
- 1 = Thickness
- 2 = Video
- 3 = MultiLayer (only IFD2451/IFD2461/IFD2471)
- 4 = VideoStream

Description: Measure mode.

11.2.2.1.3 Set_MeasurePeak (MEASPEAK)

Specify which peaks should be evaluated at controller.

Parameter: int SP_MeasurePeak

SP_MeasurePeak

Direction: Down

Valid values:

- 0 = First peak at distance mode resp. first and last peak at thickness mode (F_L)
- 1 = Last peak at distance mode resp. last and next to last peak at thickness mode (L_SL)
- 2 = First peak at distance mode resp. first and second peak at thickness mode (F_S)
- 3 = Highest peak at distance mode resp. highest and second highest peak at thickness mode (H_SH)

Description: Peaks to evaluate.

11.2.2.1.4 Get_MeasurePeak (MEASPEAK)

Retrieve which peaks are evaluated at controller.

Parameter: int SA_MeasurePeak

SA_MeasurePeak

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor

- 0= First peak at distance mode resp. first and last peak at thickness mode (F_L)
- 1= Last peak at distance mode resp. last and next to last peak at thickness mode (L_SL)
- 2= First peak at distance mode resp. first and second peak at thickness mode (F_S)
- 3= Highest peak at distance mode resp. highest and second highest peak at thickness mode (H_SH)

Description: Peaks to evalualte.

11.2.2.1.5 Set_ShutterMode (SHUTTERMODE)

Set the shutter mode.

Parameter: int SP_ShutterMode

SP_ShutterMode

Direction: Down

Valid values:

- 0= Search the best exposure time and measuring rate automatically (SEARCH)
- 1= Control exposure time, measuring rate is set manually (MEAS)
- 2= Exposure time and measuring rate are set manually (MANUAL)
- 3= Use two fixed exposure times alternately (2TIMEALT)
- 4= Use the suitable exposure time of two fixed ones (2TIMES)

Description: Shutter mode.

11.2.2.1.6 Get_ShutterMode (SHUTTERMODE)

Get the shutter mode.

Parameter: int SA_ShutterMode

SA_ShutterMode

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Search the best exposure time and measuring rate automatically (SEARCH)
- 1= Control exposure time, measuring rate is set manually (MEAS)
- 2= Exposure time and measuring rate are set manually (MANUAL)
- 3= Use two fixed exposure times alternately (2TIMEALT)
- 4= Use the suitable exposure time of two fixed ones (2TIMES)

Description: Shutter mode.

11.2.2.1.7 Set_Samplerate (MEASRATE)

Set the samplerate.

Parameter: double SP_Measrate

SP_Measrate

Direction: Down

Valid values:

- Minimum:** 0.1 for IFD2445/IFD2451/IFD2461, 0.3 for IFD2471
Maximum: 2.5 for IFD2445, 10.0 for IFD2451, 25.0 for IFD2461, 70.0 for IFD2471

Unit: kHz

Description: Samplerate of measurement. For older firmware versions (before V007.117.134.02), only discrete values are valid: 0.1, 0.2, 0.3, 1.0, 2.5, 5.0, 10.0, 25.0, 50.0, 70.0

11.2.2.1.8 Get_Samplerate (MEASRATE)

Get the samplerate.

Parameter: double SA_Measrate SA_Measrate
Direction: Up
Valid values:
Minimum: 0.1 for IFD2445/IFD2451/IFD2461, 0.3 for IFD2471
Maximum: 2.5 for IFD2445, 10.0 for IFD2451, 25.0 for IFD2461, 70.0 for IFD2471
Unit: kHz
Description: Samplerate of measurement. For older firmware versions (before V007.117.134.02), only discrete values are valid: 0.1, 0.2, 0.3, 1.0, 2.5, 5.0, 10.0, 25.0, 50.0, 70.0

11.2.2.1.9 Set_ShutterTime (SHUTTER)

Set the shutter time.

Parameter: double SP_ShutterTime1 SP_ShutterTime1
Direction: Down
Valid values:
Minimum: 0.075
Maximum: 10000.0 (IFD2445/IFD2451/IFD2461) or 3333.325 (IFD2471)
Unit: μ s
Description: First shutter time.

Parameter: double SP_ShutterTime2 SP_ShutterTime2
Direction: Down
Valid values:
Minimum: 0.075
Maximum: 10000.0 (IFD2445/IFD2451/IFD2461) or 3333.325 (IFD2471)
Unit: μ s
Description: Second shutter time. Is automatically adapted (to first shutter time), if not less or equal to first shutter time.

11.2.2.1.10 Get_ShutterTime (SHUTTER)

Get the shutter time.

Parameter: double SA_ShutterTime1 SA_ShutterTime1
Direction: Up
Valid values:
Minimum: 0.075
Maximum: 10000.0 (IFD2445/IFD2451/IFD2461) or 3333.325 (IFD2471)
Unit: μ s
Description: First shutter time.

Parameter: double SA_ShutterTime2 SA_ShutterTime2
Direction: Up
Valid values:
Minimum: 0.075
Maximum: 10000.0 (IFD2445/IFD2451/IFD2461) or 3333.325 (IFD2471)
Unit: μ s
Description: Second shutter time.

11.2.2.1.11 Get_Video (GETVIDEO)

Get recent video signals from sensor.

Parameter: Binary data SA_VideoRaw SA_VideoRaw
Direction: Up
Valid values:
 512 words (each 2 byte), each word is an intensity value.
Description: Raw video signal

Parameter: Binary data SA_VideoDark SA_VideoDark
Direction: Up
Valid values:
 512 words (each 2 byte), each word is an intensity value.
Description: Dark corrected video signal

Parameter: Binary data SA_VideoLight SA_VideoLight
Direction: Up
Valid values:
 512 words (each 2 byte), each word is an intensity value.
Description: Light corrected video signal

Parameter: Binary data SA_VideoDarkTable SA_VideoDarkTable
Direction: Up
Valid values:
 512 words (each 2 byte), each word is an intensity value.
Description: Dark table

Parameter: Binary data SA_VideoLightTable SA_VideoLightTable
Direction: Up
Valid values:
 512 words (each 2 byte), each word is an intensity value.
Description: Light table

Parameter: Binary data SA_VideoThreshold SA_VideoThreshold
Direction: Up
Valid values:
 512 words (each 2 byte), each word is an intensity value.
Description: Threshold table

Parameter: double SA_VideoTimestamp SA_VideoTimestamp
Direction: Up
Valid values:
Minimum: 0
Maximum: 1.79769e+308 (DBL_MAX)
Unit: ms
Description: Timestamp of the video signal. It starts from 1970 Jan 01 at 01:00. It is generated when the video has arrived at TCP/IP socket.

Example how to read a video signal from sensor:

```

/* Do not forget to handle potential error after each call to MEDAQLib! */
/* Create sensor instance, open sensor via TCP/IP, set output to ethernet */
/* and than switch to video mode: */
err= SetIntExecSCmd (instance, "Set_MeasureMode", "SP_MeasureMode", 2 /*Video*/);

/* Select the desired video signal: */
err= SetParameterInt (instance, "SP_OutputVideoRaw_ETH", 1);
err= SetParameterInt (instance, "SP_OutputVideoDark_ETH", 1);
err= SetParameterInt (instance, "SP_OutputVideoLight_ETH", 0);
err= SetParameterInt (instance, "SP_OutputVideoDarkTable_ETH", 0);
err= SetParameterInt (instance, "SP_OutputVideoLightTable_ETH", 0);
err= SetParameterInt (instance, "SP_OutputVideoThreshold_ETH", 0);
err= ExecSCmd (instance, "Set_OutputVideo_ETH");

/* Aquire video signals: */
err= ExecSCmd (instance, "Get_Video");
WORD videoRaw[512], videoDark[512];
DWORD maxLen= sizeof (videoRaw);
err= GetParameterBinary (instance, "SA_VideoRaw", (char *)videoRaw, &maxLen);
assert (maxlen==sizeof (videoRaw)); // additinal validity check
maxLen= sizeof (videoDark);
err= GetParameterBinary (instance, "SA_VideoDark", (char *)videoDark, &maxLen);
assert (maxlen==sizeof (videoDark)); // additinal validity check

/* Do anything with the received video signals */

```

11.2.2.1.12 Get_VideoStreamSignal

Read one video signal from video stream.

Parameter: int SP_ReadMode

SP_ReadMode

Direction: Down

Valid values:

- 0= Each video signal
- 1= Only newest video signal
- 2= Automatic

Description: This mode specifies if each video signal should be read or only the latest one. If set to automatic each video signal is read until the buffer does not overflow. If the buffer becomes full one or more video signals are discarded.

Parameter: int SP_WaitVideoTimeout

SP_WaitVideoTimeout

Direction: Down

Unit: ms

Valid values:

- Minimum:** 0
- Maximum:** 2147483647 (INT_MAX)

Description: Timeout to wait for a video signal.

Parameter: Binary data SA_VideoRaw

SA_VideoRaw

Direction: Up

Valid values:

- 512 words (each 2 byte), each word is an intensity value.

Description: Raw video signal

Parameter: Binary data SA_VideoDark	SA_VideoDark
Direction: Up	
Valid values:	512 words (each 2 byte), each word is an intensity value.
Description: Dark corrected video signal	
Parameter: Binary data SA_VideoLight	SA_VideoLight
Direction: Up	
Valid values:	512 words (each 2 byte), each word is an intensity value.
Description: Light corrected video signal	
Parameter: Binary data SA_VideoDarkTable	SA_VideoDarkTable
Direction: Up	
Valid values:	512 words (each 2 byte), each word is an intensity value.
Description: Dark table	
Parameter: Binary data SA_VideoLightTable	SA_VideoLightTable
Direction: Up	
Valid values:	512 words (each 2 byte), each word is an intensity value.
Description: Light table	
Parameter: Binary data SA_VideoThreshold	SA_VideoThreshold
Direction: Up	
Valid values:	512 words (each 2 byte), each word is an intensity value.
Description: Threshold table	
Parameter: double SA_VideoTimestamp	SA_VideoTimestamp
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 1.79769e+308 (DBL_MAX)	
Unit: ms	
Description: Timestamp of the video signal. It starts from 1970 Jan 01 at 01:00. It is generated when the video has arrived at TCP/IP socket.	
Parameter: int SA_SkippedVideo	SA_SkippedVideo
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 2147483647 (INT_MAX)	
Description: Number of skipped video signals, if SP_ReadMode is not 0.	

11.2.2.2 Video signal

11.2.2.2.1 Set_ROI (ROI)

Set the region of interest for processing video signal.

Parameter: int SP_ROIStart	SP_ROIStart
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 511	
Unit: Pixel	
Description: First position on CCD.	

Parameter: int SP_ROIEnd SP_ROIEnd
Direction: Down
Valid values:
Minimum: 0
Maximum: 511
Unit: Pixel
Description: Last position on CCD.

11.2.2.2.2 Get_ROI (ROI)

Get the region of interest for processing video signal.

Parameter: int SA_ROIStart SA_ROIStart
Direction: Up
Valid values:
Minimum: 0
Maximum: 511
Unit: Pixel
Description: First position on CCD.

Parameter: int SA_ROIEnd SA_ROIEnd
Direction: Up
Valid values:
Minimum: 0
Maximum: 511
Unit: Pixel
Description: Last position on CCD.

11.2.2.2.3 Set_VideoAverage (VSAVERAGE)

Set video averaging (before processing).

Parameter: int SP_VideoAverage SP_VideoAverage
Direction: Down
Valid values:
 0= None
 1= Recursive over 2 lines (REC2)
 2= Recursive over 4 lines (REC4)
 3= Recursive over 8 lines (REC8)
 4= Moving over 2 lines (MOV2)
 5= Moving over 3 lines (MOV3)
 6= Moving over 4 lines (MOV4)
 7= Median over 3 lines (MED3)
Description: Averaging mode.

11.2.2.2.4 Get_VideoAverage (VSAVERAGE)

Get video averaging (before processing).

Parameter: int SA_VideoAverage

SA_VideoAverage

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= Recursive over 2 lines (REC2)
- 2= Recursive over 4 lines (REC4)
- 3= Recursive over 8 lines (REC8)
- 4= Moving over 2 lines (MOV2)
- 5= Moving over 3 lines (MOV3)
- 6= Moving over 4 lines (MOV4)
- 7= Median over 3 lines (MED3)

Description: Averaging mode.

11.2.2.5 Set_Threshold (THRESHOLD)

Set threshold for video processing.

Parameter: double SP_Threshold

SP_Threshold

Direction: Down

Valid values:

- Minimum:** 0.0
- Maximum:** 99.0

Unit: %

Description: Video threshold.

11.2.2.6 Get_Threshold (THRESHOLD)

Get threshold for video processing.

Parameter: double SA_Threshold

SA_Threshold

Direction: Up

Valid values:

- Minimum:** 0.0
- Maximum:** 99.0

Unit: %

Description: Video threshold.

11.2.2.3 Material database

11.2.2.3.1 Get_MaterialTable (MATERIALTABLE)

Get a list of all materials for thickness calculation.

Parameter: String SA_MaterialTable

SA_MaterialTable

Direction: Up

Description: Whole table in one string, separated by new lines and commas.

Parameter: int SA_MaterialTableCount	SA_MaterialTableCount
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 20	
Description: Number of entries in the table. All following parameters exists from 1 to this number, e.g. SA_Pos1, SA_Pos2, ...	
Parameter: int SA_Pos1..x	SA_Pos1..x
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 19	
Description: Index of the material in the table.	
Parameter: String SA_MaterialName1..x	SA_MaterialName1..x
Direction: Up	
Description: Name of the material in the table.	
Parameter: double SA_RefractiveIndex_nF1..x	SA_RefractiveIndex_nF1..x
Direction: Up	
Valid values:	
Minimum: 1.0	
Maximum: 4.0	
Description: Refractive index of the material at 486 nm.	
Parameter: double SA_RefractiveIndex_nd1..x	SA_RefractiveIndex_nd1..x
Direction: Up	
Valid values:	
Minimum: 1.0	
Maximum: 4.0	
Description: Refractive index of the material at 587 nm.	
Parameter: double SA_RefractiveIndex_nC1..x	SA_RefractiveIndex_nC1..x
Direction: Up	
Valid values:	
Minimum: 1.0	
Maximum: 4.0	
Description: Refractive index of the material at 656 nm.	
Parameter: double SA_AbbeNumber_vd1..x	SA_AbbeNumber_vd1..x
Direction: Up	
Valid values:	
Minimum: 10.0	
Maximum: 100.0	
Description: Abbe number of the material or 0.0 if not specified.	
Parameter: String SA_Description1..x	SA_Description1..x
Direction: Up	
Description: Description of the material in the table.	

11.2.2.3.2 Set_ActiveMaterial (MATERIAL)

Set the active material for thickness calculation.

Parameter: String SP_ActiveMaterial

SP_ActiveMaterial

Direction: Down

Description: Name of material.

11.2.2.3.3 Get_ActiveMaterial (MATERIAL)

Get the active material for thickness calculation.

Parameter: String SA_ActiveMaterial

SA_ActiveMaterial

Direction: Up

Description: Name of material.

11.2.2.3.4 Get_MaterialInfo (MATERIALINFO)

Get information of active material.

Parameter: int SP_MaterialIndex

SP_MaterialIndex

Direction: Down

Valid values:

Minimum: 0

Maximum: 5

Default: Empty, means first material.

Description: Index of material to get info. If this parameter is 0 or not set, MATERIALINFO without parameter is called (for compatibility mode)

Parameter: String SA_MaterialName

SA_MaterialName

Direction: Up

Description: Name of the active material.

Parameter: String SA_Description

SA_Description

Direction: Up

Description: Description of the active material.

Parameter: double SA_RefRACTIVEINDEX_nF

SA_RefRACTIVEINDEX_nF

Direction: Up

Valid values:

Minimum: 1.0

Maximum: 4.0

Description: Refractive index of the active material at 486 nm.

Parameter: double SA_RefRACTIVEINDEX_nd

SA_RefRACTIVEINDEX_nd

Direction: Up

Valid values:

Minimum: 1.0

Maximum: 4.0

Description: Refractive index of the active material at 587 nm.

Parameter: double SA_RefractiveIndex_nC	SA_RefractiveIndex_nC
Direction: Up	
Valid values:	
Minimum: 1.0	
Maximum: 4.0	
Description: Refractive index of the active material at 656 nm.	
Parameter: double SA_AbbeNumber_vd	SA_AbbeNumber_vd
Direction: Up	
Valid values:	
Minimum: 10.0	
Maximum: 100.0	
Description: Abbe number of the active material or 0.0 if not specified.	
Parameter: int SA_Described_by	SA_Described_by
Direction: Up	
Valid values:	
0= nF, nd and nC	
1= nd and abbe value	
Description: Tells if nF, nd and nC or if nd and abbe value is valid.	

11.2.2.3.5 Edit_Material_Abbe (MATERIALEDIT)

Edit or add new material by using Abbe number.

Parameter: String SP_MaterialName	SP_MaterialName
Direction: Down	
Description: Name of the material.	
Parameter: String SP_Description	SP_Description
Direction: Down	
Description: Description of the material.	
Parameter: double SP_RefractiveIndex_nd	SP_RefractiveIndex_nd
Direction: Down	
Valid values:	
Minimum: 1.0	
Maximum: 4.0	
Description: Refractive index of the material at 587 nm.	
Parameter: double SP_AbbeNumber_vd	SP_AbbeNumber_vd
Direction: Down	
Valid values:	
Minimum: 10.0	
Maximum: 100.0	
Description: Abbe number of the material.	

11.2.2.3.6 Edit_Material_Nx (MATERIALEDIT)

Edit or add new material by using three refractive indizes.

Parameter: String SP_MaterialName	SP_MaterialName
Direction: Down	
Description: Name of the material.	

Parameter: String SP_Description	SP_Description
Direction: Down	
Description: Description of the material.	
Parameter: double SP_RefRACTIVEINDEX_nF	SP_RefRACTIVEINDEX_nF
Direction: Down	
Valid values:	
Minimum: 1.0	
Maximum: 4.0	
Description: Refractive index of the material at 486 nm.	
Parameter: double SP_RefRACTIVEINDEX_nd	SP_RefRACTIVEINDEX_nd
Direction: Down	
Valid values:	
Minimum: 1.0	
Maximum: 4.0	
Description: Refractive index of the material at 587 nm.	
Parameter: double SP_RefRACTIVEINDEX_nC	SP_RefRACTIVEINDEX_nC
Direction: Down	
Valid values:	
Minimum: 1.0	
Maximum: 4.0	
Description: Refractive index of the material at 656 nm.	

11.2.2.3.7 Delete_Material (MATERIALDELETE)

Deletes an existing material.

Parameter: String SP_MaterialName	SP_MaterialName
Direction: Down	
Description: Name of the material to delete.	

11.2.2.3.8 Clear_MaterialTable

Clear the whole material table.

11.2.2.4 Peak processing

11.2.2.4.1 Set_RefRACTIVECORR (REFRACCORR)

Specify refractive correction and number of peaks to detect.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_RefRACTIVECORR	SP_RefRACTIVECORR
Direction: Down	
Valid values:	
0= Off	
1= On	
Description: Specify if refractive correction should be enabled.	

Parameter: int SP_NumberOfPeaks SP_NumberOfPeaks
Direction: Down
Valid values:
Minimum: 2
Maximum: 6
Description: Number of peaks to detect.

11.2.2.4.2 Get_RefractiveCorrection (REFRACCORR)

Tells if refractive correction is enabled and number of peaks to detect.

Parameter: int SA_RefractiveCorrection SA_RefractiveCorrection
Direction: Up
Valid values:
-1= Unknown parameter value from sensor
0= Off
1= On
Description: Tells if refractive correction is be enabled.

Parameter: int SA_NumberOfPeaks SA_NumberOfPeaks
Direction: Up
Valid values:
Minimum: 2
Maximum: 6
Description: Number of peaks to detect.

11.2.2.4.3 Set_MaterialMultiPeak (MATERIALMP)

Set up to five materials for multipeak calculation.
Only available for IFD2451/IFD2461/IFD2471 with MultiLayer option.

Parameter: String SP_MaterialMultiPeak12 SP_MaterialMultiPeak12
Direction: Down
Default: "" (empty string, means refractive index 1.0)
Description: Name of material between first and second peak.

Parameter: String SP_MaterialMultiPeak23 SP_MaterialMultiPeak23
Direction: Down
Default: "" (empty string, means refractive index 1.0)
Description: Name of material between second and third peak.

Parameter: String SP_MaterialMultiPeak34 SP_MaterialMultiPeak34
Direction: Down
Default: "" (empty string, means refractive index 1.0)
Description: Name of material between third and fourth peak.

Parameter: String SP_MaterialMultiPeak45 SP_MaterialMultiPeak45
Direction: Down
Default: "" (empty string, means refractive index 1.0)
Description: Name of material between fourth and fifth peak.

Parameter: String SP_MaterialMultiPeak56 SP_MaterialMultiPeak56
Direction: Down
Default: "" (empty string, means refractive index 1.0)
Description: Name of material between fifth and sixth peak.

11.2.2.4.4 Get_MaterialMultiPeak (MATERIALMP)

Get all material names for multipeak calculation.
 Only available for IFD2451/IFD2461/IFD2471 with MultiLayer option.

Parameter: String SA_MaterialMultiPeak12 SA_MaterialMultiPeak12
Direction: Up
Description: Name of material between first and second peak.

Parameter: String SA_MaterialMultiPeak23 SA_MaterialMultiPeak23
Direction: Up
Description: Name of material between second and third peak.

Parameter: String SA_MaterialMultiPeak34 SA_MaterialMultiPeak34
Direction: Up
Description: Name of material between third and fourth peak.

Parameter: String SA_MaterialMultiPeak45 SA_MaterialMultiPeak45
Direction: Up
Description: Name of material between fourth and fifth peak.

Parameter: String SA_MaterialMultiPeak56 SA_MaterialMultiPeak56
Direction: Up
Description: Name of material between fifth and sixth peak.

11.2.2.5 Measurement value processing

11.2.2.5.1 Set_Averaging (AVERAGE)

Set data averaging at controller.

Parameter: int SP_AveragingType SP_AveragingType
Direction: Down
Valid values:
 0= None
 1= Moving average (MOVING)
 2= Recursive averaging (RECURSIVE)
 3= Median
Description: Averaging type.

Parameter: int SP_MovingCount SP_MovingCount

Direction: Down

Valid values:

- 2
- 4
- 8
- 16
- 32
- 64
- 128
- 256
- 512
- 1024

Description: Number of value for the averaging window. This parameter is only used at moving average.

Parameter: int SP_RecursiveCount SP_RecursiveCount

Direction: Down

Valid values:

- Minimum: 2
- Maximum: 32768

Description: Number of values for recursive averaging. This parameter is only used at recursive average.

Parameter: int SP_MedianCount SP_MedianCount

Direction: Down

Valid values:

- 3
- 5
- 7
- 9

Description: Number of values to build median. This parameter is only used at median.

11.2.2.5.2 Get_Averaging (AVERAGE)

Get data averaging at controller.

Parameter: int SA_AveragingType SA_AveragingType

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = None
- 1 = Moving average (MOVING)
- 2 = Recursive averaging (RECURSIVE)
- 3 = Median

Description: Averaging type.

Parameter: int SA_MovingCount SA_MovingCount

Direction: Up

Valid values:

- 2
- 4
- 8
- 16
- 32
- 64
- 128
- 256
- 512
- 1024

Description: Number of value for the averaging window. This parameter is only available at moving average.

Parameter: int SA_RecursiveCount SA_RecursiveCount

Direction: Up

Valid values:

- Minimum:** 2
- Maximum:** 32768

Description: Number of values for recursive averaging. This parameter is only available at recursive average.

Parameter: int SA_MedianCount SA_MedianCount

Direction: Up

Valid values:

- 3
- 5
- 7
- 9

Description: Number of values to build median. This parameter is only available at median.

11.2.2.5.3 Set_SpikeCorrection (SPIKECORR)

Set spike correction at controller.

Parameter: int SP_SpikeCorrection SP_SpikeCorrection

Direction: Down

Valid values:

- 0= off
- 1= on

Description: Spike correction.

Parameter: int SP_NbrEvaluatedValues SP_NbrEvaluatedValues

Direction: Down

Valid values:

- Minimum:** 1
- Maximum:** 10

Description: Number of values to evaluate for spike correction.

Parameter: double SP_ToleranceRange	SP_ToleranceRange
Direction: Down	
Valid values:	
Minimum: 0.0	
Maximum: 100.0	
Description: Tolerance range for spike correction.	
Parameter: int SP_NbrCorrectedValues	SP_NbrCorrectedValues
Direction: Down	
Valid values:	
Minimum: 1	
Maximum: 100	
Description: Number of values to correct at spike correction.	
11.2.2.5.4 Get_SpikeCorrection (SPIKECORR)	
Get spike correction at controller.	
Parameter: int SA_SpikeCorrection	SA_SpikeCorrection
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = off	
1 = on	
Description: Spike correction.	
Parameter: int SA_NbrEvaluatedValues	SA_NbrEvaluatedValues
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 10	
Description: Number of values to evaluate for spike correction.	
Parameter: double SA_ToleranceRange	SA_ToleranceRange
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 100.0	
Description: Tolerance range for spike correction.	
Parameter: int SA_NbrCorrectedValues	SA_NbrCorrectedValues
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 100	
Description: Number of values to correct at spike correction.	

11.2.2.5.5 Set_StatisticSignal (STATISTICSIGNAL)

Set the measured value which is used for statistic calculation.
 Only available for IFD2451/IFD2461/IFD2471 with MultiLayer option.

Parameter: int SP_StatisticSignal

SP_StatisticSignal

Direction: Down

Valid values:

- 0= Distance 1
- 1= Distance 2
- 2= Distance 3 (only IFD2451/IFD2461/IFD2471)
- 3= Distance 4 (only IFD2451/IFD2461/IFD2471)
- 4= Distance 5 (only IFD2451/IFD2461/IFD2471)
- 5= Distance 6 (only IFD2451/IFD2461/IFD2471)
- 6= Thickness 12
- 7= Thickness 13 (only IFD2451/IFD2461/IFD2471)
- 8= Thickness 14 (only IFD2451/IFD2461/IFD2471)
- 9= Thickness 15 (only IFD2451/IFD2461/IFD2471)
- 10= Thickness 16 (only IFD2451/IFD2461/IFD2471)
- 11= Thickness 23 (only IFD2451/IFD2461/IFD2471)
- 12= Thickness 24 (only IFD2451/IFD2461/IFD2471)
- 13= Thickness 25 (only IFD2451/IFD2461/IFD2471)
- 14= Thickness 26 (only IFD2451/IFD2461/IFD2471)
- 15= Thickness 34 (only IFD2451/IFD2461/IFD2471)
- 16= Thickness 35 (only IFD2451/IFD2461/IFD2471)
- 17= Thickness 36 (only IFD2451/IFD2461/IFD2471)
- 18= Thickness 45 (only IFD2451/IFD2461/IFD2471)
- 19= Thickness 46 (only IFD2451/IFD2461/IFD2471)
- 20= Thickness 56 (only IFD2451/IFD2461/IFD2471)

Description: Value which is used for statistic calculation.

11.2.2.5.6 Get_StatisticSignal (STATISTICSIGNAL)

Get the measured value which is used for statistic calculation.
 Only available for IFD2451/IFD2461/IFD2471 with MultiLayer option.

Parameter: int SA_StatisticSignal

SA_StatisticSignal

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Distance 1
- 1= Distance 2
- 2= Distance 3 (only IFD2451/IFD2461/IFD2471)
- 3= Distance 4 (only IFD2451/IFD2461/IFD2471)
- 4= Distance 5 (only IFD2451/IFD2461/IFD2471)
- 5= Distance 6 (only IFD2451/IFD2461/IFD2471)
- 6= Thickness 12
- 7= Thickness 13 (only IFD2451/IFD2461/IFD2471)
- 8= Thickness 14 (only IFD2451/IFD2461/IFD2471)
- 9= Thickness 15 (only IFD2451/IFD2461/IFD2471)
- 10= Thickness 16 (only IFD2451/IFD2461/IFD2471)
- 11= Thickness 23 (only IFD2451/IFD2461/IFD2471)

12= Thickness 24 (only IFD2451/IFD2461/IFD2471)
 13= Thickness 25 (only IFD2451/IFD2461/IFD2471)
 14= Thickness 26 (only IFD2451/IFD2461/IFD2471)
 15= Thickness 34 (only IFD2451/IFD2461/IFD2471)
 16= Thickness 35 (only IFD2451/IFD2461/IFD2471)
 17= Thickness 36 (only IFD2451/IFD2461/IFD2471)
 18= Thickness 45 (only IFD2451/IFD2461/IFD2471)
 19= Thickness 46 (only IFD2451/IFD2461/IFD2471)
 20= Thickness 56 (only IFD2451/IFD2461/IFD2471)

Description: Value which is used for statistic calculation.

11.2.2.5.7 Set_StatisticDepth (STATISTICDEPTH)

Set the window size for floating statistic calculation.

Parameter: int SP_StatisticDepth SP_StatisticDepth
Direction: Down
Valid values:
 Minimum: 2
 Maximum: 2147483647 (INT_MAX)
Description: Window size for floating statistic calculation. The value must be power of two (2, 4, 8, ..., 16384). Value greater as 16384 calculates statistic over all values.

11.2.2.5.8 Get_StatisticDepth (STATISTICDEPTH)

Get the window size for floating statistic calculation.

Parameter: int SA_StatisticDepth SA_StatisticDepth
Direction: Up
Valid values:
 Minimum: 2
 Maximum: 2147483647 (INT_MAX)
Description: Window size for floating statistic calculation. The value must be power of two (2, 4, 8, ..., 16384). Value greater as 16384 calculates statistic over all values.

11.2.2.5.9 Reset_Statistic (RESETSTATISTIC)

Reset the statistic (min, max and peak to peak values).

11.2.2.5.10 Set_MasterSignal (MASTERSIGNAL)

Set the measured value which is used for mastering.
Only available for IFD2451/IFD2461/IFD2471 with MultiLayer option.

Parameter: int SP_MasterSignal SP_MasterSignal
Direction: Down
Valid values:

0= Distance 1
 1= Distance 2
 2= Distance 3 (only IFD2451/IFD2461/IFD2471)
 3= Distance 4 (only IFD2451/IFD2461/IFD2471)
 4= Distance 5 (only IFD2451/IFD2461/IFD2471)
 5= Distance 6 (only IFD2451/IFD2461/IFD2471)
 6= Thickness 12
 7= Thickness 13 (only IFD2451/IFD2461/IFD2471)
 8= Thickness 14 (only IFD2451/IFD2461/IFD2471)
 9= Thickness 15 (only IFD2451/IFD2461/IFD2471)
 10= Thickness 16 (only IFD2451/IFD2461/IFD2471)
 11= Thickness 23 (only IFD2451/IFD2461/IFD2471)
 12= Thickness 24 (only IFD2451/IFD2461/IFD2471)
 13= Thickness 25 (only IFD2451/IFD2461/IFD2471)
 14= Thickness 26 (only IFD2451/IFD2461/IFD2471)
 15= Thickness 34 (only IFD2451/IFD2461/IFD2471)
 16= Thickness 35 (only IFD2451/IFD2461/IFD2471)
 17= Thickness 36 (only IFD2451/IFD2461/IFD2471)
 18= Thickness 45 (only IFD2451/IFD2461/IFD2471)
 19= Thickness 46 (only IFD2451/IFD2461/IFD2471)
 20= Thickness 56 (only IFD2451/IFD2461/IFD2471)

Description: Value which is used for mastering.

11.2.2.5.11 Get_MasterSignal (MASTERSIGNAL)

Get the measured value which is used for mastering.

Only available for IFD2451/IFD2461/IFD2471 with MultiLayer option.

Parameter: int SA_MasterSignal

SA_MasterSignal

Direction: Up

Valid values:

-1= Unknown parameter value from sensor
 0= Distance 1
 1= Distance 2
 2= Distance 3 (only IFD2451/IFD2461/IFD2471)
 3= Distance 4 (only IFD2451/IFD2461/IFD2471)
 4= Distance 5 (only IFD2451/IFD2461/IFD2471)
 5= Distance 6 (only IFD2451/IFD2461/IFD2471)
 6= Thickness 12
 7= Thickness 13 (only IFD2451/IFD2461/IFD2471)
 8= Thickness 14 (only IFD2451/IFD2461/IFD2471)
 9= Thickness 15 (only IFD2451/IFD2461/IFD2471)
 10= Thickness 16 (only IFD2451/IFD2461/IFD2471)
 11= Thickness 23 (only IFD2451/IFD2461/IFD2471)
 12= Thickness 24 (only IFD2451/IFD2461/IFD2471)
 13= Thickness 25 (only IFD2451/IFD2461/IFD2471)
 14= Thickness 26 (only IFD2451/IFD2461/IFD2471)
 15= Thickness 34 (only IFD2451/IFD2461/IFD2471)
 16= Thickness 35 (only IFD2451/IFD2461/IFD2471)
 17= Thickness 36 (only IFD2451/IFD2461/IFD2471)
 18= Thickness 45 (only IFD2451/IFD2461/IFD2471)
 19= Thickness 46 (only IFD2451/IFD2461/IFD2471)
 20= Thickness 56 (only IFD2451/IFD2461/IFD2471)

Description: Value which is used for mastering.

11.2.2.5.12 Set_MasterValue (MASTERMV)

Set the master value.

Parameter: int SP_Master

SP_Master

Direction: Down

Valid values:

0= no (NONE)

1= yes (MASTER=

Description: Specifies if mastering should be done or resetted.

Parameter: double SP_MasterValue

SP_MasterValue

Direction: Down

Valid values:

Minimum: -2* measuring range

Maximum: +2* measuring range

Unit: mm

Description: Master value

11.2.2.5.13 Get_MasterValue (MASTERMV)

Get the master value.

Parameter: int SA_Master

SA_Master

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= no (NONE)

1= yes (MASTER=

Description: Specifies if mastering is active.

Parameter: double SA_MasterValue

SA_MasterValue

Direction: Up

Valid values:

Minimum: -2* measuring range

Maximum: +2* measuring range

Unit: mm

Description: Master value

11.2.3 Data output

11.2.3.1 General

11.2.3.1.1 Set_DataOutInterface (OUTPUT)

Set the active interface for data output.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_DataOutInterface

SP_DataOutInterface

Direction: Down

Valid values:

0= None

1= RS422

2= Ethernet

3= HTTP

4= Ethercat

Description: Active interface for data output.

11.2.3.1.2 Get_DataOutInterface (OUTPUT)

Get the active interface for data output.

Parameter: int SA_DataOutInterface

SA_DataOutInterface

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = None
- 1 = RS422
- 2 = Ethernet
- 3 = HTTP
- 4 = Ethercat

Description: Active interface for data output.

11.2.3.1.3 Set_Resampling (OUTREDUCE)

Set resampling to reduce output data.

Parameter: int SP_Resampling

SP_Resampling

Direction: Down

Valid values:

- Minimum:** 1
- Maximum:** 4200000

Description: Resampling value.

Parameter: int SP_ResampleAnalog

SP_ResampleAnalog

Direction: Down

Valid values:

- 0 = no
- 1 = yes

Description: Specify if analog output should be resampled.

Parameter: int SP_ResampleRS422

SP_ResampleRS422

Direction: Down

Valid values:

- 0 = no
- 1 = yes

Description: Specify if RS422 output should be resampled.

Parameter: int SP_ResampleEthernet

SP_ResampleEthernet

Direction: Down

Valid values:

- 0 = no
- 1 = yes

Description: Specify if output over ethernet should be resampled.

11.2.3.1.4 Get_Resampling (OUTREDUCE)

Get resampling for reducing output data.

Parameter: int SA_Resampling

SA_Resampling

Direction: Up

Valid values:

- Minimum:** 1
- Maximum:** 4200000

Description: Resampling value.

Parameter: int SA_ResampleAnalog	SA_ResampleAnalog
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Analog output is resampled.	
Parameter: int SA_ResampleRS422	SA_ResampleRS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: RS422 output is resampled.	
Parameter: int SA_ResampleEthernet	SA_ResampleEthernet
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Output over ethernet is resampled.	

11.2.3.1.5 Set_HoldLastValid (OUTHOLD)

Set the number of values to be replaced by last valid value instead of error values.

Parameter: int SP_HoldLastValid	SP_HoldLastValid
Direction: Down	
Valid values:	
Minimum: -1	
Maximum: 1024	
Description: Values to replace by last valid value. -1 means no value to hold, 0 means never output an error value (always hold last valid value).	

11.2.3.1.6 Get_HoldLastValid (OUTHOLD)

Get the number of values to be replaced by last valid value instead of error values.

Parameter: int SA_HoldLastValid	SA_HoldLastValid
Direction: Up	
Valid values:	
Minimum: -1	
Maximum: 1024	
Description: Values to replace by last valid value. -1 means no value to hold, 0 means never output an error value (always hold last valid value).	

11.2.3.1.7 Set_FramesPerPacket_ETH (MEASCNT_ETH)

Set the maximum number of frames in ethernet packet.

Parameter: int SP_FramesPerPacket_ETH	SP_FramesPerPacket_ETH
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 1	
Description: Maximum number of frames in ethernet packet. 0 means automatic.	

11.2.3.1.8 Get_FramesPerPacket_ETH (MEASCNT_ETH)

Get the maximum number of frames in ethernet packet.

Parameter: int SA_FramesPerPacket_ETH

SA_FramesPerPacket_ETH

Direction: Up

Valid values:

Minimum: 0

Maximum: 1

Description: Maximum number of frames in ethernet packet. 0 means automatic.

11.2.3.2 Selected measurement values

11.2.3.2.1 Set_OutputDistance_RS422 (OUTDIST_RS422)

Set the distance data to be output at RS422 interface.

Parameter: int SP_OutputDistance1_RS422

SP_OutputDistance1_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if distance 1 is transmitted.

Parameter: int SP_OutputDistance2_RS422

SP_OutputDistance2_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if distance 2 is transmitted (only available at measure mode thickness, multipeak or video and not for IFD2445).

Parameter: int SP_OutputDistance3_RS422

SP_OutputDistance3_RS422

Direction: Down

Valid values:

0= no

1= yes

Valid for sensor:

IFD2451

IFD2461

IFD2471

Description: Specify if distance 3 is transmitted (only available at measure mode multipeak or video and not for IFD2445).

Parameter: int SP_OutputDistance4_RS422

SP_OutputDistance4_RS422

Direction: Down

Valid values:

0= no

1= yes

Valid for sensor:

IFD2451

IFD2461

IFD2471

Description: Specify if distance 4 is transmitted (only available at measure mode multipeak or video and not for IFD2445).

Parameter: int SP_OutputDistance5_RS422 SP_OutputDistance5_RS422

Direction: Down

Valid values:

0= no
1= yes

Valid for sensor:

IFD2451
IFD2461
IFD2471

Description: Specify if distance 5 is transmitted (only available at measure mode multipeak or video and not for IFD2445).

Parameter: int SP_OutputDistance6_RS422 SP_OutputDistance6_RS422

Direction: Down

Valid values:

0= no
1= yes

Valid for sensor:

IFD2451
IFD2461
IFD2471

Description: Specify if distance 6 is transmitted (only available at measure mode multipeak or video and not for IFD2445).

11.2.3.2.2 Get_OutputDistance_RS422 (OUTDIST_RS422)

Get the distance data to be output at RS422 interface.

Parameter: int SA_OutputDistance1_RS422 SA_OutputDistance1_RS422

Direction: Up

Valid values:

0= no
1= yes

Description: Specify if distance 1 is transmitted.

Parameter: int SA_OutputDistance2_RS422 SA_OutputDistance2_RS422

Direction: Up

Valid values:

0= no
1= yes

Description: Specify if distance 2 is transmitted (not available for IFD2445).

Parameter: int SA_OutputDistance3_RS422 SA_OutputDistance3_RS422

Direction: Up

Valid values:

0= no
1= yes

Valid for sensor:

IFD2451
IFD2461
IFD2471

Description: Specify if distance 3 is transmitted (not available for IFD2445).

Parameter: int SA_OutputDistance4_RS422	SA_OutputDistance4_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description:	Specify if distance 4 is transmitted (not available for IFD2445).
Parameter: int SA_OutputDistance5_RS422	SA_OutputDistance5_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description:	Specify if distance 5 is transmitted (not available for IFD2445).
Parameter: int SA_OutputDistance6_RS422	SA_OutputDistance6_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description:	Specify if distance 6 is transmitted (not available for IFD2445).

11.2.3.2.3 Set_OutputDistance_ETH (OUTDIST_ETH)

This command has only effect on measure mode video or multipeak. Set the distance data to be output at ethernet interface.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_OutputDistance1_ETH	SP_OutputDistance1_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description:	Specify if distance 1 is transmitted.
Parameter: int SP_OutputDistance2_ETH	SP_OutputDistance2_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description:	Specify if distance 2 is transmitted (only available at measure mode thickness, multipeak or video and not for IFD2445).

Parameter: int SP_OutputDistance3_ETH	SP_OutputDistance3_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if distance 3 is transmitted (only available at measure mode multipeak or video and not for IFD2445).	
Parameter: int SP_OutputDistance4_ETH	SP_OutputDistance4_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if distance 4 is transmitted (only available at measure mode multipeak or video and not for IFD2445).	
Parameter: int SP_OutputDistance5_ETH	SP_OutputDistance5_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if distance 5 is transmitted (only available at measure mode multipeak or video and not for IFD2445).	
Parameter: int SP_OutputDistance6_ETH	SP_OutputDistance6_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if distance 6 is transmitted (only available at measure mode multipeak or video and not for IFD2445).	

11.2.3.2.4 Get_OutputDistance_ETH (OUTDIST_ETH)

This command has only effect on measure mode video or multipeak. Get the distance data to be output at ethernet interface.

Parameter: int SA_OutputDistance1_ETH	SA_OutputDistance1_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if distance 1 is transmitted.	
Parameter: int SA_OutputDistance2_ETH	SA_OutputDistance2_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if distance 2 is transmitted (not available for IFD2445).	
Parameter: int SA_OutputDistance3_ETH	SA_OutputDistance3_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if distance 3 is transmitted (not available for IFD2445).	
Parameter: int SA_OutputDistance4_ETH	SA_OutputDistance4_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if distance 4 is transmitted (not available for IFD2445).	
Parameter: int SA_OutputDistance5_ETH	SA_OutputDistance5_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if distance 5 is transmitted (not available for IFD2445).	
Parameter: int SA_OutputDistance6_ETH	SA_OutputDistance6_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if distance 6 is transmitted (not available for IFD2445).	

11.2.3.2.5 Set_OutputThickness_RS422 (OUTTHICK_RS422)

Set the thickness data to be output at RS422 interface.

Parameter: int SP_OutputThickness12_RS422

SP_OutputThickness12_-
RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if thickness between first and second peak is transmitted
(only available at measure mode thickness, multipeak or video).

Parameter: int SP_OutputThickness13_RS422

SP_OutputThickness13_-
RS422

Direction: Down

Valid values:

0= no

1= yes

Valid for sensor:

IFD2451

IFD2461

IFD2471

Description: Specify if thickness between first and third peak is transmitted
(only available at measure mode multipeak or video).

Parameter: int SP_OutputThickness14_RS422

SP_OutputThickness14_-
RS422

Direction: Down

Valid values:

0= no

1= yes

Valid for sensor:

IFD2451

IFD2461

IFD2471

Description: Specify if thickness between first and fourth peak is transmitted
(only available at measure mode multipeak or video).

Parameter: int SP_OutputThickness15_RS422

SP_OutputThickness15_-
RS422

Direction: Down

Valid values:

0= no

1= yes

Valid for sensor:

IFD2451

IFD2461

IFD2471

Description: Specify if thickness between first and fifth peak is transmitted
(only available at measure mode multipeak or video).

Parameter: int SP_OutputThickness16_RS422	SP_OutputThickness16_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between first and sixth peak is transmitted (only available at measure mode multipeak or video).	
Parameter: int SP_OutputThickness23_RS422	SP_OutputThickness23_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and third peak is transmitted (only available at measure mode multipeak or video).	
Parameter: int SP_OutputThickness24_RS422	SP_OutputThickness24_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and fourth peak is transmitted (only available at measure mode multipeak or video).	
Parameter: int SP_OutputThickness25_RS422	SP_OutputThickness25_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and fifth peak is transmitted (only available at measure mode multipeak or video).	
Parameter: int SP_OutputThickness26_RS422	SP_OutputThickness26_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	

IFD2451
IFD2461
IFD2471

Description: Specify if thickness between second and sixth peak is transmitted (only available at measure mode multipeak or video).

Parameter: int SP_OutputThickness34_RS422

SP_OutputThickness34_-
RS422

Direction: Down

Valid values:

0 = no
1 = yes

Valid for sensor:

IFD2451
IFD2461
IFD2471

Description: Specify if thickness between third and fourth peak is transmitted (only available at measure mode multipeak or video).

Parameter: int SP_OutputThickness35_RS422

SP_OutputThickness35_-
RS422

Direction: Down

Valid values:

0 = no
1 = yes

Valid for sensor:

IFD2451
IFD2461
IFD2471

Description: Specify if thickness between third and fifth peak is transmitted (only available at measure mode multipeak or video).

Parameter: int SP_OutputThickness36_RS422

SP_OutputThickness36_-
RS422

Direction: Down

Valid values:

0 = no
1 = yes

Valid for sensor:

IFD2451
IFD2461
IFD2471

Description: Specify if thickness between third and sixth peak is transmitted (only available at measure mode multipeak or video).

Parameter: int SP_OutputThickness45_RS422

SP_OutputThickness45_-
RS422

Direction: Down

Valid values:

0 = no
1 = yes

Valid for sensor:

IFD2451
IFD2461
IFD2471

Description: Specify if thickness between fourth and fifth peak is transmitted (only available at measure mode multipeak or video).

Parameter: int SP_OutputThickness46_RS422	SP_OutputThickness46_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between fourth and sixth peak is transmitted (only available at measure mode multipeak or video).	
Parameter: int SP_OutputThickness56_RS422	SP_OutputThickness56_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between fifth and sixth peak is transmitted (only available at measure mode multipeak or video).	

11.2.3.2.6 Get_OutputThickness_RS422 (OUTTHICK_RS422)

Get the thickness data to be output at RS422 interface.

Parameter: int SA_OutputThickness12_RS422	SA_OutputThickness12_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if thickness between first and second peak is transmitted.	
Parameter: int SA_OutputThickness13_RS422	SA_OutputThickness13_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between first and third peak is transmitted.	
Parameter: int SA_OutputThickness14_RS422	SA_OutputThickness14_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between first and fourth peak is transmitted.	

Parameter: int SA_OutputThickness15_RS422	SA_OutputThickness15_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between first and fifth peak is transmitted.	
Parameter: int SA_OutputThickness16_RS422	SA_OutputThickness16_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between first and sixth peak is transmitted.	
Parameter: int SA_OutputThickness23_RS422	SA_OutputThickness23_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and third peak is transmitted.	
Parameter: int SA_OutputThickness24_RS422	SA_OutputThickness24_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and fourth peak is transmitted.	
Parameter: int SA_OutputThickness25_RS422	SA_OutputThickness25_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and fifth peak is transmitted.	

Parameter: int SA_OutputThickness26_RS422	SA_OutputThickness26_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and sixth peak is transmitted.	
Parameter: int SA_OutputThickness34_RS422	SA_OutputThickness34_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between third and fourth peak is transmitted.	
Parameter: int SA_OutputThickness35_RS422	SA_OutputThickness35_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between third and fifth peak is transmitted.	
Parameter: int SA_OutputThickness36_RS422	SA_OutputThickness36_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between third and sixth peak is transmitted.	
Parameter: int SA_OutputThickness45_RS422	SA_OutputThickness45_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between fourth and fifth peak is transmitted.	

Parameter: int SA_OutputThickness46_RS422	SA_OutputThickness46_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between fourth and sixth peak is transmitted.	
Parameter: int SA_OutputThickness56_RS422	SA_OutputThickness56_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between fifth and sixth peak is transmitted.	

11.2.3.2.7 Set_OutputThickness_ETH (OUTTHICK_ETH)

This command has only effect on measure mode multipack or video. Set the thickness data to be output at ethernet interface.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_OutputThickness12_ETH	SP_OutputThickness12_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if thickness between first and second peak is transmitted.	
Parameter: int SP_OutputThickness13_ETH	SP_OutputThickness13_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between first and third peak is transmitted.	
Parameter: int SP_OutputThickness14_ETH	SP_OutputThickness14_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between first and fourth peak is transmitted.	

Parameter: int SP_OutputThickness15_ETH	SP_OutputThickness15_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between first and fifth peak is transmitted.	
Parameter: int SP_OutputThickness16_ETH	SP_OutputThickness16_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between first and sixth peak is transmitted.	
Parameter: int SP_OutputThickness23_ETH	SP_OutputThickness23_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and third peak is transmitted.	
Parameter: int SP_OutputThickness24_ETH	SP_OutputThickness24_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and fourth peak is transmitted.	
Parameter: int SP_OutputThickness25_ETH	SP_OutputThickness25_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and fifth peak is transmitted.	

Parameter: int SP_OutputThickness26_ETH	SP_OutputThickness26_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and sixth peak is transmitted.	
Parameter: int SP_OutputThickness34_ETH	SP_OutputThickness34_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between third and fourth peak is transmitted.	
Parameter: int SP_OutputThickness35_ETH	SP_OutputThickness35_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between third and fifth peak is transmitted.	
Parameter: int SP_OutputThickness36_ETH	SP_OutputThickness36_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between third and sixth peak is transmitted.	
Parameter: int SP_OutputThickness45_ETH	SP_OutputThickness45_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between fourth and fifth peak is transmitted.	

Parameter: int SP_OutputThickness46_ETH	SP_OutputThickness46_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description:	Specify if thickness between fourth and sixth peak is transmitted.
Parameter: int SP_OutputThickness56_ETH	SP_OutputThickness56_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description:	Specify if thickness between fifth and sixth peak is transmitted.

11.2.3.2.8 Get_OutputThickness_ETH (OUTTHICK_ETH)

This command has only effect on measure mode video. Get the thickness data to be output at ethernet interface.

Parameter: int SA_OutputThickness12_ETH	SA_OutputThickness12_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description:	Specify if thickness between first and second peak is transmitted.
Parameter: int SA_OutputThickness13_ETH	SA_OutputThickness13_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description:	Specify if thickness between first and third peak is transmitted.
Parameter: int SA_OutputThickness14_ETH	SA_OutputThickness14_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description:	Specify if thickness between first and fourth peak is transmitted.

Parameter: int SA_OutputThickness15_ETH	SA_OutputThickness15_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between first and fifth peak is transmitted.	
Parameter: int SA_OutputThickness16_ETH	SA_OutputThickness16_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between first and sixth peak is transmitted.	
Parameter: int SA_OutputThickness23_ETH	SA_OutputThickness23_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and third peak is transmitted.	
Parameter: int SA_OutputThickness24_ETH	SA_OutputThickness24_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and fourth peak is transmitted.	
Parameter: int SA_OutputThickness25_ETH	SA_OutputThickness25_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and fifth peak is transmitted.	

Parameter: int SA_OutputThickness26_ETH	SA_OutputThickness26_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between second and sixth peak is transmitted.	
Parameter: int SA_OutputThickness34_ETH	SA_OutputThickness34_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between third and fourth peak is transmitted.	
Parameter: int SA_OutputThickness35_ETH	SA_OutputThickness35_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between third and fifth peak is transmitted.	
Parameter: int SA_OutputThickness36_ETH	SA_OutputThickness36_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between third and sixth peak is transmitted.	
Parameter: int SA_OutputThickness45_ETH	SA_OutputThickness45_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between fourth and fifth peak is transmitted.	

Parameter: int SA_OutputThickness46_ETH	SA_OutputThickness46_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between fourth and sixth peak is transmitted.	
Parameter: int SA_OutputThickness56_ETH	SA_OutputThickness56_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Valid for sensor:	
IFD2451	
IFD2461	
IFD2471	
Description: Specify if thickness between fifth and sixth peak is transmitted.	

11.2.3.2.9 Set_OutputStatistic_RS422 (OUTSTATISTIC_RS422)

Set the statistic data to be output at RS422 interface.

Parameter: int SP_OutputStatisticMin_RS422	SP_OutputStatisticMin_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if min value is transmitted.	
Parameter: int SP_OutputStatisticMax_RS422	SP_OutputStatisticMax_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if max value is transmitted.	
Parameter: int SP_OutputStatisticPeak2Peak_RS422	SP_OutputStatisticPeak2Peak_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if peak to peak value is transmitted.	

11.2.3.2.10 Get_OutputStatistic_RS422 (OUTSTATISTIC_RS422)

Get the statistic data to be output at RS422 interface.

Parameter: int SA_OutputStatisticMin_RS422	SA_OutputStatisticMin_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if min value is transmitted.	

Parameter: int SA_OutputStatisticMax_RS422	SA_OutputStatisticMax_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if max value is transmitted.	
Parameter: int SA_OutputStatisticPeak2Peak_RS422	SA_OutputStatisticPeak2Peak_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if peak to peak value is transmitted.	

11.2.3.2.11 Set_OutputStatistic_ETH (OUTSTATISTIC_ETH)

Set the statistic data to be output at ethernet interface.

Parameter: int SP_OutputStatisticMin_ETH	SP_OutputStatisticMin_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if min value is transmitted.	
Parameter: int SP_OutputStatisticMax_ETH	SP_OutputStatisticMax_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if max value is transmitted.	
Parameter: int SP_OutputStatisticPeak2Peak_ETH	SP_OutputStatisticPeak2Peak_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if peak to peak value is transmitted.	

11.2.3.2.12 Get_OutputStatistic_ETH (OUTSTATISTIC_ETH)

Get the statistic data to be output at ethernet interface.

Parameter: int SA_OutputStatisticMin_ETH	SA_OutputStatisticMin_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if min value is transmitted.	

Parameter: int SA_OutputStatisticMax_ETH	SA_OutputStatisticMax_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if max value is transmitted.	
Parameter: int SA_OutputStatisticPeak2Peak_ETH	SA_OutputStatisticPeak2Peak_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if peak to peak value is transmitted.	

11.2.3.2.13 Set_OutputAdditional_RS422 (OUTADD_RS422)

Set the additional data to be output at RS422 interface.

Parameter: int SP_OutputAdditionalShutterTime_RS422	SP_OutputAdditionalShutterTime_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	
Parameter: int SP_OutputAdditionalEncoder1_RS422	SP_OutputAdditionalEncoder1_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if encoder 1 is transmitted.	
Parameter: int SP_OutputAdditionalEncoder2_RS422	SP_OutputAdditionalEncoder2_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if encoder 2 is transmitted.	
Parameter: int SP_OutputAdditionalEncoder3_RS422	SP_OutputAdditionalEncoder3_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if encoder 3 is transmitted.	
Parameter: int SP_OutputAdditionalCounter_RS422	SP_OutputAdditionalCounter_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	

Parameter: int SP_OutputAdditionalTimestamp_RS422	SP_OutputAdditionalTimestamp_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SP_OutputAdditionalIntensity_RS422	SP_OutputAdditionalIntensity_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity is transmitted.	
Parameter: int SP_OutputAdditionalState_RS422	SP_OutputAdditionalState_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	
Parameter: int SP_OutputAdditionalTrgTimeDiff_RS422	SP_OutputAdditionalTrgTimeDiff_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if trigger time difference is transmitted.	
Parameter: int SP_OutputAdditionalMeasrate_RS422	SP_OutputAdditionalMeasrate_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if measrate is transmitted.	

11.2.3.2.14 Get_OutputAdditional_RS422 (OUTADD_RS422)

Get the additional data to be output at RS422 interface.

Parameter: int SA_OutputAdditionalShutterTime_RS422	SA_OutputAdditionalShutterTime_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	
Parameter: int SA_OutputAdditionalEncoder1_RS422	SA_OutputAdditionalEncoder1_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if encoder 1 is transmitted.	

Parameter: int SA_OutputAdditionalEncoder2_RS422	SA_OutputAdditionalEncoder2_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if encoder 2 is transmitted.	
Parameter: int SA_OutputAdditionalEncoder3_RS422	SA_OutputAdditionalEncoder3_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if encoder 3 is transmitted.	
Parameter: int SA_OutputAdditionalCounter_RS422	SA_OutputAdditionalCounter_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SA_OutputAdditionalTimestamp_RS422	SA_OutputAdditionalTimestamp_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SA_OutputAdditionalIntensity_RS422	SA_OutputAdditionalIntensity_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity is transmitted.	
Parameter: int SA_OutputAdditionalState_RS422	SA_OutputAdditionalState_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	
Parameter: int SA_OutputAdditionalTrgTimeDiff_RS422	SA_OutputAdditionalTrgTimeDiff_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if trigger time difference is transmitted.	
Parameter: int SA_OutputAdditionalMeasrate_RS422	SA_OutputAdditionalMeasrate_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measrate is transmitted.	

11.2.3.2.15 Set_OutputAdditional_ETH (OUTADD_ETH)

Set the additional data to be output at ethernet interface.

Parameter: int SP_OutputAdditionalShutterTime_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if shutter time is transmitted.

SP_OutputAdditionalShutterTime_ETH

Parameter: int SP_OutputAdditionalEncoder1_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if encoder 1 is transmitted.

SP_OutputAdditionalEncoder1_ETH

Parameter: int SP_OutputAdditionalEncoder2_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if encoder 2 is transmitted.

SP_OutputAdditionalEncoder2_ETH

Parameter: int SP_OutputAdditionalEncoder3_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if encoder 3 is transmitted.

SP_OutputAdditionalEncoder3_ETH

Parameter: int SP_OutputAdditionalCounter_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if counter is transmitted.

SP_OutputAdditionalCounter_ETH

Parameter: int SP_OutputAdditionalTimestamp_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if timestamp is transmitted.

SP_OutputAdditionalTimestamp_ETH

Parameter: int SP_OutputAdditionalIntensity_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if intensity is transmitted.

SP_OutputAdditionalIntensity_ETH

Parameter: int SP_OutputAdditionalState_ETH	SP_OutputAdditionalState_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	
Parameter: int SP_OutputAdditionalTrgTimeDiff_ETH	SP_OutputAdditionalTrgTimeDiff_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if trigger time difference is transmitted.	
Parameter: int SP_OutputAdditionalMeasrate_ETH	SP_OutputAdditionalMeasrate_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if measrate is transmitted.	

11.2.3.2.16 Get_OutputAdditional_ETH (OUTADD_ETH)

Get the additional data to be output at ethernet interface.

Parameter: int SA_OutputAdditionalShutterTime_ETH	SA_OutputAdditionalShutterTime_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	
Parameter: int SA_OutputAdditionalEncoder1_ETH	SA_OutputAdditionalEncoder1_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if encoder 1 is transmitted.	
Parameter: int SA_OutputAdditionalEncoder2_ETH	SA_OutputAdditionalEncoder2_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if encoder 2 is transmitted.	
Parameter: int SA_OutputAdditionalEncoder3_ETH	SA_OutputAdditionalEncoder3_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if encoder 3 is transmitted.	

Parameter: int SA_OutputAdditionalCounter_ETH	SA_OutputAdditionalCounter_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SA_OutputAdditionalTimestamp_ETH	SA_OutputAdditionalTimestamp_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SA_OutputAdditionalIntensity_ETH	SA_OutputAdditionalIntensity_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if intensity is transmitted.	
Parameter: int SA_OutputAdditionalState_ETH	SA_OutputAdditionalState_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	
Parameter: int SA_OutputAdditionalTrgTimeDiff_ETH	SA_OutputAdditionalTrgTimeDiff_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if trigger time difference is transmitted.	
Parameter: int SA_OutputAdditionalMeasrate_ETH	SA_OutputAdditionalMeasrate_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measrate is transmitted.	

11.2.3.2.17 Set_UnlinearizedMode (SWITCHMD2)

Specifiy if data should be output without linearisation.

Parameter: int SP_UnlinearizedMode	SP_UnlinearizedMode
Direction: Down	
Valid values:	
0= off	
1= on	
Description: Unlinearized data mode.	

11.2.3.2.18 Get_UnlinearizedMode (SWITCHMD2)

Retrieve if data is output without linearisation.

Parameter: int SA_UnlinearizedMode

SA_UnlinearizedMode

Direction: Up

Valid values:

0= off

1= on

Description: Unlinearized data mode.

11.2.3.2.19 Set_OutputVideo_ETH (OUTVIDEO)

Set the video signal to be output at ethernet interface.

Parameter: int SP_OutputVideoRaw_ETH

SP_OutputVideoRaw_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if raw video signal is transmitted.

Parameter: int SP_OutputVideoDark_ETH

SP_OutputVideoDark_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if dark preprocessed video signal is transmitted.

Parameter: int SP_OutputVideoLight_ETH

SP_OutputVideoLight_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if light preprocessed video signal is transmitted.

Parameter: int SP_OutputVideoDarkTable_ETH

SP_OutputVideoDarkTable_-
ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if dark table is transmitted.

Parameter: int SP_OutputVideoLightTable_ETH

SP_OutputVideoLight-
Table_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if light table is transmitted.

Parameter: int SP_OutputVideoThreshold_ETH **SP_OutputVideoThreshold_**
Direction: Down **ETH**
Valid values:
 0= no
 1= yes
Description: Specify if video threshold table is transmitted.

11.2.3.2.20 Get_OutputVideo_ETH (OUTVIDEO)

Get the video signal to be output at ethernet interface.

Parameter: int SA_OutputVideoRaw_ETH **SA_OutputVideoRaw_**
Direction: Up **ETH**
Valid values:
 0= no
 1= yes
Description: Specify if raw video signal is transmitted.

Parameter: int SA_OutputVideoDark_ETH **SA_OutputVideoDark_**
Direction: Up **ETH**
Valid values:
 0= no
 1= yes
Description: Specify if dark preprocessed video signal is transmitted.

Parameter: int SA_OutputVideoLight_ETH **SA_OutputVideoLight_**
Direction: Up **ETH**
Valid values:
 0= no
 1= yes
Description: Specify if light preprocessed video signal is transmitted.

Parameter: int SA_OutputVideoDarkTable_ETH **SA_OutputVideoDarkTable_**
Direction: Up **ETH**
Valid values:
 0= no
 1= yes
Description: Specify if dark table is transmitted.

Parameter: int SA_OutputVideoLightTable_ETH **SA_OutputVideoLight-**
Direction: Up **Table_**
Valid values:
 0= no
 1= yes
Description: Specify if light table is transmitted.

Parameter: int SA_OutputVideoThreshold_ETH **SA_OutputVideoThreshold_**
Direction: Up **ETH**
Valid values:
 0= no
 1= yes
Description: Specify if video threshold table is transmitted.

11.2.3.3 Switching outputs

11.2.3.3.1 Set_ErrorOutput1 (ERROROUT1)

Set condition to be used to set error output 1.

Parameter: int SP_ErrorOutput1

SP_ErrorOutput1

Direction: Down

Valid values:

- 0= None
- 1= Intensity error (ER1)
- 2= Out of range (ER2)
- 3= Intensity error or out of range (ER12)
- 4= Below low limit (LI1)
- 5= Above high limit (LI2)
- 6= Out of limits (LI12)

Description: Condition for error output.

11.2.3.3.2 Get_ErrorOutput1 (ERROROUT1)

Get condition to be used to set error output 1.

Parameter: int SA_ErrorOutput1

SA_ErrorOutput1

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= Intensity error (ER1)
- 2= Out of range (ER2)
- 3= Intensity error or out of range (ER12)
- 4= Below low limit (LI1)
- 5= Above high limit (LI2)
- 6= Out of limits (LI12)

Description: Condition for error output.

11.2.3.3.3 Set_ErrorOutput2 (ERROROUT2)

Set condition to be used to set error output 2.

Parameter: int SP_ErrorOutput2

SP_ErrorOutput2

Direction: Down

Valid values:

- 0= None
- 1= Intensity error (ER1)
- 2= Out of range (ER2)
- 3= Intensity error or out of range (ER12)
- 4= Below low limit (LI1)
- 5= Above high limit (LI2)
- 6= Out of limits (LI12)

Description: Condition for error output.

11.2.3.3.4 Get_ErrorOutput2 (ERRORROUT2)

Get condition to be used to set error output 2.

Parameter: int SA_ErrorOutput2

SA_ErrorOutput2

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = None
- 1 = Intensity error (ER1)
- 2 = Out of range (ER2)
- 3 = Intensity error or out of range (ER12)
- 4 = Below low limit (LI1)
- 5 = Above high limit (LI2)
- 6 = Out of limits (LI12)

Description: Condition for error output.

11.2.3.3.5 Set_ErrorLimit (ERRORLIMIT)

Set the error limits.

Parameter: int SP_DataSource

SP_DataSource

Direction: Down

Valid values:

- 0 = Distance 1
- 1 = Distance 2
- 3 = Distance 3 (only IFD2451/IFD2461/IFD2471)
- 4 = Distance 4 (only IFD2451/IFD2461/IFD2471)
- 5 = Distance 5 (only IFD2451/IFD2461/IFD2471)
- 6 = Distance 6 (only IFD2451/IFD2461/IFD2471)
- 2 = Thickness 12
- 7 = Thickness 13 (only IFD2451/IFD2461/IFD2471)
- 8 = Thickness 14 (only IFD2451/IFD2461/IFD2471)
- 9 = Thickness 15 (only IFD2451/IFD2461/IFD2471)
- 10 = Thickness 16 (only IFD2451/IFD2461/IFD2471)
- 11 = Thickness 23 (only IFD2451/IFD2461/IFD2471)
- 12 = Thickness 24 (only IFD2451/IFD2461/IFD2471)
- 13 = Thickness 25 (only IFD2451/IFD2461/IFD2471)
- 14 = Thickness 26 (only IFD2451/IFD2461/IFD2471)
- 15 = Thickness 34 (only IFD2451/IFD2461/IFD2471)
- 16 = Thickness 35 (only IFD2451/IFD2461/IFD2471)
- 17 = Thickness 36 (only IFD2451/IFD2461/IFD2471)
- 18 = Thickness 45 (only IFD2451/IFD2461/IFD2471)
- 19 = Thickness 46 (only IFD2451/IFD2461/IFD2471)
- 20 = Thickness 56 (only IFD2451/IFD2461/IFD2471)

Description: Data source to be checked.

Parameter: double SP_LowerLimit

SP_LowerLimit

Direction: Down

Valid values:

- Minimum:** -120.0
- Maximum:** 120.0

Unit: mm

Description: Lower limit.

Parameter: double SP_UpperLimit SP_UpperLimit
Direction: Down
Valid values:
Minimum: -120.0
Maximum: 120.0
Unit: mm
Description: Upper limit.

11.2.3.3.6 Get_ErrorLimit (ERRORLIMIT)

Get the error limits.

Parameter: int SA_DataSource SA_DataSource
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= Distance 1
 1= Distance 2
 3= Distance 3 (only IFD2451/IFD2461/IFD2471)
 4= Distance 4 (only IFD2451/IFD2461/IFD2471)
 5= Distance 5 (only IFD2451/IFD2461/IFD2471)
 6= Distance 6 (only IFD2451/IFD2461/IFD2471)
 2= Thickness 12
 7= Thickness 13 (only IFD2451/IFD2461/IFD2471)
 8= Thickness 14 (only IFD2451/IFD2461/IFD2471)
 9= Thickness 15 (only IFD2451/IFD2461/IFD2471)
 10= Thickness 16 (only IFD2451/IFD2461/IFD2471)
 11= Thickness 23 (only IFD2451/IFD2461/IFD2471)
 12= Thickness 24 (only IFD2451/IFD2461/IFD2471)
 13= Thickness 25 (only IFD2451/IFD2461/IFD2471)
 14= Thickness 26 (only IFD2451/IFD2461/IFD2471)
 15= Thickness 34 (only IFD2451/IFD2461/IFD2471)
 16= Thickness 35 (only IFD2451/IFD2461/IFD2471)
 17= Thickness 36 (only IFD2451/IFD2461/IFD2471)
 18= Thickness 45 (only IFD2451/IFD2461/IFD2471)
 19= Thickness 46 (only IFD2451/IFD2461/IFD2471)
 20= Thickness 56 (only IFD2451/IFD2461/IFD2471)
Description: Data source to be checked.

Parameter: double SA_LowerLimit SA_LowerLimit
Direction: Up
Valid values:
Minimum: -120.0
Maximum: 120.0
Unit: mm
Description: Lower limit.

Parameter: double SA_UpperLimit SA_UpperLimit
Direction: Up
Valid values:
Minimum: -120.0
Maximum: 120.0
Unit: mm
Description: Upper limit.

11.2.3.3.7 Set_ErrorLevel (ERRORLEVEL)

Set level of error output on error.

Parameter: int SP_ErrorLevel

SP_ErrorLevel

Direction: Down

Valid values:

0= High
1= Low

Description: Error level.

11.2.3.3.8 Get_ErrorLevel (ERRORLEVEL)

Get level of error output on error.

Parameter: int SA_ErrorLevel

SA_ErrorLevel

Direction: Up

Valid values:

0= High
1= Low

Description: Error level.

11.2.3.4 Analog output

11.2.3.4.1 Set_AnalogOutput (ANALOGOUT)

Set the data to be used for analog output.

Parameter: int SP_AnalogOutput

SP_AnalogOutput

Direction: Down

Valid values:

0= Distance 1
1= Distance 2
3= Distance 3 (only IFD2451/IFD2461/IFD2471)
4= Distance 4 (only IFD2451/IFD2461/IFD2471)
5= Distance 5 (only IFD2451/IFD2461/IFD2471)
6= Distance 6 (only IFD2451/IFD2461/IFD2471)
2= Thickness 12
7= Thickness 13 (only IFD2451/IFD2461/IFD2471)
8= Thickness 14 (only IFD2451/IFD2461/IFD2471)
9= Thickness 15 (only IFD2451/IFD2461/IFD2471)
10= Thickness 16 (only IFD2451/IFD2461/IFD2471)
11= Thickness 23 (only IFD2451/IFD2461/IFD2471)
12= Thickness 24 (only IFD2451/IFD2461/IFD2471)
13= Thickness 25 (only IFD2451/IFD2461/IFD2471)
14= Thickness 26 (only IFD2451/IFD2461/IFD2471)
15= Thickness 34 (only IFD2451/IFD2461/IFD2471)
16= Thickness 35 (only IFD2451/IFD2461/IFD2471)
17= Thickness 36 (only IFD2451/IFD2461/IFD2471)
18= Thickness 45 (only IFD2451/IFD2461/IFD2471)
19= Thickness 46 (only IFD2451/IFD2461/IFD2471)
20= Thickness 56 (only IFD2451/IFD2461/IFD2471)

Description: Data to be used for analog output.

11.2.3.4.2 Get_AnalogOutput (ANALOGOUT)

Get the data to be used for analog output.

Parameter: int SA_AnalogOutput

SA_AnalogOutput

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Distance 1
- 1= Distance 2
- 3= Distance 3 (only IFD2451/IFD2461/IFD2471)
- 4= Distance 4 (only IFD2451/IFD2461/IFD2471)
- 5= Distance 5 (only IFD2451/IFD2461/IFD2471)
- 6= Distance 6 (only IFD2451/IFD2461/IFD2471)
- 2= Thickness 12
- 7= Thickness 13 (only IFD2451/IFD2461/IFD2471)
- 8= Thickness 14 (only IFD2451/IFD2461/IFD2471)
- 9= Thickness 15 (only IFD2451/IFD2461/IFD2471)
- 10= Thickness 16 (only IFD2451/IFD2461/IFD2471)
- 11= Thickness 23 (only IFD2451/IFD2461/IFD2471)
- 12= Thickness 24 (only IFD2451/IFD2461/IFD2471)
- 13= Thickness 25 (only IFD2451/IFD2461/IFD2471)
- 14= Thickness 26 (only IFD2451/IFD2461/IFD2471)
- 15= Thickness 34 (only IFD2451/IFD2461/IFD2471)
- 16= Thickness 35 (only IFD2451/IFD2461/IFD2471)
- 17= Thickness 36 (only IFD2451/IFD2461/IFD2471)
- 18= Thickness 45 (only IFD2451/IFD2461/IFD2471)
- 19= Thickness 46 (only IFD2451/IFD2461/IFD2471)
- 20= Thickness 56 (only IFD2451/IFD2461/IFD2471)

Description: Data to be used for analog output.

11.2.3.4.3 Set_AnalogRange (ANALOG RANGE)

Set the analog output range.

Parameter: int SP_AnalogRange

SP_AnalogRange

Direction: Down

Valid values:

- 0= None
- 1= 0 - 5V
- 2= 0 - 10V
- 3= -5 - 5V
- 4= -10 - 10V
- 5= 4 - 20mA

Description: Analog output range.

11.2.3.4.4 Get_AnalogRange (ANALOG RANGE)

Get the analog output range.

Parameter: int SA_AnalogRange

SA_AnalogRange

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = None
- 1 = 0 - 5V
- 2 = 0 - 10V
- 3 = -5 - 5V
- 4 = -10 - 10V
- 5 = 4 - 20mA

Description: Analog output range.

11.2.3.4.5 Set_AnalogScale (ANALOGSCALE)

Set the scaling factor for analog output. If both parameters are zero, the analog output is scaled to range of sensor head.

Parameter: int SP_AnalogScaleMode

SP_AnalogScaleMode

Direction: Down

Valid values:

- 0 = Standard
- 1 = Two point (TWOPOINT)

Description: Analog scale mode.

Parameter: double SP_MinValue

SP_MinValue

Direction: Down

Valid values:

- Minimum:** -120.0
- Maximum:** 120.0

Unit: mm

Description: Value which represents lowest voltage/current (at two point scaling).

Parameter: double SP_MaxValue

SP_MaxValue

Direction: Down

Valid values:

- Minimum:** -120.0
- Maximum:** 120.0

Unit: mm

Description: Value which represents highest voltage/current (at two point scaling).

11.2.3.4.6 Get_AnalogScale (ANALOGSCALE)

Get the scaling factor for analog output. If both parameters are zero, the analog output is scaled to range of sensor head.

Parameter: int SA_AnalogScaleMode

SA_AnalogScaleMode

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = Standard
- 1 = Two point (TWOPOINT)

Description: Analog scale mode.

Parameter: double SA_MinValue SA_MinValue

Direction: Up

Valid values:

Minimum: -120.0

Maximum: 120.0

Unit: mm

Description: Value which represents lowest voltage/current (at two point scaling).

Parameter: double SA_MaxValue SA_MaxValue

Direction: Up

Valid values:

Minimum: -120.0

Maximum: 120.0

Unit: mm

Description: Value which represents highest voltage/current (at two point scaling).

11.2.4 Internal commands

11.2.4.1 Set_PeakParameter (PEAKPARAM)

Set peak detection parameters. This is an internal command. It should not be used by the customer.

Parameter: int SP_HalfMinDist SP_HalfMinDist

Direction: Down

Valid values:

Minimum: 0

Maximum: 255

Description: Half of the minimum distance between two peaks.

Parameter: int SP_MinGap SP_MinGap

Direction: Down

Valid values:

Minimum: 0

Maximum: 255

Description: Minimum gap between two peaks (signal below threshold).

11.2.4.2 Get_PeakParameter (PEAKPARAM)

Get peak detection parameters. This is an internal command. It should not be used by the customer.

Parameter: int SA_HalfMinDist SA_HalfMinDist

Direction: Up

Valid values:

Minimum: 0

Maximum: 255

Description: Half of the minimum distance between two peaks.

Parameter: int SA_MinGap SA_MinGap
Direction: Up
Valid values:
Minimum: 0
Maximum: 255
Description: Minimum gap between two peaks (signal below threshold).

11.2.4.3 Set_BoreControlParam (BORECTRL)

Set firmware parameters for boreCONTROL.
 Only available at Option 002 and 201.
 This is an internal command. It should not be used by the customer.

Parameter: int SP_SplitPixel SP_SplitPixel
Direction: Down
Valid values:
Minimum: 0
Maximum: 511
Description: DarkCorr evaluates only CCD right of this pixel.

Parameter: int SP_DarkCorrAverage SP_DarkCorrAverage
Direction: Down
Valid values:
Minimum: 0
Maximum: 2048
Description: Video averaging for DarkCorr.

11.2.4.4 Get_BoreControlParam (BORECTRL)

Get firmware parameters for boreCONTROL.
 Only available at Option 002 and 201.
 This is an internal command. It should not be used by the customer.

Parameter: int SA_SplitPixel SA_SplitPixel
Direction: Up
Valid values:
Minimum: 0
Maximum: 511
Description: DarkCorr evaluates only CCD right of this pixel.

Parameter: int SA_DarkCorrAverage SA_DarkCorrAverage
Direction: Up
Valid values:
Minimum: 0
Maximum: 2048
Description: Video averaging for DarkCorr.

11.2.4.5 Get_SpecCal (SPECCAL)

This is an internal command. It should not be used by the customer.

Parameter: double SA_lc0 SA_lc0

Direction: Up

Valid values:

Minimum: -1.79769e+308 (-DBL_MAX)

Maximum: 1.79769e+308 (DBL_MAX)

Description: lc0.

Parameter: double SA_lc1 SA_lc1

Direction: Up

Valid values:

Minimum: -1.79769e+308 (-DBL_MAX)

Maximum: 1.79769e+308 (DBL_MAX)

Description: lc1.

Parameter: double SA_lc2 SA_lc2

Direction: Up

Valid values:

Minimum: -1.79769e+308 (-DBL_MAX)

Maximum: 1.79769e+308 (DBL_MAX)

Description: lc2.

11.2.4.6 Del_LinearisationTable (DELLINTAB)

Delete a linearisation table from sensor. This is an internal command. It should not be used by the customer.

Parameter: int SP_SensorTable SP_SensorTable

Direction: Down

Valid values:

Minimum: 0

Maximum: 19

Description: Index of sensor table.

11.2.4.7 Save_ParametersTemp (SAVE2TMP)

Save parameters temporary. This is an internal command. It should not be used by the customer.

11.2.4.8 Load_ParametersTemp (RESTORETMP)

Restore parameters previously saved from temporary buffer. This is an internal command. It should not be used by the customer.

11.2.4.9 Set_CalibrationDefaults (SETCALDEFAULT)

Set default parameter set for calibration. This is an internal command. It should not be used by the customer.

Parameter: int SP_CalDefaultType SP_CalDefaultType
Direction: Down
Valid values:
 0= All settings (ALL)
 1= Only measurement settings (MEAS)
Description: Specifies which settings should be set to calibration defaults.

11.2.4.10 Get_RegHyst (REGHYST)

Get the destination value for regulation (peak height in raw signal). This is an internal command. It should not be used by the customer.

Parameter: int SA_RegHyst SA_RegHyst
Direction: Up
Valid values:
Minimum: 0
Maximum: 100
Unit: %
Description: Destination value for regulation.

11.2.4.11 Upload_CalibTable

Send a calibration table for selected sensor to the controller. This is an internal command. It should not be used by the customer.

Parameter: Binary data SP_CalibTable SP_CalibTable
Direction: Down
Description: Calibration table for selected sensor.

11.2.4.12 Save_DarkCorrReference (SAVEDARKCORRREF)

Save the darkcorr reference value. This is an internal command. It should not be used by the customer.

12 Commands for ODC sensors

12.1 Commands for ODC1202

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [RS232](#) (native).
- [TCP/IP](#) (additional, e.g. RS232/RS422 to TCP/IP comm server and RS232 high level interface).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_Version](#) and [Get_ParamSet2](#) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib timeout checking and to scale data.

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor (including two status bits, mask it out by raw&0xffff), from 0 to 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28) (without segment bits).
- Scaled values are scaled using sensor range, from 0 to 75.0 (ODC1202-75) or 100.0 (ODC1202-100) or 28.0 (ODC1220-28), error values are scaled depending of [IP_ScaleErrorValues](#).

12.1.1 General commands

12.1.1.1 General

12.1.1.1.1 Get_Version

Read sensor version and other information.

Parameter: String SA_Sensor SA_Sensor
Direction: Up
Description: Name of the sensor.

Parameter: double SA_Range SA_Range
Direction: Up
Unit: mm
Valid values:
 75
 100
 28
Description: Sensor range.

Parameter: String SA_Softwareversion SA_Softwareversion
Direction: Up
Description: Sensor firmware version.

Parameter: String SA_Date SA_Date
Direction: Up
Description: Sensor firmware date.

12.1.1.2 User level

12.1.1.2.1 Get_Echo

Checks if the sensor is responding.

Parameter: int SA_EchoValue SA_EchoValue
Direction: Up
Valid values:
 0x00aa
Description: Echo value, returned from sensor.

12.1.1.3 Interfaces

12.1.1.3.1 Change_Baudrate

Change baudrate at sensor. After receiving the answer of the command, the baudrate of the local serial interface is changed, too.

Parameter: int SP_RS232Baudrate SP_RS232Baudrate
Direction: Down
Valid values:
 0= 9600 Baud
 1= 19200 Baud
 2= 38400 Baud
 3= 57600 Baud
 4= 115200 Baud
Description: Baudrate of RS232 interface.

12.1.1.4 Parameter management

12.1.1.4.1 RefreshVideoThr

Refresh auto video threshold at sensor.

Parameter: int SP_Location SP_Location
Direction: Down
Valid values:
 0= RAM
 1= EEPROM
Description: Specify if threshold should be stored in temporary (RAM) until sensor is switch off or permanently (EEPROM).

Parameter: int SA_VideoThrAuto SA_VideoThrAuto
Direction: Up
Valid values:
Minimum: 0
Maximum: 100
Description: Video threshold (in percent) of full ADC range for automatic video threshold mode.

12.1.2 Measurement

12.1.2.1 Set_ParamSet1

Set first part of parameters.

Parameter: int SP_Location SP_Location
Direction: Down
Valid values:
 0= RAM
 1= EEPROM
Description: Specify if parameters should be stored in temporary (RAM) until sensor is switch off or permanently (EEPROM).

Parameter: int SP_Power SP_Power
Direction: Down
Valid values:
Minimum: 0
Maximum: 1000
Description: Laser intensity.

Parameter: int SP_PowerMode SP_PowerMode
Direction: Down
Valid values:
 0= Static
 1= Dynamic
Description: Transmitter mode, currently not used.

Parameter: int SP_Polarity SP_Polarity
Direction: Down
Valid values:
 0= Direct
 1= Inverse
Description: Polarity setting for digital outputs OUT0 to OUT3.

Parameter: int SP_EvalMode SP_EvalMode
Direction: Down
Valid values:
 0= Left edge
 1= Right edge
 2= Width
 3= Center
Description: Evaluation mode for sensor.

Parameter: int SP_EvalBegin SP_EvalBegin

Direction: Down

Valid values:

Minimum: 0

Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)

Description: Evaluation start pixel.

Parameter: int SP_EvalEnd SP_EvalEnd

Direction: Down

Valid values:

Minimum: 0

Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)

Description: Evaluation end pixel. SP_EvalEnd must be greater as SP_EvalBegin.

Parameter: int SP_TeachValue SP_TeachValue

Direction: Down

Valid values:

Minimum: 1

Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)

Description: Teach value.

Parameter: int SP_ToleranceHigh SP_ToleranceHigh

Direction: Down

Valid values:

Minimum: 0

Maximum: 2360/2 (ODC1202-75) or 3072/2 (ODC1202-100) or 16384/2 (ODC1220-28)

Description: Upper tolerance value.

Parameter: int SP_ToleranceLow SP_ToleranceLow

Direction: Down

Valid values:

Minimum: 0

Maximum: 2360/2 (ODC1202-75) or 3072/2 (ODC1202-100) or 16384/2 (ODC1220-28)

Description: Lower tolerance value.

Parameter: int SP_Averaging SP_Averaging

Direction: Down

Valid values:

1

2

4

8

16

32

64

128

256

Description: Average setting for measurement value.

12.1. Commands for ODC1202

Parameter: int SP_TriggerMode	SP_TriggerMode
Direction: Down	
Valid values:	
0= Continuous	
1= In0 rising edge	
2= In0 high level	
3= Save video threshold	
4= Output width	
5= Laser on/off	
Description: Trigger mode.	
Parameter: int SP_AnalogOutput	SP_AnalogOutput
Direction: Down	
Valid values:	
0= Direct 0..10V	
1= Maxima	
2= Minima	
3= Max-Min	
Description: Analog output mode.	
Parameter: int SP_OperationMode	SP_OperationMode
Direction: Down	
Valid values:	
0= ADC	
1= Comperator	
Description: CCD operation mode. It can be sampled using the ADC or only an comperator (1 bit).	
Parameter: int SP_HWMode	SP_HWMode
Direction: Down	
Valid values:	
0= Disable all	
1= Enable all	
2= Enable button	
3= Enable poti	
Description: Enabe or disable tolerance poti and button at sensor.	
Parameter: int SP_VideoThrMode	SP_VideoThrMode
Direction: Down	
Valid values:	
0= Fix	
1= Auto	
Description: Video threshold mode.	

12.1.2.2 Get_ParamSet1

Get first part of parameters.

Parameter: int SP_Location	SP_Location
Direction: Down	
Valid values:	
0= RAM	
1= EEPROM	
Description: Specify if parameters should be read from RAM or EEPROM.	

Parameter: int SA_Power	SA_Power
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 1000	
Description: Laser intensity.	
Parameter: int SA_PowerMode	SA_PowerMode
Direction: Up	
Valid values:	
0= Static	
1= Dynamic	
Description: Transmitter mode, currently not used.	
Parameter: int SA_Polarity	SA_Polarity
Direction: Up	
Valid values:	
0= Direct	
1= Inverse	
Description: Polarity setting for digital outputs OUT0 to OUT3.	
Parameter: int SA_EvalMode	SA_EvalMode
Direction: Up	
Valid values:	
0= Left edge	
1= Right edge	
2= Width	
3= Center	
Description: Evaluation mode for sensor.	
Parameter: int SA_EvalBegin	SA_EvalBegin
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)	
Description: Evaluation start pixel.	
Parameter: int SA_EvalEnd	SA_EvalEnd
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)	
Description: Evaluation end pixel.	
Parameter: int SA_TeachValue	SA_TeachValue
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)	
Description: Teach value.	

12.1. Commands for ODC1202

Parameter: int SA_ToleranceHigh	SA_ToleranceHigh
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 2360/2 (ODC1202-75) or 3072/2 (ODC1202-100) or 16384/2 (ODC1220-28)	
Description: Upper tolerance value.	
Parameter: int SA_ToleranceLow	SA_ToleranceLow
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 2360/2 (ODC1202-75) or 3072/2 (ODC1202-100) or 16384/2 (ODC1220-28)	
Description: Lower tolerance value.	
Parameter: int SA_Averaging	SA_Averaging
Direction: Up	
Valid values:	
1	
2	
4	
8	
16	
32	
64	
128	
256	
Description: Average setting for measurement value.	
Parameter: int SA_TriggerMode	SA_TriggerMode
Direction: Up	
Valid values:	
0= Continuous	
1= In0 rising edge	
2= In0 high level	
3= Save video threshold	
4= Output width	
5= Laser on/off	
Description: Trigger mode.	
Parameter: int SA_AnalogOutput	SA_AnalogOutput
Direction: Up	
Valid values:	
0= Direct 0..10V	
1= Maxima	
2= Minima	
3= Max-Min	
Description: Analog output mode.	
Parameter: int SA_OperationMode	SA_OperationMode
Direction: Up	
Valid values:	
0= ADC	
1= Comperator	
Description: CCD operation mode. It can be sampled using the ADC or only an comperator (1 bit).	

Parameter: int SA_HWMode SA_HWMode

Direction: Up

Valid values:

- 0= Disable all
- 1= Enable all
- 2= Enable button
- 3= Enable poti

Description: Enable or disable tolerance poti and button at sensor.

Parameter: int SA_VideoThrMode SA_VideoThrMode

Direction: Up

Valid values:

- 0= Fix
- 1= Auto

Description: Video threshold mode.

12.1.2.3 Set_ParamSet2

Set second part of parameters.

Parameter: int SP_Location SP_Location

Direction: Down

Valid values:

- 0= RAM
- 1= EEPROM

Description: Specify if parameters should be stored in temporary (RAM) until sensor is switch off or permanently (EEPROM).

Parameter: int SP_VideoThrFix SP_VideoThrFix

Direction: Down

Valid values:

- Minimum:** 0
- Maximum:** 100

Description: Video threshold (in percent) of full ADC range for fix video threshold mode.

Parameter: int SP_VideoThrAuto SP_VideoThrAuto

Direction: Down

Valid values:

- Minimum:** 0
- Maximum:** 100

Description: Video threshold (in percent) of full ADC range for automatic video threshold mode.

Parameter: int SP_RS232Mode SP_RS232Mode

Direction: Down

Valid values:

- 0= Static (record)
- 1= In0 rising edge (record)
- 2= In0 rising edge (3 byte)
- 3= Continous (3 byte)

Description: Transfer mode at RS232 interface.

Parameter: int SP_RS232Baudrate	SP_RS232Baudrate
Direction: Down	
Valid values:	
0= 9600 Baud	
1= 19200 Baud	
2= 38400 Baud	
3= 57600 Baud	
4= 115200 Baud	
Description: Baudrate of RS232 interface.	
Parameter: int SP_SmoothVideo	SP_SmoothVideo
Direction: Down	
Valid values:	
1	
2	
4	
8	
12	
16	
24	
32	
48	
64	
Description: Smooth video signal.	
Parameter: int SP_AnalogZoom	SP_AnalogZoom
Direction: Down	
Valid values:	
0= Direct	
1= Zoom x1	
2= Zoom x2	
3= Zoom x4	
4= Zoom x8	
5= Zoom x15	
6= Tolerance Window 5V	
7= Tolerance Window 10V	
Description: Zoom mode for analog output.	
Parameter: int SP_InternalTriggerMode	SP_InternalTriggerMode
Direction: Down	
Valid values:	
0= Disable	
1= Enable	
Description: Internal object trigger.	
Parameter: int SP_DarkPixel	SP_DarkPixel
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)	
Description: Number of dark pixels for object trigger.	

12.1. Commands for ODC1202

Parameter: int SP_OutMode	SP_OutMode
Direction: Down	
Valid values:	
0= [-]OUT0, [+]OUT1, [OK]OUT2	
Description: Output mode.	
Parameter: int SP_Parameter10	SP_Parameter10
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved for further use.	
Parameter: int SP_Parameter11	SP_Parameter11
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved for further use.	
Parameter: double SP_SlopeValue	SP_SlopeValue
Direction: Down	
Unit: $\mu\text{m}/\text{pixel}$	
Valid values:	
Minimum: 1.0	
Maximum: 1000.0	
Description: Slope value for calibration.	
Parameter: double SP_RefOffset	SP_RefOffset
Direction: Down	
Unit: μm	
Valid values:	
Minimum: -32767.0	
Maximum: 32767.0	
Description: Intersection parameter for calibration.	

12.1.2.4 Get_ParamSet2

Get second part of parameters.

Parameter: int SP_Location	SP_Location
Direction: Down	
Valid values:	
0= RAM	
1= EEPROM	
Description: Specify if parameters should be read from RAM or EEPROM.	
Parameter: int SA_VideoThrFix	SA_VideoThrFix
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 100	
Description: Video threshold (in percent) of full ADC range for fix video threshold mode.	

12.1. Commands for ODC1202

Parameter: int SA_VideoThrAuto	SA_VideoThrAuto
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 100	
Description: Video threshold (in percent) of full ADC range for automatic video threshold mode.	
Parameter: int SA_RS232Mode	SA_RS232Mode
Direction: Up	
Valid values:	
0= Static (record)	
1= In0 rising edge (record)	
2= In0 rising edge (3 byte)	
3= Continous (3 byte)	
Description: Transfer mode at RS232 interface.	
Parameter: int SA_RS232Baudrate	SA_RS232Baudrate
Direction: Up	
Valid values:	
0= 9600 Baud	
1= 19200 Baud	
2= 38400 Baud	
3= 57600 Baud	
4= 115200 Baud	
Description: Baudrate of RS232 interface.	
Parameter: int SA_SmoothVideo	SA_SmoothVideo
Direction: Up	
Valid values:	
1	
2	
4	
8	
12	
16	
24	
32	
48	
64	
Description: Smooth video signal.	
Parameter: int SA_AnalogZoom	SA_AnalogZoom
Direction: Up	
Valid values:	
0= Direct	
1= Zoom x1	
2= Zoom x2	
3= Zoom x4	
4= Zoom x8	
5= Zoom x15	
6= Tolerance Window 5V	
7= Tolerance Window 10V	
Description: Zoom mode for analog output.	

Parameter: int SA_InternalTriggerMode	SA_InternalTriggerMode
Direction: Up	
Valid values:	
0= Disable	
1= Enable	
Description: Internal object trigger.	
Parameter: int SA_DarkPixel	SA_DarkPixel
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)	
Description: Number of dark pixels for object trigger.	
Parameter: int SA_OutMode	SA_OutMode
Direction: Up	
Valid values:	
0= [-]OUT0, [+]OUT1, [OK]OUT2	
Description: Output mode.	
Parameter: int SA_Parameter10	SA_Parameter10
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved for further use.	
Parameter: int SA_Parameter11	SA_Parameter11
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved for further use.	
Parameter: double SA_SlopeValue	SA_SlopeValue
Direction: Up	
Unit: $\mu\text{m}/\text{pixel}$	
Valid values:	
Minimum: 1.0	
Maximum: 1000.0	
Description: Slope value for calibration.	
Parameter: double SA_RefOffset	SA_RefOffset
Direction: Up	
Unit: μm	
Valid values:	
Minimum: -32767.0	
Maximum: 32767.0	
Description: Intersection parameter for calibration.	

12.1.2.5 Get_Video

Get video signal from sensor. When starting reading video signal (first call to Get_Video), the whole video signal is stored in a buffer at sensor and can be read completely by further calls to Get_Video (sample and hold). If another command is executed, the buffer is released.

Parameter: int SP_BlockIndex SP_BlockIndex

Direction: Down

Valid values:

Minimum: -1

Maximum: 3

Description: Video signal cannot be transmitted at once, so it is devided into four blocks (0..3). If block index is -1, MEDAQLib automatically reads all four blocks and concatenates the video signal.

Parameter: Binary data SA_VideoSignal SA_VideoSignal

Direction: Up

Valid values:

Each block (0..3) contains 128 bytes, convertible to 64 words.

Description: Raw video signal.

12.1.2.6 Get_MeasValues

Get measured values from sensor.

Parameter: int SA_LeftEdge SA_LeftEdge

Direction: Up

Valid values:

Minimum: 0

Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)

Description: Pixel position of left edge.

Parameter: int SA_RightEdge SA_RightEdge

Direction: Up

Valid values:

Minimum: 0

Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)

Description: Pixel position of right edge.

Parameter: int SA_MeasValue SA_MeasValue

Direction: Up

Valid values:

Minimum: 0

Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)

Description: Unscaled measurement value.

Parameter: double SA_ScaledValue	SA_ScaledValue
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 75.0 (ODC1202-75) or 100.0 (ODC1202-100) or 28.0 (ODC1220-28)	
Description: Scaled measurement value.	
Parameter: int SA_TeachValue	SA_TeachValue
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)	
Description: Teach value.	
Parameter: int SA_ToleranceValue	SA_ToleranceValue
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 1024	
Description: Tolerance value of potentiometer.	
Parameter: int SA_EdgeCount	SA_EdgeCount
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 15	
Description: Number of detected edges.	
Parameter: int SA_MV-First-8	SA_MV-First-8
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 255	
Description: Mean value of first eight pixel after begin of evaluation in video signal.	
Parameter: int SA_MV-Last-8	SA_MV-Last-8
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 255	
Description: Mean value of last eight pixel before end of evaluation in video signal.	
Parameter: int SA_FreePara1	SA_FreePara1
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 65534	
Description: Reserved for further use.	

Parameter: int SA_MaxValue SA_MaxValue

Direction: Up

Valid values:

Minimum: 0

Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)

Description: Maximum value.

Parameter: int SA_MinValue SA_MinValue

Direction: Up

Valid values:

Minimum: 0

Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)

Description: Minimum value.

Parameter: int SA_InState SA_InState

Direction: Up

Valid values:

Bit 0 shows digital input In0

Bit 1 shows digital input In1

Bit 2 shows button at sensor

Description: Bit coded state of digital input and button at sensor.

Parameter: int SA_VideoMax SA_VideoMax

Direction: Up

Valid values:

Minimum: 0

Maximum: 255

Description: Maximum value of video signal.

Parameter: int SA_DarkPixel SA_DarkPixel

Direction: Up

Valid values:

Minimum: 0

Maximum: 65535

Description: Dark pixel.

12.1.2.7 Get_Value_DataRecord

Get measured values from sensor (less as at Get_MeasValues).

Parameter: int SA_MeasValue SA_MeasValue

Direction: Up

Valid values:

Minimum: 0

Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)

Description: Unscaled measurement value.

12.1. Commands for ODC1202

Parameter: int SA_LeftEdge	SA_LeftEdge
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)	
Description: Pixel position of left edge.	
Parameter: int SA_RightEdge	SA_RightEdge
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)	
Description: Pixel position of right edge.	
Parameter: double SA_ScaledValue	SA_ScaledValue
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 75.0 (ODC1202-75) or 100.0 (ODC1202-100) or 28.0 (ODC1220-28)	
Description: Scaled measurement value.	
Parameter: int SA_EdgeCount	SA_EdgeCount
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 15	
Description: Number of detected edges.	

12.1.3 Data output

12.1.3.1 General

12.1.3.1.1 AutoDataTransfer

Starts or stops auto data transfer at sensor.

Parameter: int SP_Active	SP_Active
Direction: Down	
Valid values:	
0= Stop	
1= Start	
Description: This mode can be activated only if parameter SP_RS232Mode is set to 3 (Continous, 3 byte) at command Set_ParamSet2.	

12.1.3.2 Switching outputs

12.1.3.2.1 Activate_Teach

Activate teaching mode at sensor.

Parameter: int SA_Power SA_Power

Direction: Up

Valid values:

Minimum: 0

Maximum: 1000

Description: Laser intensity.

Parameter: int SA_PowerMode SA_PowerMode

Direction: Up

Valid values:

0= Static

1= Dynamic

Description: Transmitter mode, currently not used.

Parameter: int SA_Polarity SA_Polarity

Direction: Up

Valid values:

0= Direct

1= Inverse

Description: Polarity setting for digital outputs OUT0 to OUT3.

Parameter: int SA_EvalMode SA_EvalMode

Direction: Up

Valid values:

0= Left edge

1= Right edge

2= Width

3= Center

Description: Evaluation mode for sensor.

Parameter: int SA_EvalBegin SA_EvalBegin

Direction: Up

Valid values:

Minimum: 0

Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)

Description: Evaluation start pixel.

Parameter: int SA_EvalEnd SA_EvalEnd

Direction: Up

Valid values:

Minimum: 0

Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)

Description: Evaluation end pixel.

Parameter: int SA_TeachValue	SA_TeachValue
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 2360 (ODC1202-75) or 3072 (ODC1202-100) or 16384 (ODC1220-28)	
Description: Teach value.	
Parameter: int SA_ToleranceHigh	SA_ToleranceHigh
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 2360/2 (ODC1202-75) or 3072/2 (ODC1202-100) or 16384/2 (ODC1220-28)	
Description: Upper tolerance value.	
Parameter: int SA_ToleranceLow	SA_ToleranceLow
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 2360/2 (ODC1202-75) or 3072/2 (ODC1202-100) or 16384/2 (ODC1220-28)	
Description: Lower tolerance value.	
Parameter: int SA_Averaging	SA_Averaging
Direction: Up	
Valid values:	
1	
2	
4	
8	
16	
32	
64	
128	
256	
Description: Average setting for measurement value.	
Parameter: int SA_TriggerMode	SA_TriggerMode
Direction: Up	
Valid values:	
0= Continuous	
1= In0 rising edge	
2= In0 high level	
3= Save video threshold	
4= Output width	
5= Laser on/off	
Description: Trigger mode.	
Parameter: int SA_AnalogOutput	SA_AnalogOutput
Direction: Up	
Valid values:	
0= Direct 0..10V	
1= Maxima	
2= Minima	
3= Max-Min	
Description: Analog output mode.	

Parameter: int SA_OperationMode	SA_OperationMode
Direction: Up	
Valid values:	
0= ADC	
1= Comperator	
Description:	CCD operation mode. It can be sampled using the ADC or only an comperator (1 bit).
Parameter: int SA_HWMode	SA_HWMode
Direction: Up	
Valid values:	
0= Disable all	
1= Enable all	
2= Enable button	
3= Enable poti	
Description:	Enabe or disable tolerance poti and button at sensor.
Parameter: int SA_VideoThrMode	SA_VideoThrMode
Direction: Up	
Valid values:	
0= Fix	
1= Auto	
Description:	Video threshold mode.

12.1.3.3 Analog output

12.1.3.3.1 Reset_MinMax_Analog

Reset maximum/minimum values at analog output.

12.2 Commands for ODC2500

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [RS232](#) (native).
- [IF2004](#) (native).
- [TCP/IP](#) (additional, e.g. RS232/RS422 to TCP/IP comm server and RS232 high level interface).
- [IF2004_USB](#) (native).
- [IF2008](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_Info](#) and [Read_OptionData](#) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib to calculate datarate, to interpret data and to assign values.

If second bit of [IP_AutomaticMode](#) is set (2), MEDAQLib calls sensor command [Dat_Out_On](#) at [OpenSensor](#).

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor (including two segment bits, mask it out by raw&0xffff), from 0 to 65535 (without segment bits).
- Scaled values are scaled using sensor range, error values are scaled depending of IP_ScaleErrorValues.

The values of selected segments are filled in the arrays one after another. Each array always starts with first selected segment.

12.2.1 General commands

12.2.1.1 General

12.2.1.1.1 Get_Info (INFO)

Retrieve some information about the sensor.

Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the sensor.	
Parameter: String SA_Option	SA_Option
Direction: Up	
Valid values:	
Numeric value	
Description: Option of the sensor.	
Parameter: double SA_Range	SA_Range
Direction: Up	
Unit: mm	
Valid values:	
34.0	
Description: Range of the sensor.	
Parameter: int SA_Reserve_1	SA_Reserve_1
Direction: Up	
Valid values:	
0	
Description: Reserved.	
Parameter: String SA_SoftArtBoot	SA_SoftArtBoot
Direction: Up	
Description: Sensor software versions.	
Parameter: String SA_SoftArtArm	SA_SoftArtArm
Direction: Up	
Description: Sensor software versions.	
Parameter: String SA_SoftArtDSP	SA_SoftArtDSP
Direction: Up	
Description: Sensor software versions.	

Parameter: int SA_SoftVerBoot	SA_SoftVerBoot
Direction: Up	
Valid values:	
Numeric value	
Description: Sensor software versions.	
Parameter: int SA_SoftVerArm	SA_SoftVerArm
Direction: Up	
Valid values:	
Numeric value	
Description: Sensor software versions.	
Parameter: int SA_SoftVerDSP	SA_SoftVerDSP
Direction: Up	
Valid values:	
Numeric value	
Description: Sensor software versions.	

12.2.1.1.2 Reset_Boot (RESET)

Resets the sensor. This command has no parameters.

12.2.1.2 Parameter management

12.2.1.2.1 Save_OptionData (SAVE_OPT_RAM_TO_FLASH)

Save the option data to flash.

12.2.1.2.2 Save_MeasProgData (SAVE_MPR_RAM_TO_FLASH)

Save the measure program data to flash.

12.2.2 Measurement

12.2.2.1 Choose_MeasProg (CHOOSE_MP)

Select the measurement program of sensor.

Parameter: int SP_MeasProgNumber	SP_MeasProgNumber
Direction: Down	
Valid values:	
0= EdgeHL	
1= EdgeLH	
2= DIA	
3= GAP	
4= SEG_2_4	
5= 2-SEG	
6= USER1	
7= USER2	
8= USER3	
9= USER4	
Description: Measure program of sensor.	

12.2.2.2 Switch_Edge (SWITCH_EDGE)

Select the edges to measure.

Parameter: int SP_FrontEdge_Seg1	SP_FrontEdge_Seg1
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 32	
Description: Front edge of segment 1.	
Parameter: int SP_FrontEdge_Seg2	SP_FrontEdge_Seg2
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 32	
Description: Front edge of segment 2.	
Parameter: int SP_RearEdge_Seg1	SP_RearEdge_Seg1
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 32	
Description: Rear edge of segment 1.	
Parameter: int SP_RearEdge_Seg2	SP_RearEdge_Seg2
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 32	
Description: Rear edge of segment 2.	

12.2.2.3 Write_OptionData (WR_OPT_TO_RAM)

Write the option data to sensor.

Parameter: int SP_MeasProgNumber	SP_MeasProgNumber
Direction: Down	
Valid values:	
0= EdgeHL	
1= EdgeLH	
2= DIA	
3= GAP	
4= SEG_2_4	
5= 2-SEG	
6= USER1	
7= USER2	
8= USER3	
9= USER4	
Description: Measure program of sensor.	

Parameter: int SP_Language SP_Language

Direction: Down

Valid values:

0= German

1= English

Description: Language of sensor.

Parameter: int SP_DisMeasUnit SP_DisMeasUnit

Direction: Down

Valid values:

0= mm

1= inch

Description: Display measurement unit of sensor.

Parameter: int SP_ErrorHandler SP_ErrorHandler

Direction: Down

Valid values:

0= error output

1= retain last value

Description: If the sensor cannot measure values, it can output the last valid value or it can output an error values.

Parameter: int SP_Reserve_2 SP_Reserve_2

Direction: Down

Valid values:

Minimum: 0

Maximum: 65535

Description: Reserved.

Parameter: int SP_Ext_LaserSwitch SP_Ext_LaserSwitch

Direction: Down

Valid values:

0= not active

1= active

Description: Enable/Disable external laser switch.

Parameter: int SP_LaserIntensity SP_LaserIntensity

Direction: Down

Valid values:

Minimum: 0

Maximum: 100

Unit: %

Description: No effect.

Parameter: int SP_Contrast SP_Contrast

Direction: Down

Valid values:

Minimum: 0

Maximum: 100

Unit: %

Description: Display contrast.

Parameter: int SP_Reserve_3	SP_Reserve_3
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: Reserved.	
Parameter: int SP_ActiveSerialIf	SP_ActiveSerialIf
Direction: Down	
Valid values:	
0 = RS422	
1 = RS232	
Description: Active serial interface of sensor.	
Parameter: int SP_RS232_Baudrate	SP_RS232_Baudrate
Direction: Down	
Valid values:	
9600	
19200	
38400	
115200	
Unit: Baud	
Description: Baudrate of RS232 interface of sensor.	
Parameter: int SP_RS232_Parity	SP_RS232_Parity
Direction: Down	
Valid values:	
0 = none	
1 = even	
2 = odd	
Description: Parity of RS232 interface of sensor.	
Parameter: int SP_RS232_StopBits	SP_RS232_StopBits
Direction: Down	
Valid values:	
1	
2	
Description: Stop bits of RS232 interface of sensor.	
Parameter: int SP_RS232_TimeoutSend	SP_RS232_TimeoutSend
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	
Parameter: int SP_RS232_TimeoutRecv	SP_RS232_TimeoutRecv
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	

Parameter: int SP_RS422_Baudrate	SP_RS422_Baudrate
Direction: Down	
Valid values:	
9600	
19200	
38400	
115200	
691200	
Unit: Baud	
Description: Baudrate of RS422 interface of sensor.	
Parameter: int SP_RS422_Parity	SP_RS422_Parity
Direction: Down	
Valid values:	
0= none	
1= even	
2= odd	
Description: Parity of RS422 interface of sensor.	
Parameter: int SP_RS422_StopBits	SP_RS422_StopBits
Direction: Down	
Valid values:	
1	
2	
Description: Stop bits of RS422 interface of sensor.	
Parameter: int SP_RS422_TimeoutSend	SP_RS422_TimeoutSend
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	
Parameter: int SP_RS422_TimeoutRecv	SP_RS422_TimeoutRecv
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	

12.2.2.4 Read_OptionData (RD_OPT_RAM)

Read the option data from sensor.

Parameter: int SA_MeasProgNumber	SA_MeasProgNumber
Direction: Up	
Valid values:	
0= EdgeHL	
1= EdgeLH	
2= DIA	
3= GAP	
4= SEG_2_4	
5= 2-SEG	
6= USER1	
7= USER2	
8= USER3	
9= USER4	
Description: Measure program of sensor.	

Parameter: int SA_Language SA_Language

Direction: Up

Valid values:

0= German

1= English

Description: Language of sensor.

Parameter: int SA_DisMeasUnit SA_DisMeasUnit

Direction: Up

Valid values:

0= mm

1= inch

Description: Display measurement unit of sensor.

Parameter: int SA_ErrorHandler SA_ErrorHandler

Direction: Up

Valid values:

0= error output

1= retain last value

Description: If the sensor cannot measure values, it can output the last valid value or it can output an error values.

Parameter: int SA_Reserve_2 SA_Reserve_2

Direction: Up

Valid values:

Minimum: 0

Maximum: 65535

Description: Reserved.

Parameter: int SA_Ext_LaserSwitch SA_Ext_LaserSwitch

Direction: Up

Valid values:

0= not active

1= active

Description: Enable/Disable external laser switch.

Parameter: int SA_LaserIntensity SA_LaserIntensity

Direction: Up

Valid values:

Minimum: 0

Maximum: 100

Unit: %

Description: No effect.

Parameter: int SA_Contrast SA_Contrast

Direction: Up

Valid values:

Minimum: 0

Maximum: 100

Unit: %

Description: Display contrast.

Parameter: int SA_Reserve_3 SA_Reserve_3

Direction: Up

Valid values:

Minimum: 0

Maximum: 65535

Description: Reserved.

Parameter: int SA_ActiveSerialIf SA_ActiveSerialIf

Direction: Up

Valid values:

0 = RS422

1 = RS232

Description: Active serial interface of sensor.

Parameter: int SA_RS232_Baudrate SA_RS232_Baudrate

Direction: Up

Valid values:

9600

19200

38400

115200

Unit: Baud

Description: Baudrate of RS232 interface of sensor.

Parameter: int SA_RS232_Parity SA_RS232_Parity

Direction: Up

Valid values:

0 = none

1 = even

2 = odd

Description: Parity of RS232 interface of sensor.

Parameter: int SA_RS232_StopBits SA_RS232_StopBits

Direction: Up

Valid values:

1

2

Description: Stop bits of RS232 interface of sensor.

Parameter: int SA_RS232_TimeoutSend SA_RS232_TimeoutSend

Direction: Up

Valid values:

Minimum: 0

Maximum: 65535

Description: No effect.

Parameter: int SA_RS232_TimeoutRecv SA_RS232_TimeoutRecv

Direction: Up

Valid values:

Minimum: 0

Maximum: 65535

Description: No effect.

Parameter: int SA_RS422_Baudrate	SA_RS422_Baudrate
Direction: Up	
Valid values:	
9600	
19200	
38400	
115200	
691200	
Unit: Baud	
Description: Baudrate of RS422 interface of sensor.	
Parameter: int SA_RS422_Parity	SA_RS422_Parity
Direction: Up	
Valid values:	
0= none	
1= even	
2= odd	
Description: Parity of RS422 interface of sensor.	
Parameter: int SA_RS422_StopBits	SA_RS422_StopBits
Direction: Up	
Valid values:	
1	
2	
Description: Stop bits of RS422 interface of sensor.	
Parameter: int SA_RS422_TimeoutSend	SA_RS422_TimeoutSend
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	
Parameter: int SA_RS422_TimeoutRecv	SA_RS422_TimeoutRecv
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	

12.2.2.5 Update_OptionData

Update option data at sensor.

This is a meta command which first calls [Read_OptionData](#), renames all answer parameters (SA_...) to command parameters (SP_...), overwrites the parameter set by the parameters from user and finally calls [Write_OptionData](#).

The parameters of this command are NOT obligatory. Missing parameters are not overwritten, so the sensor keeps the original values.

Parameter: int SP_MeasProgNumber	SP_MeasProgNumber
Direction: Down	
Valid values:	
0= EdgeHL	

1= EdgeLH
 2= DIA
 3= GAP
 4= SEG_2_4
 5= 2-SEG
 6= USER1
 7= USER2
 8= USER3
 9= USER4

Description: Measure program of sensor.

Parameter: int SP_Language

SP_Language

Direction: Down

Valid values:

0= German
1= English

Description: Language of sensor.

Parameter: int SP_DisMeasUnit

SP_DisMeasUnit

Direction: Down

Valid values:

0= mm
1= inch

Description: Display measurement unit of sensor.

Parameter: int SP_ErrorHandler

SP_ErrorHandler

Direction: Down

Valid values:

0= error output
1= retain last value

Description: If the sensor cannot measure values, it can output the last valid value or it can output an error values.

Parameter: int SP_Reserve_2

SP_Reserve_2

Direction: Down

Valid values:

Minimum: 0
Maximum: 65535

Description: Reserved.

Parameter: int SP_Ext_LaserSwitch

SP_Ext_LaserSwitch

Direction: Down

Valid values:

0= not active
1= active

Description: Enable/Disable external laser switch.

Parameter: int SP_LaserIntensity

SP_LaserIntensity

Direction: Down

Valid values:

Minimum: 0
Maximum: 100

Unit: %

Description: No effect.

Parameter: int SP_Contrast	SP_Contrast
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 100	
Unit: %	
Description: Display contrast.	
Parameter: int SP_Reserve_3	SP_Reserve_3
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: Reserved.	
Parameter: int SP_ActiveSerialIf	SP_ActiveSerialIf
Direction: Down	
Valid values:	
0 = RS422	
1 = RS232	
Description: Active serial interface of sensor.	
Parameter: int SP_RS232_Baudrate	SP_RS232_Baudrate
Direction: Down	
Valid values:	
9600	
19200	
38400	
115200	
Unit: Baud	
Description: Baudrate of RS232 interface of sensor.	
Parameter: int SP_RS232_Parity	SP_RS232_Parity
Direction: Down	
Valid values:	
0 = none	
1 = even	
2 = odd	
Description: Parity of RS232 interface of sensor.	
Parameter: int SP_RS232_StopBits	SP_RS232_StopBits
Direction: Down	
Valid values:	
1	
2	
Description: Stop bits of RS232 interface of sensor.	
Parameter: int SP_RS232_TimeoutSend	SP_RS232_TimeoutSend
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	

Parameter: int SP_RS232_TimeoutRecv	SP_RS232_TimeoutRecv
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	
Parameter: int SP_RS422_Baudrate	SP_RS422_Baudrate
Direction: Down	
Valid values:	
9600	
19200	
38400	
115200	
691200	
Unit: Baud	
Description: Baudrate of RS422 interface of sensor.	
Parameter: int SP_RS422_Parity	SP_RS422_Parity
Direction: Down	
Valid values:	
0= none	
1= even	
2= odd	
Description: Parity of RS422 interface of sensor.	
Parameter: int SP_RS422_StopBits	SP_RS422_StopBits
Direction: Down	
Valid values:	
1	
2	
Description: Stop bits of RS422 interface of sensor.	
Parameter: int SP_RS422_TimeoutSend	SP_RS422_TimeoutSend
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	
Parameter: int SP_RS422_TimeoutRecv	SP_RS422_TimeoutRecv
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	

12.2.2.6 Write_MeasProgData (WR_MPR_TO_RAM)

Write the measurement program data to sensor.

Parameter: int SP_UserMeasProgNumber	SP_UserMeasProgNumber
Direction: Down	
Valid values:	
6= USER1	
7= USER2	
8= USER3	
9= USER4	
Description: User measure program of sensor.	

Parameter: String SP_MeasProgName	SP_MeasProgName
Direction: Down	
Description: Measure program name of sensor.	
Parameter: double SP_AnalogOffset	SP_AnalogOffset
Direction: Down	
Valid values:	
Minimum: -10.0	
Maximum: 10.0	
Unit: V	
Description: Analog output offset of sensor.	
Parameter: double SP_AnalogGain	SP_AnalogGain
Direction: Down	
Valid values:	
Minimum: -3.4	
Maximum: 3.4	
Description: Analog output gain of sensor.	
Parameter: double SP_DisplayOffset	SP_DisplayOffset
Direction: Down	
Valid values:	
Minimum: -99.99	
Maximum: 99.99	
Unit: mm	
Description: Display offset of sensor.	
Parameter: double SP_DisplayGain	SP_DisplayGain
Direction: Down	
Valid values:	
Minimum: -2.0	
Maximum: 2.0	
Description: Display gain of sensor.	
Parameter: double SP_UpperLimit	SP_UpperLimit
Direction: Down	
Valid values:	
Minimum: -168.876	
Maximum: 168.876	
Unit: mm	
Description: Upper limit of sensor.	
Parameter: double SP_LowerLimit	SP_LowerLimit
Direction: Down	
Valid values:	
Minimum: -168.876	
Maximum: 168.876	
Unit: mm	
Description: Lower limit of sensor.	
Parameter: double SP_UpperWarning	SP_UpperWarning
Direction: Down	
Valid values:	
Minimum: -168.876	
Maximum: 168.876	
Unit: mm	
Description: Upper warning of sensor.	

Parameter: double SP_LowerWarning	SP_LowerWarning
Direction: Down	
Valid values:	
Minimum: -168.876	
Maximum: 168.876	
Unit: mm	
Description: Lower warning of sensor.	
Parameter: int SP_Target_Distance	SP_Target_Distance
Direction: Down	
Valid values:	
0= 20 mm	
1= 50 mm	
2= 100 mm	
3= 150 mm	
Description: Target distance of sensor.	
Parameter: int SP_Average_for_reading	SP_Average_for_reading
Direction: Down	
Valid values:	
1 to 128 sliding	
129 to 4096 recursive	
Description: Averaging mode and number of sensor.	
Parameter: int SP_Reserve_4	SP_Reserve_4
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: Reserved.	
Parameter: int SP_MeasObject	SP_MeasObject
Direction: Down	
Valid values:	
1= EdgeHL	
2= EdgeLH	
3= DIA	
4= GAP	
5= SEG_2_4	
6= 2-SEG	
Description: Measurement program.	
Parameter: int SP_FrontEdge_Seg1	SP_FrontEdge_Seg1
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 32	
Description: Front edge of segment 1.	
Parameter: int SP_FrontEdge_Seg2	SP_FrontEdge_Seg2
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 32	
Description: Front edge of segment 2.	

Parameter: int SP_RearEdge_Seg1	SP_RearEdge_Seg1
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 32	
Description: Rear edge of segment 1.	
Parameter: int SP_RearEdge_Seg2	SP_RearEdge_Seg2
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 32	
Description: Rear edge of segment 2.	
Parameter: double SP_MasterValue	SP_MasterValue
Direction: Down	
Valid values:	
Minimum: -34.0	
Maximum: 34.0	
Unit: mm	
Description: Master value of sensor.	

12.2.2.7 Read_MeasProgData (RD_MPR_RAM)

Read the measurement program data from sensor.

Parameter: int SA_MeasProgNumber	SA_MeasProgNumber
Direction: Up	
Valid values:	
0= EdgeHL	
1= EdgeLH	
2= DIA	
3= GAP	
4= SEG 2_4	
5= 2-SEG	
6= USER1	
7= USER2	
8= USER3	
9= USER4	
Description: User measure program of sensor.	
Parameter: String SA_MeasProgName	SA_MeasProgName
Direction: Up	
Description: Measure program name of sensor.	
Parameter: double SA_AnalogOffset	SA_AnalogOffset
Direction: Up	
Valid values:	
Minimum: -10.0	
Maximum: 10.0	
Unit: V	
Description: Analog output offset of sensor.	

Parameter: double SA_AnalogGain	SA_AnalogGain
Direction: Up	
Valid values:	
Minimum: -3.4	
Maximum: 3.4	
Description: Analog output gain of sensor.	
Parameter: double SA_DisplayOffset	SA_DisplayOffset
Direction: Up	
Valid values:	
Minimum: -99.99	
Maximum: 99.99	
Unit: mm	
Description: Display offset of sensor.	
Parameter: double SA_DisplayGain	SA_DisplayGain
Direction: Up	
Valid values:	
Minimum: -2.0	
Maximum: 2.0	
Description: Display gain of sensor.	
Parameter: double SA_UpperLimit	SA_UpperLimit
Direction: Up	
Valid values:	
Minimum: -168.876	
Maximum: 168.876	
Unit: mm	
Description: Upper limit of sensor.	
Parameter: double SA_LowerLimit	SA_LowerLimit
Direction: Up	
Valid values:	
Minimum: -168.876	
Maximum: 168.876	
Unit: mm	
Description: Lower limit of sensor.	
Parameter: double SA_UpperWarning	SA_UpperWarning
Direction: Up	
Valid values:	
Minimum: -168.876	
Maximum: 168.876	
Unit: mm	
Description: Upper warning of sensor.	
Parameter: double SA_LowerWarning	SA_LowerWarning
Direction: Up	
Valid values:	
Minimum: -168.876	
Maximum: 168.876	
Unit: mm	
Description: Lower warning of sensor.	

Parameter: int SA_Target_Distance	SA_Target_Distance
Direction: Up	
Valid values:	
0= 20 mm	
1= 50 mm	
2= 100 mm	
3= 150 mm	
Description: Target distance of sensor.	
Parameter: int SA_Average_for_reading	SA_Average_for_reading
Direction: Up	
Valid values:	
1 to 128 sliding	
129 to 4096 recursive	
Description: Averaging mode and number of sensor.	
Parameter: int SA_Reserve_4	SA_Reserve_4
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: Reserved.	
Parameter: int SA_MeasObject	SA_MeasObject
Direction: Up	
Valid values:	
1= EdgeHL	
2= EdgeLH	
3= DIA	
4= GAP	
5= SEG_2_4	
6= 2-SEG	
Description: Measurement program.	
Parameter: int SA_FrontEdge_Seg1	SA_FrontEdge_Seg1
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 32	
Description: Front edge of segment 1.	
Parameter: int SA_FrontEdge_Seg2	SA_FrontEdge_Seg2
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 32	
Description: Front edge of segment 2.	
Parameter: int SA_RearEdge_Seg1	SA_RearEdge_Seg1
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 32	
Description: Rear edge of segment 1.	

Parameter: int SA_RearEdge_Seg2 SA_RearEdge_Seg2

Direction: Up

Valid values:

Minimum: 0

Maximum: 32

Description: Rear edge of segment 2.

Parameter: double SA_MasterValue SA_MasterValue

Direction: Up

Valid values:

Minimum: -34.0

Maximum: 34.0

Unit: mm

Description: Master value of sensor.

12.2.2.8 Read_MinMax (RD_MINMAX)

Read the minimum and maximum values from sensor.

Parameter: int SA_MinRaw SA_MinRaw

Direction: Up

Valid values:

Minimum: 0

Maximum: 65535

Description: Minimum raw value.

Parameter: int SA_MaxRaw SA_MaxRaw

Direction: Up

Valid values:

Minimum: 0

Maximum: 65535

Description: Maximum raw value.

Parameter: double SA_MinScaled SA_MinScaled

Direction: Up

Valid values:

Minimum: 0.0

Maximum: 34.0

Unit: mm

Description: Minimum scaled value.

Parameter: double SA_MaxScaled SA_MaxScaled

Direction: Up

Valid values:

Minimum: 0.0

Maximum: 34.0

Unit: mm

Description: Maximum scaled value.

12.2.2.9 Read_MinMaxReset (RD_MINMAX_RESET)

Read the minimum and maximum values from sensor and reset the values.

Parameter: int SA_MinRaw SA_MinRaw

Direction: Up

Valid values:

Minimum: 0

Maximum: 65535

Description: Minimum raw value.

Parameter: int SA_MaxRaw SA_MaxRaw

Direction: Up

Valid values:

Minimum: 0

Maximum: 65535

Description: Maximum raw value.

Parameter: double SA_MinScaled SA_MinScaled

Direction: Up

Valid values:

Minimum: 0.0

Maximum: 34.0

Unit: mm

Description: Minimum scaled value.

Parameter: double SA_MaxScaled SA_MaxScaled

Direction: Up

Valid values:

Minimum: 0.0

Maximum: 34.0

Unit: mm

Description: Maximum scaled value.

12.2.3 Data output

12.2.3.1 General

12.2.3.1.1 Dat_Out_Off (STOP)

Switch off data output from sensor.

12.2.3.1.2 Dat_Out_On (START)

Switch on data output from sensor.

12.3 Commands for ODC2520

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [RS232](#) (additional, e.g. IF2001_USB (RS422) and RS232 high level interface).
- [IF2004](#) (native).
- [TCP/IP](#) (native).
- [IF2004_USB](#) (native).
- [IF2008](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_AllParameters](#) (SP_Additional= 1) after [OpenSensor](#). Otherwise, you have to call it manually to allow MEDAQLib timeout checking, to calculate datarate, to interpret and scale data and to assign values. If second bit of [IP_AutomaticMode](#) is set (2), MEDAQLib calls optionally sensor command [Set_IPDataTransferMode](#) and [Set_DataOutInterface](#) at [OpenSensor](#).

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor.
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command [Get_AllParameters](#) (SP_Additional= 1)).

The values of one data frame are filled in the arrays one after another. Each array always starts with a new data frame.

12.3.1 General commands

12.3.1.1 General

12.3.1.1.1 Get_Help (HELP)

Retrieve a help text from controller for a specific command.

Parameter: String SP_Command SP_Command
Direction: Down
Valid values:
 " " (empty string, means general help)
 or any command name
Description: Name of the command.

Parameter: String SA_HelpText SA_HelpText
Direction: Up
Description: Help text to the command.

12.3.1.1.2 Get_Info (GETINFO)

Retrieve information about the controller.

Parameter: String SA_Sensor	SA_Sensor
Direction: Up	
Description: Name of the controller.	
Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the controller.	
Parameter: String SA_Option	SA_Option
Direction: Up	
Valid values:	
Numeric value	
Description: Option of the controller.	
Parameter: String SA_ArticleNumber	SA_ArticleNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Article number of the controller.	
Parameter: String SA_MacAddress	SA_MacAddress
Direction: Up	
Valid values:	
Valid MAC address in form of xx-xx-xx-xx-xx-xx	
Description: MAC address (low level ethernet address) of the controller.	
Parameter: String SA_Softwareversion	SA_Softwareversion
Direction: Up	
Description: Software version of firmware in the controller.	
Parameter: String SA_Image	SA_Image
Direction: Up	
Description: Active software, ethernet or ethercat.	
Parameter: String SA_ImageType	SA_ImageType
Direction: Up	
Description: Firmware image type used by the controller.	
Parameter: String SA_HardwareRevision	SA_HardwareRevision
Direction: Up	
Valid values:	
Numeric value	
Description: Revision index of the controller board.	

12.3.1.1.3 Set_Echo (ECHO)

Set echo for sensor commands.

Parameter: int SP_Echo SP_Echo
Direction: Down
Valid values:
 0= Off
 1= On
Description: Echo mode.

12.3.1.1.4 Get_Echo (ECHO)

Get the echo mode.

Parameter: int SA_Echo SA_Echo
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= Off
 1= On
Description: Echo mode.

12.3.1.1.5 Get_AllParameters (PRINT)

Get all parameters from controller.

Parameter: int SP_Additional SP_Additional
Direction: Down
Valid values:
 0= No
 1= Yes
Description: If set, additional information about controller, sensor and material is output.

Parameter: int SA_SyncMode SA_SyncMode
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= None
 1= Slave
 2= Master
Description: Synchronisation mode.

Parameter: int SA_SyncTermination SA_SyncTermination
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= Off (TERMOFF)
 1= On 120 Ohm (TERMON)
Description: Termination of external input.

Parameter: int SA_UserLevel	SA_UserLevel
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = User	
1 = Professional	
Description: Actual user level.	
Parameter: int SA_DefaultUser	SA_DefaultUser
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = User	
1 = Professional	
Description: Default user level.	
Parameter: int SA_Echo	SA_Echo
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Off	
1 = On	
Description: Echo mode.	
Parameter: int SA_DisplayUnit	SA_DisplayUnit
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = mm	
1 = Inch	
Description: Unit.	
Parameter: int SA_TriggerMode	SA_TriggerMode
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = None	
1 = Edge	
2 = Level (PULSE)	
3 = Software	
Description: Trigger mode.	
Parameter: int SA_TriggerTermination	SA_TriggerTermination
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Off (TERMOFF)	
1 = On 120 Ohm (TERMON)	
Description: Termination of external input.	
Parameter: int SA_TriggerMoment	SA_TriggerMoment
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Input	
1 = Output	
Description: Trigger moment.	

Parameter: int SA_TriggerLevel	SA_TriggerLevel
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = High	
1 = Low	
Description: Trigger level.	
Parameter: int SA_TriggerCount	SA_TriggerCount
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 16383	
Description: Number of values to measure. 0 means no trigger, 16383 means endless measurement.	
Parameter: int SA_EthernetMode	SA_EthernetMode
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Ethernet	
1 = Ethercat	
Description: Ethernet mode.	
Parameter: int SA_DHCPEnabled	SA_DHCPEnabled
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = FALSE	
1 = TRUE	
Description: Get settings if controller should use a static IP address ask for IP at DHCP server (dynamic IP address).	
Parameter: String SA_Address	SA_Address
Direction: Up	
Valid values:	
Valid IP address in form of xxx.xxx.xxx.xxx	
Description: IP address of the controller. If DHCP is enabled it returns the currently assigned IP address.	
Parameter: String SA_SubnetMask	SA_SubnetMask
Direction: Up	
Valid values:	
Valid network mask (e.g. 255.255.255.0 for a Class C network)	
Description: Network mask of the controller. If DHCP is enabled it returns the currently assigned network mask.	
Parameter: String SA_Gateway	SA_Gateway
Direction: Up	
Valid values:	
Valid IP address of default gateway in form of xxx.xxx.xxx.xxx	
Description: Address of the default gateway. If DHCP is enabled it returns the currently assigned default gateway.	

Parameter: int SA_Protocol SA_Protocol

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= TCP server (SERVER/TCP)
- 1= TCP client (CLIENT/TCP)
- 2= UDP sender (CLIENT/UDP)
- 3= None

Description: Specifies if data should be send using TCP or UDP.

Parameter: String SA_RemoteAddress SA_RemoteAddress

Direction: Up

Valid values:

Valid IP address of receiver of data

Description: Address of remote computer to send data to.

Parameter: int SA_Port SA_Port

Direction: Up

Valid values:

- Minimum: 1024
- Maximum: 65535

Description: Port to send data to or to listen for incoming requests.

Parameter: int SA_SensorBaudrate SA_SensorBaudrate

Direction: Up

Valid values:

- 4000000
- 3500000
- 3000000
- 2500000
- 2000000
- 1500000
- 921600
- 691200
- 460800
- 230400
- 115200
- 9600

Unit: Baud

Description: Baudrate of controller.

Parameter: int SA_MeasureMode SA_MeasureMode

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Edge Light-Dark (EDGEHL)
- 1= Edge Dark-Light (EDGELH)
- 2= Diameter (DIA)
- 3= Gap
- 4= Segments (SEGMENT)

Description: Measure mode.

Parameter: int SA_SearchDirection	SA_SearchDirection
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Standard	
1 = Inverse	
Description: Search direction for edges.	
Parameter: int SA_MeasureDirection	SA_MeasureDirection
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Standard	
1 = Inverse	
Description: Reaference point.	
Parameter: int SA_ExpectedEdges	SA_ExpectedEdges
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 64	
Description: Number of expected edges.	
Parameter: int SA_Edge1_A	SA_Edge1_A
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 64	
Description: First edge.	
Parameter: int SA_Edge1_B	SA_Edge1_B
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 64	
Description: Second edge.	
Parameter: int SA_Edge2_A	SA_Edge2_A
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 64	
Description: First edge.	
Parameter: int SA_Edge2_B	SA_Edge2_B
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 64	
Description: Second edge.	
Parameter: int SA_Edge3_A	SA_Edge3_A
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 64	
Description: First edge.	

Parameter: int SA_Edge3_B	SA_Edge3_B
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 64	
Description: Second edge.	
Parameter: int SA_Edge4_A	SA_Edge4_A
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 64	
Description: First edge.	
Parameter: int SA_Edge4_B	SA_Edge4_B
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 64	
Description: Second edge.	
Parameter: int SA_Edge5_A	SA_Edge5_A
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 64	
Description: First edge.	
Parameter: int SA_Edge5_B	SA_Edge5_B
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 64	
Description: Second edge.	
Parameter: int SA_Edge6_A	SA_Edge6_A
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 64	
Description: First edge.	
Parameter: int SA_Edge6_B	SA_Edge6_B
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 64	
Description: Second edge.	
Parameter: int SA_Edge7_A	SA_Edge7_A
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 64	
Description: First edge.	

Parameter: int SA_Edge7_B	SA_Edge7_B
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 64	
Description: Second edge.	
Parameter: int SA_Edge8_A	SA_Edge8_A
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 64	
Description: First edge.	
Parameter: int SA_Edge8_B	SA_Edge8_B
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 64	
Description: Second edge.	
Parameter: double SA_Threshold	SA_Threshold
Direction: Up	
Valid values:	
Minimum: 1.0	
Maximum: 99.0	
Unit: %	
Description: Video threshold.	
Parameter: int SA_ROIStart	SA_ROIStart
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 767	
Unit: Pixel	
Description: First position on CCD.	
Parameter: int SA_ROIEnd	SA_ROIEnd
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 767	
Unit: Pixel	
Description: Last position on CCD.	
Parameter: int SA_AveragingType	SA_AveragingType
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = None	
1 = Moving average (MOVING)	
2 = Recursive averaging (RECURSIVE)	
3 = Median	
Description: Averaging type.	

Parameter: int SA_MovingCount SA_MovingCount

Direction: Up

Valid values:

- 2
- 4
- 8
- 16
- 32
- 64
- 128

Description: Number of value for the averaging window. This parameter is only available at moving average.

Parameter: int SA_RecursiveCount SA_RecursiveCount

Direction: Up

Valid values:

- Minimum:** 2
- Maximum:** 32768

Description: Number of values for recursive averaging. This parameter is only available at recursive average.

Parameter: int SA_MedianCount SA_MedianCount

Direction: Up

Valid values:

- 3
- 5
- 7
- 9

Description: Number of values to build median. This parameter is only available at median.

Parameter: int SA_SpikeCorrection SA_SpikeCorrection

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = off
- 1 = on

Description: Spike correction.

Parameter: int SA_NbrEvaluatedValues SA_NbrEvaluatedValues

Direction: Up

Valid values:

- Minimum:** 1
- Maximum:** 10

Description: Number of values to evaluate for spike correction.

Parameter: double SA_ToleranceRange SA_ToleranceRange

Direction: Up

Valid values:

- Minimum:** 0.0
- Maximum:** 100.0

Description: Tolerance range for spike correction.

Parameter: int SA_NbrCorrectedValues	SA_NbrCorrectedValues
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 100	
Description: Number of values to correct at spike correction.	
Parameter: int SA_StatisticSignal	SA_StatisticSignal
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Edge, Light-Dark (EHL)	
1= Edge, Dark-Light (ELH)	
2= Diameter, Edge A (DA)	
3= Diameter, Edge B (DB)	
4= Diameter, Difference (DD)	
5= Diameter, Center axis (DC)	
6= Gap, Edge A (GA)	
7= Gap, Edge B (GB)	
8= Gap, Difference (GD)	
9= Gap, Center axis (GC)	
10= Segment 1, Edge A (S1A)	
11= Segment 1, Edge B (S1B)	
12= Segment 1, Difference (S1D)	
13= Segment 1, Center axis (S1C)	
14= Segment 2, Edge A (S2A)	
15= Segment 2, Edge B (S2B)	
16= Segment 2, Difference (S2D)	
17= Segment 2, Center axis(S2C)	
18= Segment 3, Edge A (S3A)	
19= Segment 3, Edge B (S3B)	
20= Segment 3, Difference (S3D)	
21= Segment 3, Center axis (S3C)	
22= Segment 4, Edge A (S4A)	
23= Segment 4, Edge B (S4B)	
24= Segment 4, Difference (S4D)	
25= Segment 4, Center axis (S4C)	
26= Segment 5, Edge A (S5A)	
27= Segment 5, Edge B (S5B)	
28= Segment 5, Difference (S5D)	
29= Segment 5, Center axis (S5C)	
30= Segment 6, Edge A (S6A)	
31= Segment 6, Edge B (S6B)	
32= Segment 6, Difference (S6D)	
33= Segment 6, Center axis (S6C)	
34= Segment 7, Edge A (S7A)	
35= Segment 7, Edge B (S7B)	
36= Segment 7, Difference (S7D)	
37= Segment 7, Center axis (S7C)	
38= Segment 8, Edge A (S8A)	
39= Segment 8, Edge B (S8B)	
40= Segment 8, Difference (S8D)	
41= Segment 8, Center axis (S8C)	
Description: Value which is used for statistic calculation.	

Parameter: int SA_Statistic2Signal SA_Statistic2Signal

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Edge, Light-Dark (EHL)
- 1= Edge, Dark-Light (ELH)
- 2= Diameter, Edge A (DA)
- 3= Diameter, Edge B (DB)
- 4= Diameter, Difference (DD)
- 5= Diameter, Center axis (DC)
- 6= Gap, Edge A (GA)
- 7= Gap, Edge B (GB)
- 8= Gap, Difference (GD)
- 9= Gap, Center axis (GC)
- 10= Segment 1, Edge A (S1A)
- 11= Segment 1, Edge B (S1B)
- 12= Segment 1, Difference (S1D)
- 13= Segment 1, Center axis (S1C)
- 14= Segment 2, Edge A (S2A)
- 15= Segment 2, Edge B (S2B)
- 16= Segment 2, Difference (S2D)
- 17= Segment 2, Center axis (S2C)
- 18= Segment 3, Edge A (S3A)
- 19= Segment 3, Edge B (S3B)
- 20= Segment 3, Difference (S3D)
- 21= Segment 3, Center axis (S3C)
- 22= Segment 4, Edge A (S4A)
- 23= Segment 4, Edge B (S4B)
- 24= Segment 4, Difference (S4D)
- 25= Segment 4, Center axis (S4C)
- 26= Segment 5, Edge A (S5A)
- 27= Segment 5, Edge B (S5B)
- 28= Segment 5, Difference (S5D)
- 29= Segment 5, Center axis (S5C)
- 30= Segment 6, Edge A (S6A)
- 31= Segment 6, Edge B (S6B)
- 32= Segment 6, Difference (S6D)
- 33= Segment 6, Center axis (S6C)
- 34= Segment 7, Edge A (S7A)
- 35= Segment 7, Edge B (S7B)
- 36= Segment 7, Difference (S7D)
- 37= Segment 7, Center axis (S7C)
- 38= Segment 8, Edge A (S8A)
- 39= Segment 8, Edge B (S8B)
- 40= Segment 8, Difference (S8D)
- 41= Segment 8, Center axis (S8C)

Description: Value which is used for second statistic calculation.

Parameter: int SA_StatisticDepth SA_StatisticDepth

Direction: Up

Valid values:

Minimum: 2

Maximum: 2147483647 (INT_MAX)

Description: Window size for floating statistic calculation. The value must be power of two (2, 4, 8, ..., 8192). Value greater as 8192 calculates statistic over all values.

Parameter: int SA_MasterSignal

SA_MasterSignal

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Edge, Light-Dark (EHL)
- 1= Edge, Dark-Light (ELH)
- 2= Diameter, Edge A (DA)
- 3= Diameter, Edge B (DB)
- 4= Diameter, Difference (DD)
- 5= Diameter, Center axis (DC)
- 6= Gap, Edge A (GA)
- 7= Gap, Edge B (GB)
- 8= Gap, Difference (GD)
- 9= Gap, Center axis (GC)
- 10= Segment 1, Edge A (S1A)
- 11= Segment 1, Edge B (S1B)
- 12= Segment 1, Difference (S1D)
- 13= Segment 1, Center axis (S1C)
- 14= Segment 2, Edge A (S2A)
- 15= Segment 2, Edge B (S2B)
- 16= Segment 2, Difference (S2D)
- 17= Segment 2, Center axis (S2C)
- 18= Segment 3, Edge A (S3A)
- 19= Segment 3, Edge B (S3B)
- 20= Segment 3, Difference (S3D)
- 21= Segment 3, Center axis (S3C)
- 22= Segment 4, Edge A (S4A)
- 23= Segment 4, Edge B (S4B)
- 24= Segment 4, Difference (S4D)
- 25= Segment 4, Center axis (S4C)
- 26= Segment 5, Edge A (S5A)
- 27= Segment 5, Edge B (S5B)
- 28= Segment 5, Difference (S5D)
- 29= Segment 5, Center axis (S5C)
- 30= Segment 6, Edge A (S6A)
- 31= Segment 6, Edge B (S6B)
- 32= Segment 6, Difference (S6D)
- 33= Segment 6, Center axis (S6C)
- 34= Segment 7, Edge A (S7A)
- 35= Segment 7, Edge B (S7B)
- 36= Segment 7, Difference (S7D)
- 37= Segment 7, Center axis (S7C)
- 38= Segment 8, Edge A (S8A)
- 39= Segment 8, Edge B (S8B)
- 40= Segment 8, Difference (S8D)
- 41= Segment 8, Center axis (S8C)

Description: Value which is used for mastering.

Parameter: int SA_Master

SA_Master

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= no (NONE)
- 1= yes (MASTER=

Description: Specifies if mastering is active.

Parameter: double SA_MasterValue

SA_MasterValue

Direction: Up

Valid values:

- Minimum:** -measuring range
- Maximum:** +measuring range

Unit: mm

Description: Master value

Parameter: int SA_DataOutInterface

SA_DataOutInterface

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= RS422
- 2= Ethernet
- 3= HTTP

Description: Active interface for data output.

Parameter: int SA_Resampling

SA_Resampling

Direction: Up

Valid values:

- Minimum:** 1
- Maximum:** 150000

Description: Resampling value.

Parameter: int SA_ResampleAnalog

SA_ResampleAnalog

Direction: Up

Valid values:

- 0= no
- 1= yes

Description: Analog output is resampled.

Parameter: int SA_ResampleRS422

SA_ResampleRS422

Direction: Up

Valid values:

- 0= no
- 1= yes

Description: RS422 output is resampled.

Parameter: int SA_ResampleEthernet

SA_ResampleEthernet

Direction: Up

Valid values:

- 0= no
- 1= yes

Description: Output over ethernet is resampled.

Parameter: int SA_HoldLastValid	SA_HoldLastValid
Direction: Up	
Valid values:	
Minimum: -1	
Maximum: 1024	
Description: Values to replace by last valid value. -1 means no value to hold, 0 means never output an error value (always hold last valid value).	
Parameter: int SA_OutputEdgeLightDark_RS422	SA_OutputEdgeLightDark_-RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge light-dark is transmitted.	
Parameter: int SA_OutputEdgeLightDark_ETH	SA_OutputEdgeLightDark_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge light-dark is transmitted.	
Parameter: int SA_OutputEdgeDarkLight_RS422	SA_OutputEdgeDarkLight_-RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge dark-light is transmitted.	
Parameter: int SA_OutputEdgeDarkLight_ETH	SA_OutputEdgeDarkLight_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge dark-light is transmitted.	
Parameter: int SA_OutputDiameterEdgeA_RS422	SA_OutputDiameterEdgeA_-RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A (first light-dark edge) is transmitted.	
Parameter: int SA_OutputDiameterEdgeB_RS422	SA_OutputDiameterEdgeB_-RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B (last dark-light edge) is transmitted.	
Parameter: int SA_OutputDiameterDifference_RS422	SA_OutputDiameterDifference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) is transmitted.	

Parameter: int SA_OutputDiameterCenterAxis_RS422	SA_OutputDiameterCenterAxis_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) is transmitted.	
Parameter: int SA_OutputDiameterEdgeA_ETH	SA_OutputDiameterEdgeA_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A (first light-dark edge) is transmitted.	
Parameter: int SA_OutputDiameterEdgeB_ETH	SA_OutputDiameterEdgeB_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B (last dark-light edge) is transmitted.	
Parameter: int SA_OutputDiameterDifference_ETH	SA_OutputDiameterDifference_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) is transmitted.	
Parameter: int SA_OutputDiameterCenterAxis_ETH	SA_OutputDiameterCenterAxis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) is transmitted.	
Parameter: int SA_OutputGapEdgeA_RS422	SA_OutputGapEdgeA_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A (first dark-light edge) is transmitted.	
Parameter: int SA_OutputGapEdgeB_RS422	SA_OutputGapEdgeB_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B (following edge after A) is transmitted.	
Parameter: int SA_OutputGapDifference_RS422	SA_OutputGapDifference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) is transmitted.	

Parameter: int SA_OutputGapCenterAxis_RS422	SA_OutputGapCenterAxis_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) is transmitted.	
Parameter: int SA_OutputGapEdgeA_ETH	SA_OutputGapEdgeA_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A (first dark-light edge) is transmitted.	
Parameter: int SA_OutputGapEdgeB_ETH	SA_OutputGapEdgeB_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B (following edge after A) is transmitted.	
Parameter: int SA_OutputGapDifference_ETH	SA_OutputGapDifference_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) is transmitted.	
Parameter: int SA_OutputGapCenterAxis_ETH	SA_OutputGapCenterAxis_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) is transmitted.	
Parameter: int SA_OutputSegment1EdgeA_RS422	SA_OutputSegment1EdgeA_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 1 is transmitted.	
Parameter: int SA_OutputSegment1EdgeB_RS422	SA_OutputSegment1EdgeB_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 1 is transmitted.	
Parameter: int SA_OutputSegment1Difference_RS422	SA_OutputSeg- ment1Difference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 1 is transmitted.	

Parameter: int SA_OutputSegment1CenterAxis_RS422	SA_OutputSegment1CenterAxis_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 1 is transmitted.	
Parameter: int SA_OutputSegment2EdgeA_RS422	SA_OutputSegment2EdgeA_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 2 is transmitted.	
Parameter: int SA_OutputSegment2EdgeB_RS422	SA_OutputSegment2EdgeB_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 2 is transmitted.	
Parameter: int SA_OutputSegment2Difference_RS422	SA_OutputSegment2Difference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 2 is transmitted.	
Parameter: int SA_OutputSegment2CenterAxis_RS422	SA_OutputSegment2CenterAxis_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 2 is transmitted.	
Parameter: int SA_OutputSegment3EdgeA_RS422	SA_OutputSegment3EdgeA_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 3 is transmitted.	
Parameter: int SA_OutputSegment3EdgeB_RS422	SA_OutputSegment3EdgeB_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 3 is transmitted.	
Parameter: int SA_OutputSegment3Difference_RS422	SA_OutputSegment3Difference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 3 is transmitted.	

Parameter: int SA_OutputSegment3CenterAxis_RS422	SA_OutputSegment3CenterAxis_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 3 is transmitted.	
Parameter: int SA_OutputSegment4EdgeA_RS422	SA_OutputSegment4EdgeA_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 4 is transmitted.	
Parameter: int SA_OutputSegment4EdgeB_RS422	SA_OutputSegment4EdgeB_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 4 is transmitted.	
Parameter: int SA_OutputSegment4Difference_RS422	SA_OutputSegment4Difference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 4 is transmitted.	
Parameter: int SA_OutputSegment4CenterAxis_RS422	SA_OutputSegment4CenterAxis_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 4 is transmitted.	
Parameter: int SA_OutputSegment5EdgeA_RS422	SA_OutputSegment5EdgeA_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 5 is transmitted.	
Parameter: int SA_OutputSegment5EdgeB_RS422	SA_OutputSegment5EdgeB_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 5 is transmitted.	
Parameter: int SA_OutputSegment5Difference_RS422	SA_OutputSegment5Difference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 5 is transmitted.	

Parameter: int SA_OutputSegment5CenterAxis_RS422	SA_OutputSegment5CenterAxis_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 5 is transmitted.	
Parameter: int SA_OutputSegment6EdgeA_RS422	SA_OutputSegment6EdgeA_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 6 is transmitted.	
Parameter: int SA_OutputSegment6EdgeB_RS422	SA_OutputSegment6EdgeB_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 6 is transmitted.	
Parameter: int SA_OutputSegment6Difference_RS422	SA_OutputSegment6Difference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 6 is transmitted.	
Parameter: int SA_OutputSegment6CenterAxis_RS422	SA_OutputSegment6CenterAxis_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 6 is transmitted.	
Parameter: int SA_OutputSegment7EdgeA_RS422	SA_OutputSegment7EdgeA_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 7 is transmitted.	
Parameter: int SA_OutputSegment7EdgeB_RS422	SA_OutputSegment7EdgeB_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 7 is transmitted.	
Parameter: int SA_OutputSegment7Difference_RS422	SA_OutputSegment7Difference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 7 is transmitted.	

Parameter: int SA_OutputSegment7CenterAxis_RS422	SA_OutputSegment7CenterAxis_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 7 is transmitted.	
Parameter: int SA_OutputSegment8EdgeA_RS422	SA_OutputSegment8EdgeA_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 8 is transmitted.	
Parameter: int SA_OutputSegment8EdgeB_RS422	SA_OutputSegment8EdgeB_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 8 is transmitted.	
Parameter: int SA_OutputSegment8Difference_RS422	SA_OutputSegment8Difference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 8 is transmitted.	
Parameter: int SA_OutputSegment8CenterAxis_RS422	SA_OutputSegment8CenterAxis_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 8 is transmitted.	
Parameter: int SA_OutputSegment1EdgeA_ETH	SA_OutputSegment1EdgeA_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 1 is transmitted.	
Parameter: int SA_OutputSegment1EdgeB_ETH	SA_OutputSegment1EdgeB_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 1 is transmitted.	
Parameter: int SA_OutputSegment1Difference_ETH	SA_OutputSegment1Difference_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 1 is transmitted.	

Parameter: int SA_OutputSegment1CenterAxis_ETH	SA_OutputSegment1CenterAxis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 1 is transmitted.	
Parameter: int SA_OutputSegment2EdgeA_ETH	SA_OutputSegment2EdgeA_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 2 is transmitted.	
Parameter: int SA_OutputSegment2EdgeB_ETH	SA_OutputSegment2EdgeB_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 2 is transmitted.	
Parameter: int SA_OutputSegment2Difference_ETH	SA_OutputSegment2Difference_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 2 is transmitted.	
Parameter: int SA_OutputSegment2CenterAxis_ETH	SA_OutputSegment2CenterAxis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 2 is transmitted.	
Parameter: int SA_OutputSegment3EdgeA_ETH	SA_OutputSegment3EdgeA_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 3 is transmitted.	
Parameter: int SA_OutputSegment3EdgeB_ETH	SA_OutputSegment3EdgeB_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 3 is transmitted.	
Parameter: int SA_OutputSegment3Difference_ETH	SA_OutputSegment3Difference_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 3 is transmitted.	

Parameter: int SA_OutputSegment3CenterAxis_ETH	SA_OutputSegment3CenterAxis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 3 is transmitted.	
Parameter: int SA_OutputSegment4EdgeA_ETH	SA_OutputSegment4EdgeA_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 4 is transmitted.	
Parameter: int SA_OutputSegment4EdgeB_ETH	SA_OutputSegment4EdgeB_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 4 is transmitted.	
Parameter: int SA_OutputSegment4Difference_ETH	SA_OutputSegment4Difference_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 4 is transmitted.	
Parameter: int SA_OutputSegment4CenterAxis_ETH	SA_OutputSegment4CenterAxis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 4 is transmitted.	
Parameter: int SA_OutputSegment5EdgeA_ETH	SA_OutputSegment5EdgeA_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 5 is transmitted.	
Parameter: int SA_OutputSegment5EdgeB_ETH	SA_OutputSegment5EdgeB_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 5 is transmitted.	
Parameter: int SA_OutputSegment5Difference_ETH	SA_OutputSegment5Difference_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 5 is transmitted.	

Parameter: int SA_OutputSegment5CenterAxis_ETH	SA_OutputSegment5CenterAxis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 5 is transmitted.	
Parameter: int SA_OutputSegment6EdgeA_ETH	SA_OutputSegment6EdgeA_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 6 is transmitted.	
Parameter: int SA_OutputSegment6EdgeB_ETH	SA_OutputSegment6EdgeB_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 6 is transmitted.	
Parameter: int SA_OutputSegment6Difference_ETH	SA_OutputSegment6Difference_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 6 is transmitted.	
Parameter: int SA_OutputSegment6CenterAxis_ETH	SA_OutputSegment6CenterAxis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 6 is transmitted.	
Parameter: int SA_OutputSegment7EdgeA_ETH	SA_OutputSegment7EdgeA_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 7 is transmitted.	
Parameter: int SA_OutputSegment7EdgeB_ETH	SA_OutputSegment7EdgeB_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 7 is transmitted.	
Parameter: int SA_OutputSegment7Difference_ETH	SA_OutputSegment7Difference_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 7 is transmitted.	

Parameter: int SA_OutputSegment7CenterAxis_ETH	SA_OutputSegment7CenterAxis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 7 is transmitted.	
Parameter: int SA_OutputSegment8EdgeA_ETH	SA_OutputSegment8EdgeA_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 8 is transmitted.	
Parameter: int SA_OutputSegment8EdgeB_ETH	SA_OutputSegment8EdgeB_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 8 is transmitted.	
Parameter: int SA_OutputSegment8Difference_ETH	SA_OutputSegment8Difference_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 8 is transmitted.	
Parameter: int SA_OutputSegment8CenterAxis_ETH	SA_OutputSegment8CenterAxis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 8 is transmitted.	
Parameter: int SA_OutputStatisticMin_RS422	SA_OutputStatisticMin_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statictic min value is transmitted.	
Parameter: int SA_OutputStatisticMax_RS422	SA_OutputStatisticMax_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statictic max value is transmitted.	
Parameter: int SA_OutputStatisticPeak2Peak_RS422	SA_OutputStatisticPeak2Peak_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statictic peak to peak value is transmitted.	

Parameter: int SA_OutputStatistic2Min_RS422	SA_OutputStatistic2Min_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if second statictic min value is transmitted.	
Parameter: int SA_OutputStatistic2Max_RS422	SA_OutputStatistic2Max_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if second statictic max value is transmitted.	
Parameter: int SA_OutputStatistic2Peak2Peak_RS422	SA_OutputStatis- tic2Peak2Peak_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if second statictic peak to peak value is transmitted.	
Parameter: int SA_OutputStatisticMin_ETH	SA_OutputStatisticMin_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statictic min value is transmitted.	
Parameter: int SA_OutputStatisticMax_ETH	SA_OutputStatisticMax_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statictic max value is transmitted.	
Parameter: int SA_OutputStatisticPeak2Peak_ETH	SA_OutputStatistic- Peak2Peak_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statictic peak to peak value is transmitted.	
Parameter: int SA_OutputStatistic2Min_ETH	SA_OutputStatistic2Min_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if second statictic min value is transmitted.	
Parameter: int SA_OutputStatistic2Max_ETH	SA_OutputStatistic2Max_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if second statictic max value is transmitted.	

Parameter: int SA_OutputStatistic2Peak2Peak_ETH	SA_OutputStatistic2Peak2Peak_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if second staticic peak to peak value is transmitted.	
Parameter: int SA_OutputAdditionalNbrEdges_RS422	SA_OutputAdditionalNbrEdges_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if number of edges is transmitted.	
Parameter: int SA_OutputAdditionalNbrPins_RS422	SA_OutputAdditionalNbrPins_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if number of pins (dark areas) is transmitted.	
Parameter: int SA_OutputAdditionalNbrGaps_RS422	SA_OutputAdditionalNbrGaps_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if number of gaps (light areas) is transmitted.	
Parameter: int SA_OutputAdditionalCounter_RS422	SA_OutputAdditionalCounter_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SA_OutputAdditionalTimestamp_RS422	SA_OutputAdditionalTimestamp_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SA_OutputAdditionalState_RS422	SA_OutputAdditionalState_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	
Parameter: int SA_OutputAdditionalNbrEdges_ETH	SA_OutputAdditionalNbrEdges_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if number of edges is transmitted.	

Parameter: int SA_OutputAdditionalNbrPins_ETH	SA_OutputAdditionalNbrPins_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if number of pins (dark areas) is transmitted.	
Parameter: int SA_OutputAdditionalNbrGaps_ETH	SA_OutputAdditionalNbrGaps_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if number of gaps (light areas) is transmitted.	
Parameter: int SA_OutputAdditionalCounter_ETH	SA_OutputAdditionalCounter_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SA_OutputAdditionalTimestamp_ETH	SA_OutputAdditionalTimestamp_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SA_OutputAdditionalState_ETH	SA_OutputAdditionalState_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	
Parameter: int SA_OutputVideoRaw_ETH	SA_OutputVideoRaw_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if raw video signal is transmitted.	
Parameter: int SA_OutputVideoLight_ETH	SA_OutputVideoLight_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if light preprocessed video signal is transmitted.	
Parameter: int SA_OutputVideoLightTable_ETH	SA_OutputVideoLightTable_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if light table is transmitted.	

Parameter: int SA_OutputVideoThreshold_ETH	SA_OutputVideoThreshold_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if video threshold table is transmitted.	
Parameter: int SA_ErrorOutput1	SA_ErrorOutput1
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Wrong number of edges (ER1)	
2= Measurement error (ER2)	
3= Below low limit (LI1)	
4= Above high limit (LI2)	
5= Out of limits (LI12)	
Description: Condition for error output.	
Parameter: int SA_ErrorOutput2	SA_ErrorOutput2
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Wrong number of edges (ER1)	
2= Measurement error (ER2)	
3= Below low limit (LI1)	
4= Above high limit (LI2)	
5= Out of limits (LI12)	
Description: Condition for error output.	
Parameter: int SA_DataSource	SA_DataSource
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Edge, Light-Dark (EHL)	
1= Edge, Dark-Light (ELH)	
2= Diameter, Edge A (DA)	
3= Diameter, Edge B (DB)	
4= Diameter, Difference (DD)	
5= Diameter, Center axis (DC)	
6= Gap, Edge A (GA)	
7= Gap, Edge B (GB)	
8= Gap, Difference (GD)	
9= Gap, Center axis (GC)	
10= Segment 1, Edge A (S1A)	
11= Segment 1, Edge B (S1B)	
12= Segment 1, Difference (S1D)	
13= Segment 1, Center axis (S1C)	
14= Segment 2, Edge A (S2A)	
15= Segment 2, Edge B (S2B)	
16= Segment 2, Difference (S2D)	
17= Segment 2, Center axis (S2C)	
18= Segment 3, Edge A (S3A)	

19= Segment 3, Edge B (S3B)
 20= Segment 3, Difference (S3D)
 21= Segment 3, Center axis (S3C)
 22= Segment 4, Edge A (S4A)
 23= Segment 4, Edge B (S4B)
 24= Segment 4, Difference (S4D)
 25= Segment 4, Center axis (S4C)
 26= Segment 5, Edge A (S5A)
 27= Segment 5, Edge B (S5B)
 28= Segment 5, Difference (S5D)
 29= Segment 5, Center axis (S5C)
 30= Segment 6, Edge A (S6A)
 31= Segment 6, Edge B (S6B)
 32= Segment 6, Difference (S6D)
 33= Segment 6, Center axis (S6C)
 34= Segment 7, Edge A (S7A)
 35= Segment 7, Edge B (S7B)
 36= Segment 7, Difference (S7D)
 37= Segment 7, Center axis (S7C)
 38= Segment 8, Edge A (S8A)
 39= Segment 8, Edge B (S8B)
 40= Segment 8, Difference (S8D)
 41= Segment 8, Center axis (S8C)

Description: Data source to be checked.

Parameter: double SA_LowerLimit SA_LowerLimit

Direction: Up

Valid values:

Minimum: -120.0

Maximum: 120.0

Unit: mm

Description: Lower limit.

Parameter: double SA_UpperLimit SA_UpperLimit

Direction: Up

Valid values:

Minimum: -120.0

Maximum: 120.0

Unit: mm

Description: Upper limit.

Parameter: int SA_ErrorLevelOut1 SA_ErrorLevelOut1

Direction: Up

Valid values:

0= NPN

1= PNP

2= Push-Pull (PUSHPULL)

3= Push-Pull negated (PUSHPULLNEG)

Description: Error level for out 1.

Parameter: int SA_ErrorLevelOut2	SA_ErrorLevelOut2
Direction: Up	
Valid values:	
0= NPN	
1= PNP	
2= Push-Pull (PUSHPULL)	
3= Push-Pull negated (PUSHPULLNEG)	
Description: Error level for out 2.	
 Parameter: int SA_AnalogOutput	SA_AnalogOutput
Direction: Up	
Valid values:	
0= Edge, Light-Dark (EHL)	
1= Edge, Dark-Light (ELH)	
2= Diameter, Edge A (DA)	
3= Diameter, Edge B (DB)	
4= Diameter, Difference (DD)	
5= Diameter, Center axis (DC)	
6= Gap, Edge A (GA)	
7= Gap, Edge B (GB)	
8= Gap, Difference (GD)	
9= Gap, Center axis (GC)	
10= Segment 1, Edge A (S1A)	
11= Segment 1, Edge B (S1B)	
12= Segment 1, Difference (S1D)	
13= Segment 1, Center axis (S1C)	
14= Segment 2, Edge A (S2A)	
15= Segment 2, Edge B (S2B)	
16= Segment 2, Difference (S2D)	
17= Segment 2, Center axis (S2C)	
18= Segment 3, Edge A (S3A)	
19= Segment 3, Edge B (S3B)	
20= Segment 3, Difference (S3D)	
21= Segment 3, Center axis (S3C)	
22= Segment 4, Edge A (S4A)	
23= Segment 4, Edge B (S4B)	
24= Segment 4, Difference (S4D)	
25= Segment 4, Center axis (S4C)	
26= Segment 5, Edge A (S5A)	
27= Segment 5, Edge B (S5B)	
28= Segment 5, Difference (S5D)	
29= Segment 5, Center axis (S5C)	
30= Segment 6, Edge A (S6A)	
31= Segment 6, Edge B (S6B)	
32= Segment 6, Difference (S6D)	
33= Segment 6, Center axis (S6C)	
34= Segment 7, Edge A (S7A)	
35= Segment 7, Edge B (S7B)	
36= Segment 7, Difference (S7D)	
37= Segment 7, Center axis (S7C)	
38= Segment 8, Edge A (S8A)	
39= Segment 8, Edge B (S8B)	
40= Segment 8, Difference (S8D)	

41 = Segment 8, Center axis (S8C)	
Description: Data to be used for analog output.	
Parameter: int SA_AnalogRange	SA_AnalogRange
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = None	
1 = 0 - 10V	
Description: Analog output range.	
Parameter: int SA_AnalogScaleMode	SA_AnalogScaleMode
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Standard	
1 = Two point (TWOPOINT)	
Description: Analog scale mode.	
Parameter: double SA_MinValue	SA_MinValue
Direction: Up	
Valid values:	
Minimum: -120.0	
Maximum: 120.0	
Unit: mm	
Description: Value which represents lowest voltage/current (at two point scaling).	
Parameter: double SA_MaxValue	SA_MaxValue
Direction: Up	
Valid values:	
Minimum: -120.0	
Maximum: 120.0	
Unit: mm	
Description: Value which represents highest voltage/current (at two point scaling).	
Parameter: String SA_Sensor	SA_Sensor
Direction: Up	
Description: Name of the controller.	
Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the controller.	
Parameter: String SA_Option	SA_Option
Direction: Up	
Valid values:	
Numeric value	
Description: Option of the controller.	

Parameter: String SA_ArticleNumber	SA_ArticleNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Article number of the controller.	
Parameter: String SA_MacAddress	SA_MacAddress
Direction: Up	
Valid values:	
Valid MAC address in form of xx-xx-xx-xx-xx-xx	
Description: MAC address (low level ethernet address) of the controller.	
Parameter: String SA_Softwareversion	SA_Softwareversion
Direction: Up	
Description: Software version of firmware in the controller.	
Parameter: String SA_Image	SA_Image
Direction: Up	
Description: Active software, ethernet or ethercat.	
Parameter: String SA_ImageType	SA_ImageType
Direction: Up	
Description: Firmware image type used by the controller.	
Parameter: String SA_HardwareRevision	SA_HardwareRevision
Direction: Up	
Valid values:	
Numeric value	
Description: Revision index of the controller board.	
Parameter: int SA_MeasDistIndex	SA_MeasDistIndex
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 8	
Description: Index of the active measurement distance in the table.	
Parameter: double SA_MeasDistance	SA_MeasDistance
Direction: Up	
Description: Active measurement distance.	
Parameter: String SA_Unit	SA_Unit
Direction: Up	
Description: Unit of the measurement distance.	
Parameter: String SA_MeasDistTable	SA_MeasDistTable
Direction: Up	
Description: Whole table in one string, separated by new lines and commas.	

12.3.1.1.6 Set_SyncMode (SYNC)

Set the synchronisation mode.

Parameter: int SP_SyncMode

SP_SyncMode

Direction: Down

Valid values:

0= None

1= Slave

2= Master

Description: Synchronisation mode.

Parameter: int SP_SyncTermination

SP_SyncTermination

Direction: Down

Valid values:

0= Off (TERMOFF)

1= On 120 Ohm (TERMON)

Description: Termination of external input.

12.3.1.1.7 Get_SyncMode (SYNC)

Get the synchronisation mode.

Parameter: int SA_SyncMode

SA_SyncMode

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= None

1= Slave

2= Master

Description: Synchronisation mode.

Parameter: int SA_SyncTermination

SA_SyncTermination

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= Off (TERMOFF)

1= On 120 Ohm (TERMON)

Description: Termination of external input.

12.3.1.1.8 Set_Unit (UNIT)

Set the unit for configuration and display in the web diagram.

Parameter: int SP_DisplayUnit

SP_DisplayUnit

Direction: Down

Valid values:

0= mm

1= Inch

Description: Unit.

12.3.1.1.9 Get_Unit (UNIT)

Get the unit for configuration and display in the web diagram.

Parameter: int SA_DisplayUnit SA_DisplayUnit
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = mm
 1 = Inch
Description: Unit.

12.3.1.1.10 Reset_Boot (RESET)

Resets the sensor.

12.3.1.2 User level

12.3.1.2.1 Logout (LOGOUT)

Change user level to user.

12.3.1.2.2 Login (LOGIN)

Change user level to professional.

Parameter: String SP_Password SP_Password
Direction: Down
Description: Valid password to login.

12.3.1.2.3 Get_UserLevel (GETUSERLEVEL)

Retrieve actual user level.

Parameter: int SA_UserLevel SA_UserLevel
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = User
 1 = Professional
Description: Actual user level.

12.3.1.2.4 Set_DefaultUser (STDUSER)

Set the default user level after booting the system.

Parameter: int SP_DefaultUser SP_DefaultUser
Direction: Down
Valid values:
 0 = User
 1 = Professional
Description: Default user level.

12.3.1.2.5 Get_DefaultUser (STDUSER)

Get the default user level after booting the system.

Parameter: int SA_DefaultUser SA_DefaultUser
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = User
 1 = Professional
Description: Default user level.

12.3.1.2.6 Set_Password (PASSWD)

Change the password for login.

Parameter: String SP_OldPassword SP_OldPassword
Direction: Down
Description: Old password.

Parameter: String SP_NewPassword SP_NewPassword
Direction: Down
Description: New password.

12.3.1.3 Measure distance

12.3.1.3.1 Get_MeasDistTable (MEASDISTTABLE)

Get a list of all calibrated measure distances.

Parameter: String SA_MeasDistTable SA_MeasDistTable
Direction: Up
Description: Whole table in one string, separated by new lines and commas.

Parameter: int SA_MeasDistTableCount SA_MeasDistTableCount
Direction: Up
Valid values:
Minimum: 0
Maximum: 8
Description: Number of entries in the table. All following parameters exists from 1 to this number, e.g. SA_MeasDistIndex1, SA_MeasDistIndex2, ...

Parameter: int SA_MeasDistIndex1..x SA_MeasDistIndex1..x
Direction: Up
Valid values:
Minimum: 0
Maximum: 8
Description: Index of the sensor head in the table.

Parameter: double SA_MeasDistance1..x SA_MeasDistance1..x
Direction: Up
Description: Measurement distance of receiver in the table.

Parameter: String SA_Unit1..x SA_Unit1..x
Direction: Up
Description: Unit of the measurement distance.

12.3.1.3.2 Set_ActiveMeasDist (MEASDIST)

Change to another measurement distance.

Parameter: int SP_MeasDistIndex SP_MeasDistIndex
Direction: Down
Valid values:
 Minimum: 0
 Maximum: 8
Description: Index of new measurement distance.

12.3.1.3.3 Get_ActiveMeasDist (MEASDIST)

Get active measurement distance.

Parameter: int SA_MeasDistIndex SA_MeasDistIndex
Direction: Up
Valid values:
 Minimum: 0
 Maximum: 8
Description: Index of active measurement distance.

12.3.1.3.4 Get_MeasDistInfo (MEASDISTINFO)

Get information of active measurement distance.

Parameter: int SA_MeasDistIndex SA_MeasDistIndex
Direction: Up
Valid values:
 Minimum: 0
 Maximum: 8
Description: Index of the active measurement distance in the table.

Parameter: double SA_MeasDistance SA_MeasDistance
Direction: Up
Description: Active measurement distance.

Parameter: String SA_Unit SA_Unit
Direction: Up
Description: Unit of the measurement distance.

12.3.1.3.5 LightCorr (LIGHTCORR)

Make a light correction.

12.3.1.4 Triggering

12.3.1.4.1 Set_TriggerMode (TRIGGER)

Set the trigger mode.

Parameter: int SP_TriggerMode

SP_TriggerMode

Direction: Down

Valid values:

- 0= None
- 1= Edge
- 2= Level (PULSE)
- 3= Software

Description: Trigger mode.

Parameter: int SP_TriggerTermination

SP_TriggerTermination

Direction: Down

Valid values:

- 0= Off (TERMOFF)
- 1= On 120 Ohm (TERMON)

Description: Termination of external input.

12.3.1.4.2 Get_TriggerMode (TRIGGER)

Get the active trigger mode.

Parameter: int SA_TriggerMode

SA_TriggerMode

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= Edge
- 2= Level (PULSE)
- 3= Software

Description: Trigger mode.

Parameter: int SA_TriggerTermination

SA_TriggerTermination

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Off (TERMOFF)
- 1= On 120 Ohm (TERMON)

Description: Termination of external input.

12.3.1.4.3 Set_TriggerMoment (TRIGGERAT)

Set the trigger time.

Parameter: int SP_TriggerMoment

SP_TriggerMoment

Direction: Down

Valid values:

- 0= Input
- 1= Output

Description: Trigger moment.

12.3.1.4.4 Get_TriggerMoment (TRIGGERAT)

Get the active trigger time.

Parameter: int SA_TriggerMoment SA_TriggerMoment
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= Input
 1= Output
Description: Trigger moment.

12.3.1.4.5 Set_TriggerLevel (TRIGGERLEVEL)

Set the trigger level.

Parameter: int SP_TriggerLevel SP_TriggerLevel
Direction: Down
Valid values:
 0= High
 1= Low
Description: Trigger level.

12.3.1.4.6 Get_TriggerLevel (TRIGGERLEVEL)

Get the active trigger level.

Parameter: int SA_TriggerLevel SA_TriggerLevel
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= High
 1= Low
Description: Trigger level.

12.3.1.4.7 Set_TriggerCount (TRIGGERCOUNT)

Set the number of values to measure at trigger.

Parameter: int SP_TriggerCount SP_TriggerCount
Direction: Down
Valid values:
Minimum: 0
Maximum: 16383
Description: Number of values to measure. 0 means no trigger, 16383 means endless measurement.

12.3.1.4.8 Get_TriggerCount (TRIGGERCOUNT)

Get the number of values to measure at trigger.

Parameter: int SA_TriggerCount SA_TriggerCount
Direction: Up
Valid values:
Minimum: 0
Maximum: 16383
Description: Number of values to measure. 0 means no trigger, 16383 means endless measurement.

12.3.1.4.9 Software_Trigger (TRIGGERSW)

Execute a software trigger.

12.3.1.5 Interfaces

12.3.1.5.1 Set_IPConfiguration (IPCONFIG)

Set the IP configuration at controller.

Parameter: int SP_DHCPEnabled SP_DHCPEnabled
Direction: Down
Valid values:
 0 = FALSE
 1 = TRUE
Description: Specify if controller should use a static IP address or ask for IP at DHCP server (dynamic IP address).

Parameter: String SP_Address SP_Address
Direction: Down
Valid values:
 Valid IP address in form of xxx.xxx.xxx.xxx
Description: IP address of the controller. This parameter is only evaluated on static IP assignment.

Parameter: String SP_SubnetMask SP_SubnetMask
Direction: Down
Valid values:
 Valid network mask (e.g. 255.255.255.0 for a Class C network)
Description: Network mask of the controller. This parameter is only evaluated on static IP assignment.

Parameter: String SP_Gateway SP_Gateway
Direction: Down
Valid values:
 Valid IP address of default gateway in form of xxx.xxx.xxx.xxx
Description: The default gateway must be specified if the controller should communicate with peers in foreign subnets. This parameter is only evaluated on static IP assignment.

12.3.1.5.2 Get_IPConfiguration (IPCONFIG)

Get the IP configuration at controller.

Parameter: int SA_DHCPEnabled SA_DHCPEnabled

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = FALSE
- 1 = TRUE

Description: Get settings if controller should use a static IP address ask for IP at DHCP server (dynamic IP address).

Parameter: String SA_Address SA_Address

Direction: Up

Valid values:

Valid IP address in form of xxx.xxx.xxx.xxx

Description: IP address of the controller. If DHCP is enabled it returns the currently assigned IP address.

Parameter: String SA_SubnetMask SA_SubnetMask

Direction: Up

Valid values:

Valid network mask (e.g. 255.255.255.0 for a Class C network)

Description: Network mask of the controller. If DHCP is enabled it returns the currently assigned network mask.

Parameter: String SA_Gateway SA_Gateway

Direction: Up

Valid values:

Valid IP address of default gateway in form of xxx.xxx.xxx.xxx

Description: Address of the default gateway. If DHCP is enabled it returns the currently assigned default gateway.

12.3.1.5.3 Set_IPDataTransferMode (MEATRANSFER)

Set IP protocol at controller.

Parameter: int SP_Protocol SP_Protocol

Direction: Down

Valid values:

- 0 = TCP server (SERVER/TCP)
- 1 = TCP client (CLIENT/TCP)
- 2 = UDP sender (CLIENT/UDP)
- 3 = None

Description: Specifies if data should be send using TCP or UDP.

Parameter: String SP_RemoteAddress SP_RemoteAddress

Direction: Down

Valid values:

Valid IP address of receiver of data

Description: Address of remote computer to send data to. On TCP server this parameter is ignored.

12.3. Commands for ODC2520

Parameter: int SP_Port SP_Port
Direction: Down
Valid values:
Minimum: 1024
Maximum: 65535
Description: Port to send data to or to listen for incoming requests.

12.3.1.5.4 Get_IPDataTransferMode (MEASTRANSFER)

Get IP protocol at controller.

Parameter: int SA_Protocol SA_Protocol
Direction: Up
Valid values:
-1= Unknown parameter value from sensor
0= TCP server (SERVER/TCP)
1= TCP client (CLIENT/TCP)
2= UDP sender (CLIENT/UDP)
3= None
Description: Specifies if data should be send using TCP or UDP.

Parameter: String SA_RemoteAddress SA_RemoteAddress
Direction: Up
Valid values:
Valid IP address of receiver of data
Description: Address of remote computer to send data to.
Parameter: int SA_Port SA_Port
Direction: Up
Valid values:
Minimum: 1024
Maximum: 65535
Description: Port to send data to or to listen for incoming requests.

12.3.1.5.5 Set_EthernetMode (ETHERMODE)

Switches ethernet mode between Ethernet and Ethercat.

Parameter: int SP_EthernetMode SP_EthernetMode
Direction: Down
Valid values:
0= Ethernet
1= Ethercat
Description: Ethernet mode.

12.3.1.5.6 Get_EthernetMode (ETHERMODE)

Get ethernet mode of controller.

Parameter: int SA_EthernetMode SA_EthernetMode
Direction: Up
Valid values:
-1= Unknown parameter value from sensor
0= Ethernet
1= Ethercat
Description: Ethernet mode.

12.3.1.5.7 Set_Baudrate (BAUDRATE)

Set baudrate of controller for serial RS422 communication. After receiving the answer of the command, the baudrate of the local serial interface is changed, too.

Parameter: int SP_SensorBaudrate SP_SensorBaudrate
Direction: Down
Valid values:
 4000000
 3500000
 3000000
 2500000
 2000000
 1500000
 921600
 691200
 460800
 230400
 115200
 9600
Unit: Baud
Description: Baudrate of controller.

12.3.1.5.8 Get_Baudrate (BAUDRATE)

Get baudrate of controller for serial RS422 communication.

Parameter: int SA_SensorBaudrate SA_SensorBaudrate
Direction: Up
Valid values:
 4000000
 3500000
 3000000
 2500000
 2000000
 1500000
 921600
 691200
 460800
 230400
 115200
 9600
Unit: Baud
Description: Baudrate of controller.

12.3.1.6 Parameter management

12.3.1.6.1 Save_Parameters (STORE)

Save actual parameters at controller. There can be saved several settings on different locations. So it is easy to switch to another setting.

Parameter: int SP_ParameterSet SP_ParameterSet
Direction: Down
Valid values:
Minimum: 1
Maximum: 8
Description: Location to save the settings.

12.3.1.6.2 Load_Parameters (READ)

Load actual parameters into controller RAM. There can be loaded several settings from different locations. So it is easy to switch to another setting.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_ParameterType SP_ParameterType
Direction: Down
Valid values:
 0= All settings (ALL)
 1= Device settings (DEVICE)
 2= Measurement settings (MEAS)
Description: Specifies which settings should be loaded.

Parameter: int SP_ParameterSet SP_ParameterSet
Direction: Down
Valid values:
Minimum: 1
Maximum: 8
Description: Location from where the settings should be loaded.

12.3.1.6.3 Set_Default (SETDEFAULT)

Reset the controller to default settings.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_DefaultType SP_DefaultType
Direction: Down
Valid values:
 0= All settings (ALL)
 1= Just the current setting (CURRENT)
Description: Specifies which settings should be resetted.

Parameter: int SP_KeepDevice SP_KeepDevice
Direction: Down
Valid values:
 0= no
 1= yes
Description: Specifies if device settings should be kept temporary.

12.3.2 Measurement

12.3.2.1 General

12.3.2.1.1 Set_MeasureMode (MEASMODE)

Set the measure mode.

If first bit of **IP_AutomaticMode** is set (1), **Get_AllParameters** (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_MeasureMode

SP_MeasureMode

Direction: Down

Valid values:

- 0= Edge Light-Dark (EDGEHL)
- 1= Edge Dark-Light (EDGELH)
- 2= Diameter (DIA)
- 3= Gap
- 4= Segments (SEGMENT)

Description: Measure mode.

12.3.2.1.2 Get_MeasureMode (MEASMODE)

Get the measure mode.

Parameter: int SA_MeasureMode

SA_MeasureMode

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Edge Light-Dark (EDGEHL)
- 1= Edge Dark-Light (EDGELH)
- 2= Diameter (DIA)
- 3= Gap
- 4= Segments (SEGMENT)

Description: Measure mode.

12.3.2.1.3 Set_SearchDirection (SEARCHDIR)

Specifiy the direction to search for edges.

Parameter: int SP_SearchDirection

SP_SearchDirection

Direction: Down

Valid values:

- 0= Standard
- 1= Inverse

Description: Search direction for edges.

12.3.2.1.4 Get_SearchDirection (SEARCHDIR)

Retrieve the direction to search for edges.

Parameter: int SA_SearchDirection SA_SearchDirection
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = Standard
 1 = Inverse
Description: Search direction for edges.

12.3.2.1.5 Set_MeasureDirection (MEASDIR)

Specifiy the reference point of measured value.

Parameter: int SP_MeasureDirection SP_MeasureDirection
Direction: Down
Valid values:
 0 = Standard
 1 = Inverse
Description: Reaference point.

12.3.2.1.6 Get_MeasureDirection (MEASDIR)

Retrieve the reference point of measured value.

Parameter: int SA_MeasureDirection SA_MeasureDirection
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = Standard
 1 = Inverse
Description: Reaference point.

12.3.2.1.7 Set_ExpectedEdges (EXPEDGES)

Set the number of expected edges.

Parameter: int SP_ExpectedEdges SP_ExpectedEdges
Direction: Down
Valid values:
Minimum: 1
Maximum: 64
Description: Number of expected edges.

12.3.2.1.8 Get_ExpectedEdges (EXPEDGES)

Get the number of expected edges.

Parameter: int SA_ExpectedEdges SA_ExpectedEdges
Direction: Up
Valid values:
Minimum: 1
Maximum: 64
Description: Number of expected edges.

12.3.2.1.9 Set_SegmentDefinition1 (DEFSEG1)

Define a segment.

Parameter: int SP_Edge1_A SP_Edge1_A
Direction: Down
Valid values:
Minimum: 0
Maximum: 64
Description: First edge.

Parameter: int SP_Edge1_B SP_Edge1_B
Direction: Down
Valid values:
Minimum: 0
Maximum: 64
Description: Second edge.

12.3.2.1.10 Get_SegmentDefinition1 (DEFSEG1)

Retrieve a segment definition.

Parameter: int SA_Edge1_A SA_Edge1_A
Direction: Up
Valid values:
Minimum: 0
Maximum: 64
Description: First edge.

Parameter: int SA_Edge1_B SA_Edge1_B
Direction: Up
Valid values:
Minimum: 0
Maximum: 64
Description: Second edge.

12.3.2.1.11 Set_SegmentDefinition2 (DEFSEG2)

Define a segment.

Parameter: int SP_Edge2_A SP_Edge2_A

Direction: Down

Valid values:

Minimum: 0

Maximum: 64

Description: First edge.

Parameter: int SP_Edge2_B SP_Edge2_B

Direction: Down

Valid values:

Minimum: 0

Maximum: 64

Description: Second edge.

12.3.2.1.12 Get_SegmentDefinition2 (DEFSEG2)

Retrieve a segment definition.

Parameter: int SA_Edge2_A SA_Edge2_A

Direction: Up

Valid values:

Minimum: 0

Maximum: 64

Description: First edge.

Parameter: int SA_Edge2_B SA_Edge2_B

Direction: Up

Valid values:

Minimum: 0

Maximum: 64

Description: Second edge.

12.3.2.1.13 Set_SegmentDefinition3 (DEFSEG3)

Define a segment.

Parameter: int SP_Edge3_A SP_Edge3_A

Direction: Down

Valid values:

Minimum: 0

Maximum: 64

Description: First edge.

Parameter: int SP_Edge3_B SP_Edge3_B

Direction: Down

Valid values:

Minimum: 0

Maximum: 64

Description: Second edge.

12.3.2.1.14 Get_SegmentDefinition3 (DEFSEG3)

Retrieve a segment definition.

Parameter: int SA_Edge3_A SA_Edge3_A

Direction: Up

Valid values:

Minimum: 0

Maximum: 64

Description: First edge.

Parameter: int SA_Edge3_B SA_Edge3_B

Direction: Up

Valid values:

Minimum: 0

Maximum: 64

Description: Second edge.

12.3.2.1.15 Set_SegmentDefinition4 (DEFSEG4)

Define a segment.

Parameter: int SP_Edge4_A SP_Edge4_A

Direction: Down

Valid values:

Minimum: 0

Maximum: 64

Description: First edge.

Parameter: int SP_Edge4_B SP_Edge4_B

Direction: Down

Valid values:

Minimum: 0

Maximum: 64

Description: Second edge.

12.3.2.1.16 Get_SegmentDefinition4 (DEFSEG4)

Retrieve a segment definition.

Parameter: int SA_Edge4_A SA_Edge4_A

Direction: Up

Valid values:

Minimum: 0

Maximum: 64

Description: First edge.

Parameter: int SA_Edge4_B SA_Edge4_B

Direction: Up

Valid values:

Minimum: 0

Maximum: 64

Description: Second edge.

12.3.2.1.17 Set_SegmentDefinition5 (DEFSEG5)

Define a segment.

Parameter: int SP_Edge5_A SP_Edge5_A

Direction: Down

Valid values:

Minimum: 0

Maximum: 64

Description: First edge.

Parameter: int SP_Edge5_B SP_Edge5_B

Direction: Down

Valid values:

Minimum: 0

Maximum: 64

Description: Second edge.

12.3.2.1.18 Get_SegmentDefinition5 (DEFSEG5)

Retrieve a segment definition.

Parameter: int SA_Edge5_A SA_Edge5_A

Direction: Up

Valid values:

Minimum: 0

Maximum: 64

Description: First edge.

Parameter: int SA_Edge5_B SA_Edge5_B

Direction: Up

Valid values:

Minimum: 0

Maximum: 64

Description: Second edge.

12.3.2.1.19 Set_SegmentDefinition6 (DEFSEG6)

Define a segment.

Parameter: int SP_Edge6_A SP_Edge6_A

Direction: Down

Valid values:

Minimum: 0

Maximum: 64

Description: First edge.

Parameter: int SP_Edge6_B SP_Edge6_B

Direction: Down

Valid values:

Minimum: 0

Maximum: 64

Description: Second edge.

12.3.2.1.20 Get_SegmentDefinition6 (DEFSEG6)

Retrieve a segment definition.

Parameter: int SA_Edge6_A SA_Edge6_A

Direction: Up

Valid values:

Minimum: 0

Maximum: 64

Description: First edge.

Parameter: int SA_Edge6_B SA_Edge6_B

Direction: Up

Valid values:

Minimum: 0

Maximum: 64

Description: Second edge.

12.3.2.1.21 Set_SegmentDefinition7 (DEFSEG7)

Define a segment.

Parameter: int SP_Edge7_A SP_Edge7_A

Direction: Down

Valid values:

Minimum: 0

Maximum: 64

Description: First edge.

Parameter: int SP_Edge7_B SP_Edge7_B

Direction: Down

Valid values:

Minimum: 0

Maximum: 64

Description: Second edge.

12.3.2.1.22 Get_SegmentDefinition7 (DEFSEG7)

Retrieve a segment definition.

Parameter: int SA_Edge7_A SA_Edge7_A

Direction: Up

Valid values:

Minimum: 0

Maximum: 64

Description: First edge.

Parameter: int SA_Edge7_B SA_Edge7_B

Direction: Up

Valid values:

Minimum: 0

Maximum: 64

Description: Second edge.

12.3.2.1.23 Set_SegmentDefinition8 (DEFSEG8)

Define a segment.

Parameter: int SP_Edge8_A SP_Edge8_A

Direction: Down

Valid values:

Minimum: 0

Maximum: 64

Description: First edge.

Parameter: int SP_Edge8_B SP_Edge8_B

Direction: Down

Valid values:

Minimum: 0

Maximum: 64

Description: Second edge.

12.3.2.1.24 Get_SegmentDefinition8 (DEFSEG8)

Retrieve a segment definition.

Parameter: int SA_Edge8_A SA_Edge8_A

Direction: Up

Valid values:

Minimum: 0

Maximum: 64

Description: First edge.

Parameter: int SA_Edge8_B SA_Edge8_B

Direction: Up

Valid values:

Minimum: 0

Maximum: 64

Description: Second edge.

12.3.2.1.25 Get_VideoStreamSignal

Read one video signal from video stream.

Parameter: int SP_ReadMode SP_ReadMode

Direction: Down

Valid values:

0= Each video signal

1= Only newest video signal

2= Automatic

Description: This mode specifies if each video signal should be read or only the latest one. If set to automatic each video signal is read until the buffer does not overflow. If the buffer becomes full one or more video signals are discarded.

Parameter: int SP_WaitVideoTimeout	SP_WaitVideoTimeout
Direction: Down	
Unit: ms	
Valid values:	
Minimum: 0	
Maximum: 2147483647 (INT_MAX)	
Description: Timeout to wait for a video signal.	
Parameter: Binary data SA_VideoRaw	SA_VideoRaw
Direction: Up	
Valid values:	
768 words (each 2 byte), each word is an intensity value.	
Description: Raw video signal	
Parameter: Binary data SA_VideoLight	SA_VideoLight
Direction: Up	
Valid values:	
768 words (each 2 byte), each word is an intensity value.	
Description: Light corrected video signal	
Parameter: Binary data SA_VideoLightTable	SA_VideoLightTable
Direction: Up	
Valid values:	
768 words (each 2 byte), each word is an intensity value.	
Description: Light table	
Parameter: Binary data SA_VideoThreshold	SA_VideoThreshold
Direction: Up	
Valid values:	
768 words (each 2 byte), each word is an intensity value.	
Description: Threshold table	
Parameter: double SA_VideoTimestamp	SA_VideoTimestamp
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 1.79769e+308 (DBL_MAX)	
Unit: ms	
Description: Timestamp of the video signal. It starts from 1970 Jan 01 at 01:00. It is generated when the video has arrived at TCP/IP socket.	
Parameter: int SA_SkippedVideo	SA_SkippedVideo
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 2147483647 (INT_MAX)	
Description: Number of skipped video signals, if SP_ReadMode is not 0.	

12.3.2.2 Video signal

12.3.2.2.1 Set_ROI (ROI)

Set the region of interest for processing video signal.

Parameter: int SP_ROIStart SP_ROIStart

Direction: Down

Valid values:

Minimum: 0

Maximum: 767

Unit: Pixel

Description: First position on CCD.

Parameter: int SP_ROIEnd SP_ROIEnd

Direction: Down

Valid values:

Minimum: 0

Maximum: 767

Unit: Pixel

Description: Last position on CCD.

12.3.2.2.2 Get_ROI (ROI)

Get the region of interest for processing video signal.

Parameter: int SA_ROIStart SA_ROIStart

Direction: Up

Valid values:

Minimum: 0

Maximum: 767

Unit: Pixel

Description: First position on CCD.

Parameter: int SA_ROIEnd SA_ROIEnd

Direction: Up

Valid values:

Minimum: 0

Maximum: 767

Unit: Pixel

Description: Last position on CCD.

12.3.2.2.3 Set_Threshold (THRESHOLD)

Set threshold for video processing.

Parameter: double SP_Threshold SP_Threshold

Direction: Down

Valid values:

Minimum: 1.0

Maximum: 99.0

Unit: %

Description: Video threshold.

12.3.2.2.4 Get_Threshold (THRESHOLD)

Get threshold for video processing.

Parameter: double SA_Threshold SA_Threshold
Direction: Up
Valid values:
Minimum: 1.0
Maximum: 99.0
Unit: %
Description: Video threshold.

12.3.2.3 Measurement value processing

12.3.2.3.1 Set_Averaging (AVERAGE)

Set data averaging at controller.

Parameter: int SP_AveragingType SP_AveragingType
Direction: Down
Valid values:
 0= None
 1= Moving average (MOVING)
 2= Recursive averaging (RECURSIVE)
 3= Median
Description: Averaging type.

Parameter: int SP_MovingCount SP_MovingCount
Direction: Down
Valid values:
 2
 4
 8
 16
 32
 64
 128

Description: Number of value for the averaging window. This parameter is only used at moving average.

Parameter: int SP_RecursiveCount SP_RecursiveCount
Direction: Down
Valid values:
Minimum: 2
Maximum: 32768
Description: Number of values for recursive averaging. This parameter is only used at recursive average.

Parameter: int SP_MedianCount SP_MedianCount

Direction: Down

Valid values:

- 3
- 5
- 7
- 9

Description: Number of values to build median. This parameter is only used at median.

12.3.2.3.2 Get_Averaging (AVERAGE)

Get data averaging at controller.

Parameter: int SA_AveragingType SA_AveragingType

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= Moving average (MOVING)
- 2= Recursive averaging (RECURSIVE)
- 3= Median

Description: Averaging type.

Parameter: int SA_MovingCount SA_MovingCount

Direction: Up

Valid values:

- 2
- 4
- 8
- 16
- 32
- 64
- 128

Description: Number of value for the averaging window. This parameter is only available at moving average.

Parameter: int SA_RecursiveCount SA_RecursiveCount

Direction: Up

Valid values:

- Minimum:** 2
- Maximum:** 32768

Description: Number of values for recursive averaging. This parameter is only available at recursive average.

Parameter: int SA_MedianCount SA_MedianCount

Direction: Up

Valid values:

- 3
- 5
- 7
- 9

Description: Number of values to build median. This parameter is only available at median.

12.3.2.3.3 Set_SpikeCorrection (SPIKECORR)

Set spike correction at controller.

Parameter: int SP_SpikeCorrection	SP_SpikeCorrection
Direction: Down	
Valid values:	
0= off	
1= on	
Description: Spike correction.	
Parameter: int SP_NbrEvaluatedValues	SP_NbrEvaluatedValues
Direction: Down	
Valid values:	
Minimum: 1	
Maximum: 10	
Description: Number of values to evaluate for spike correction.	
Parameter: double SP_ToleranceRange	SP_ToleranceRange
Direction: Down	
Valid values:	
Minimum: 0.0	
Maximum: 46.0	
Description: Tolerance range for spike correction.	
Parameter: int SP_NbrCorrectedValues	SP_NbrCorrectedValues
Direction: Down	
Valid values:	
Minimum: 1	
Maximum: 100	
Description: Number of values to correct at spike correction.	

12.3.2.3.4 Get_SpikeCorrection (SPIKECORR)

Get spike correction at controller.

Parameter: int SA_SpikeCorrection	SA_SpikeCorrection
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= off	
1= on	
Description: Spike correction.	
Parameter: int SA_NbrEvaluatedValues	SA_NbrEvaluatedValues
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 10	
Description: Number of values to evaluate for spike correction.	

Parameter: double SA_ToleranceRange	SA_ToleranceRange
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 100.0	
Description: Tolerance range for spike correction.	
Parameter: int SA_NbrCorrectedValues	SA_NbrCorrectedValues
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 100	
Description: Number of values to correct at spike correction.	

12.3.2.3.5 Set_StatisticSignal (STATISTICSIGNAL)

Set the measured value which is used for statistic calculation.

Parameter: int SP_StatisticSignal	SP_StatisticSignal
Direction: Down	
Valid values:	
0= Edge, Light-Dark (EHL)	
1= Edge, Dark-Light (ELH)	
2= Diameter, Edge A (DA)	
3= Diameter, Edge B (DB)	
4= Diameter, Difference (DD)	
5= Diameter, Center axis (DC)	
6= Gap, Edge A (GA)	
7= Gap, Edge B (GB)	
8= Gap, Difference (GD)	
9= Gap, Center axis (GC)	
10= Segment 1, Edge A (S1A)	
11= Segment 1, Edge B (S1B)	
12= Segment 1, Difference (S1D)	
13= Segment 1, Center axis (S1C)	
14= Segment 2, Edge A (S2A)	
15= Segment 2, Edge B (S2B)	
16= Segment 2, Difference (S2D)	
17= Segment 2, Center axis (S2C)	
18= Segment 3, Edge A (S3A)	
19= Segment 3, Edge B (S3B)	
20= Segment 3, Difference (S3D)	
21= Segment 3, Center axis (S3C)	
22= Segment 4, Edge A (S4A)	
23= Segment 4, Edge B (S4B)	
24= Segment 4, Difference (S4D)	
25= Segment 4, Center axis (S4C)	
26= Segment 5, Edge A (S5A)	
27= Segment 5, Edge B (S5B)	
28= Segment 5, Difference (S5D)	
29= Segment 5, Center axis (S5C)	
30= Segment 6, Edge A (S6A)	

31= Segment 6, Edge B (S6B)
 32= Segment 6, Difference (S6D)
 33= Segment 6, Center axis (S6C)
 34= Segment 7, Edge A (S7A)
 35= Segment 7, Edge B (S7B)
 36= Segment 7, Difference (S7D)
 37= Segment 7, Center axis (S7C)
 38= Segment 8, Edge A (S8A)
 39= Segment 8, Edge B (S8B)
 40= Segment 8, Difference (S8D)
 41= Segment 8, Center axis (S8C)

Description: Value which is used for statistic calculation.

12.3.2.3.6 Get_StatisticSignal (STATISTICSIGNAL)

Get the measured value which is used for statistic calculation.

Parameter: int SA_StatisticSignal

SA_StatisticSignal

Direction: Up

Valid values:

-1= Unknown parameter value from sensor
 0= Edge, Light-Dark (EHL)
 1= Edge, Dark-Light (ELH)
 2= Diameter, Edge A (DA)
 3= Diameter, Edge B (DB)
 4= Diameter, Difference (DD)
 5= Diameter, Center axis (DC)
 6= Gap, Edge A (GA)
 7= Gap, Edge B (GB)
 8= Gap, Difference (GD)
 9= Gap, Center axis (GC)
 10= Segment 1, Edge A (S1A)
 11= Segment 1, Edge B (S1B)
 12= Segment 1, Difference (S1D)
 13= Segment 1, Center axis (S1C)
 14= Segment 2, Edge A (S2A)
 15= Segment 2, Edge B (S2B)
 16= Segment 2, Difference (S2D)
 17= Segment 2, Center axis (S2C)
 18= Segment 3, Edge A (S3A)
 19= Segment 3, Edge B (S3B)
 20= Segment 3, Difference (S3D)
 21= Segment 3, Center axis (S3C)
 22= Segment 4, Edge A (S4A)
 23= Segment 4, Edge B (S4B)
 24= Segment 4, Difference (S4D)
 25= Segment 4, Center axis (S4C)
 26= Segment 5, Edge A (S5A)
 27= Segment 5, Edge B (S5B)
 28= Segment 5, Difference (S5D)
 29= Segment 5, Center axis (S5C)
 30= Segment 6, Edge A (S6A)

31= Segment 6, Edge B (S6B)
 32= Segment 6, Difference (S6D)
 33= Segment 6, Center axis (S6C)
 34= Segment 7, Edge A (S7A)
 35= Segment 7, Edge B (S7B)
 36= Segment 7, Difference (S7D)
 37= Segment 7, Center axis (S7C)
 38= Segment 8, Edge A (S8A)
 39= Segment 8, Edge B (S8B)
 40= Segment 8, Difference (S8D)
 41= Segment 8, Center axis (S8C)

Description: Value which is used for statistic calculation.

12.3.2.3.7 Set_Statistic2Signal (STATISTIC2SIGNAL)

Set the measured value which is used for second statistic calculation.

Parameter: int SP_Statistic2Signal

SP_Statistic2Signal

Direction: Down

Valid values:

0= Edge, Light-Dark (EHL)
 1= Edge, Dark-Light (ELH)
 2= Diameter, Edge A (DA)
 3= Diameter, Edge B (DB)
 4= Diameter, Difference (DD)
 5= Diameter, Center axis (DC)
 6= Gap, Edge A (GA)
 7= Gap, Edge B (GB)
 8= Gap, Difference (GD)
 9= Gap, Center axis (GC)
 10= Segment 1, Edge A (S1A)
 11= Segment 1, Edge B (S1B)
 12= Segment 1, Difference (S1D)
 13= Segment 1, Center axis (S1C)
 14= Segment 2, Edge A (S2A)
 15= Segment 2, Edge B (S2B)
 16= Segment 2, Difference (S2D)
 17= Segment 2, Center axis (S2C)
 18= Segment 3, Edge A (S3A)
 19= Segment 3, Edge B (S3B)
 20= Segment 3, Difference (S3D)
 21= Segment 3, Center axis (S3C)
 22= Segment 4, Edge A (S4A)
 23= Segment 4, Edge B (S4B)
 24= Segment 4, Difference (S4D)
 25= Segment 4, Center axis (S4C)
 26= Segment 5, Edge A (S5A)
 27= Segment 5, Edge B (S5B)
 28= Segment 5, Difference (S5D)
 29= Segment 5, Center axis (S5C)
 30= Segment 6, Edge A (S6A)
 31= Segment 6, Edge B (S6B)

32= Segment 6, Difference (S6D)
 33= Segment 6, Center axis (S6C)
 34= Segment 7, Edge A (S7A)
 35= Segment 7, Edge B (S7B)
 36= Segment 7, Difference (S7D)
 37= Segment 7, Center axis (S7C)
 38= Segment 8, Edge A (S8A)
 39= Segment 8, Edge B (S8B)
 40= Segment 8, Difference (S8D)
 41= Segment 8, Center axis (S8C)

Description: Value which is used for second statistic calculation.

12.3.2.3.8 Get_Statistic2Signal (STATISTIC2SIGNAL)

Get the measured value which is used for second statistic calculation.

Parameter: int SA_Statistic2Signal

SA_Statistic2Signal

Direction: Up

Valid values:

-1= Unknown parameter value from sensor
 0= Edge, Light-Dark (EHL)
 1= Edge, Dark-Light (ELH)
 2= Diameter, Edge A (DA)
 3= Diameter, Edge B (DB)
 4= Diameter, Difference (DD)
 5= Diameter, Center axis (DC)
 6= Gap, Edge A (GA)
 7= Gap, Edge B (GB)
 8= Gap, Difference (GD)
 9= Gap, Center axis (GC)
 10= Segment 1, Edge A (S1A)
 11= Segment 1, Edge B (S1B)
 12= Segment 1, Difference (S1D)
 13= Segment 1, Center axis (S1C)
 14= Segment 2, Edge A (S2A)
 15= Segment 2, Edge B (S2B)
 16= Segment 2, Difference (S2D)
 17= Segment 2, Center axis (S2C)
 18= Segment 3, Edge A (S3A)
 19= Segment 3, Edge B (S3B)
 20= Segment 3, Difference (S3D)
 21= Segment 3, Center axis (S3C)
 22= Segment 4, Edge A (S4A)
 23= Segment 4, Edge B (S4B)
 24= Segment 4, Difference (S4D)
 25= Segment 4, Center axis (S4C)
 26= Segment 5, Edge A (S5A)
 27= Segment 5, Edge B (S5B)
 28= Segment 5, Difference (S5D)
 29= Segment 5, Center axis (S5C)
 30= Segment 6, Edge A (S6A)
 31= Segment 6, Edge B (S6B)

32= Segment 6, Difference (S6D)
 33= Segment 6, Center axis (S6C)
 34= Segment 7, Edge A (S7A)
 35= Segment 7, Edge B (S7B)
 36= Segment 7, Difference (S7D)
 37= Segment 7, Center axis (S7C)
 38= Segment 8, Edge A (S8A)
 39= Segment 8, Edge B (S8B)
 40= Segment 8, Difference (S8D)
 41= Segment 8, Center axis (S8C)

Description: Value which is used for second statistic calculation.

12.3.2.3.9 Set_StatisticDepth (STATISTICDEPTH)

Set the window size for floating statistic calculation.

Parameter: int SP_StatisticDepth

SP_StatisticDepth

Direction: Down

Valid values:

Minimum: 2

Maximum: 2147483647 (INT_MAX)

Description: Window size for floating statistic calculation. The value must be power of two (2, 4, 8, ..., 8192). Value greater as 8192 calculates statistic over all values.

12.3.2.3.10 Get_StatisticDepth (STATISTICDEPTH)

Get the window size for floating statistic calculation.

Parameter: int SA_StatisticDepth

SA_StatisticDepth

Direction: Up

Valid values:

Minimum: 2

Maximum: 2147483647 (INT_MAX)

Description: Window size for floating statistic calculation. The value must be power of two (2, 4, 8, ..., 8192). Value greater as 8192 calculates statistic over all values.

12.3.2.3.11 Reset_Statistic (RESETSTATISTIC)

Reset the statistic (min, max and peak to peak values).

12.3.2.3.12 Set_EdgeFilter1 (EDGEFILTER1)

Set first edge filter.

Only available at Option 201.

Parameter: int SP_EdgeFilter1

SP_EdgeFilter1

Direction: Down

Valid values:

0= off

1= on

Description: Specifies if filter should be active.

Parameter: double SP_LowerBound1	SP_LowerBound1
Direction: Down	
Valid values:	
Minimum: 0.0	
Maximum: measuring range	
Unit: mm	
Description: Lower bound for deactivation.	
Parameter: double SP_UpperBound1	SP_UpperBound1
Direction: Down	
Valid values:	
Minimum: 0.0	
Maximum: measuring range	
Unit: mm	
Description: Upper bound for activation.	
Parameter: int SP_IgnoreValues1	SP_IgnoreValues1
Direction: Down	
Valid values:	
Minimum: 1	
Maximum: 32768	
Description: Number of values to ignore.	

12.3.2.3.13 Get_EdgeFilter1 (EDGEFILTER1)

Get first edge filter.

Only available at Option 201.

Parameter: int SA_EdgeFilter1	SA_EdgeFilter1
Direction: Up	
Valid values:	
0= off	
1= on	
Description: Specifies if filter should be active.	
Parameter: double SA_LowerBound1	SA_LowerBound1
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: measuring range	
Unit: mm	
Description: Lower bound for deactivation.	
Parameter: double SA_UpperBound1	SA_UpperBound1
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: measuring range	
Unit: mm	
Description: Upper bound for activation.	

Parameter: int SA_IgnoreValues1 SA_IgnoreValues1
Direction: Up
Valid values:
Minimum: 1
Maximum: 32768
Description: Number of values to ignore.

12.3.2.3.14 Set_EdgeFilter2 (EDGEFILTER2)

Set second edge filter.
 Only available at Option 201.

Parameter: int SP_EdgeFilter2 SP_EdgeFilter2
Direction: Down
Valid values:
 0= off
 1= on
Description: Specifies if filter should be active.

Parameter: double SP_LowerBound2 SP_LowerBound2
Direction: Down
Valid values:
Minimum: 0.0
Maximum: measuring range
Unit: mm
Description: Lower bound for deactivation.

Parameter: double SP_UpperBound2 SP_UpperBound2
Direction: Down
Valid values:
Minimum: 0.0
Maximum: measuring range
Unit: mm
Description: Upper bound for activation.

Parameter: int SP_IgnoreValues2 SP_IgnoreValues2
Direction: Down
Valid values:
Minimum: 1
Maximum: 32768
Description: Number of values to ignore.

12.3.2.3.15 Get_EdgeFilter2 (EDGEFILTER1)

Get second edge filter.
 Only available at Option 201.

Parameter: int SA_EdgeFilter2 SA_EdgeFilter2
Direction: Up
Valid values:
 0= off
 1= on
Description: Specifies if filter should be active.

Parameter: double SA_LowerBound2	SA_LowerBound2
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: measuring range	
Unit: mm	
Description: Lower bound for deactivation.	
Parameter: double SA_UpperBound2	SA_UpperBound2
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: measuring range	
Unit: mm	
Description: Upper bound for activation.	
Parameter: int SA_IgnoreValues2	SA_IgnoreValues2
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 32768	
Description: Number of values to ignore.	

12.3.2.3.16 Set_EdgeFilter1Signal (EDGEFILTER1SIGNAL)

Set signal for first edge filter.
Only available at Option 201.

Parameter: int SP_EdgeFilter1Signal	SP_EdgeFilter1Signal
Direction: Down	
Valid values:	
0= Edge, Light-Dark (EHL)	
1= Edge, Dark-Light (ELH)	
2= Diameter, Edge A (DA)	
3= Diameter, Edge B (DB)	
4= Diameter, Difference (DD)	
5= Diameter, Center axis (DC)	
6= Gap, Edge A (GA)	
7= Gap, Edge B (GB)	
8= Gap, Difference (GD)	
9= Gap, Center axis (GC)	
10= Segment 1, Edge A (S1A)	
11= Segment 1, Edge B (S1B)	
12= Segment 1, Difference (S1D)	
13= Segment 1, Center axis (S1C)	
14= Segment 2, Edge A (S2A)	
15= Segment 2, Edge B (S2B)	
16= Segment 2, Difference (S2D)	
17= Segment 2, Center axis(S2C)	
18= Segment 3, Edge A (S3A)	
19= Segment 3, Edge B (S3B)	
20= Segment 3, Difference (S3D)	
21= Segment 3, Center axis (S3C)	

22= Segment 4, Edge A (S4A)
 23= Segment 4, Edge B (S4B)
 24= Segment 4, Difference (S4D)
 25= Segment 4, Center axis (S4C)
 26= Segment 5, Edge A (S5A)
 27= Segment 5, Edge B (S5B)
 28= Segment 5, Difference (S5D)
 29= Segment 5, Center axis (S5C)
 30= Segment 6, Edge A (S6A)
 31= Segment 6, Edge B (S6B)
 32= Segment 6, Difference (S6D)
 33= Segment 6, Center axis (S6C)
 34= Segment 7, Edge A (S7A)
 35= Segment 7, Edge B (S7B)
 36= Segment 7, Difference (S7D)
 37= Segment 7, Center axis (S7C)
 38= Segment 8, Edge A (S8A)
 39= Segment 8, Edge B (S8B)
 40= Segment 8, Difference (S8D)
 41= Segment 8, Center axis (S8C)

Description: Value which is used for filter.

12.3.2.3.17 Get_EdgeFilter1Signal (EDGEFILTER1SIGNAL)

Get signal for first edge filter.

Only available at Option 201.

Parameter: int SA_EdgeFilter1Signal

SA_EdgeFilter1Signal

Direction: Up

Valid values:

0= Edge, Light-Dark (EHL)
 1= Edge, Dark-Light (ELH)
 2= Diameter, Edge A (DA)
 3= Diameter, Edge B (DB)
 4= Diameter, Difference (DD)
 5= Diameter, Center axis (DC)
 6= Gap, Edge A (GA)
 7= Gap, Edge B (GB)
 8= Gap, Difference (GD)
 9= Gap, Center axis (GC)
 10= Segment 1, Edge A (S1A)
 11= Segment 1, Edge B (S1B)
 12= Segment 1, Difference (S1D)
 13= Segment 1, Center axis (S1C)
 14= Segment 2, Edge A (S2A)
 15= Segment 2, Edge B (S2B)
 16= Segment 2, Difference (S2D)
 17= Segment 2, Center axis (S2C)
 18= Segment 3, Edge A (S3A)
 19= Segment 3, Edge B (S3B)
 20= Segment 3, Difference (S3D)
 21= Segment 3, Center axis (S3C)

22= Segment 4, Edge A (S4A)
 23= Segment 4, Edge B (S4B)
 24= Segment 4, Difference (S4D)
 25= Segment 4, Center axis (S4C)
 26= Segment 5, Edge A (S5A)
 27= Segment 5, Edge B (S5B)
 28= Segment 5, Difference (S5D)
 29= Segment 5, Center axis (S5C)
 30= Segment 6, Edge A (S6A)
 31= Segment 6, Edge B (S6B)
 32= Segment 6, Difference (S6D)
 33= Segment 6, Center axis (S6C)
 34= Segment 7, Edge A (S7A)
 35= Segment 7, Edge B (S7B)
 36= Segment 7, Difference (S7D)
 37= Segment 7, Center axis (S7C)
 38= Segment 8, Edge A (S8A)
 39= Segment 8, Edge B (S8B)
 40= Segment 8, Difference (S8D)
 41= Segment 8, Center axis (S8C)

Description: Value which is used for filter.

12.3.2.3.18 Set_EdgeFilter2Signal (EDGEFILTER2SIGNAL)

Set signal for second edge filter.

Only available at Option 201.

Parameter: int SP_EdgeFilter2Signal

SP_EdgeFilter2Signal

Direction: Down

Valid values:

0= Edge, Light-Dark (EHL)
 1= Edge, Dark-Light (ELH)
 2= Diameter, Edge A (DA)
 3= Diameter, Edge B (DB)
 4= Diameter, Difference (DD)
 5= Diameter, Center axis (DC)
 6= Gap, Edge A (GA)
 7= Gap, Edge B (GB)
 8= Gap, Difference (GD)
 9= Gap, Center axis (GC)
 10= Segment 1, Edge A (S1A)
 11= Segment 1, Edge B (S1B)
 12= Segment 1, Difference (S1D)
 13= Segment 1, Center axis (S1C)
 14= Segment 2, Edge A (S2A)
 15= Segment 2, Edge B (S2B)
 16= Segment 2, Difference (S2D)
 17= Segment 2, Center axis (S2C)
 18= Segment 3, Edge A (S3A)
 19= Segment 3, Edge B (S3B)
 20= Segment 3, Difference (S3D)
 21= Segment 3, Center axis (S3C)

22= Segment 4, Edge A (S4A)
 23= Segment 4, Edge B (S4B)
 24= Segment 4, Difference (S4D)
 25= Segment 4, Center axis (S4C)
 26= Segment 5, Edge A (S5A)
 27= Segment 5, Edge B (S5B)
 28= Segment 5, Difference (S5D)
 29= Segment 5, Center axis (S5C)
 30= Segment 6, Edge A (S6A)
 31= Segment 6, Edge B (S6B)
 32= Segment 6, Difference (S6D)
 33= Segment 6, Center axis (S6C)
 34= Segment 7, Edge A (S7A)
 35= Segment 7, Edge B (S7B)
 36= Segment 7, Difference (S7D)
 37= Segment 7, Center axis (S7C)
 38= Segment 8, Edge A (S8A)
 39= Segment 8, Edge B (S8B)
 40= Segment 8, Difference (S8D)
 41= Segment 8, Center axis (S8C)

Description: Value which is used for filter.

12.3.2.3.19 Get_EdgeFilter2Signal (EDGEFILTER2SIGNAL)

Get signal for second edge filter.

Only available at Option 201.

Parameter: int SA_EdgeFilter2Signal

SA_EdgeFilter2Signal

Direction: Up

Valid values:

0= Edge, Light-Dark (EHL)
 1= Edge, Dark-Light (ELH)
 2= Diameter, Edge A (DA)
 3= Diameter, Edge B (DB)
 4= Diameter, Difference (DD)
 5= Diameter, Center axis (DC)
 6= Gap, Edge A (GA)
 7= Gap, Edge B (GB)
 8= Gap, Difference (GD)
 9= Gap, Center axis (GC)
 10= Segment 1, Edge A (S1A)
 11= Segment 1, Edge B (S1B)
 12= Segment 1, Difference (S1D)
 13= Segment 1, Center axis (S1C)
 14= Segment 2, Edge A (S2A)
 15= Segment 2, Edge B (S2B)
 16= Segment 2, Difference (S2D)
 17= Segment 2, Center axis (S2C)
 18= Segment 3, Edge A (S3A)
 19= Segment 3, Edge B (S3B)
 20= Segment 3, Difference (S3D)
 21= Segment 3, Center axis (S3C)

22= Segment 4, Edge A (S4A)
 23= Segment 4, Edge B (S4B)
 24= Segment 4, Difference (S4D)
 25= Segment 4, Center axis (S4C)
 26= Segment 5, Edge A (S5A)
 27= Segment 5, Edge B (S5B)
 28= Segment 5, Difference (S5D)
 29= Segment 5, Center axis (S5C)
 30= Segment 6, Edge A (S6A)
 31= Segment 6, Edge B (S6B)
 32= Segment 6, Difference (S6D)
 33= Segment 6, Center axis (S6C)
 34= Segment 7, Edge A (S7A)
 35= Segment 7, Edge B (S7B)
 36= Segment 7, Difference (S7D)
 37= Segment 7, Center axis (S7C)
 38= Segment 8, Edge A (S8A)
 39= Segment 8, Edge B (S8B)
 40= Segment 8, Difference (S8D)
 41= Segment 8, Center axis (S8C)

Description: Value which is used for filter.

12.3.2.3.20 Set_MasterSignal (MASTERSIGNAL)

Set the measured value which is used for mastering.

Parameter: int SP_MasterSignal

SP_MasterSignal

Direction: Down

Valid values:

0= Edge, Light-Dark (EHL)
 1= Edge, Dark-Light (ELH)
 2= Diameter, Edge A (DA)
 3= Diameter, Edge B (DB)
 4= Diameter, Difference (DD)
 5= Diameter, Center axis (DC)
 6= Gap, Edge A (GA)
 7= Gap, Edge B (GB)
 8= Gap, Difference (GD)
 9= Gap, Center axis (GC)
 10= Segment 1, Edge A (S1A)
 11= Segment 1, Edge B (S1B)
 12= Segment 1, Difference (S1D)
 13= Segment 1, Center axis (S1C)
 14= Segment 2, Edge A (S2A)
 15= Segment 2, Edge B (S2B)
 16= Segment 2, Difference (S2D)
 17= Segment 2, Center axis (S2C)
 18= Segment 3, Edge A (S3A)
 19= Segment 3, Edge B (S3B)
 20= Segment 3, Difference (S3D)
 21= Segment 3, Center axis (S3C)
 22= Segment 4, Edge A (S4A)

23= Segment 4, Edge B (S4B)
 24= Segment 4, Difference (S4D)
 25= Segment 4, Center axis (S4C)
 26= Segment 5, Edge A (S5A)
 27= Segment 5, Edge B (S5B)
 28= Segment 5, Difference (S5D)
 29= Segment 5, Center axis (S5C)
 30= Segment 6, Edge A (S6A)
 31= Segment 6, Edge B (S6B)
 32= Segment 6, Difference (S6D)
 33= Segment 6, Center axis (S6C)
 34= Segment 7, Edge A (S7A)
 35= Segment 7, Edge B (S7B)
 36= Segment 7, Difference (S7D)
 37= Segment 7, Center axis (S7C)
 38= Segment 8, Edge A (S8A)
 39= Segment 8, Edge B (S8B)
 40= Segment 8, Difference (S8D)
 41= Segment 8, Center axis (S8C)

Description: Value which is used for mastering.

12.3.2.3.21 Get_MasterSignal (MASTERSIGNAL)

Get the measured value which is used for mastering.

Parameter: int SA_MasterSignal

SA_MasterSignal

Direction: Up

Valid values:

-1= Unknown parameter value from sensor
 0= Edge, Light-Dark (EHL)
 1= Edge, Dark-Light (ELH)
 2= Diameter, Edge A (DA)
 3= Diameter, Edge B (DB)
 4= Diameter, Difference (DD)
 5= Diameter, Center axis (DC)
 6= Gap, Edge A (GA)
 7= Gap, Edge B (GB)
 8= Gap, Difference (GD)
 9= Gap, Center axis (GC)
 10= Segment 1, Edge A (S1A)
 11= Segment 1, Edge B (S1B)
 12= Segment 1, Difference (S1D)
 13= Segment 1, Center axis (S1C)
 14= Segment 2, Edge A (S2A)
 15= Segment 2, Edge B (S2B)
 16= Segment 2, Difference (S2D)
 17= Segment 2, Center axis (S2C)
 18= Segment 3, Edge A (S3A)
 19= Segment 3, Edge B (S3B)
 20= Segment 3, Difference (S3D)
 21= Segment 3, Center axis (S3C)
 22= Segment 4, Edge A (S4A)

23= Segment 4, Edge B (S4B)
 24= Segment 4, Difference (S4D)
 25= Segment 4, Center axis (S4C)
 26= Segment 5, Edge A (S5A)
 27= Segment 5, Edge B (S5B)
 28= Segment 5, Difference (S5D)
 29= Segment 5, Center axis (S5C)
 30= Segment 6, Edge A (S6A)
 31= Segment 6, Edge B (S6B)
 32= Segment 6, Difference (S6D)
 33= Segment 6, Center axis (S6C)
 34= Segment 7, Edge A (S7A)
 35= Segment 7, Edge B (S7B)
 36= Segment 7, Difference (S7D)
 37= Segment 7, Center axis (S7C)
 38= Segment 8, Edge A (S8A)
 39= Segment 8, Edge B (S8B)
 40= Segment 8, Difference (S8D)
 41= Segment 8, Center axis (S8C)

Description: Value which is used for mastering.

12.3.2.3.22 Set_MasterValue (MASTERMV)

Set the master value.

Parameter: int SP_Master

SP_Master

Direction: Down

Valid values:

0= no (NONE)
1= yes (MASTER=

Description: Specifies if mastering should be done or resetted.

Parameter: double SP_MasterValue

SP_MasterValue

Direction: Down

Valid values:

Minimum: -measuring range
Maximum: +measuring range

Unit: mm

Description: Master value

12.3.2.3.23 Get_MasterValue (MASTERMV)

Get the master value.

Parameter: int SA_Master

SA_Master

Direction: Up

Valid values:

-1= Unknown parameter value from sensor
0= no (NONE)
1= yes (MASTER=

Description: Specifies if mastering is active.

Parameter: double SA_MasterValue SA_MasterValue
Direction: Up
Valid values:
 Minimum: -measuring range
 Maximum: +measuring range
Unit: mm
Description: Master value

12.3.3 Data output

12.3.3.1 General

12.3.3.1.1 Set_DataOutInterface (OUTPUT)

Set the active interface for data output.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_DataOutInterface SP_DataOutInterface
Direction: Down
Valid values:
 0= None
 1= RS422
 2= Ethernet
 3= HTTP
Description: Active interface for data output.

12.3.3.1.2 Get_DataOutInterface (OUTPUT)

Get the active interface for data output.

Parameter: int SA_DataOutInterface SA_DataOutInterface
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= None
 1= RS422
 2= Ethernet
 3= HTTP
Description: Active interface for data output.

12.3.3.1.3 Set_Resampling (OUTREDUCE)

Set resampling to reduce output data.

Parameter: int SP_Resampling SP_Resampling
Direction: Down
Valid values:
 Minimum: 1
 Maximum: 150000
Description: Resampling value.

Parameter: int SP_ResampleAnalog	SP_ResampleAnalog
Direction: Down	
Valid values:	
0= no	
1= yes	
Description:	Specify if analog output should be resampled.
Parameter: int SP_ResampleRS422	SP_ResampleRS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description:	Specify if RS422 output should be resampled.
Parameter: int SP_ResampleEthernet	SP_ResampleEthernet
Direction: Down	
Valid values:	
0= no	
1= yes	
Description:	Specify if output over ethernet should be resampled.

12.3.3.1.4 Get_Resampling (OUTREDUCE)

Get resampling for reducing output data.

Parameter: int SA_Resampling	SA_Resampling
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 150000	
Description:	Resampling value.
Parameter: int SA_ResampleAnalog	SA_ResampleAnalog
Direction: Up	
Valid values:	
0= no	
1= yes	
Description:	Analog output is resampled.
Parameter: int SA_ResampleRS422	SA_ResampleRS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description:	RS422 output is resampled.
Parameter: int SA_ResampleEthernet	SA_ResampleEthernet
Direction: Up	
Valid values:	
0= no	
1= yes	
Description:	Output over ethernet is resampled.

12.3.3.1.5 Set_HoldLastValid (OUTHOLD)

Set the number of values to be replaced by last valid value instead of error values.

Parameter: int SP_HoldLastValid

SP_HoldLastValid

Direction: Down

Valid values:

Minimum: -1

Maximum: 1024

Description: Values to replace by last valid value. -1 means no value to hold, 0 means never output an error value (always hold last valid value).

12.3.3.1.6 Get_HoldLastValid (OUTHOLD)

Get the number of values to be replaced by last valid value instead of error values.

Parameter: int SA_HoldLastValid

SA_HoldLastValid

Direction: Up

Valid values:

Minimum: -1

Maximum: 1024

Description: Values to replace by last valid value. -1 means no value to hold, 0 means never output an error value (always hold last valid value).

12.3.3.2 Selected measurement values

12.3.3.2.1 Set_OutputEdgeLightDark_RS422 (OUTEDGEHL_RS422)

Set the edge light-dark to be output at RS422 interface (at measure mode edge light-dark).

Parameter: int SP_OutputEdgeLightDark_RS422

SP_OutputEdgeLightDark_-
RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if edge light-dark is transmitted.

12.3.3.2.2 Get_OutputEdgeLightDark_RS422 (OUTEDGEHL_RS422)

Retrieve if the edge light-dark to be output at RS422 interface (at measure mode edge light-dark).

Parameter: int SA_OutputEdgeLightDark_RS422

SA_OutputEdgeLightDark_-
RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if edge light-dark is transmitted.

12.3.3.2.3 Set_OutputEdgeLightDark_ETH (OUTEDGEHL_ETH)

Set the edge light-dark to be output at ethernet interface (at measure mode edge light-dark).

Parameter: int SP_OutputEdgeLightDark_ETH

SP_OutputEdgeLightDark_-
ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if edge light-dark is transmitted.

12.3.3.2.4 Get_OutputEdgeLightDark_ETH (OUTEDGEHL_ETH)

Retrieve if the edge light-dark to be output at ethernet interface (at measure mode edge light-dark).

Parameter: int SA_OutputEdgeLightDark_ETH

SA_OutputEdgeLightDark_-
ETH

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if edge light-dark is transmitted.

12.3.3.2.5 Set_OutputEdgeDarkLight_RS422 (OUTEDGEHL_RS422)

Set the edge dark-light to be output at RS422 interface (at measure mode edge dark-light).

Parameter: int SP_OutputEdgeDarkLight_RS422

SP_OutputEdgeDarkLight_-
RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if edge dark-light is transmitted.

12.3.3.2.6 Get_OutputEdgeDarkLight_RS422 (OUTEDGEHL_RS422)

Retrieve if the edge dark-light to be output at RS422 interface (at measure mode edge dark-light).

Parameter: int SA_OutputEdgeDarkLight_RS422

SA_OutputEdgeDarkLight_-
RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if edge dark-light is transmitted.

12.3.3.2.7 Set_OutputEdgeDarkLight_ETH (OUTEDGEETH_ETH)

Set the edge dark-light to be output at ethernet interface (at measure mode edge dark-light).

Parameter: int SP_OutputEdgeDarkLight_ETH

SP_OutputEdgeDarkLight_-
ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if edge dark-light is transmitted.

12.3.3.2.8 Get_OutputEdgeDarkLight_ETH (OUTEDGEETH_ETH)

Retrieve if the edge dark-light to be output at ethernet interface (at measure mode edge dark-light).

Parameter: int SA_OutputEdgeDarkLight_ETH

SA_OutputEdgeDarkLight_-
ETH

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if edge dark-light is transmitted.

12.3.3.2.9 Set_OutputDiameter_RS422 (OUTDIA_RS422)

Set the diameter data to be output at RS422 interface (at measure mode diameter).

Parameter: int SP_OutputDiameterEdgeA_RS422

SP_OutputDiameterEdgeA_-
RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if edge A (first light-dark edge) is transmitted.

Parameter: int SP_OutputDiameterEdgeB_RS422

SP_OutputDiameterEdgeB_-
RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if edge B (last dark-light edge) is transmitted.

Parameter: int SP_OutputDiameterDifference_RS422

SP_OutputDiameterDiffer-
ence_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if difference ($|A-B|$) is transmitted.

Parameter: int SP_OutputDiameterCenterAxis_RS422

SP_OutputDiameterCenter-
Axis_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if center axis ((A+B)/2) is transmitted.

12.3.3.2.10 Get_OutputDiameter_RS422 (OUTDIA_RS422)

Get the diameter data to be output at RS422 interface (at measure mode diameter).

Parameter: int SA_OutputDiameterEdgeA_RS422

SA_OutputDiameterEdgeA-
RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if edge A (first light-dark edge) is transmitted.

Parameter: int SA_OutputDiameterEdgeB_RS422

SA_OutputDiameterEdgeB-
RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if edge B (last dark-light edge) is transmitted.

Parameter: int SA_OutputDiameterDifference_RS422

SA_OutputDiameterDiffer-
ence_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if difference (|A-B|) is transmitted.

Parameter: int SA_OutputDiameterCenterAxis_RS422

SA_OutputDiameterCenter-
Axis_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if center axis ((A+B)/2) is transmitted.

12.3.3.2.11 Set_OutputDiameter_ETH (OUTDIA_ETH)

Set the diameter data to be output at ethernet interface (at measure mode diameter).

Parameter: int SP_OutputDiameterEdgeA_ETH

SP_OutputDiameterEdgeA-
ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if edge A (first light-dark edge) is transmitted.

Parameter: int SP_OutputDiameterEdgeB_ETH	SP_OutputDiameterEdgeB_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B (last dark-light edge) is transmitted.	
Parameter: int SP_OutputDiameterDifference_ETH	SP_OutputDiameterDiffer- ence_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) is transmitted.	
Parameter: int SP_OutputDiameterCenterAxis_ETH	SP_OutputDiameterCenter- Axis_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) is transmitted.	

12.3.3.2.12 Get_OutputDiameter_ETH (OUTDIA_ETH)

Get the diameter data to be output at ethernet interface (at measure mode diameter).

Parameter: int SA_OutputDiameterEdgeA_ETH	SA_OutputDiameterEdgeA_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A (first light-dark edge) is transmitted.	
Parameter: int SA_OutputDiameterEdgeB_ETH	SA_OutputDiameterEdgeB_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B (last dark-light edge) is transmitted.	
Parameter: int SA_OutputDiameterDifference_ETH	SA_OutputDiameterDiffer- ence_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) is transmitted.	
Parameter: int SA_OutputDiameterCenterAxis_ETH	SA_OutputDiameterCenter- Axis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) is transmitted.	

12.3.3.2.13 Set_OutputGap_RS422 (OUTGAP_RS422)

Set the gap data to be output at RS422 interface (at measure mode gap).

Parameter: int SP_OutputGapEdgeA_RS422

SP_OutputGapEdgeA_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if edge A (first dark-light edge) is transmitted.

Parameter: int SP_OutputGapEdgeB_RS422

SP_OutputGapEdgeB_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if edge B (following edge after A) is transmitted.

Parameter: int SP_OutputGapDifference_RS422

SP_OutputGapDifference_-
RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if difference (|A-B|) is transmitted.

Parameter: int SP_OutputGapCenterAxis_RS422

SP_OutputGapCenterAxis_-
RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if center axis ((A+B)/2) is transmitted.

12.3.3.2.14 Get_OutputGap_RS422 (OUTGAP_RS422)

Get the gap data to be output at RS422 interface (at measure mode gap).

Parameter: int SA_OutputGapEdgeA_RS422

SA_OutputGapEdgeA_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if edge A (first dark-light edge) is transmitted.

Parameter: int SA_OutputGapEdgeB_RS422

SA_OutputGapEdgeB_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if edge B (following edge after A) is transmitted.

12.3. Commands for ODC2520

Parameter: int SA_OutputGapDifference_RS422	SA_OutputGapDifference_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) is transmitted.	
Parameter: int SA_OutputGapCenterAxis_RS422	SA_OutputGapCenterAxis_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) is transmitted.	

12.3.3.2.15 Set_OutputGap_ETH (OUTGAP_ETH)

Set the gap data to be output at ethernet interface (at measure mode gap).

Parameter: int SP_OutputGapEdgeA_ETH	SP_OutputGapEdgeA_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A (first dark-light edge) is transmitted.	
Parameter: int SP_OutputGapEdgeB_ETH	SP_OutputGapEdgeB_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B (following edge after A) is transmitted.	
Parameter: int SP_OutputGapDifference_ETH	SP_OutputGapDifference_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) is transmitted.	
Parameter: int SP_OutputGapCenterAxis_ETH	SP_OutputGapCenterAxis_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) is transmitted.	

12.3.3.2.16 Get_OutputGap_ETH (OUTGAP_ETH)

Get the gap data to be output at ethernet interface (at measure mode gap).

Parameter: int SA_OutputGapEdgeA_ETH	SA_OutputGapEdgeA_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A (first dark-light edge) is transmitted.	

Parameter: int SA_OutputGapEdgeB_ETH	SA_OutputGapEdgeB_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B (following edge after A) is transmitted.	
Parameter: int SA_OutputGapDifference_ETH	SA_OutputGapDifference_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) is transmitted.	
Parameter: int SA_OutputGapCenterAxis_ETH	SA_OutputGapCenterAxis_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) is transmitted.	

12.3.3.2.17 Set_OutputSegment_RS422 (OUTSEG_RS422)

Set the segment data to be output at RS422 interface (at measure mode segments).

Parameter: int SP_OutputSegment1EdgeA_RS422	SP_OutputSegment1EdgeA_-RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 1 is transmitted.	
Parameter: int SP_OutputSegment1EdgeB_RS422	SP_OutputSegment1EdgeB_-RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 1 is transmitted.	
Parameter: int SP_OutputSegment1Difference_RS422	SP_OutputSeg-ment1Difference_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 1 is transmitted.	
Parameter: int SP_OutputSegment1CenterAxis_RS422	SP_OutputSeg-ment1CenterAxis_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 1 is transmitted.	

Parameter: int SP_OutputSegment2EdgeA_RS422	SP_OutputSegment2EdgeA_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 2 is transmitted.	
Parameter: int SP_OutputSegment2EdgeB_RS422	SP_OutputSegment2EdgeB_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 2 is transmitted.	
Parameter: int SP_OutputSegment2Difference_RS422	SP_OutputSeg- ment2Difference_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 2 is transmitted.	
Parameter: int SP_OutputSegment2CenterAxis_RS422	SP_OutputSeg- ment2CenterAxis_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 2 is transmitted.	
Parameter: int SP_OutputSegment3EdgeA_RS422	SP_OutputSegment3EdgeA_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 3 is transmitted.	
Parameter: int SP_OutputSegment3EdgeB_RS422	SP_OutputSegment3EdgeB_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 3 is transmitted.	
Parameter: int SP_OutputSegment3Difference_RS422	SP_OutputSeg- ment3Difference_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 3 is transmitted.	
Parameter: int SP_OutputSegment3CenterAxis_RS422	SP_OutputSeg- ment3CenterAxis_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 3 is transmitted.	

Parameter: int SP_OutputSegment4EdgeA_RS422	SP_OutputSegment4EdgeA_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 4 is transmitted.	
Parameter: int SP_OutputSegment4EdgeB_RS422	SP_OutputSegment4EdgeB_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 4 is transmitted.	
Parameter: int SP_OutputSegment4Difference_RS422	SP_OutputSeg- ment4Difference_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 4 is transmitted.	
Parameter: int SP_OutputSegment4CenterAxis_RS422	SP_OutputSeg- ment4CenterAxis_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 4 is transmitted.	
Parameter: int SP_OutputSegment5EdgeA_RS422	SP_OutputSegment5EdgeA_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 5 is transmitted.	
Parameter: int SP_OutputSegment5EdgeB_RS422	SP_OutputSegment5EdgeB_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 5 is transmitted.	
Parameter: int SP_OutputSegment5Difference_RS422	SP_OutputSeg- ment5Difference_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 5 is transmitted.	
Parameter: int SP_OutputSegment5CenterAxis_RS422	SP_OutputSeg- ment5CenterAxis_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 5 is transmitted.	

Parameter: int SP_OutputSegment6EdgeA_RS422	SP_OutputSegment6EdgeA_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 6 is transmitted.	
Parameter: int SP_OutputSegment6EdgeB_RS422	SP_OutputSegment6EdgeB_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 6 is transmitted.	
Parameter: int SP_OutputSegment6Difference_RS422	SP_OutputSeg- ment6Difference_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 6 is transmitted.	
Parameter: int SP_OutputSegment6CenterAxis_RS422	SP_OutputSeg- ment6CenterAxis_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 6 is transmitted.	
Parameter: int SP_OutputSegment7EdgeA_RS422	SP_OutputSegment7EdgeA_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 7 is transmitted.	
Parameter: int SP_OutputSegment7EdgeB_RS422	SP_OutputSegment7EdgeB_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 7 is transmitted.	
Parameter: int SP_OutputSegment7Difference_RS422	SP_OutputSeg- ment7Difference_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 7 is transmitted.	
Parameter: int SP_OutputSegment7CenterAxis_RS422	SP_OutputSeg- ment7CenterAxis_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 7 is transmitted.	

Parameter: int SP_OutputSegment8EdgeA_RS422	SP_OutputSegment8EdgeA_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 8 is transmitted.	
Parameter: int SP_OutputSegment8EdgeB_RS422	SP_OutputSegment8EdgeB_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 8 is transmitted.	
Parameter: int SP_OutputSegment8Difference_RS422	SP_OutputSeg- ment8Difference_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 8 is transmitted.	
Parameter: int SP_OutputSegment8CenterAxis_RS422	SP_OutputSeg- ment8CenterAxis_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 8 is transmitted.	

12.3.3.2.18 Get_OutputSegment_RS422 (OUTSEG_RS422)

Get the segment data to be output at RS422 interface (at measure mode segments).

Parameter: int SA_OutputSegment1EdgeA_RS422	SA_OutputSegment1EdgeA_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 1 is transmitted.	
Parameter: int SA_OutputSegment1EdgeB_RS422	SA_OutputSegment1EdgeB_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 1 is transmitted.	
Parameter: int SA_OutputSegment1Difference_RS422	SA_OutputSeg- ment1Difference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 1 is transmitted.	

Parameter: int SA_OutputSegment1CenterAxis_RS422	SA_OutputSegment1CenterAxis_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 1 is transmitted.	
Parameter: int SA_OutputSegment2EdgeA_RS422	SA_OutputSegment2EdgeA_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 2 is transmitted.	
Parameter: int SA_OutputSegment2EdgeB_RS422	SA_OutputSegment2EdgeB_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 2 is transmitted.	
Parameter: int SA_OutputSegment2Difference_RS422	SA_OutputSegment2Difference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 2 is transmitted.	
Parameter: int SA_OutputSegment2CenterAxis_RS422	SA_OutputSegment2CenterAxis_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 2 is transmitted.	
Parameter: int SA_OutputSegment3EdgeA_RS422	SA_OutputSegment3EdgeA_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 3 is transmitted.	
Parameter: int SA_OutputSegment3EdgeB_RS422	SA_OutputSegment3EdgeB_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 3 is transmitted.	
Parameter: int SA_OutputSegment3Difference_RS422	SA_OutputSegment3Difference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 3 is transmitted.	

Parameter: int SA_OutputSegment3CenterAxis_RS422	SA_OutputSegment3CenterAxis_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 3 is transmitted.	
Parameter: int SA_OutputSegment4EdgeA_RS422	SA_OutputSegment4EdgeA_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 4 is transmitted.	
Parameter: int SA_OutputSegment4EdgeB_RS422	SA_OutputSegment4EdgeB_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 4 is transmitted.	
Parameter: int SA_OutputSegment4Difference_RS422	SA_OutputSegment4Difference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 4 is transmitted.	
Parameter: int SA_OutputSegment4CenterAxis_RS422	SA_OutputSegment4CenterAxis_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 4 is transmitted.	
Parameter: int SA_OutputSegment5EdgeA_RS422	SA_OutputSegment5EdgeA_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 5 is transmitted.	
Parameter: int SA_OutputSegment5EdgeB_RS422	SA_OutputSegment5EdgeB_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 5 is transmitted.	
Parameter: int SA_OutputSegment5Difference_RS422	SA_OutputSegment5Difference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 5 is transmitted.	

Parameter: int SA_OutputSegment5CenterAxis_RS422	SA_OutputSegment5CenterAxis_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 5 is transmitted.	
Parameter: int SA_OutputSegment6EdgeA_RS422	SA_OutputSegment6EdgeA_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 6 is transmitted.	
Parameter: int SA_OutputSegment6EdgeB_RS422	SA_OutputSegment6EdgeB_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 6 is transmitted.	
Parameter: int SA_OutputSegment6Difference_RS422	SA_OutputSegment6Difference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 6 is transmitted.	
Parameter: int SA_OutputSegment6CenterAxis_RS422	SA_OutputSegment6CenterAxis_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 6 is transmitted.	
Parameter: int SA_OutputSegment7EdgeA_RS422	SA_OutputSegment7EdgeA_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 7 is transmitted.	
Parameter: int SA_OutputSegment7EdgeB_RS422	SA_OutputSegment7EdgeB_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 7 is transmitted.	
Parameter: int SA_OutputSegment7Difference_RS422	SA_OutputSegment7Difference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 7 is transmitted.	

Parameter: int SA_OutputSegment7CenterAxis_RS422	SA_OutputSegment7CenterAxis_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 7 is transmitted.	
Parameter: int SA_OutputSegment8EdgeA_RS422	SA_OutputSegment8EdgeA_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 8 is transmitted.	
Parameter: int SA_OutputSegment8EdgeB_RS422	SA_OutputSegment8EdgeB_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 8 is transmitted.	
Parameter: int SA_OutputSegment8Difference_RS422	SA_OutputSegment8Difference_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference (A-B) of segment 8 is transmitted.	
Parameter: int SA_OutputSegment8CenterAxis_RS422	SA_OutputSegment8CenterAxis_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 8 is transmitted.	

12.3.3.2.19 Set_OutputSegment_ETH (OUTSEG_ETH)

Set the segment data to be output at ethernet interface (at measure mode segments).

Parameter: int SP_OutputSegment1EdgeA_ETH	SP_OutputSegment1EdgeA_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 1 is transmitted.	
Parameter: int SP_OutputSegment1EdgeB_ETH	SP_OutputSegment1EdgeB_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 1 is transmitted.	

Parameter: int SP_OutputSegment1Difference_ETH	SP_OutputSegment1Difference_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 1 is transmitted.	
Parameter: int SP_OutputSegment1CenterAxis_ETH	SP_OutputSegment1CenterAxis_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 1 is transmitted.	
Parameter: int SP_OutputSegment2EdgeA_ETH	SP_OutputSegment2EdgeA_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 2 is transmitted.	
Parameter: int SP_OutputSegment2EdgeB_ETH	SP_OutputSegment2EdgeB_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 2 is transmitted.	
Parameter: int SP_OutputSegment2Difference_ETH	SP_OutputSegment2Difference_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 2 is transmitted.	
Parameter: int SP_OutputSegment2CenterAxis_ETH	SP_OutputSegment2CenterAxis_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 2 is transmitted.	
Parameter: int SP_OutputSegment3EdgeA_ETH	SP_OutputSegment3EdgeA_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 3 is transmitted.	
Parameter: int SP_OutputSegment3EdgeB_ETH	SP_OutputSegment3EdgeB_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 3 is transmitted.	

Parameter: int SP_OutputSegment3Difference_ETH	SP_OutputSegment3Difference_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 3 is transmitted.	
Parameter: int SP_OutputSegment3CenterAxis_ETH	SP_OutputSegment3CenterAxis_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis (($A+B$)/2) of segment 3 is transmitted.	
Parameter: int SP_OutputSegment4EdgeA_ETH	SP_OutputSegment4EdgeA_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 4 is transmitted.	
Parameter: int SP_OutputSegment4EdgeB_ETH	SP_OutputSegment4EdgeB_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 4 is transmitted.	
Parameter: int SP_OutputSegment4Difference_ETH	SP_OutputSegment4Difference_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 4 is transmitted.	
Parameter: int SP_OutputSegment4CenterAxis_ETH	SP_OutputSegment4CenterAxis_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis (($A+B$)/2) of segment 4 is transmitted.	
Parameter: int SP_OutputSegment5EdgeA_ETH	SP_OutputSegment5EdgeA_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 5 is transmitted.	
Parameter: int SP_OutputSegment5EdgeB_ETH	SP_OutputSegment5EdgeB_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 5 is transmitted.	

Parameter: int SP_OutputSegment5Difference_ETH	SP_OutputSegment5Difference_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 5 is transmitted.	
Parameter: int SP_OutputSegment5CenterAxis_ETH	SP_OutputSegment5CenterAxis_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 5 is transmitted.	
Parameter: int SP_OutputSegment6EdgeA_ETH	SP_OutputSegment6EdgeA_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 6 is transmitted.	
Parameter: int SP_OutputSegment6EdgeB_ETH	SP_OutputSegment6EdgeB_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 6 is transmitted.	
Parameter: int SP_OutputSegment6Difference_ETH	SP_OutputSegment6Difference_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 6 is transmitted.	
Parameter: int SP_OutputSegment6CenterAxis_ETH	SP_OutputSegment6CenterAxis_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 6 is transmitted.	
Parameter: int SP_OutputSegment7EdgeA_ETH	SP_OutputSegment7EdgeA_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 7 is transmitted.	
Parameter: int SP_OutputSegment7EdgeB_ETH	SP_OutputSegment7EdgeB_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 7 is transmitted.	

Parameter: int SP_OutputSegment7Difference_ETH	SP_OutputSegment7Difference_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 7 is transmitted.	
Parameter: int SP_OutputSegment7CenterAxis_ETH	SP_OutputSegment7CenterAxis_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 7 is transmitted.	
Parameter: int SP_OutputSegment8EdgeA_ETH	SP_OutputSegment8EdgeA_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 8 is transmitted.	
Parameter: int SP_OutputSegment8EdgeB_ETH	SP_OutputSegment8EdgeB_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 8 is transmitted.	
Parameter: int SP_OutputSegment8Difference_ETH	SP_OutputSegment8Difference_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 8 is transmitted.	
Parameter: int SP_OutputSegment8CenterAxis_ETH	SP_OutputSegment8CenterAxis_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 8 is transmitted.	

12.3.3.2.20 Get_OutputSegment_ETH (OUTSEG_ETH)

Get the segment data to be output at ethernet interface (at measure mode segments).

Parameter: int SA_OutputSegment1EdgeA_ETH	SA_OutputSegment1EdgeA_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 1 is transmitted.	

Parameter: int SA_OutputSegment1EdgeB_ETH	SA_OutputSegment1EdgeB_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 1 is transmitted.	
Parameter: int SA_OutputSegment1Difference_ETH	SA_OutputSeg- ment1Difference_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 1 is transmitted.	
Parameter: int SA_OutputSegment1CenterAxis_ETH	SA_OutputSeg- ment1CenterAxis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 1 is transmitted.	
Parameter: int SA_OutputSegment2EdgeA_ETH	SA_OutputSegment2EdgeA_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 2 is transmitted.	
Parameter: int SA_OutputSegment2EdgeB_ETH	SA_OutputSegment2EdgeB_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 2 is transmitted.	
Parameter: int SA_OutputSegment2Difference_ETH	SA_OutputSeg- ment2Difference_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 2 is transmitted.	
Parameter: int SA_OutputSegment2CenterAxis_ETH	SA_OutputSeg- ment2CenterAxis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 2 is transmitted.	
Parameter: int SA_OutputSegment3EdgeA_ETH	SA_OutputSegment3EdgeA_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 3 is transmitted.	

Parameter: int SA_OutputSegment3EdgeB_ETH	SA_OutputSegment3EdgeB_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 3 is transmitted.	
Parameter: int SA_OutputSegment3Difference_ETH	SA_OutputSeg- ment3Difference_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 3 is transmitted.	
Parameter: int SA_OutputSegment3CenterAxis_ETH	SA_OutputSeg- ment3CenterAxis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 3 is transmitted.	
Parameter: int SA_OutputSegment4EdgeA_ETH	SA_OutputSegment4EdgeA_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 4 is transmitted.	
Parameter: int SA_OutputSegment4EdgeB_ETH	SA_OutputSegment4EdgeB_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 4 is transmitted.	
Parameter: int SA_OutputSegment4Difference_ETH	SA_OutputSeg- ment4Difference_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 4 is transmitted.	
Parameter: int SA_OutputSegment4CenterAxis_ETH	SA_OutputSeg- ment4CenterAxis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 4 is transmitted.	
Parameter: int SA_OutputSegment5EdgeA_ETH	SA_OutputSegment5EdgeA_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 5 is transmitted.	

Parameter: int SA_OutputSegment5EdgeB_ETH	SA_OutputSegment5EdgeB_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 5 is transmitted.	
Parameter: int SA_OutputSegment5Difference_ETH	SA_OutputSeg- ment5Difference_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 5 is transmitted.	
Parameter: int SA_OutputSegment5CenterAxis_ETH	SA_OutputSeg- ment5CenterAxis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 5 is transmitted.	
Parameter: int SA_OutputSegment6EdgeA_ETH	SA_OutputSegment6EdgeA_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 6 is transmitted.	
Parameter: int SA_OutputSegment6EdgeB_ETH	SA_OutputSegment6EdgeB_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 6 is transmitted.	
Parameter: int SA_OutputSegment6Difference_ETH	SA_OutputSeg- ment6Difference_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 6 is transmitted.	
Parameter: int SA_OutputSegment6CenterAxis_ETH	SA_OutputSeg- ment6CenterAxis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 6 is transmitted.	
Parameter: int SA_OutputSegment7EdgeA_ETH	SA_OutputSegment7EdgeA_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 7 is transmitted.	

Parameter: int SA_OutputSegment7EdgeB_ETH	SA_OutputSegment7EdgeB_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 7 is transmitted.	
Parameter: int SA_OutputSegment7Difference_ETH	SA_OutputSeg- ment7Difference_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 7 is transmitted.	
Parameter: int SA_OutputSegment7CenterAxis_ETH	SA_OutputSeg- ment7CenterAxis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 7 is transmitted.	
Parameter: int SA_OutputSegment8EdgeA_ETH	SA_OutputSegment8EdgeA_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge A of segment 8 is transmitted.	
Parameter: int SA_OutputSegment8EdgeB_ETH	SA_OutputSegment8EdgeB_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if edge B of segment 8 is transmitted.	
Parameter: int SA_OutputSegment8Difference_ETH	SA_OutputSeg- ment8Difference_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if difference ($ A-B $) of segment 8 is transmitted.	
Parameter: int SA_OutputSegment8CenterAxis_ETH	SA_OutputSeg- ment8CenterAxis_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if center axis ((A+B)/2) of segment 8 is transmitted.	

12.3.3.2.21 Set_OutputStatistic_RS422 (OUTSTATIC_RS422)

Set the statistic data to be output at RS422 interface.

Parameter: int SP_OutputStatisticMin_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if statictic min value is transmitted.

SP_OutputStatisticMin_-
RS422

Parameter: int SP_OutputStatisticMax_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if statictic max value is transmitted.

SP_OutputStatisticMax_-
RS422

Parameter: int SP_OutputStatisticPeak2Peak_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if statictic peak to peak value is transmitted.

SP_OutputStatistic-
Peak2Peak_RS422

Parameter: int SP_OutputStatistic2Min_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if second staticic min value is transmitted.

SP_OutputStatistic2Min_-
RS422

Parameter: int SP_OutputStatistic2Max_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if second staticic max value is transmitted.

SP_OutputStatistic2Max_-
RS422

Parameter: int SP_OutputStatistic2Peak2Peak_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if second staticic peak to peak value is transmitted.

SP_OutputStatis-
tic2Peak2Peak_RS422

12.3.3.2.22 Get_OutputStatistic_RS422 (OUTSTATIC_RS422)

Get the statistic data to be output at RS422 interface.

Parameter: int SA_OutputStatisticMin_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if statictic min value is transmitted.

SA_OutputStatisticMin_-
RS422

Parameter: int SA_OutputStatisticMax_RS422	SA_OutputStatisticMax_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statictic max value is transmitted.	
Parameter: int SA_OutputStatisticPeak2Peak_RS422	SA_OutputStatistic- Peak2Peak_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statictic peak to peak value is transmitted.	
Parameter: int SA_OutputStatistic2Min_RS422	SA_OutputStatistic2Min_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if second statictic min value is transmitted.	
Parameter: int SA_OutputStatistic2Max_RS422	SA_OutputStatistic2Max_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if second statictic max value is transmitted.	
Parameter: int SA_OutputStatistic2Peak2Peak_RS422	SA_OutputStatis- tic2Peak2Peak_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if second statictic peak to peak value is transmitted.	

12.3.3.2.23 Set_OutputStatistic_ETH (OUTSTATISTIC_ETH)

Set the statistic data to be output at ethernet interface.

Parameter: int SP_OutputStatisticMin_ETH	SP_OutputStatisticMin_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if statictic min value is transmitted.	
Parameter: int SP_OutputStatisticMax_ETH	SP_OutputStatisticMax_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if statictic max value is transmitted.	

Parameter: int SP_OutputStatisticPeak2Peak_ETH	SP_OutputStatisticPeak2Peak_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if statictic peak to peak value is transmitted.	
Parameter: int SP_OutputStatistic2Min_ETH	SP_OutputStatistic2Min_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if second statictic min value is transmitted.	
Parameter: int SP_OutputStatistic2Max_ETH	SP_OutputStatistic2Max_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if second statictic max value is transmitted.	
Parameter: int SP_OutputStatistic2Peak2Peak_ETH	SP_OutputStatistic2Peak2Peak_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if second statictic peak to peak value is transmitted.	

12.3.3.2.24 Get_OutputStatistic_ETH (OUTSTATISTIC_ETH)

Get the statistic data to be output at ethernet interface.

Parameter: int SA_OutputStatisticMin_ETH	SA_OutputStatisticMin_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statictic min value is transmitted.	
Parameter: int SA_OutputStatisticMax_ETH	SA_OutputStatisticMax_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statictic max value is transmitted.	
Parameter: int SA_OutputStatisticPeak2Peak_ETH	SA_OutputStatisticPeak2Peak_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statictic peak to peak value is transmitted.	

Parameter: int SA_OutputStatistic2Min_ETH	SA_OutputStatistic2Min_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if second statictic min value is transmitted.	
Parameter: int SA_OutputStatistic2Max_ETH	SA_OutputStatistic2Max_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if second statictic max value is transmitted.	
Parameter: int SA_OutputStatistic2Peak2Peak_ETH	SA_OutputStatis- tic2Peak2Peak_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if second statictic peak to peak value is transmitted.	

12.3.3.2.25 Set_OutputAdditional_RS422 (OUTADD_RS422)

Set the additional data to be output at RS422 interface.

Parameter: int SP_OutputAdditionalNbrEdges_RS422	SP_OutputAddition- alNbrEdges_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if number of edges is transmitted.	
Parameter: int SP_OutputAdditionalNbrPins_RS422	SP_OutputAdditionalNbr- Pins_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if number of pins (dark areas) is transmitted.	
Parameter: int SP_OutputAdditionalNbrGaps_RS422	SP_OutputAdditionalNbr- Gaps_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if number of gaps (light areas) is transmitted.	
Parameter: int SP_OutputAdditionalCounter_RS422	SP_OutputAdditional- Counter_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	

12.3. Commands for ODC2520

Parameter: int SP_OutputAdditionalTimestamp_RS422	SP_OutputAdditionalTimestamp_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SP_OutputAdditionalState_RS422	SP_OutputAdditionalState_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	

12.3.3.2.26 Get_OutputAdditional_RS422 (OUTADD_RS422)

Get the additional data to be output at RS422 interface.

Parameter: int SA_OutputAdditionalNbrEdges_RS422	SA_OutputAdditionalNbrEdges_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if number of edges is transmitted.	
Parameter: int SA_OutputAdditionalNbrPins_RS422	SA_OutputAdditionalNbrPins_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if number of pins (dark areas) is transmitted.	
Parameter: int SA_OutputAdditionalNbrGaps_RS422	SA_OutputAdditionalNbrGaps_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if number of gaps (light areas) is transmitted.	
Parameter: int SA_OutputAdditionalCounter_RS422	SA_OutputAdditionalCounter_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SA_OutputAdditionalTimestamp_RS422	SA_OutputAdditionalTimestamp_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	

12.3. Commands for ODC2520

Parameter: int SA_OutputAdditionalState_RS422
Direction: Up
Valid values:
 0 = no
 1 = yes
Description: Specify if state is transmitted.

12.3.3.2.27 Set_OutputAdditional_ETH (OUTADD_ETH)

Set the additional data to be output at ethernet interface.

Parameter: int SP_OutputAdditionalNbrEdges_ETH	SP_OutputAdditionalNbrEdges_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if number of edges is transmitted.	
Parameter: int SP_OutputAdditionalNbrPins_ETH	SP_OutputAdditionalNbrPins_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if number of pins (dark areas) is transmitted.	
Parameter: int SP_OutputAdditionalNbrGaps_ETH	SP_OutputAdditionalNbrGaps_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if number of gaps (light areas) is transmitted.	
Parameter: int SP_OutputAdditionalCounter_ETH	SP_OutputAdditionalCounter_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SP_OutputAdditionalTimestamp_ETH	SP_OutputAdditionalTimestamp_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SP_OutputAdditionalState_ETH	SP_OutputAdditionalState_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	

12.3.3.2.28 Get_OutputAdditional_ETH (OUTADD_ETH)

Get the additional data to be output at ethernet interface.

Parameter: int SA_OutputAdditionalNbrEdges_ETH	SA_OutputAdditionalNbrEdges_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if number of edges is transmitted.	
Parameter: int SA_OutputAdditionalNbrPins_ETH	SA_OutputAdditionalNbrPins_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if number of pins (dark areas) is transmitted.	
Parameter: int SA_OutputAdditionalNbrGaps_ETH	SA_OutputAdditionalNbrGaps_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if number of gaps (light areas) is transmitted.	
Parameter: int SA_OutputAdditionalCounter_ETH	SA_OutputAdditionalCounter_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SA_OutputAdditionalTimestamp_ETH	SA_OutputAdditionalTimestamp_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SA_OutputAdditionalState_ETH	SA_OutputAdditionalState_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if state is transmitted.	

12.3.3.2.29 Set_OutputVideo_ETH (OUTVID_ETH)

Set the video signal to be output at ethernet interface.

Parameter: int SP_OutputVideoRaw_ETH	SP_OutputVideoRaw_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if raw video signal is transmitted.	

Parameter: int SP_OutputVideoLight_ETH	SP_OutputVideoLight_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if light preprocessed video signal is transmitted.	
Parameter: int SP_OutputVideoLightTable_ETH	SP_OutputVideoLight-
Direction: Down	Table_ETH
Valid values:	
0= no	
1= yes	
Description: Specify if light table is transmitted.	
Parameter: int SP_OutputVideoThreshold_ETH	SP_OutputVideoThreshold_-
Direction: Down	ETH
Valid values:	
0= no	
1= yes	
Description: Specify if video threshold table is transmitted.	
Parameter: int IP_TimerResolution	IP_TimerResolution
Direction: Down	
Unit: ms	
Valid values:	
-1= Do not set timer resolution.	
0= Use greatest possible accuracy.	
1..255= Resolution in milliseconds.	
Unit: ms	
Default: 0 if any video signal is enabled, otherwise -1	
Description: This parameter is necessary at video stream mode. Unless MEDAQLib waits for new video signals (using Sleep API function) and the default resolution for this function is 15 milli seconds there may be a time jitter at processing video signals (e.g. 15, 0, 0, 0 ..., 0, 15, 0, 0, ...). So this parameter changes the Windows timer resolution (for Windows scheduler, set by timeBeginPeriod). If it is not resetted manually when switching off all video signals (set to 0), it is automatically resetted at CloseSensor.	

12.3.3.2.30 Get_OutputVideo_ETH (OUTVID_ETH)

Get the video signal to be output at ethernet interface.

Parameter: int SA_OutputVideoRaw_ETH	SA_OutputVideoRaw_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if raw video signal is transmitted.	

Parameter: int SA_OutputVideoLight_ETH	SA_OutputVideoLight_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if light preprocessed video signal is transmitted.	
Parameter: int SA_OutputVideoLightTable_ETH	SA_OutputVideoLight-
Direction: Up	Table_ETH
Valid values:	
0= no	
1= yes	
Description: Specify if light table is transmitted.	
Parameter: int SA_OutputVideoThreshold_ETH	SA_OutputVideoThreshold_-
Direction: Up	ETH
Valid values:	
0= no	
1= yes	
Description: Specify if video threshold table is transmitted.	

12.3.3.3 Switching outputs

12.3.3.3.1 Set_ErrorOutput1 (ERROROUT1)

Set condition to be used to set error output 1.

Parameter: int SP_ErrorOutput1	SP_ErrorOutput1
Direction: Down	
Valid values:	
0= None	
1= Wrong number of edges (ER1)	
2= Measurement error (ER2)	
3= Below low limit (LI1)	
4= Above high limit (LI2)	
5= Out of limits (LI12)	
Description: Condition for error output.	

12.3.3.3.2 Get_ErrorOutput1 (ERROROUT1)

Get condition to be used to set error output 1.

Parameter: int SA_ErrorOutput1	SA_ErrorOutput1
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Wrong number of edges (ER1)	
2= Measurement error (ER2)	
3= Below low limit (LI1)	
4= Above high limit (LI2)	
5= Out of limits (LI12)	
Description: Condition for error output.	

12.3.3.3.3 Set_ErrorOutput2 (ERROROUT2)

Set condition to be used to set error output 2.

Parameter: int SP_ErrorOutput2

SP_ErrorOutput2

Direction: Down

Valid values:

- 0= None
- 1= Wrong number of edges (ER1)
- 2= Measurement error (ER2)
- 3= Below low limit (LI1)
- 4= Above high limit (LI2)
- 5= Out of limits (LI12)

Description: Condition for error output.

12.3.3.3.4 Get_ErrorOutput2 (ERROROUT2)

Get condition to be used to set error output 2.

Parameter: int SA_ErrorOutput2

SA_ErrorOutput2

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= Wrong number of edges (ER1)
- 2= Measurement error (ER2)
- 3= Below low limit (LI1)
- 4= Above high limit (LI2)
- 5= Out of limits (LI12)

Description: Condition for error output.

12.3.3.3.5 Set_ErrorLimit (ERRORLIMIT)

Set the error limits.

Parameter: int SP_DataSource

SP_DataSource

Direction: Down

Valid values:

- 0= Edge, Light-Dark (EHL)
- 1= Edge, Dark-Light (ELH)
- 2= Diameter, Edge A (DA)
- 3= Diameter, Edge B (DB)
- 4= Diameter, Difference (DD)
- 5= Diameter, Center axis (DC)
- 6= Gap, Edge A (GA)
- 7= Gap, Edge B (GB)
- 8= Gap, Difference (GD)
- 9= Gap, Center axis (GC)
- 10= Segment 1, Edge A (S1A)
- 11= Segment 1, Edge B (S1B)
- 12= Segment 1, Difference (S1D)

13= Segment 1, Center axis (S1C)
 14= Segment 2, Edge A (S2A)
 15= Segment 2, Edge B (S2B)
 16= Segment 2, Difference (S2D)
 17= Segment 2, Center axis(S2C)
 18= Segment 3, Edge A (S3A)
 19= Segment 3, Edge B (S3B)
 20= Segment 3, Difference (S3D)
 21= Segment 3, Center axis (S3C)
 22= Segment 4, Edge A (S4A)
 23= Segment 4, Edge B (S4B)
 24= Segment 4, Difference (S4D)
 25= Segment 4, Center axis (S4C)
 26= Segment 5, Edge A (S5A)
 27= Segment 5, Edge B (S5B)
 28= Segment 5, Difference (S5D)
 29= Segment 5, Center axis (S5C)
 30= Segment 6, Edge A (S6A)
 31= Segment 6, Edge B (S6B)
 32= Segment 6, Difference (S6D)
 33= Segment 6, Center axis (S6C)
 34= Segment 7, Edge A (S7A)
 35= Segment 7, Edge B (S7B)
 36= Segment 7, Difference (S7D)
 37= Segment 7, Center axis (S7C)
 38= Segment 8, Edge A (S8A)
 39= Segment 8, Edge B (S8B)
 40= Segment 8, Difference (S8D)
 41= Segment 8, Center axis (S8C)

Description: Data source to be checked.

Parameter: double SP_LowerLimit SP_LowerLimit

Direction: Down

Valid values:

Minimum: -100.0

Maximum: 100.0

Unit: mm

Description: Lower limit.

Parameter: double SP_UpperLimit SP_UpperLimit

Direction: Down

Valid values:

Minimum: -100.0

Maximum: 100.0

Unit: mm

Description: Upper limit.

12.3.3.6 Get_ErrorLimit (ERRORLIMIT)

Get the error limits.

Parameter: int SA_DataSource SA_DataSource

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = Edge, Light-Dark (EHL)
- 1 = Edge, Dark-Light (ELH)
- 2 = Diameter, Edge A (DA)
- 3 = Diameter, Edge B (DB)
- 4 = Diameter, Difference (DD)
- 5 = Diameter, Center axis (DC)
- 6 = Gap, Edge A (GA)
- 7 = Gap, Edge B (GB)
- 8 = Gap, Difference (GD)
- 9 = Gap, Center axis (GC)
- 10 = Segment 1, Edge A (S1A)
- 11 = Segment 1, Edge B (S1B)
- 12 = Segment 1, Difference (S1D)
- 13 = Segment 1, Center axis (S1C)
- 14 = Segment 2, Edge A (S2A)
- 15 = Segment 2, Edge B (S2B)
- 16 = Segment 2, Difference (S2D)
- 17 = Segment 2, Center axis (S2C)
- 18 = Segment 3, Edge A (S3A)
- 19 = Segment 3, Edge B (S3B)
- 20 = Segment 3, Difference (S3D)
- 21 = Segment 3, Center axis (S3C)
- 22 = Segment 4, Edge A (S4A)
- 23 = Segment 4, Edge B (S4B)
- 24 = Segment 4, Difference (S4D)
- 25 = Segment 4, Center axis (S4C)
- 26 = Segment 5, Edge A (S5A)
- 27 = Segment 5, Edge B (S5B)
- 28 = Segment 5, Difference (S5D)
- 29 = Segment 5, Center axis (S5C)
- 30 = Segment 6, Edge A (S6A)
- 31 = Segment 6, Edge B (S6B)
- 32 = Segment 6, Difference (S6D)
- 33 = Segment 6, Center axis (S6C)
- 34 = Segment 7, Edge A (S7A)
- 35 = Segment 7, Edge B (S7B)
- 36 = Segment 7, Difference (S7D)
- 37 = Segment 7, Center axis (S7C)
- 38 = Segment 8, Edge A (S8A)
- 39 = Segment 8, Edge B (S8B)
- 40 = Segment 8, Difference (S8D)
- 41 = Segment 8, Center axis (S8C)

Description: Data source to be checked.

Parameter: double SA_LowerLimit

SA_LowerLimit

Direction: Up

Valid values:

Minimum: -100.0

Maximum: 100.0

Unit: mm

Description: Lower limit.

Parameter: double SA_UpperLimit SA_UpperLimit
Direction: Up
Valid values:
 Minimum: -100.0
 Maximum: 100.0
Unit: mm
Description: Upper limit.

12.3.3.3.7 Set_ErrorLevelOut1 (ERRORLEVELOUT1)

Set level of error output 1 on error.

Parameter: int SP_ErrorLevelOut1 SP_ErrorLevelOut1
Direction: Down
Valid values:
 0= NPN
 1= PNP
 2= Push-Pull (PUSHPULL)
 3= Push-Pull negated (PUSHPULLNEG)
Description: Error level for out 1.

12.3.3.3.8 Get_ErrorLevelOut1 (ERRORLEVELOUT1)

Get level of error output 1 on error.

Parameter: int SA_ErrorLevelOut1 SA_ErrorLevelOut1
Direction: Up
Valid values:
 0= NPN
 1= PNP
 2= Push-Pull (PUSHPULL)
 3= Push-Pull negated (PUSHPULLNEG)
Description: Error level for out 1.

12.3.3.3.9 Set_ErrorLevelOut2 (ERRORLEVELOUT2)

Set level of error output 2 on error.

Parameter: int SP_ErrorLevelOut2 SP_ErrorLevelOut2
Direction: Down
Valid values:
 0= NPN
 1= PNP
 2= Push-Pull (PUSHPULL)
 3= Push-Pull negated (PUSHPULLNEG)
Description: Error level for out 2.

12.3.3.3.10 Get_ErrorLevelOut2 (ERRORLEVELOUT2)

Get level of error output 2 on error.

Parameter: int SA_ErrorLevelOut2 SA_ErrorLevelOut2
Direction: Up
Valid values:
 0= NPN
 1= PNP
 2= Push-Pull (PUSHPULL)
 3= Push-Pull negated (PUSHPULLNEG)
Description: Error level for out 2.

12.3.3.4 Analog output

12.3.3.4.1 Set_AnalogOutput (ANALOGOUT)

Set the data to be used for analog output.

Parameter: int SP_AnalogOutput SP_AnalogOutput
Direction: Down
Valid values:
 0= Edge, Light-Dark (EHL)
 1= Edge, Dark-Light (ELH)
 2= Diameter, Edge A (DA)
 3= Diameter, Edge B (DB)
 4= Diameter, Difference (DD)
 5= Diameter, Center axis (DC)
 6= Gap, Edge A (GA)
 7= Gap, Edge B (GB)
 8= Gap, Difference (GD)
 9= Gap, Center axis (GC)
 10= Segment 1, Edge A (S1A)
 11= Segment 1, Edge B (S1B)
 12= Segment 1, Difference (S1D)
 13= Segment 1, Center axis (S1C)
 14= Segment 2, Edge A (S2A)
 15= Segment 2, Edge B (S2B)
 16= Segment 2, Difference (S2D)
 17= Segment 2, Center axis (S2C)
 18= Segment 3, Edge A (S3A)
 19= Segment 3, Edge B (S3B)
 20= Segment 3, Difference (S3D)
 21= Segment 3, Center axis (S3C)
 22= Segment 4, Edge A (S4A)
 23= Segment 4, Edge B (S4B)
 24= Segment 4, Difference (S4D)
 25= Segment 4, Center axis (S4C)
 26= Segment 5, Edge A (S5A)
 27= Segment 5, Edge B (S5B)
 28= Segment 5, Difference (S5D)
 29= Segment 5, Center axis (S5C)

30= Segment 6, Edge A (S6A)
 31= Segment 6, Edge B (S6B)
 32= Segment 6, Difference (S6D)
 33= Segment 6, Center axis (S6C)
 34= Segment 7, Edge A (S7A)
 35= Segment 7, Edge B (S7B)
 36= Segment 7, Difference (S7D)
 37= Segment 7, Center axis (S7C)
 38= Segment 8, Edge A (S8A)
 39= Segment 8, Edge B (S8B)
 40= Segment 8, Difference (S8D)
 41= Segment 8, Center axis (S8C)

Description: Data to be used for analog output.

12.3.3.4.2 Get_AnalogOutput (ANALOGOUT)

Get the data to be used for analog output.

Parameter: int SA_AnalogOutput

SA_AnalogOutput

Direction: Up

Valid values:

0= Edge, Light-Dark (EHL)
 1= Edge, Dark-Light (ELH)
 2= Diameter, Edge A (DA)
 3= Diameter, Edge B (DB)
 4= Diameter, Difference (DD)
 5= Diameter, Center axis (DC)
 6= Gap, Edge A (GA)
 7= Gap, Edge B (GB)
 8= Gap, Difference (GD)
 9= Gap, Center axis (GC)
 10= Segment 1, Edge A (S1A)
 11= Segment 1, Edge B (S1B)
 12= Segment 1, Difference (S1D)
 13= Segment 1, Center axis (S1C)
 14= Segment 2, Edge A (S2A)
 15= Segment 2, Edge B (S2B)
 16= Segment 2, Difference (S2D)
 17= Segment 2, Center axis (S2C)
 18= Segment 3, Edge A (S3A)
 19= Segment 3, Edge B (S3B)
 20= Segment 3, Difference (S3D)
 21= Segment 3, Center axis (S3C)
 22= Segment 4, Edge A (S4A)
 23= Segment 4, Edge B (S4B)
 24= Segment 4, Difference (S4D)
 25= Segment 4, Center axis (S4C)
 26= Segment 5, Edge A (S5A)
 27= Segment 5, Edge B (S5B)
 28= Segment 5, Difference (S5D)
 29= Segment 5, Center axis (S5C)
 30= Segment 6, Edge A (S6A)

12.3. Commands for ODC2520

31= Segment 6, Edge B (S6B)
 32= Segment 6, Difference (S6D)
 33= Segment 6, Center axis (S6C)
 34= Segment 7, Edge A (S7A)
 35= Segment 7, Edge B (S7B)
 36= Segment 7, Difference (S7D)
 37= Segment 7, Center axis (S7C)
 38= Segment 8, Edge A (S8A)
 39= Segment 8, Edge B (S8B)
 40= Segment 8, Difference (S8D)
 41= Segment 8, Center axis (S8C)

Description: Data to be used for analog output.

12.3.3.4.3 Set_AnalogRange (ANALOG RANGE)

Set the analog output range.

Parameter: int SP_AnalogRange

SP_AnalogRange

Direction: Down

Valid values:

0= None
1= 0 - 10V

Description: Analog output range.

12.3.3.4.4 Get_AnalogRange (ANALOG RANGE)

Get the analog output range.

Parameter: int SA_AnalogRange

SA_AnalogRange

Direction: Up

Valid values:

-1= Unknown parameter value from sensor
0= None
1= 0 - 10V

Description: Analog output range.

12.3.3.4.5 Set_AnalogScale (ANALOG SCALE)

Set the scaling factor for analog output. If both parameters are zero, the analog output is scaled to default range.

Parameter: int SP_AnalogScaleMode

SP_AnalogScaleMode

Direction: Down

Valid values:

0= Standard
1= Two point (TWOPOINT)

Description: Analog scale mode.

Parameter: double SP_MinValue SP_MinValue

Direction: Down

Valid values:

Minimum: -100.0

Maximum: 100.0

Unit: mm

Description: Value which represents lowest voltage/current (at two point scaling).

Parameter: double SP_MaxValue SP_MaxValue

Direction: Down

Valid values:

Minimum: -100.0

Maximum: 100.0

Unit: mm

Description: Value which represents highest voltage/current (at two point scaling).

12.3.3.4.6 Get_AnalogScale (ANALOGSCALE)

Get the scaling factor for analog output. If both parameters are zero, the analog output is scaled to default range.

Parameter: int SA_AnalogScaleMode SA_AnalogScaleMode

Direction: Up

Valid values:

 -1= Unknown parameter value from sensor

 0= Standard

 1= Two point (TWOPOINT)

Description: Analog scale mode.

Parameter: double SA_MinValue SA_MinValue

Direction: Up

Valid values:

Minimum: -100.0

Maximum: 100.0

Unit: mm

Description: Value which represents lowest voltage/current (at two point scaling).

Parameter: double SA_MaxValue SA_MaxValue

Direction: Up

Valid values:

Minimum: -100.0

Maximum: 100.0

Unit: mm

Description: Value which represents highest voltage/current (at two point scaling).

12.4 Commands for ODC2600

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [RS232](#) (native).
- [IF2004](#) (native).
- [TCP/IP](#) (additional, e.g. RS232/RS422 to TCP/IP comm server and RS232 high level interface).
- [IF2004_USB](#) (native).
- [IF2008](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_Info](#) and [Read_OptionData](#) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib to calculate datarate, to interpret data and to assign values.

If second bit of [IP_AutomaticMode](#) is set (2), MEDAQLib calls sensor command [Dat_Out_On](#) at [OpenSensor](#).

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor (including two segment bits, mask it out by raw&0xffff), from 0 to 65535 (without segment bits).
- Scaled values are scaled using sensor range, error values are scaled depending of [IP_ScaleErrorValues](#).

The values of selected segments are filled in the arrays one after another. Each array always starts with first selected segment.

12.4.1 General commands

12.4.1.1 General

12.4.1.1.1 Get_Info (INFO)

Retrieve some information about the sensor.

Parameter: String SA_ArticleNumber

SA_ArticleNumber

Direction: Up

Valid values:

Numeric value

Description: Article number of the sensor.

Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the sensor.	
Parameter: String SA_Option	SA_Option
Direction: Up	
Valid values:	
Numeric value	
Description: Option of the sensor.	
Parameter: double SA_Range	SA_Range
Direction: Up	
Unit: mm	
Valid values:	
40.0	
Description: Range of the sensor.	
Parameter: int SA_Reserve_1	SA_Reserve_1
Direction: Up	
Description: Reserved.	
Parameter: String SA_SoftArtBoot	SA_SoftArtBoot
Direction: Up	
Description: Sensor software versions.	
Parameter: String SA_SoftArtArm	SA_SoftArtArm
Direction: Up	
Description: Sensor software versions.	
Parameter: String SA_SoftArtDSP	SA_SoftArtDSP
Direction: Up	
Description: Sensor software versions.	
Parameter: int SA_SoftVerBoot	SA_SoftVerBoot
Direction: Up	
Valid values:	
Numeric value	
Description: Sensor software versions.	
Parameter: int SA_SoftVerArm	SA_SoftVerArm
Direction: Up	
Valid values:	
Numeric value	
Description: Sensor software versions.	
Parameter: int SA_SoftVerDSP	SA_SoftVerDSP
Direction: Up	
Valid values:	
Numeric value	
Description: Sensor software versions.	

12.4.1.1.2 Reset_Boot (RESET)

Resets the sensor. This command has no parameters.

12.4.1.2 Triggering

12.4.1.2.1 Triggermode_Reset (TRIGGERMODE RESET)

Resets the values at trigger mode.

12.4.1.2.2 Triggermode_Trigger (TRIGGERMODE TRIGGER)

Activate output in trigger mode.

12.4.1.3 Parameter management

12.4.1.3.1 Save_OptionData (SAVE OPT RAM TO FLASH)

Save the option data to flash.

12.4.1.3.2 Save_MeasProgData (SAVE MPR RAM TO FLASH)

Save the measure program data to flash.

12.4.2 Measurement

12.4.2.1 Choose_MeasProg (CHOOSE MP)

Select the measurement program of sensor.

Parameter: int SP_MeasProgNumber

SP_MeasProgNumber

Direction: Down

Valid values:

- 0= EdgeHL
- 1= EdgeLH
- 2= DIA
- 3= GAP
- 4= SEG_2_4
- 5= MULTISEG
- 6= USER1
- 7= USER2
- 8= USER3
- 9= USER4

Description: Measure program of sensor.

12.4.2.2 Switch_Edge (SWITCH EDGE)

Select the edges to measure.

Parameter: int SP_FrontEdge_Seg1	SP_FrontEdge_Seg1
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Front edge of segment 1.	
Parameter: int SP_FrontEdge_Seg2	SP_FrontEdge_Seg2
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Front edge of segment 2.	
Parameter: int SP_FrontEdge_Seg3	SP_FrontEdge_Seg3
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Front edge of segment 3.	
Parameter: int SP_FrontEdge_Seg4	SP_FrontEdge_Seg4
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Front edge of segment 4.	
Parameter: int SP_RearEdge_Seg1	SP_RearEdge_Seg1
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Rear edge of segment 1.	
Parameter: int SP_RearEdge_Seg2	SP_RearEdge_Seg2
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Rear edge of segment 2.	
Parameter: int SP_RearEdge_Seg3	SP_RearEdge_Seg3
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Rear edge of segment 3.	

Parameter: int SP_RearEdge_Seg4 SP_RearEdge_Seg4
Direction: Down
Valid values:
Minimum: 0
Maximum: 80
Description: Rear edge of segment 4.

12.4.2.3 Write_OptionData (WR OPT TO RAM)

Write option data to sensor.

Parameter: int SP_MeasProgNumber SP_MeasProgNumber
Direction: Down
Valid values:
 0= EdgeHL
 1= EdgeLH
 2= DIA
 3= GAP
 4= SEG_2_4
 5= MULTISEG
 6= USER1
 7= USER2
 8= USER3
 9= USER4
Description: Measure program of sensor.

Parameter: int SP_Language SP_Language
Direction: Down
Valid values:
 0= German
 1= English
Description: Language of sensor.

Parameter: int SP_DispmesUnit SP_DispmesUnit
Direction: Down
Valid values:
 0= mm
 1= inch
Description: Display measurement unit of sensor.

Parameter: int SP_ErrorHandler SP_ErrorHandler
Direction: Down
Valid values:
 0= error output
 1= retain last value
Description: If the sensor cannot measure values, it can output the last valid value or it can output an error values.

Parameter: int SP_SerialOutFormat SP_SerialOutFormat
Direction: Down
Valid values:
 0= binary
 1= ASCII
Description: Serial output format of sensor.

Parameter: int SP_Ext_LaserSwitch	SP_Ext_LaserSwitch
Direction: Down	
Valid values:	
0= not active	
1= active	
Description: Enable/Disable external laser switch.	
Parameter: int SP_LaserIntensity	SP_LaserIntensity
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 100	
Unit: %	
Description: No effect.	
Parameter: int SP_Contrast	SP_Contrast
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 100	
Unit: %	
Description: Display contrast.	
Parameter: int SP_EdgeDetectThreshold	SP_EdgeDetectThreshold
Direction: Down	
Valid values:	
Minimum: 20	
Maximum: 90	
Unit: %	
Description: Contrast for edge detection.	
Parameter: int SP_Reserve_2	SP_Reserve_2
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved.	
Parameter: int SP_ActiveSerialIf	SP_ActiveSerialIf
Direction: Down	
Valid values:	
0= RS422	
1= RS232	
Description: Active serial interface of sensor.	
Parameter: int SP_RS232_Baudrate	SP_RS232_Baudrate
Direction: Down	
Valid values:	
9600	
19200	
38400	
115200	
Unit: Baud	
Description: Baudrate of RS232 interface of sensor.	

Parameter: int SP_RS232_Parity	SP_RS232_Parity
Direction: Down	
Valid values:	
0= none	
1= even	
2= odd	
Description: Parity of RS232 interface of sensor.	
Parameter: int SP_RS232_StopBits	SP_RS232_StopBits
Direction: Down	
Valid values:	
1	
2	
Description: Stop bits of RS232 interface of sensor.	
Parameter: int SP_RS232_TimeoutSend	SP_RS232_TimeoutSend
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	
Parameter: int SP_RS232_TimeoutRecv	SP_RS232_TimeoutRecv
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	
Parameter: int SP_RS422_Baudrate	SP_RS422_Baudrate
Direction: Down	
Valid values:	
9600	
19200	
38400	
115200	
691200	
Unit: Baud	
Description: Baudrate of RS422 interface of sensor.	
Parameter: int SP_RS422_Parity	SP_RS422_Parity
Direction: Down	
Valid values:	
0= none	
1= even	
2= odd	
Description: Parity of RS422 interface of sensor.	
Parameter: int SP_RS422_StopBits	SP_RS422_StopBits
Direction: Down	
Valid values:	
1	
2	
Description: Stop bits of RS422 interface of sensor.	

Parameter: int SP_RS422_TimeoutSend	SP_RS422_TimeoutSend
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	
Parameter: int SP_RS422_TimeoutRecv	SP_RS422_TimeoutRecv
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	

12.4.2.4 Read_OptionData (RD OPT RAM)

Read the option data from sensor.

Parameter: int SA_MeasProgNumber	SA_MeasProgNumber
Direction: Up	
Valid values:	
0= EdgeHL	
1= EdgeLH	
2= DIA	
3= GAP	
4= SEG_2_4	
5= MULTISEG	
6= USER1	
7= USER2	
8= USER3	
9= USER4	
Description: Measure program of sensor.	
Parameter: int SA_Language	SA_Language
Direction: Up	
Valid values:	
0= German	
1= English	
Description: Language of sensor.	
Parameter: int SA_DispmesUnit	SA_DispmesUnit
Direction: Up	
Valid values:	
0= mm	
1= inch	
Description: Display measurement unit of sensor.	
Parameter: int SA_ErrorHandler	SA_ErrorHandler
Direction: Up	
Valid values:	
0= error output	
1= retain last value	
Description: If the sensor cannot measure values, it can output the last valid value or it can output an error values.	

Parameter: int SA_SerialOutFormat	SA_SerialOutFormat
Direction: Up	
Valid values:	
0= binary	
1= ASCII	
Description: Serial output format of sensor.	
Parameter: int SA_Ext_LaserSwitch	SA_Ext_LaserSwitch
Direction: Up	
Valid values:	
0= not active	
1= active	
Description: Enable/Disable external laser switch.	
Parameter: int SA_LaserIntensity	SA_LaserIntensity
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 100	
Unit: %	
Description: No effect.	
Parameter: int SA_Contrast	SA_Contrast
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 100	
Unit: %	
Description: Display contrast.	
Parameter: int SA_EdgeDetectThreshold	SA_EdgeDetectThreshold
Direction: Up	
Valid values:	
Minimum: 20	
Maximum: 90	
Unit: %	
Description: Contrast for edge detection.	
Parameter: int SA_Reserve_2	SA_Reserve_2
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved.	
Parameter: int SA_ActiveSerialIf	SA_ActiveSerialIf
Direction: Up	
Valid values:	
0= RS422	
1= RS232	
Description: Active serial interface of sensor.	

Parameter: int SA_RS232_Baudrate	SA_RS232_Baudrate
Direction: Up	
Valid values:	
9600	
19200	
38400	
115200	
Unit: Baud	
Description: Baudrate of RS232 interface of sensor.	
Parameter: int SA_RS232_Parity	SA_RS232_Parity
Direction: Up	
Valid values:	
0= none	
1= even	
2= odd	
Description: Parity of RS232 interface of sensor.	
Parameter: int SA_RS232_StopBits	SA_RS232_StopBits
Direction: Up	
Valid values:	
1	
2	
Description: Stop bits of RS232 interface of sensor.	
Parameter: int SA_RS232_TimeoutSend	SA_RS232_TimeoutSend
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	
Parameter: int SA_RS232_TimeoutRecv	SA_RS232_TimeoutRecv
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	
Parameter: int SA_RS422_Baudrate	SA_RS422_Baudrate
Direction: Up	
Valid values:	
9600	
19200	
38400	
115200	
691200	
Unit: Baud	
Description: Baudrate of RS422 interface of sensor.	
Parameter: int SA_RS422_Parity	SA_RS422_Parity
Direction: Up	
Valid values:	
0= none	
1= even	
2= odd	
Description: Parity of RS422 interface of sensor.	

Parameter: int SA_RS422_StopBits	SA_RS422_StopBits
Direction: Up	
Valid values:	
1	
2	
Description:	Stop bits of RS422 interface of sensor.
Parameter: int SA_RS422_TimeoutSend	SA_RS422_TimeoutSend
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description:	No effect.
Parameter: int SA_RS422_TimeoutRecv	SA_RS422_TimeoutRecv
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description:	No effect.

12.4.2.5 Update_OptionData

Update option data at sensor.

This is a meta command which first calls [Read_OptionData](#), renames all answer parameters (SA_...) to command parameters (SP_...), overwrites the parameter set by the parameters from user and finally calls [Write_OptionData](#).

The parameters of this command are NOT obligatory. Missing parameters are not overwritten, so the sensor keeps the original values.

Parameter: int SP_MeasProgNumber	SP_MeasProgNumber
Direction: Down	
Valid values:	
0= EdgeHL	
1= EdgeLH	
2= DIA	
3= GAP	
4= SEG_2_4	
5= MULTISEG	
6= USER1	
7= USER2	
8= USER3	
9= USER4	
Description:	Measure program of sensor.
Parameter: int SP_Language	SP_Language
Direction: Down	
Valid values:	
0= German	
1= English	
Description:	Language of sensor.

Parameter: int SP_DispmesUnit	SP_DispmesUnit
Direction: Down	
Valid values:	
0= mm	
1= inch	
Description: Display measurement unit of sensor.	
Parameter: int SP_ErrorHandler	SP_ErrorHandler
Direction: Down	
Valid values:	
0= error output	
1= retain last value	
Description: If the sensor cannot measure values, it can output the last valid value or it can output an error values.	
Parameter: int SP_SerialOutFormat	SP_SerialOutFormat
Direction: Down	
Valid values:	
0= binary	
1= ASCII	
Description: Serial output format of sensor.	
Parameter: int SP_Ext_LaserSwitch	SP_Ext_LaserSwitch
Direction: Down	
Valid values:	
0= not active	
1= active	
Description: Enable/Disable external laser switch.	
Parameter: int SP_LaserIntensity	SP_LaserIntensity
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 100	
Unit: %	
Description: No effect.	
Parameter: int SP_Contrast	SP_Contrast
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 100	
Unit: %	
Description: Display contrast.	
Parameter: int SP_EdgeDetectThreshold	SP_EdgeDetectThreshold
Direction: Down	
Valid values:	
Minimum: 20	
Maximum: 90	
Unit: %	
Description: Contrast for edge detection.	

Parameter: int SP_Reserve_2	SP_Reserve_2
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved.	
Parameter: int SP_ActiveSerialIf	SP_ActiveSerialIf
Direction: Down	
Valid values:	
0 = RS422	
1 = RS232	
Description: Active serial interface of sensor.	
Parameter: int SP_RS232_Baudrate	SP_RS232_Baudrate
Direction: Down	
Valid values:	
9600	
19200	
38400	
115200	
Unit: Baud	
Description: Baudrate of RS232 interface of sensor.	
Parameter: int SP_RS232_Parity	SP_RS232_Parity
Direction: Down	
Valid values:	
0 = none	
1 = even	
2 = odd	
Description: Parity of RS232 interface of sensor.	
Parameter: int SP_RS232_StopBits	SP_RS232_StopBits
Direction: Down	
Valid values:	
1	
2	
Description: Stop bits of RS232 interface of sensor.	
Parameter: int SP_RS232_TimeoutSend	SP_RS232_TimeoutSend
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	
Parameter: int SP_RS232_TimeoutRecv	SP_RS232_TimeoutRecv
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	

Parameter: int SP_RS422_Baudrate	SP_RS422_Baudrate
Direction: Down	
Valid values:	
9600	
19200	
38400	
115200	
691200	
Unit: Baud	
Description: Baudrate of RS422 interface of sensor.	
Parameter: int SP_RS422_Parity	SP_RS422_Parity
Direction: Down	
Valid values:	
0= none	
1= even	
2= odd	
Description: Parity of RS422 interface of sensor.	
Parameter: int SP_RS422_StopBits	SP_RS422_StopBits
Direction: Down	
Valid values:	
1	
2	
Description: Stop bits of RS422 interface of sensor.	
Parameter: int SP_RS422_TimeoutSend	SP_RS422_TimeoutSend
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	
Parameter: int SP_RS422_TimeoutRecv	SP_RS422_TimeoutRecv
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 65535	
Description: No effect.	

12.4.2.6 Write_MeasProgData (WR MPR TO RAM)

Write measurement program data to sensor.

Parameter: int SP_UserMeasProgNumber	SP_UserMeasProgNumber
Direction: Down	
Valid values:	
6= USER1	
7= USER2	
8= USER3	
9= USER4	
Description: User measure program of sensor.	

Parameter: String SP_MeasProgName	SP_MeasProgName
Direction: Down	
Description: Measure program name of sensor.	
Parameter: double SP_AnalogOffset	SP_AnalogOffset
Direction: Down	
Valid values:	
Minimum: -10.0	
Maximum: 10.0	
Unit: V	
Description: Analog output offset of sensor.	
Parameter: double SP_AnalogGain	SP_AnalogGain
Direction: Down	
Valid values:	
Minimum: -4.0	
Maximum: 4.0	
Description: Analog output gain of sensor.	
Parameter: double SP_DisplayOffset	SP_DisplayOffset
Direction: Down	
Valid values:	
Minimum: -99.99	
Maximum: 99.99	
Unit: mm	
Description: Display offset of sensor.	
Parameter: double SP_DisplayGain	SP_DisplayGain
Direction: Down	
Valid values:	
Minimum: -2.0	
Maximum: 2.0	
Description: Display gain of sensor.	
Parameter: double SP_UpperLimit	SP_UpperLimit
Direction: Down	
Valid values:	
Minimum: -168.876	
Maximum: 168.876	
Unit: mm	
Description: Upper limit of sensor.	
Parameter: double SP_LowerLimit	SP_LowerLimit
Direction: Down	
Valid values:	
Minimum: -168.876	
Maximum: 168.876	
Unit: mm	
Description: Lower limit of sensor.	
Parameter: double SP_UpperWarning	SP_UpperWarning
Direction: Down	
Valid values:	
Minimum: -168.876	
Maximum: 168.876	
Unit: mm	
Description: Upper warning of sensor.	

Parameter: double SP_LowerWarning	SP_LowerWarning
Direction: Down	
Valid values:	
Minimum: -168.876	
Maximum: 168.876	
Unit: mm	
Description: Lower warning of sensor.	
Parameter: int SP_Reserve_3	SP_Reserve_3
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved.	
Parameter: int SP_MeasMode	SP_MeasMode
Direction: Down	
Valid values:	
0= Normal	
1= Max_Cont	
2= Min_Cont	
3= P-P_Cont	
4= Max_Trg	
5= Min_Trg	
6= P-P_Trg	
7= SC1_Trg	
Description: Measure Mode.	
Parameter: int SP_Median	SP_Median
Direction: Down	
Valid values:	
0= no Median	
3	
5	
7	
9	
Description: Median over n values.	
Parameter: int SP_Average_for_reading	SP_Average_for_reading
Direction: Down	
Valid values:	
1 to 128 sliding	
129 to 4096 recursive	
Description: Averaging mode and number of sensor.	
Parameter: int SP_Reserve_4	SP_Reserve_4
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved.	

Parameter: int SP_MeasObject	SP_MeasObject
Direction: Down	
Valid values:	
1= EdgeHL	
2= EdgeLH	
3= DIA	
4= GAP	
5= SEG_2_4	
6= MULTISEG	
Description: Measurement program.	
Parameter: int SP_NumberOfSegments	SP_NumberOfSegments
Direction: Down	
Valid values:	
1 for EdgeHL, EdgeLH, DIA, GAP and SEG_2_4	
2..4 for MULTISEG	
Description: Number of segments.	
Parameter: int SP_FrontEdge_Seg1	SP_FrontEdge_Seg1
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Front edge of segment 1.	
Parameter: int SP_FrontEdge_Seg2	SP_FrontEdge_Seg2
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Front edge of segment 2.	
Parameter: int SP_FrontEdge_Seg3	SP_FrontEdge_Seg3
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Front edge of segment 3.	
Parameter: int SP_FrontEdge_Seg4	SP_FrontEdge_Seg4
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Front edge of segment 4.	
Parameter: int SP_Reserve_6	SP_Reserve_6
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved.	

Parameter: int SP_Reserve_7	SP_Reserve_7
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved.	
Parameter: int SP_RearEdge_Seg1	SP_RearEdge_Seg1
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Rear edge of segment 1.	
Parameter: int SP_RearEdge_Seg2	SP_RearEdge_Seg2
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Rear edge of segment 2.	
Parameter: int SP_RearEdge_Seg3	SP_RearEdge_Seg3
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Rear edge of segment 3.	
Parameter: int SP_RearEdge_Seg4	SP_RearEdge_Seg4
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Rear edge of segment 4.	
Parameter: int SP_Reserve_9	SP_Reserve_9
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved.	
Parameter: int SP_Reserve_10	SP_Reserve_10
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved.	
Parameter: double SP_MasterValue	SP_MasterValue
Direction: Down	
Valid values:	
Minimum: -40.0	
Maximum: 40.0	
Unit: mm	
Description: Master value of sensor.	

12.4.2.7 Read_MeasProgData (RD MPR RAM)

Read measurement program data from sensor

Parameter: int SA_MeasProgNumber	SA_MeasProgNumber
Direction: Up	
Valid values:	
0= EdgeHL	
1= EdgeLH	
2= DIA	
3= GAP	
4= SEG_2_4	
5= MULTISEG	
6= USER1	
7= USER2	
8= USER3	
9= USER4	
Description: User measure program of sensor.	
Parameter: String SA_MeasProgName	SA_MeasProgName
Direction: Up	
Description: Measure program name of sensor.	
Parameter: double SA_AnalogOffset	SA_AnalogOffset
Direction: Up	
Valid values:	
Minimum: -10.0	
Maximum: 10.0	
Unit: V	
Description: Analog output offset of sensor.	
Parameter: double SA_AnalogGain	SA_AnalogGain
Direction: Up	
Valid values:	
Minimum: -4.0	
Maximum: 4.0	
Description: Analog output gain of sensor.	
Parameter: double SA_DisplayOffset	SA_DisplayOffset
Direction: Up	
Valid values:	
Minimum: -99.99	
Maximum: 99.99	
Unit: mm	
Description: Display offset of sensor.	
Parameter: double SA_DisplayGain	SA_DisplayGain
Direction: Up	
Valid values:	
Minimum: -2.0	
Maximum: 2.0	
Description: Display gain of sensor.	

Parameter: double SA_UpperLimit	SA_UpperLimit
Direction: Up	
Valid values:	
Minimum: -168.876	
Maximum: 168.876	
Unit: mm	
Description: Upper limit of sensor.	
Parameter: double SA_LowerLimit	SA_LowerLimit
Direction: Up	
Valid values:	
Minimum: -168.876	
Maximum: 168.876	
Unit: mm	
Description: Lower limit of sensor.	
Parameter: double SA_UpperWarning	SA_UpperWarning
Direction: Up	
Valid values:	
Minimum: -168.876	
Maximum: 168.876	
Unit: mm	
Description: Upper warning of sensor.	
Parameter: double SA_LowerWarning	SA_LowerWarning
Direction: Up	
Valid values:	
Minimum: -168.876	
Maximum: 168.876	
Unit: mm	
Description: Lower warning of sensor.	
Parameter: int SA_Reserve_3	SA_Reserve_3
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved.	
Parameter: int SA_MeasMode	SA_MeasMode
Direction: Up	
Valid values:	
0= Normal	
1= Max_Cont	
2= Min_Cont	
3= P-P_Cont	
4= Max_Trg	
5= Min_Trg	
6= P-P_Trg	
7= SC1_Trg	
Description: Measure Mode.	

Parameter: int SA_Median	SA_Median
Direction: Up	
Valid values:	
0= no Median	
3	
5	
7	
9	
Description: Median over n values.	
Parameter: int SA_Average_for_reading	SA_Average_for_reading
Direction: Up	
Valid values:	
1 to 128 sliding	
129 to 4096 recursive	
Description: Averaging mode and number of sensor.	
Parameter: int SA_Reserve_4	SA_Reserve_4
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved.	
Parameter: int SA_MeasObject	SA_MeasObject
Direction: Up	
Valid values:	
1= EdgeHL	
2= EdgeLH	
3= DIA	
4= GAP	
5= SEG_2_4	
6= MULTISEG	
Description: Measurement program.	
Parameter: int SA_NumberOfSegments	SA_NumberOfSegments
Direction: Up	
Valid values:	
1 for EdgeHL, EdgeLH, DIA, GAP and SEG_2_4	
2..4 for MULTISEG	
Description: Number of segments.	
Parameter: int SA_FrontEdge_Seg1	SA_FrontEdge_Seg1
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Front edge of segment 1.	
Parameter: int SA_FrontEdge_Seg2	SA_FrontEdge_Seg2
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Front edge of segment 2.	

Parameter: int SA_FrontEdge_Seg3	SA_FrontEdge_Seg3
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Front edge of segment 3.	
Parameter: int SA_FrontEdge_Seg4	SA_FrontEdge_Seg4
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Front edge of segment 4.	
Parameter: int SA_Reserve_6	SA_Reserve_6
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved.	
Parameter: int SA_Reserve_7	SA_Reserve_7
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved.	
Parameter: int SA_RearEdge_Seg1	SA_RearEdge_Seg1
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Rear edge of segment 1.	
Parameter: int SA_RearEdge_Seg2	SA_RearEdge_Seg2
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Rear edge of segment 2.	
Parameter: int SA_RearEdge_Seg3	SA_RearEdge_Seg3
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Rear edge of segment 3.	
Parameter: int SA_RearEdge_Seg4	SA_RearEdge_Seg4
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 80	
Description: Rear edge of segment 4.	

Parameter: int SA_Reserve_9	SA_Reserve_9
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved.	
Parameter: int SA_Reserve_10	SA_Reserve_10
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 0	
Description: Reserved.	
Parameter: double SA_MasterValue	SA_MasterValue
Direction: Up	
Valid values:	
Minimum: -40.0	
Maximum: 40.0	
Unit: mm	
Description: Master value of sensor.	

12.4.2.8 Set_LightRef (SET LIGHT REFERENCE TUNING)

Set the light reference.

12.4.2.9 Reset_LightRef (RESET LIGHT REFERENCE TUNING)

Reset the light reference.

12.4.2.10 Read_MinMax (RD MINMAX)

Read the minimum and maximum values from sensor.

Parameter: int SA_MinRaw	SA_MinRaw
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 65519	
Description: Minimum raw value.	
Parameter: int SA_MaxRaw	SA_MaxRaw
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 65519	
Description: Maximum raw value.	

Parameter: double SA_MinScaled SA_MinScaled

Direction: Up

Valid values:

Minimum: -0.4204872

Maximum: 40.4035128

Unit: mm

Description: Minimum scaled value.

Parameter: double SA_MaxScaled SA_MaxScaled

Direction: Up

Valid values:

Minimum: -0.4204872

Maximum: 40.4035128

Unit: mm

Description: Maximum scaled value.

12.4.2.11 Read_MinMax_Reset (RD MINMAX RESET)

Read the minimum and maximum values from sensor and reset the values.

Parameter: int SA_MinRaw SA_MinRaw

Direction: Up

Valid values:

Minimum: 0

Maximum: 65519

Description: Minimum raw value.

Parameter: int SA_MaxRaw SA_MaxRaw

Direction: Up

Valid values:

Minimum: 0

Maximum: 65519

Description: Maximum raw value.

Parameter: double SA_MinScaled SA_MinScaled

Direction: Up

Valid values:

Minimum: -0.4204872

Maximum: 40.4035128

Unit: mm

Description: Minimum scaled value.

Parameter: double SA_MaxScaled SA_MaxScaled

Direction: Up

Valid values:

Minimum: -0.4204872

Maximum: 40.4035128

Unit: mm

Description: Maximum scaled value.

12.4.3 Data output

12.4.3.1 General

12.4.3.1.1 Dat_Out_Off (STOP)

Switch off data output from sensor.

12.4.3.1.2 Dat_Out_On (START)

Switch on data output from sensor.

13 Commands for Eddy sensors

DT3909 and ESC4912 are no longer supported.

13.1 Commands for DT3100

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [TCP/IP](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_Settings](#) and [Get_SensorInfo](#) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib to calculate datarate and to scale data.

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor, from 0 to 16777215.
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command [Get_SensorInfo](#)).

13.1.1 Information

13.1.1.1 Get_Status (STS)

Retrieve information about the controller.

Parameter: int SA_CableCnt

SA_CableCnt

Direction: Up

Valid values:

Minimum: 0

Maximum: 2

Description: Number of cables.

Parameter: int SA_AvailableTargets

SA_AvailableTargets

Direction: Up

Valid values:

Bit combination of:

1 (Bit 0)= Ferro magnetic

2 (Bit 1)= Non ferro magnetic

4 (Bit 2)= Custom1

8 (Bit 3)= Custom2

128 (Bit 7)= Custom (EPxx-LC)

Description: Available targets at controller.

13.1.1.2 Get_Settings (SET)

Retrieve detailed information about the controller settings.

Parameter: int SA_MeasureMode	SA_MeasureMode
Direction: Up	
Valid values:	
0= Off	
1= Continuous	
2= Rising edge	
3= Falling edge	
4= High level	
5= Low level	
Description: Measure mode.	
Parameter: int SA_SRIndex	SA_SRIndex
Direction: Up	
Valid values:	
0= 3600 Hz	
1= 7200 Hz	
2= 14400 Hz	
Description: Samplerate index.	
Parameter: int SA_AvrType	SA_AvrType
Direction: Up	
Valid values:	
0= off	
1= Moving average	
2= Recursive average	
3= Median	
Description: Averaging type at controller.	
Parameter: int SA_AvrNbr	SA_AvrNbr
Direction: Up	
Valid values:	
0= 4 (moving + recursive), 3 (median)	
1= 8 (moving + recursive), 5 (median)	
2= 16 (moving + recursive), 7 (median)	
3= 32 (moving + recursive), 9 (median)	
Description: Averaging number at controller.	
Parameter: int SA.MeasureCount	SA.MeasureCount
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 9999	
Description: Number of values to measure.	
Parameter: int SA_CalibTarget	SA_CalibTarget
Direction: Up	
Valid values:	
1= Ferro magnetic	
2= Non ferro magnetic	
4= Custom1	
8= Custom2	
Description: Calibrated target at controller.	

13.1. Commands for DT3100

Parameter: String SA_TextField SA_TextField
Direction: Up
Valid values:
 Up to 32 ASCII characters, no semicolon or new line.
Description: ASCII text.

13.1.1.3 Get_Error (ERR)

Get error information from controller.

Parameter: int SA_Error SA_Error
Direction: Up
Valid values:
 Bit combination of:
 1 (Bit 0)= 1 wire shortcut
 2 (Bit 1)= No EEPROM or 1 wire at +5V
 4 (Bit 2)= Error reading EEPROM
 8 (Bit 3)= More than one EEPROM
 16 (Bit 4)= Invalid data at EEPROM
 32 (Bit 5)= Error at Three point calibration: Poti reached lower limit
 64 (Bit 6)= Error at Three point calibration: Poti reached upper limit
 128 (Bit 7)= Invalid data at internal EEPROM
Description: Error field.

13.1.1.4 Get_ControllerInfo (IND)

Get information about controller.

Parameter: String SA_ControllerSerialNumber SA_ControllerSerialNumber
Direction: Up
Valid values:
 Numeric value
Description: Serial number of controller.

Parameter: String SA_ControllerProductCode SA_ControllerProductCode
Direction: Up
Valid values:
 Numeric value
Description: Product code of controller.

Parameter: String SA_ControllerRevisionIndex SA_ControllerRevisionIndex
Direction: Up
Description: Revision index of controller.

Parameter: String SA_ControllerSoftwareVersion SA_ControllerSoftwareVersion
Direction: Up
Description: Software version of controller.

Parameter: int SA_ControllerOption SA_ControllerOption
Direction: Up
Description: Option of controller.

13.1. Commands for DT3100

Parameter: String SA_ControllerName	SA_ControllerName
Direction: Up	
Valid values:	
DT3100	
Description: Name of controller.	
13.1.1.5 Get_SensorInfo (SEN)	
Get information about attached sensor.	
Parameter: String SA_SensorSerialNumber	SA_SensorSerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of sensor.	
Parameter: String SA_SensorProductCode	SA_SensorProductCode
Direction: Up	
Valid values:	
Numeric value	
Description: Product code of sensor.	
Parameter: String SA_SensorRevisionIndex	SA_SensorRevisionIndex
Direction: Up	
Description: Revision index of sensor.	
Parameter: int SA_SensorOption	SA_SensorOption
Direction: Up	
Description: Option of sensor.	
Parameter: String SA_SensorName	SA_SensorName
Direction: Up	
Description: Name of sensor.	
Parameter: int SA_SensorCableLength	SA_SensorCableLength
Direction: Up	
Description: Length of integrated sensor cable.	
Parameter: int SA_SensorStartMeasRange	SA_SensorStartMeasRange
Direction: Up	
Description: Start of measurement range of sensor.	
Parameter: int SA_SensorMidMeasRange	SA_SensorMidMeasRange
Direction: Up	
Description: Mid of measurement range of sensor.	
Parameter: int SA_SensorEndMeasRange	SA_SensorEndMeasRange
Direction: Up	
Description: End of measurement range of sensor.	

13.1. Commands for DT3100

13.1.1.6 Get_CableInfo1 (CB1)

Get information about cable 1.

Parameter: String SA_CableSerialNumber	SA_CableSerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of cable.	
Parameter: String SA_CableProductCode	SA_CableProductCode
Direction: Up	
Valid values:	
Numeric value	
Description: Product code of cable.	
Parameter: String SA_CableRevisionIndex	SA_CableRevisionIndex
Direction: Up	
Valid values:	
Numeric value	
Description: Revision index of cable.	
Parameter: int SA_CableOption	SA_CableOption
Direction: Up	
Description: Option of cable.	
Parameter: int SA_CableLength	SA_CableLength
Direction: Up	
Description: Length of cable.	

13.1.1.7 Get_CableInfo2 (CB2)

Get information about cable 2.

Parameter: String SA_CableSerialNumber	SA_CableSerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of cable.	
Parameter: String SA_CableProductCode	SA_CableProductCode
Direction: Up	
Valid values:	
Numeric value	
Description: Product code of cable.	
Parameter: String SA_CableRevisionIndex	SA_CableRevisionIndex
Direction: Up	
Valid values:	
Numeric value	
Description: Revision index of cable.	

13.1. Commands for DT3100

Parameter: int SA_CableOption SA_CableOption

Direction: Up

Description: Option of cable.

Parameter: int SA_CableLength SA_CableLength

Direction: Up

Description: Length of cable.

13.1.1.8 DetectSensorChange (DSC)

Check if sensor was disconnected or reconnected since last check.

Parameter: int SA_SensorChanged SA_SensorChanged

Direction: Up

Valid values:

0= Sensor was not changed

1= Sensor was changed

Description: Flag if sensor was changed.

13.1.1.9 Set_TextField (ETF)

Set a text field at controller.

Parameter: String SP_TextField SP_TextField

Direction: Down

Valid values:

Up to 32 ASCII characters, no semicolon or new line.

Description: ASCII text.

13.1.1.10 Get_TextField (ETF?)

Get a text field at controller.

Parameter: String SA_TextField SA_TextField

Direction: Up

Valid values:

Up to 32 ASCII characters, no semicolon or new line.

Description: ASCII text.

13.1.2 Parameter Management

13.1.2.1 Save_Settings (SSE)

Save settings at controller.

13.1.2.2 Read_Settings (RSE)

Read settings at controller.

13.1.2.3 Set_Defaults (DSE)

Set and retrieve default settings at controller.

Parameter: int SA_MeasureMode	SA_MeasureMode
Direction: Up	
Valid values:	
0= Off	
1= Continuous	
2= Rising edge	
3= Falling edge	
4= High level	
5= Low level	
Description: Measure mode.	
Parameter: int SA_SRIndex	SA_SRIndex
Direction: Up	
Valid values:	
0= 3600 Hz	
1= 7200 Hz	
2= 14400 Hz	
Description: Samplerate index.	
Parameter: int SA_AvrType	SA_AvrType
Direction: Up	
Valid values:	
0= off	
1= Moving average	
2= Recursive average	
3= Median	
Description: Averaging type at controller.	
Parameter: int SA_AvrNbr	SA_AvrNbr
Direction: Up	
Valid values:	
0= 4 (moving + recursive), 3 (median)	
1= 8 (moving + recursive), 5 (median)	
2= 16 (moving + recursive), 7 (median)	
3= 32 (moving + recursive), 9 (median)	
Description: Averaging number at controller.	
Parameter: int SA.MeasureCount	SA.MeasureCount
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 9999	
Description: Number of values to measure.	
Parameter: int SA_CalibTarget	SA_CalibTarget
Direction: Up	
Valid values:	
1= Ferro magnetic	
2= Non ferro magnetic	
4= Custom1	
8= Custom1	
Description: Calibrated target at controller.	

Parameter: String SA_TextField SA_TextField

Direction: Up

Valid values:

Up to 32 ASCII characters, no semicolon or new line.

Description: ASCII text.

13.1.3 Display

13.1.3.1 Set_AppLanguage (LNG)

Set language of java applet at controller.

Parameter: int SP_ApplicationLanguage

SP_ApplicationLanguage

Direction: Down

Valid values:

0= System

1= English

2= German

3..255 reserved

Description: Language of java applet.

13.1.3.2 Get_AppLanguage (LNG?)

Get language of java applet from controller.

Parameter: int SA_ApplicationLanguage

SA_ApplicationLanguage

Direction: Up

Valid values:

0= System

1= English

2= German

3..255 reserved

Description: Language of java applet.

13.1.3.3 Set_AppYScale (SCA)

Set display scaling settings of java applet at controller.

Parameter: int SP_ApplicationScaleMode

SP_ApplicationScaleMode

Direction: Down

Valid values:

0= User defined

1= Full scale

Description: Scaling mode of java applet.

Parameter: int SP_ApplicationYMin

SP_ApplicationYMin

Direction: Down

Valid values:

Minimum: 0

Maximum: 65535

Description: Lower bound at display of java applet.

13.1. Commands for DT3100

Parameter: int SP_ApplicationYMax SP_ApplicationYMax
Direction: Down
Valid values:
Minimum: 0
Maximum: 65535
Description: Upper bound at display of java applet.

13.1.3.4 Get_ApplYScale (SCA?)

Get display scaling settings of java applet at controller.

Parameter: int SA_ApplicationScaleMode SA_ApplicationScaleMode
Direction: Up
Valid values:
0= User defined
1= Full scale
Description: Scaling mode of java applet.

Parameter: int SA_ApplicationYMin SA_ApplicationYMin
Direction: Up
Valid values:
Minimum: 0
Maximum: 65535
Description: Lower bound at display of java applet.

Parameter: int SA_ApplicationYMax SA_ApplicationYMax
Direction: Up
Valid values:
Minimum: 0
Maximum: 65535
Description: Upper bound at display of java applet.

13.1.3.5 Set_ApplDispMode (DMD)

Set display mode of java applet at controller.

Parameter: int SP_ApplicationDisplayMode SP_ApplicationDisplayMode
Direction: Down
Valid values:
0= Roll
1= Single frame
Description: Display mode of java applet.

13.1.3.6 Get_ApplDispMode (DMD?)

Get display mode of java applet at controller.

Parameter: int SA_ApplicationDisplayMode SA_ApplicationDisplayMode
Direction: Up
Valid values:
0= Roll
1= Single frame
Description: Display mode of java applet.

13.1.4 Measurement

13.1.4.1 Set_SRIndex (SRA)

Set the samplerate for data acquisition.

Parameter: int SP_SRIndex

SP_SRIndex

Direction: Down

Valid values:

0= 3600 Hz
1= 7200 Hz
2= 14400 Hz

Description: Samplerate index

13.1.4.2 Get_SRIndex (SRA?)

Get the current samplerate.

Parameter: int SA_SRIndex

SA_SRIndex

Direction: Up

Valid values:

0= 3600 Hz
1= 7200 Hz
2= 14400 Hz

Description: Samplerate index.

13.1.4.3 Get_Measure (GMD)

Retrieve one value from sensor, even if trigger is active.

13.1.4.4 Set_MeasureCnt (VTT)

Set the number of values to measure at several measure modes.

Parameter: int SP_MeasureCount

SP_MeasureCount

Direction: Down

Valid values:

Minimum: 1
Maximum: 9999

Description: Number of values to measure.

13.1.4.5 Get_MeasureCnt (VTT?)

Get the number of values to measure at several measure modes.

Parameter: int SA_MeasureCount

SA_MeasureCount

Direction: Up

Valid values:

Minimum: 1
Maximum: 9999

Description: Number of values to measure.

13.1.4.6 Set_MeasureMode (MMD)

Set the measure mode at sensor.

Parameter: int SP_MeasureMode

SP_MeasureMode

Direction: Down

Valid values:

- 0= Off
- 1= Continuous
- 2= Rising edge
- 3= Falling edge
- 4= High level
- 5= Low level

Description: Measure mode.

13.1.4.7 Get_MeasureMode (MMD?)

Retrieve the measure mode from sensor.

Parameter: int SA_MeasureMode

SA_MeasureMode

Direction: Up

Valid values:

- 0= Off
- 1= Continuous
- 2= Rising edge
- 3= Falling edge
- 4= High level
- 5= Low level

Description: Measure mode.

13.1.4.8 Set_AvrType (AVT)

Set the averaging type at controller.

Parameter: int SP_AvrType

SP_AvrType

Direction: Down

Valid values:

- 0= off
- 1= Moving average
- 2= Recursive average
- 3= Median

Description: Averaging type.

13.1.4.9 Get_AvrType (AVT?)

Retrieve the averaging type at controller.

Parameter: int SA_AvrType

SA_AvrType

Direction: Up

Valid values:

- 0= off
- 1= Moving average
- 2= Recursive average
- 3= Median

Description: Averaging type at controller.

13.1. Commands for DT3100

13.1.4.10 Set_AvrNbr (AVN)

Set the averaging number at controller.

Parameter: int SP_AvrNbr SP_AvrNbr
Direction: Down
Valid values:
 0= 4 (moving + recursive), 3 (median)
 1= 8 (moving + recursive), 5 (median)
 2= 16 (moving + recursive), 7 (median)
 3= 32 (moving + recursive), 9 (median)
Description: Averaging number.

13.1.4.11 Get_AvrNbr (AVN?)

Retrieve the averaging number at controller.

Parameter: int SA_AvrNbr SA_AvrNbr
Direction: Up
Valid values:
 0= 4 (moving + recursive), 3 (median)
 1= 8 (moving + recursive), 5 (median)
 2= 16 (moving + recursive), 7 (median)
 3= 32 (moving + recursive), 9 (median)
Description: Averaging number at controller.

13.1.4.12 Get_ControllerTemp (GCT)

Get temperature at controller.

Parameter: double SA_ControllerTemperature SA_ControllerTemperature
Direction: Up
Valid values:
Minimum: -55.0
Maximum: 125.0
Unit: °C
Description: Temperature at controller.

13.1.4.13 Get_SensorTemp (GST)

Get temperature at sensor.

Parameter: double SA_SensorTemperature SA_SensorTemperature
Direction: Up
Unit: °C
Description: Temperature at sensor.

13.1.5 Linearization

13.1.5.1 Get_CalibState (CST)

Get state of calibration progress.

Parameter: int SA_CalibState

SA_CalibState

Direction: Up

Valid values:

Minimum: 0

Maximum: 6

Description: Calibration state.

13.1.5.2 Set_CalibTarget (TAR)

Set target for calibration.

Parameter: int SP_CalibTarget

SP_CalibTarget

Direction: Down

Valid values:

1= Ferro magnetic

2= Non ferro magnetic

4= Custom1

8= Custom2

Description: Calibration target.

13.1.5.3 Get_CalibTarget (TAR?)

Get target for calibration.

Parameter: int SA_CalibTarget

SA_CalibTarget

Direction: Up

Valid values:

1= Ferro magnetic

2= Non ferro magnetic

4= Custom1

8= Custom2

Description: Calibration target.

13.1.5.4 Get_CalibProgress (BSY)

Get calibration progress.

Parameter: int SA_CalibProgress

SA_CalibProgress

Direction: Up

Valid values:

Minimum: 0

Maximum: 255

Description: Calibration progress.

13.1.5.5 Set_CalibStart (SMR)

Set start value of measurement range for calibration.

Parameter: int SP_CalibStart SP_CalibStart
Direction: Down
Valid values:
Minimum: -1
Maximum: 10000
Unit: μm
Description: Start value of measurement range. -1 means do not send this parameter.

13.1.5.6 Set_CalibEnd (EMR)

Set end value of measurement range for calibration.

Parameter: int SP_CalibEnd SP_CalibEnd
Direction: Down
Valid values:
Minimum: -1
Maximum: 10000
Unit: μm
Description: End value of measurement range. -1 means do not send this parameter.

13.1.5.7 Set_CalibMid (MMR)

Set mid value of measurement range for calibration.

Parameter: int SP_CalibMid SP_CalibMid
Direction: Down
Valid values:
Minimum: -1
Maximum: 10000
Unit: μm
Description: Mid value of measurement range. -1 means do not send this parameter.

13.1.5.8 Exit_Calib (ECA)

Abort active calibration progress.

13.1.5.9 Set_FactoryCalib (FCA)

Load factory default calibration.

13.1. Commands for DT3100

13.1.5.10 Set_Poti (WPT)

Set values of digital potis at controller.

Parameter: int SP_NullPoti SP_NullPoti

Direction: Down

Valid values:

Minimum: 1

Maximum: 4095

Description: Value of NullPoti.

Parameter: int SP_GainPoti SP_GainPoti

Direction: Down

Valid values:

Minimum: 1

Maximum: 4095

Description: Value of GainPoti.

Parameter: int SP_LinPoti SP_LinPoti

Direction: Down

Valid values:

Minimum: 1

Maximum: 4095

Description: Value of LinPoti.

13.1.5.11 Get_Poti (RPT)

Get values of digital potis from controller.

Parameter: int SA_NullPoti SA_NullPoti

Direction: Up

Valid values:

Minimum: 1

Maximum: 4095

Description: Value of NullPoti.

Parameter: int SA_GainPoti SA_GainPoti

Direction: Up

Valid values:

Minimum: 1

Maximum: 4095

Description: Value of GainPoti.

Parameter: int SA_LinPoti SA_LinPoti

Direction: Up

Valid values:

Minimum: 1

Maximum: 4095

Description: Value of LinPoti.

13.1.5.12 Save_Poti (SPT)

Save values of digital potis at controller.

13.1. Commands for DT3100

13.1.6 Internal commands

13.1.6.1 Update_Firmware

Update firmware version at controller.

Attention! This function can take approx. 3 minutes.

Parameter: Binary data SP_FirmwareFile

SP_FirmwareFile

Direction: Down

Valid values:

Firmware file as binary data.

Description: Firmware version, read from file.

Parameter: int SA_Result

SA_Result

Direction: Up

Valid values:

0= Failed

1= Success

Description: Result of firmware update.

13.1.6.2 Get_BootLoaderMode (BOT)

Get state of boot loader at controller. This is an internal command. It should not be used by the customer.

Parameter: int SA_BootLoaderMode

SA_BootLoaderMode

Direction: Up

Valid values:

0= Inactive

1= Active

Description: State of boot loader.

13.1.6.3 Get_BootLoaderVer (VBL)

Get version of boot loader at controller.

Parameter: String SA_BootLoaderVersion

SA_BootLoaderVersion

Direction: Up

Valid values:

Numeric value

Description: Version of boot loader.

13.1.6.4 Get_FirmwareVer (VFW)

Get version of firmware at controller.

Parameter: String SA_FirmwareVersion

SA_FirmwareVersion

Direction: Up

Valid values:

Numeric value

Description: Version of firmware.

13.1. Commands for DT3100

13.1.6.5 Start_BootLoader (SBL)

Start boot loader at controller. This is an internal command. It should not be used by the customer.

13.1.6.6 Erase_Flash (EFL)

Erase flash at controller. This is an internal command. It should not be used by the customer.

13.1.6.7 Write_Sensor (WDS)

Write data to Sensor EEPROM at controller. This is an internal command. It should not be used by the customer.

Parameter: Binary data SP_SensorData

SP_SensorData

Direction: Down

Valid values:

128 bytes

Description: Binary data.

13.1.6.8 Read_Sensor (RDS)

Read data from Sensor EEPROM at controller. This is an internal command. It should not be used by the customer.

Parameter: Binary data SA_SensorData

SA_SensorData

Direction: Up

Valid values:

128 bytes

Description: Binary data.

13.1.6.9 Write_AnalogEEPROM (WA1+2)

Write data to analog EEPROM at controller. This is an internal command. It should not be used by the customer.

Parameter: Binary data SP_AnalogEEPROMData

SP_AnalogEEPROMData

Direction: Down

Valid values:

256 bytes

Description: Binary data.

13.1.6.10 Read_AnalogEEPROM (RA1+2)

Read data from analog EEPROM at controller. This is an internal command. It should not be used by the customer.

Parameter: Binary data SA_AnalogEEPROMData

SA_AnalogEEPROMData

Direction: Up

Valid values:

256 bytes

Description: Binary data.

13.1. Commands for DT3100

13.1.6.11 Write_SensorParam (WSP)

Write sensor parameter controller. This is an internal command. It should not be used by the customer.

Parameter: Binary data SP_SensorParamData SP_SensorParamData
Direction: Down
Valid values:
 78 bytes
Description: Binary data.

13.1.6.12 Read_SensorParam (RSP)

Read sensor parameter from controller. This is an internal command. It should not be used by the customer.

Parameter: Binary data SA_SensorParamData SA_SensorParamData
Direction: Up
Valid values:
 78 bytes
Description: Binary data.

13.1.6.13 Write_AnalogParam (WP1+2)

Write analog parameter to controller. This is an internal command. It should not be used by the customer.

Parameter: Binary data SP_AnalogParamData SP_AnalogParamData
Direction: Down
Valid values:
 256 bytes
Description: Binary data.

13.1.6.14 Read_AnalogParam (RP1+2)

Read analog parameter from controller. This is an internal command. It should not be used by the customer.

Parameter: Binary data SA_AnalogParamData SA_AnalogParamData
Direction: Up
Valid values:
 256 bytes
Description: Binary data.

13.1.6.15 Exec_SysCalibOff (SCO)

Assign system calibration offset. This is an internal command. It should not be used by the customer.

13.1.6.16 Exec_SysCalibGain (SCG)

Assign system calibration gain. This is an internal command. It should not be used by the customer.

13.1.6.17 Write_Index (WRI)

Write index data to controller. This is an internal command. It should not be used by the customer.

Parameter: Binary data SP_IndexData SP_IndexData
Direction: Down
Valid values:
 20 bytes
Description: Binary data.

13.1.6.18 Read_Index (RDI)

Read index data from controller. This is an internal command. It should not be used by the customer.

Parameter: Binary data SA_IndexData SA_IndexData
Direction: Up
Valid values:
 20 bytes
Description: Binary data.

13.1.6.19 Write_IndexDigital (WRD)

Write digital index data to controller. This is an internal command. It should not be used by the customer.

Parameter: Binary data SP_DigitalIndexData SP_DigitalIndexData
Direction: Down
Valid values:
 128 bytes
Description: Binary data.

13.1.6.20 Read_IndexDigital (RDD)

Read digital index data from controller. This is an internal command. It should not be used by the customer.

Parameter: Binary data SA_DigitalIndexData SA_DigitalIndexData
Direction: Up
Valid values:
 128 bytes
Description: Binary data.

13.1. Commands for DT3100

13.1.6.21 Get_ButtonState (CBT)

Read button states at controller. This is an internal command. It should not be used by the customer.

Parameter: int SA_ButtonState SA_ButtonState
Direction: Up
Valid values:
 Bit combination of:
 0= Nothing pressed
 1 (Bit 0)= Button 1 pressed
 2 (Bit 1)= Button 2 pressed
 4 (Bit 2)= Button 3 pressed
Description: Button states.

13.1.6.22 Set_Multiplexer (MUX)

Set multiplexer state at controller. This is an internal command. It should not be used by the customer.

Parameter: int SP_Multiplexer SP_Multiplexer
Direction: Down
Valid values:
 0= -U-DEMO
 1= -U-TEMP
 2= U-PHASE
 3= -U-LOG-A
Description: Multiplexer state.

Parameter: double SA_RawValue SA_RawValue
Direction: Up
Valid values:
Minimum: 0.0
Maximum: 65535.0
Description: Raw value of multiplexer channel.

Parameter: double SA_ScaledValue SA_ScaledValue
Direction: Up
Valid values:
Minimum: -5.0
Maximum: +5.0
Unit: V
Description: Scaled value of multiplexer channel.

14 Commands for Capa sensors

14.1 Commands for DT6100

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [TCP/IP](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_Status](#) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib to calculate datarate.

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor, from 0 to 16777215.
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command [Use_Defaults](#) with parameter [IP_Range](#)).

14.1.1 Measurement

14.1.1.1 Set_SRIndex (SRA)

Set the samplerate for data acquisition.

Parameter: int SP_SRIndex

SP_SRIndex

Direction: Down

Valid values:

0= 2.60 Hz
 1= 5.21 Hz
 2= 10.42 Hz
 3= 15.63 Hz
 4= 26.04 Hz
 5= 31.25 Hz
 6= 52.08 Hz
 7= 62.50 Hz
 8= 104.17 Hz
 9= 520.83 Hz
 10= 1041.67 Hz
 11= 2083.33 Hz
 12= 3906.25 Hz
 13= 7812.50 Hz

Description: Samplerate index

14.1. Commands for DT6100

14.1.1.2 Get_SRIndex (SRA?)

Get the current samplerate.

Parameter: int SA_SRIndex SA_SRIndex

Direction: Up

Valid values:

- 0= 2.60 Hz
- 1= 5.21 Hz
- 2= 10.42 Hz
- 3= 15.63 Hz
- 4= 26.04 Hz
- 5= 31.25 Hz
- 6= 52.08 Hz
- 7= 62.50 Hz
- 8= 104.17 Hz
- 9= 520.83 Hz
- 10= 1041.67 Hz
- 11= 2083.33 Hz
- 12= 3906.25 Hz
- 13= 7812.50 Hz

Description: Samplerate index.

14.1.1.3 Set_Trigger (TRG)

Activate/disable the trigger at sensor.

Parameter: int SP_TrgMode SP_TrgMode

Direction: Down

Valid values:

- 0= disabled
- 1= active

Description: Trigger active/disabled.

14.1.1.4 Get_Trigger (TRG?)

Retrieve the trigger mode at sensor.

Parameter: int SA_TrgMode SA_TrgMode

Direction: Up

Valid values:

- 0= disabled
- 1= active

Description: Trigger active/disabled.

14.1.1.5 Get_Measure (GMD)

Retrieve one value from sensor, even if trigger is active.

14.1. Commands for DT6100

14.1.1.6 Set_AvrType (AVT)

Set the averaging type at controller.

Parameter: int SP_AvrType SP_AvrType
Direction: Down
Valid values:
 0= off
 1= Moving average
 2= Mean (arithmetic)
 3= Median
Description: Averaging type.

14.1.1.7 Get_AvrType (AVT?)

Retrieve the averaging type at controller.

Parameter: int SA_AvrType SA_AvrType
Direction: Up
Valid values:
 0= off
 1= Moving average
 2= Mean (arithmetic)
 3= Median
Description: Averaging type at controller.

14.1.1.8 Set_AvrNbr (AVN)

Set the averaging number at controller.

Parameter: int SP_AvrNbr SP_AvrNbr
Direction: Down
Valid values:
Minimum: 2
Maximum: 8
Description: Averaging number.

14.1.1.9 Get_AvrNbr (AVN?)

Retrieve the averaging number at controller.

Parameter: int SA_AvrNbr SA_AvrNbr
Direction: Up
Valid values:
Minimum: 2
Maximum: 8
Description: Averaging number at controller.

14.1. Commands for DT6100

14.1.2 Linearization

14.1.2.1 Set_LinMode (LIN)

Set the linearisation mode for sensor.

Parameter: int SP_LinMode

SP_LinMode

Direction: Down

Valid values:

- 0= off
- 1= offset correction
- 2= 2 point linearization
- 3= 3 point linearization
- 4= 5 point linearization
- 5= 10 point linearization

Description: Linearisation mode.

14.1.2.2 Get_LinMode (LIN?)

Retrieve the linearisation mode for sensor.

Parameter: int SA_LinMode

SA_LinMode

Direction: Up

Valid values:

- 0= off
- 1= offset correction
- 2= 2 point linearization
- 3= 3 point linearization
- 4= 5 point linearization
- 5= 10 point linearization

Description: Linearisation mode.

14.1.2.3 Set_LinPoint (SLP)

Set a linearisation point at sensor.

Parameter: int SP_LinPos

SP_LinPos

Direction: Down

Valid values:

- 1= at 10%
- 2= at 20%
- 3= at 30%
- 4= at 40%
- 5= at 50%
- 6= at 60%
- 7= at 70%
- 8= at 80%
- 9= at 90%
- 10= at 100%

Description: Linearisation position to be set for.

14.1.2.4 Get_LinPoint (GLP)

Get a linearisation point at sensor.

Parameter: int SP_LinPos

SP_LinPos

Direction: Down

Valid values:

- 1= at 10%
- 2= at 20%
- 3= at 30%
- 4= at 40%
- 5= at 50%
- 6= at 60%
- 7= at 70%
- 8= at 80%
- 9= at 90%
- 10= at 100%

Description: Linearisation position to be get for.

Parameter: double SA_LinPoint

SA_LinPoint

Direction: Up

Valid values:

- Minimum:** 0
- Maximum:** 16777215 (0xffffffff)

Description: Linearisation point.

14.1.3 Information

14.1.3.1 Get_Status (STS)

Retrieve detailed information about the sensor.

Parameter: int SA_SRIndex

SA_SRIndex

Direction: Up

Valid values:

- 0= 2.60 Hz
- 1= 5.21 Hz
- 2= 10.42 Hz
- 3= 15.63 Hz
- 4= 26.04 Hz
- 5= 31.25 Hz
- 6= 52.08 Hz
- 7= 62.50 Hz
- 8= 104.17 Hz
- 9= 520.83 Hz
- 10= 1041.67 Hz
- 11= 2083.33 Hz
- 12= 3906.25 Hz
- 13= 7812.50 Hz

Description: Samplerate index.

14.1. Commands for DT6100

Parameter: int SA_AvrType	SA_AvrType
Direction: Up	
Valid values:	
0= off	
1= Moving average	
2= Mean (arithmetic)	
3= Median	
Description: Averaging type at controller.	
Parameter: int SA_AvrNbr	SA_AvrNbr
Direction: Up	
Valid values:	
Minimum: 2	
Maximum: 8	
Description: Averaging number at controller.	
Parameter: int SA_TrgMode	SA_TrgMode
Direction: Up	
Valid values:	
0= disabled	
1= active	
Description: Trigger active/disabled.	
Parameter: int SA_LinMode	SA_LinMode
Direction: Up	
Valid values:	
0= off	
1= offset correction	
2= 2 point linearization	
3= 3 point linearization	
4= 5 point linearization	
5= 10 point linearization	
Description: Linearisation mode.	

14.1.3.2 Get_Version (VER)

Retrieve the sensor software version.

Parameter: String SA_Version	SA_Version
Direction: Up	
Description: Software version of the controller.	

14.1.4 Parameter management

14.1.4.1 Save_Setup (SSU)

Save the current setup of controller to flash.

14.1.4.2 Read_Setup (RSU)

Read the setup from controller flash.

14.1.4.3 Factory_Defaults (FDE)

Restore the sensor to factory defaults.

To save the default values call Save_Setup too. The new parameters are returned.

Parameter: int SA_SRIndex

SA_SRIndex

Direction: Up

Valid values:

- 0= 2.60 Hz
- 1= 5.21 Hz
- 2= 10.42 Hz
- 3= 15.63 Hz
- 4= 26.04 Hz
- 5= 31.25 Hz
- 6= 52.08 Hz
- 7= 62.50 Hz
- 8= 104.17 Hz
- 9= 520.83 Hz
- 10= 1041.67 Hz
- 11= 2083.33 Hz
- 12= 3906.25 Hz
- 13= 7812.50 Hz

Description: Samplerate index.

Parameter: int SA_AvrType

SA_AvrType

Direction: Up

Valid values:

- 0= off
- 1= Moving average
- 2= Mean (arithmetic)
- 3= Median

Description: Averaging type at controller.

Parameter: int SA_AvrNbr

SA_AvrNbr

Direction: Up

Valid values:

- Minimum:** 2
- Maximum:** 8

Description: Averaging number at controller.

Parameter: int SA_TrgMode

SA_TrgMode

Direction: Up

Valid values:

- 0= disabled
- 1= active

Description: Trigger active/disabled.

Parameter: int SA_LinMode

SA_LinMode

Direction: Up

Valid values:

- 0= off
- 1= offset correction
- 2= 2 point linearization
- 3= 3 point linearization
- 4= 5 point linearization
- 5= 10 point linearization

Description: Linearisation mode.

14.1.5 Internal commands

14.1.5.1 Set_Coefficient (SCO)

Set coefficient at controller. This is an internal command. It should not be used by the customer.

Parameter: int SP_CoeffParam

SP_CoeffParam

Direction: Down

Valid values:

Minimum: 0

Maximum: 16

Description: Coeffizient parameter.

Parameter: double SP_Coeffizient

SP_Coeffizient

Direction: Down

Valid values:

Minimum: 0

Maximum: 16777215 (0xffffffff)

Description: Coeffizient.

14.1.5.2 Get_Coefficient (GCO)

Get coefficient from controller. This is an internal command. It should not be used by the customer.

Parameter: int SP_CoeffParam

SP_CoeffParam

Direction: Down

Valid values:

Minimum: 0

Maximum: 16

Description: Coeffizient parameter.

Parameter: double SA_Coeffizient

SA_Coeffizient

Direction: Up

Valid values:

Minimum: 0

Maximum: 16777215 (0xffffffff)

Description: Coeffizient.

14.1.6 Special commands

14.1.6.1 Use_Defaults

This command tells the driver to use default values to operate with sensor data. If some parameters are not specified they are not changed. The sensor is not affected by this command.

Parameter: double IP_Range

IP_Range

Direction: Down

Valid values:

Minimum: -1.79769e+308 (-DBL_MAX)

Maximum: 1.79769e+308 (DBL_MAX)

Unit: μm or mm

Description: Tells the driver the range of sensor. It is used to scale the raw sensor values into μm or mm. If it is zero, no scaling is done.

14.1.6.2 Get_DrvSetting

Returns the current settings of the driver used for operating with sensor data. It is the opposite of Use_Defaults. The sensor is not affected by this command.

Parameter: double IA_Range

IA_Range

Direction: Up

Valid values:

Minimum: -1.79769e+308 (-DBL_MAX)

Maximum: 1.79769e+308 (DBL_MAX)

Unit: μm or mm

Description: The range of sensor, used by driver to scale values into μm or mm.

14.2 Commands for DT6120

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- RS232 (RS485/USB converter).

If first bit of IP_AutomaticMode is set (1), MEDAQLib calls automatically sensor command [Read_AllBlocks](#) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib to calculate datarate and to assign values.

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor, range is depending on data format.
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command [Read_AllBlocks](#)).

The values of selected channels are filled in the arrays one after another. Each array always starts with first selected channel.

14.2.1 Measurement

14.2.1.1 Set_Samplerate

Set the samplerate for data acquisition.

Parameter: double SP_Measrate

SP_Measrate

Direction: Down

Valid values:

5

10

20

40

80

160

320

639.79526551503523

1000

2000

Unit: Hz

Description: Desired samplerate

14.2. Commands for DT6120

Parameter: double SA_Measrate SA_Measrate

Direction: Up

Valid values:

- 5
- 10
- 20
- 40
- 80
- 160
- 320
- 639.79526551503523
- 1000
- 2000

Unit: Hz

Description: Real samplerate at sensor

14.2.1.2 Get_Samplerate

Get the current samplerate.

Parameter: double SA_Measrate SA_Measrate

Direction: Up

Valid values:

- 5
- 10
- 20
- 40
- 80
- 160
- 320
- 639.79526551503523
- 1000
- 2000

Unit: Hz

Description: Real samplerate at sensor

14.2.1.3 Set_Averaging

Set data averaging at sensor.

Parameter: int SP_AveragingType SP_AveragingType

Direction: Down

Valid values:

- 0= off
- 1= Moving average
- 2= Mean (arithmetic)
- 3= Median
- 4= Dynamic noise rejection

Description: Averaging type.

Parameter: int SP_MovingCount SP_MovingCount

Direction: Down

Valid values:

Minimum: 2

Maximum: 8

Description: Number of value for the averaging window. This parameter is only used at moving average.

Parameter: int SP_MeanCount SP_MeanCount

Direction: Down

Valid values:

Minimum: 2

Maximum: 8

Description: Number of values for arithmetic mean. This parameter is only used at arithmetic mean.

Parameter: int SP_MedianCount SP_MedianCount

Direction: Down

Valid values:

Minimum: 2

Maximum: 8

Description: Number of values to build median. This parameter is only used at median.

14.2.1.4 Get_Averaging

Retrieve the averaging type at sensor.

Parameter: int SA_AveragingType SA_AveragingType

Direction: Up

Valid values:

0= off

1= Moving average

2= Mean (arithmetic)

3= Median

4= Dynamic noise rejection

Description: Averaging type at sensor.

Parameter: int SA_MovingCount SA_MovingCount

Direction: Up

Valid values:

Minimum: 2

Maximum: 8

Description: Number of value for the averaging window. This parameter is only available at moving average.

Parameter: int SA_MeanCount SA_MeanCount

Direction: Up

Valid values:

Minimum: 2

Maximum: 8

Description: Number of values for arithmetic mean. This parameter is only available at arithmetic mean.

Parameter: int SA_MedianCount SA_MedianCount
Direction: Up
Valid values:
Minimum: 2
Maximum: 8
Description: Number of values to build median. This parameter is only available at median.

14.2.1.5 Get_Measure

Retrieve a data block from controller. To get the values from MEDAQLib, use the functions [Poll](#) or [TransferData](#).

14.2.2 Data output

14.2.2.1 Set_Range

Write the measurement range to sensor.

Parameter: double SP_Range SP_Range
Direction: Down
Valid values:
Minimum: -3.40282e+38 (-FLT_MAX)
Maximum: 3.40282e+38 (FLT_MAX)
Unit: μm
Description: Range of sensor.

14.2.3 Interfaces

14.2.3.1 Set_Baudrate

Set baudrate of sensor for serial RS485 communication. After receiving the answer of the command, the baudrate of the local serial interface is changed, too.

Parameter: int SP_SensorBaudrate SP_SensorBaudrate
Direction: Down
Valid values:
 921600
 460800
 230400
 115200
 9600
Unit: Baud
Description: Baudrate of sensor.

14.2. Commands for DT6120

Parameter: int SA_SensorBaudrate SA_SensorBaudrate
Direction: Up
Valid values:
 921600
 460800
 230400
 115200
 9600
Unit: Baud
Description: Real baudrate at sensor.

14.2.3.2 Get_Baudrate

Get baudrate of controller for serial RS485 communication.

Parameter: int SA_SensorBaudrate SA_SensorBaudrate
Direction: Up
Valid values:
 921600
 460800
 230400
 115200
 9600
Unit: Baud
Description: Real baudrate at sensor.

14.2.3.3 Set_SensorAddress

Changes the address at sensor which is used to communicate over RS485 bus.

Parameter: int SP_SensorAddress SP_SensorAddress
Direction: Down
Valid values:
Minimum: 1
Maximum: 126
Description: Address of sensor.

14.2.4 Information

14.2.4.1 Get_SensorInfo

Retrieve information about the connected sensor.

Parameter: String SA_ArticleNumber SA_ArticleNumber
Direction: Up
Valid values:
 Numeric value
Description: Article number of the connected sensor.

Parameter: String SA_SensorName	SA_SensorName
Direction: Up	
Description: Name of the connected sensor.	
Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the connected sensor.	
Parameter: double SA_Offset	SA_Offset
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Unit: See parameter SA_Unit	
Description: Offset of the connected sensor.	
Parameter: double SA_Range	SA_Range
Direction: Up	
Valid values:	
Minimum: -3.40282e+38 (-FLT_MAX)	
Maximum: 3.40282e+38 (FLT_MAX)	
Unit: See parameter SA_Unit	
Description: Range of connected sensor.	
Parameter: String SA_Unit	SA_Unit
Direction: Up	
Description: Unit of parameter SA_Offset and SA_Range, e.g. μm or mm.	

14.2.4.2 Get_ControllerInfo

Retrieve information about the controller part of sensor.

Parameter: String SA_ArticleNumber	SA_ArticleNumber
Direction: Up	
Description: Article number of the controller part of sensor.	
Parameter: String SA_ControllerName	SA_ControllerName
Direction: Up	
Description: Name of the controller part of sensor.	
Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the controller part of sensor.	
Parameter: String SA_Option	SA_Option
Direction: Up	
Valid values:	
Numeric value	
Description: Option of the controller part of sensor.	

Parameter: String SA_Softwareversion SA_Softwareversion
Direction: Up
Description: Software version of firmware in the controller part of sensor.

14.2.4.3 Read_AllBlocks

Tell MEDAQLib to read all sensor information.

14.3 Commands for DT6200

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [TCP/IP](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_Status](#), [Get_RawDataRanges](#) and [Get_ChannelInfos](#) after [OpenSensor](#). Otherwise, you have to call it manually to allow MEDAQLib to calculate datarate and to assign values.

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor, either from -2147483648 ([INT_MIN](#)) to 2147483647 ([INT_MAX](#)) or from 0 to 4294967295 ([UINT_MAX](#)) or from -3.402823466e+38 (-[FLT_MAX](#)) to 3.402823466e+38 ([FLT_MAX](#)).
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command [Get_RawDataRanges](#) and [Get_ChannelInfos](#)).

The values of selected channels are filled in the arrays one after another. Each array always starts with first selected channel.

14.3.1 User Level

14.3.1.1 Logout (LGO)

Change user level to user at web interface.

14.3.1.2 Login (LGI)

Change user level to setup at web interface.

Parameter: String SP_Password SP_Password
Direction: Down
Description: Valid password to login.

14.3.1.3 Set_Password (PWD)

Change the password for login.

Parameter: String SP_OldPassword SP_OldPassword
Direction: Down
Description: Old password.

Parameter: String SP_NewPassword SP_NewPassword
Direction: Down
Description: New password.

14.3.2 Measurement

14.3.2.1 Set_SampleTime (STI)

Set the sample time for data acquisition.

Parameter: double SP_SampleTime SP_SampleTime
Direction: Down
Valid values:
Minimum: 0.0
Maximum: 999999999.9999988
Unit: μs
Description: Desired sample time

Parameter: double SA_SampleTime SA_SampleTime
Direction: Up
Valid values:
Minimum: 0.0
Maximum: 1000000000.0
Unit: μs
Description: Real sample time at controller

14.3.2.2 Get_SampleTime (STI?)

Get the current sample time.

Parameter: double SA_SampleTime SA_SampleTime
Direction: Up
Valid values:
Minimum: 0.0
Maximum: 1000000000.0
Unit: μs
Description: Real sample time at controller

14.3.2.3 Set_Trigger (TRG)

Activate/disable the trigger at controller.

Parameter: int SP_TrMode SP_TrMode
Direction: Down
Valid values:
 0= Off
 1= Rising edge
 2= High level
 3= Gate at rising edge
Description: Trigger active/disabled.

14.3.2.4 Get_Trigger (TRG?)

Retrieve the trigger mode at controller.

Parameter: int SA_TrMode SA_TrMode
Direction: Up
Valid values:
 0= Off
 1= Rising edge
 2= High level
 3= Gate at rising edge
Description: Trigger active/disabled.

14.3.2.5 Get_Measure (GMD)

Retrieve one value from controller, even if trigger is active. To get the values from MEDAQLib, use the functions [Poll](#) or [TransferData](#).

14.3.2.6 Set_AvrType (AVT)

Set the averaging type at controller.

Parameter: int SP_AvrType SP_AvrType
Direction: Down
Valid values:
 0= off
 1= Moving average
 2= Mean (arithmetic)
 3= Median
 4= Dynamic noise rejection
Description: Averaging type.

14.3.2.7 Get_AvrType (AVT?)

Retrieve the averaging type at controller.

Parameter: int SA_AvrType SA_AvrType
Direction: Up
Valid values:
 0= off
 1= Moving average
 2= Mean (arithmetic)
 3= Median
 4= Dynamic noise rejection
Description: Averaging type at controller.

14.3.2.8 Set_AvrNbr (AVN)

Set the averaging number at controller.

Parameter: int SP_AvrNbr SP_AvrNbr
Direction: Down
Valid values:
Minimum: 2
Maximum: 8
Description: Averaging number.

14.3.2.9 Get_AvrNbr (AVN?)

Retrieve the averaging number at controller.

Parameter: int SA_AvrNbr SA_AvrNbr
Direction: Up
Valid values:
Minimum: 2
Maximum: 8
Description: Averaging number at controller.

14.3.2.10 Set_AnalogLowPass (ALP)

Set the analog lowpass filter at controller.

Parameter: int SP_LowPass SP_LowPass
Direction: Down
Valid values:
 0= off
 1= on
Description: Analog lowpass filter.

14.3.2.11 Get_AnalogLowPass (ALP?)

Retrieve the analog lowpass filter from controller.

Parameter: int SA_LowPass SA_LowPass
Direction: Up
Valid values:
 0= off
 1= on
Description: Analog lowpass filter.

14.3.3 Data output

14.3.3.1 ChannelStatus (CHS)

Retrieve the available channels at controller.

Parameter: int SA_ChExist1..4 SA_ChExist1..4
Direction: Up
Valid values:
 0= Channel not available
 1= Measured channel
 2= Mathematic channel
Description: Channel 1 to 4 is available at controller.

14.3.3.2 Set_Range (MRA)

Write the measurement range for a channel to controller.

Parameter: int SP_Chан SP_Chан
Direction: Down
Valid values:
Minimum: 1
Maximum: 4
Description: Sensor channel to set it's range.

Parameter: double SP_Range SP_Range
Direction: Down
Valid values:
Minimum: -999999999.9999988
Maximum: 999999999.9999988
Unit: μm
Description: Range of sensor channel.

14.3.3.3 Get_RawDataRange (MDF)

Read the raw data range for a channel from controller. The range is used by MEDAQLib for scaling raw data.

Parameter: int SP_Chан SP_Chан
Direction: Down
Valid values:
Minimum: 1
Maximum: 4
Description: Channel to read the raw data range for.

14.3. Commands for DT6200

Parameter: double SA_RawRangeMin	SA_RawRangeMin
Direction: Up	
Valid values:	
Minimum: -2147483648.0 (INT_MIN)	
Maximum: 2147483647.0 (INT_MAX)	
Description: Minimum raw data range of channel.	
Parameter: double SA_RawRangeMax	SA_RawRangeMax
Direction: Up	
Valid values:	
Minimum: -2147483648.0 (INT_MIN)	
Maximum: 2147483647.0 (INT_MAX)	
Description: Maximum raw data range of channel.	

14.3.3.4 Get_RawDataRanges

Calls the sensor command Get_RawDataRange for any requested channel.

Parameter: int SP_Complete	SP_Complete
Direction: Down	
Valid values:	
0 = FALSE	
1 = TRUE	
Description: Specifies if any possible channel should be requested or only known channels (from former call to ChannelStatus or Get_Status).	
Parameter: double SA_RawRangeMin1..4	SA_RawRangeMin1..4
Direction: Up	
Valid values:	
Minimum: -2147483648.0 (INT_MIN)	
Maximum: 2147483647.0 (INT_MAX)	
Description: Minimum raw data range of channel 1 to 4.	
Parameter: double SA_RawRangeMax1..4	SA_RawRangeMax1..4
Direction: Up	
Valid values:	
Minimum: -2147483648.0 (INT_MIN)	
Maximum: 2147483647.0 (INT_MAX)	
Description: Maximum raw data range of channel 1 to 4.	

14.3.4 Math

14.3.4.1 Set_MathFunction (SMF)

Set mathematic function at controller.

The result of the math function is transmitted like normal sensor data at a specific channel.

Parameter: int SP_Chан	SP_Chан
Direction: Down	
Valid values:	
Minimum: 1	
Maximum: 4	
Description: Channels which returns the result.	

14.3. Commands for DT6200

Parameter: double SP_Offset SP_Offset

Direction: Down

Valid values:

Minimum: -16777215 (-0xffffffff)

Maximum: 16777215 (0xffffffff)

Description: Offset to be added to result.

Parameter: double SP_FactorCh1..4 SP_FactorCh1..4

Direction: Down

Valid values:

Minimum: -9.9

Maximum: 9.9

Description: Multiplication factor for channel 1 to 4.

14.3.4.2 Get_MathFunction (GMF)

Get mathematic function from controller.

Parameter: int SP_Chан SP_Chан

Direction: Down

Valid values:

Minimum: 1

Maximum: 4

Description: Channels for which the function should be read.

Parameter: double SA_Offset SA_Offset

Direction: Up

Valid values:

Minimum: -16777215 (-0xffffffff)

Maximum: 16777215 (0xffffffff)

Description: Offset to be added to result.

Parameter: double SA_FactorCh1..4 SA_FactorCh1..4

Direction: Up

Valid values:

Minimum: -9.9

Maximum: 9.9

Description: Multiplication factor for channel 1 to 4.

14.3.4.3 Clr_MathFunction (CMF)

Clears mathematic function at controller.

Parameter: int SP_Chан SP_Chан

Direction: Down

Valid values:

Minimum: 1

Maximum: 4

Description: Channels for which the function should be cleared.

14.3.5 Interfaces

14.3.5.1 Set_DataPort (SDP)

Set the TCP/IP data port at controller.

Parameter: int SP_DataPort SP_DataPort
Direction: Down
Valid values:
Minimum: 1024
Maximum: 65535
Description: TCP/IP data port at controller.

14.3.5.2 Get_DataPort (GDP)

Retrieve the TCP/IP data port from controller.

Parameter: int SA_DataPort SA_DataPort
Direction: Up
Valid values:
Minimum: 1024
Maximum: 65535
Description: TCP/IP data port at controller.

14.3.5.3 Set_IPConfiguration (IPS)

Set the IP configuration at controller.

Parameter: int SP_DHCPEnabled SP_DHCPEnabled
Direction: Down
Valid values:
 0 = FALSE
 1 = TRUE
Description: Specify if controller should use a static IP address or ask for IP at DHCP server (dynamic IP address).

Parameter: String SP_Address SP_Address
Direction: Down
Valid values:
 Valid IP address in form of xxx.xxx.xxx.xxx
Description: IP address of the controller. This parameter is only evaluated on static IP assignment.

Parameter: String SP_SubnetMask SP_SubnetMask
Direction: Down
Valid values:
 Valid network mask (e.g. 255.255.255.0 for a Class C network)
Description: Network mask of the controller. This parameter is only evaluated on static IP assignment.

Parameter: String SP_Gateway SP_Gateway

Direction: Down

Valid values:

Valid IP address of default gateway in form of xxx.xxx.xxx.xxx

Description: The default gateway must be specified if the controller should communicate with peers in foreign subnets. This parameter is only evaluated on static IP assignment.

14.3.5.4 Get_IPConfiguration (IPS?)

Get the IP configuration at controller.

Parameter: int SA_DHCPEnabled SA_DHCPEnabled

Direction: Up

Valid values:

0= FALSE

1= TRUE

Description: Get settings if controller should use a static IP address ask for IP at DHCP server (dynamic IP address).

Parameter: String SA_Address SA_Address

Direction: Up

Valid values:

Valid IP address in form of xxx.xxx.xxx.xxx

Description: IP address of the controller. If DHCP is enabled it returns the currently assigned IP address.

Parameter: String SA_SubnetMask SA_SubnetMask

Direction: Up

Valid values:

Valid network mask (e.g. 255.255.255.0 for a Class C network)

Description: Network mask of the controller. If DHCP is enabled it returns the currently assigned network mask.

Parameter: String SA_Gateway SA_Gateway

Direction: Up

Valid values:

Valid IP address of default gateway in form of xxx.xxx.xxx.xxx

Description: Address of the default gateway. If DHCP is enabled it returns the currently assigned default gateway.

14.3.5.5 Set_EthernetMode (IFC)

Switches ethernet mode between Ethernet and Ethercat. The controller must be rebooted to apply this setting.

Parameter: int SP_EthernetMode SP_EthernetMode

Direction: Down

Valid values:

0= Ethernet

1= Ethercat

Description: Ethernet mode.

14.3.5.6 Get_EthernetMode (IFC?)

Get ethernet mode of controller.

Parameter: int SA_EthernetMode

SA_EthernetMode

Direction: Up

Valid values:

0= Ethernet

1= Ethercat

Description: Ethernet mode.

14.3.5.7 Set_AppLanguage (LNG)

Set language of web interface at controller.

Parameter: int SP_ApplicationLanguage

SP_ApplicationLanguage

Direction: Down

Valid values:

0= System

1= English

2= German

Description: Language of web interface.

14.3.5.8 Get_AppLanguage (LNG?)

Get language of web interface from controller.

Parameter: int SA_ApplicationLanguage

SA_ApplicationLanguage

Direction: Up

Valid values:

0= System

1= English

2= German

Description: Language of web interface.

14.3.6 Linearization

14.3.6.1 Set_LinMode (LIN)

Set the linearisation mode for a channel at controller.

Parameter: int SP_Chан

SP_Chан

Direction: Down

Valid values:

Minimum: 1

Maximum: 4

Description: Channels to be linearized.

14.3. Commands for DT6200

Parameter: int SP_LinMode SP_LinMode

Direction: Down

Valid values:

- 0= off
- 1= offset correction
- 2= 2 point linearization
- 3= 3 point linearization
- 4= 5 point linearization
- 5= 10 point linearization

Description: Linearisation mode.

14.3.6.2 Get_LinMode (LIN?)

Retrieve the linearisation mode for all channels at controller.

Parameter: int SA_LinMode1..4 SA_LinMode1..4

Direction: Up

Valid values:

- 0= off
- 1= offset correction
- 2= 2 point linearization
- 3= 3 point linearization
- 4= 5 point linearization
- 5= 10 point linearization

Description: Linearisation mode for channel 1 to 4.

14.3.6.3 Set_LinPoint (SLP)

Set a linearisation point for a channel at controller.

Parameter: int SP_Chан SP_Chан

Direction: Down

Valid values:

- Minimum:** 1
Maximum: 4

Description: Channel to be set for.

Parameter: int SP_LinPos SP_LinPos

Direction: Down

Valid values:

- 0= at 0%
- 1= at 10%
- 2= at 20%
- 3= at 30%
- 4= at 40%
- 5= at 50%
- 6= at 60%
- 7= at 70%
- 8= at 80%
- 9= at 90%
- 10= at 100%

Description: Linearisation position to be set for.

14.3.6.4 Set_LinPointOffline (SLP)

Set a linearisation point for a channel at controller (specified at command).

Parameter: int SP_Chан SP_Chан

Direction: Down

Valid values:

Minimum: 1

Maximum: 4

Description: Channel to be set for.

Parameter: int SP_LinPos SP_LinPos

Direction: Down

Valid values:

0= at 0%

1= at 10%

2= at 20%

3= at 30%

4= at 40%

5= at 50%

6= at 60%

7= at 70%

8= at 80%

9= at 90%

10= at 100%

Description: Linearisation position to be set for.

Parameter: double SP_LinPoint SP_LinPoint

Direction: Down

Valid values:

Minimum: -16777215 (-0xffffffff)

Maximum: 16777215 (0xffffffff)

Description: Linearisation point.

14.3.6.5 Get_LinPoint (GLP)

Get a linearisation point for a channel at controller.

Parameter: int SP_Chан SP_Chан

Direction: Down

Valid values:

Minimum: 1

Maximum: 4

Description: Channel to be get for.

Parameter: int SP_LinPos SP_LinPos

Direction: Down

Valid values:

- 0= at 0%
- 1= at 10%
- 2= at 20%
- 3= at 30%
- 4= at 40%
- 5= at 50%
- 6= at 60%
- 7= at 70%
- 8= at 80%
- 9= at 90%
- 10= at 100%

Description: Linearisation position to be get for.

Parameter: double SA_LinPoint SA_LinPoint

Direction: Up

Valid values:

- Minimum:** -16777215 (-0xfffffff)
- Maximum:** 16777215 (0xfffffff)

Description: Linearisation point.

14.3.7 Information

14.3.7.1 Get_Status (STS)

Retrieve detailed information about the controller.

Parameter: double SA_SampleTime SA_SampleTime

Direction: Up

Valid values:

- Minimum:** 0.0
- Maximum:** 1000000000.0

Unit: μs

Description: Real sample time at controller

Parameter: int SA_AvrType SA_AvrType

Direction: Up

Valid values:

- 0= off
- 1= Moving average
- 2= Mean (arithmetic)
- 3= Median
- 4= Dynamic noise rejection

Description: Averaging type at controller.

Parameter: int SA_AvrNbr SA_AvrNbr

Direction: Up

Valid values:

- Minimum:** 2
- Maximum:** 8

Description: Averaging number at controller.

14.3. Commands for DT6200

Parameter: int SA_ChExist1..4 SA_ChExist1..4

Direction: Up

Valid values:

- 0= Channel not available
- 1= Measured channel
- 2= Mathematic channel

Description: Channel 1 to 4 is available at controller.

Parameter: int SA_TrMode SA_TrMode

Direction: Up

Valid values:

- 0= Off
- 1= Rising edge
- 2= High level
- 3= Gate at rising edge

Description: Trigger active/disabled.

14.3.7.2 Get_Version (VER)

Retrieve the sensor software version.

Parameter: String SA_Version SA_Version

Direction: Up

Description: Software version of the controller.

14.3.7.3 Get_ChannelInfo (CHI)

Retrieve information about a sensor channel.

Parameter: int SP_Chан SP_Chан

Direction: Down

Valid values:

- Minimum: 1
- Maximum: 4

Description: Channels to get information for.

Parameter: String SA_ArticleNumber SA_ArticleNumber

Direction: Up

Valid values:

- Numeric value

Description: Article number of the sensor channel.

Parameter: String SA_SensorName SA_SensorName

Direction: Up

Description: Name of the sensor channel.

Parameter: String SA_SerialNumber SA_SerialNumber

Direction: Up

Valid values:

- Numeric value

Description: Serial number of the sensor channel.

Parameter: double SA_Offset	SA_Offset
Direction: Up	
Valid values:	
Minimum: -1000000000.0	
Maximum: 1000000000.0	
Unit: See parameter SA_Unit	
Description: Offset of the sensor channel.	
Parameter: double SA_Range	SA_Range
Direction: Up	
Valid values:	
Minimum: -1000000000.0	
Maximum: 1000000000.0	
Unit: See parameter SA_Unit	
Description: Range of sensor channel.	
Parameter: String SA_Unit	SA_Unit
Direction: Up	
Description: Unit of parameter SA_Offset and SA_Range, e.g. μm or mm.	

14.3.7.4 Get_ChannelInfos

Calls the sensor command Get_ChannelInfo for any requested channel.

Parameter: int SP_Complete	SP_Complete
Direction: Down	
Valid values:	
0= FALSE	
1= TRUE	
Description: Specifies if any possible channel should be requested or only known channels (from former call to ChannelStatus or Get_Status).	
Parameter: String SA_ArticleNumber1..4	SA_ArticleNumber1..4
Direction: Up	
Valid values:	
Numeric value	
Description: Article number of the sensor channel 1 to 4.	
Parameter: String SA_SensorName1..4	SA_SensorName1..4
Direction: Up	
Description: Name of the sensor channel 1 to 4.	
Parameter: String SA_SerialNumber1..4	SA_SerialNumber1..4
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the sensor channel 1 to 4.	
Parameter: double SA_Offset1..4	SA_Offset1..4
Direction: Up	
Valid values:	
Minimum: -1000000000.0	
Maximum: 1000000000.0	
Unit: See parameter SA_Unit	
Description: Offset of the sensor channel 1 to 4.	

14.3. Commands for DT6200

Parameter: double SA_Range1..4	SA_Range1..4
Direction: Up	
Valid values:	
Minimum: -1000000000.0	
Maximum: 1000000000.0	
Unit: See parameter SA_Unit	
Description: Range of sensor channel 1 to 4.	
Parameter: String SA_Unit1..4	SA_Unit1..4
Direction: Up	
Description: Unit of parameter SA_Offset and SA_Range, e.g. μm or mm.	

14.3.7.5 Get_ControllerInfo (COI)

Retrieve information about the controller.

Parameter: String SA_ArticleNumber	SA_ArticleNumber
Direction: Up	
Description: Article number of the controller.	
Parameter: String SA_ControllerName	SA_ControllerName
Direction: Up	
Description: Name of the controller.	
Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the controller.	
Parameter: String SA_Option	SA_Option
Direction: Up	
Valid values:	
Numeric value	
Description: Option of the controller.	
Parameter: String SA_Softwareversion	SA_Softwareversion
Direction: Up	
Description: Software version of firmware in the controller.	

14.3.8 Internal commands

14.3.8.1 Update_Firmware

Update firmware version at controller.

Attention! This function can take approx. 1 minute. This is an internal command. It should not be used by the customer.

Parameter: Binary data SP_FirmwareFile	SP_FirmwareFile
Direction: Down	
Valid values:	
Firmware file as binary data.	
Description: Firmware version, read from file.	

Parameter: int SA_Result SA_Result
Direction: Up
Valid values:
 0= Failed
 1= Success
Description: Result of firmware update.

14.4 Commands for KSS6380

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [TCP/IP](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_Status](#) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib to calculate datarate and to assign values.

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor, from 0 to 16777215.
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command [Use_Defaults](#) parameter [IP_Range1..4](#)).

The values of selected channels are filled in the arrays one after another. Each array always starts with first selected channel.

14.4.1 Measurement

14.4.1.1 Set_SRIndex (SRA)

Set the samplerate for data acquisition.

Parameter: int SP_SRIndex SP_SRIndex
Direction: Down
Valid values:
 0= 26 Hz
 1= 104 Hz
 2= 520 Hz
 3= 1040 Hz
Description: Samplerate index

14.4.1.2 Get_SRIndex (SRA?)

Get the current samplerate.

Parameter: int SA_SRIndex SA_SRIndex
Direction: Up
Valid values:
 0= 26 Hz
 1= 104 Hz
 2= 520 Hz
 3= 1040 Hz
Description: Samplerate index.

14.4.1.3 Set_Trigger (TRG)

Activate/disable the trigger at controller.

Parameter: int SP_TrgMode SP_TrgMode
Direction: Down
Valid values:
 0= disabled
 1= active
Description: Trigger active/disabled.

14.4.1.4 Get_Trigger (TRG?)

Retrieve the trigger mode at controller.

Parameter: int SA_TrgMode SA_TrgMode
Direction: Up
Valid values:
 0= disabled
 1= active
Description: Trigger active/disabled.

14.4.1.5 Get_Measure (GMD)

Retrieve one value from controller, even if trigger is active.

14.4.1.6 Set_AvrType (AVT)

Set the averaging type at controller.

Parameter: int SP_AvrType SP_AvrType
Direction: Down
Valid values:
 0= off
 1= Moving average
Description: Averaging type.

14.4.1.7 Get_AvrType (AVT?)

Retrieve the averaging type at controller.

Parameter: int SA_AvrType SA_AvrType

Direction: Up

Valid values:

0= off

1= Moving average

Description: Averaging type at controller.

14.4.1.8 Set_AvrNbr (AVN)

Set the averaging number at controller.

Parameter: int SP_AvrNbr SP_AvrNbr

Direction: Down

Valid values:

Minimum: 2

Maximum: 4

Description: Averaging number.

14.4.1.9 Get_AvrNbr (AVN?)

Retrieve the averaging number at controller.

Parameter: int SA_AvrNbr SA_AvrNbr

Direction: Up

Valid values:

Minimum: 2

Maximum: 4

Description: Averaging number at controller.

14.4.2 Data output

14.4.2.1 ChannelStatus (CHS)

Retrieve the available channels at controller.

Parameter: int SA_ChExist1..4 SA_ChExist1..4

Direction: Up

Valid values:

0= Channel not available

1= Measured channel

2= Mathematic channel (only for channel 1 to 3)

Description: Channel 1 to 4 is available at controller.

14.4.2.2 Set_Channel (CHT)

Set the channels to transmit from controller.

Parameter: int SP_ChTransmit1..4 SP_ChTransmit1..4
Direction: Down
Valid values:
 0= no
 1= yes
Description: Channel 1 to 4 is transmitted.

14.4.2.3 Get_Channel (CHT?)

Get the channels transmitted from controller.

Parameter: int SA_ChTransmit1..4 SA_ChTransmit1..4
Direction: Up
Valid values:
 0= no
 1= yes
Description: Channel 1 to 4 is transmitted.

14.4.3 Math

14.4.3.1 Set_MathFunction (SMF)

Set mathematic function at controller.

The result of the math function is transmitted like normal sensor data at a specific channel.

Because of the mathematic operations (multiplication, addition) the raw values can exceed the range of 24 bit. In this case, it is set to maximum value.

Parameter: int SP_Chан SP_Chан
Direction: Down
Valid values:
Minimum: 1
Maximum: 3
Description: Channels which returns the result. Only channel 1 to 3 supports math function.

Parameter: double SP_Offset SP_Offset
Direction: Down
Valid values:
Minimum: -16777215 (-0xffffffff)
Maximum: 16777215 (0xffffffff)
Description: Offset to be added to result.

Parameter: double SP_FactorCapa SP_FactorCapa
Direction: Down
Valid values:
Minimum: -9.9
Maximum: 9.9
Description: Multiplication factor for capa.

Parameter: double SP_FactorEddy SP_FactorEddy
Direction: Down
Valid values:
Minimum: -9.9
Maximum: 9.9
Description: Multiplication factor for eddy.

14.4.3.2 Get_MathFunction (GMF)

Get mathematic function from controller.

Parameter: int SP_Chан SP_Chан
Direction: Down
Valid values:
Minimum: 1
Maximum: 3
Description: Channels for which the function should be read. Only channel 1 to 3 supports math function.

Parameter: double SA_Offset SA_Offset
Direction: Up
Valid values:
Minimum: -16777215 (-0xffffffff)
Maximum: 16777215 (0xffffffff)
Description: Offset to be added to result.

Parameter: double SA_FactorCapa SA_FactorCapa
Direction: Up
Valid values:
Minimum: -9.9
Maximum: 9.9
Description: Multiplication factor for capa.

Parameter: double SA_FactorEddy SA_FactorEddy
Direction: Up
Valid values:
Minimum: -9.9
Maximum: 9.9
Description: Multiplication factor for eddy.

14.4.3.3 Clr_MathFunction (CMF)

Clears mathematic function at controller.

Parameter: int SP_Chан SP_Chан
Direction: Down
Valid values:
Minimum: 1
Maximum: 3
Description: Channels for which the function should be cleared. Only channel 1 to 3 supports math function.

14.4.4 Information

14.4.4.1 Get_Status (STS)

Retrieve detailed information about the controller.

Parameter: int SA_SRIndex

SA_SRIndex

Direction: Up

Valid values:

- 0= 26 Hz
- 1= 104 Hz
- 2= 520 Hz
- 3= 1040 Hz

Description: Samplerate index.

Parameter: int SA_AvrType

SA_AvrType

Direction: Up

Valid values:

- 0= off
- 1= Moving average

Description: Averaging type at controller.

Parameter: int SA_AvrNbr

SA_AvrNbr

Direction: Up

Valid values:

- Minimum: 2
- Maximum: 4

Description: Averaging number at controller.

Parameter: int SA_ChExist1..4

SA_ChExist1..4

Direction: Up

Valid values:

- 0= Channel not available
- 1= Measured channel
- 2= Mathematic channel (only for channel 1 to 3)

Description: Channel 1 to 4 is available at controller.

Parameter: int SA_ChTransmit1..4

SA_ChTransmit1..4

Direction: Up

Valid values:

- 0= no
- 1= yes

Description: Channel 1 to 4 is transmitted.

Parameter: int SA_TrgMode

SA_TrgMode

Direction: Up

Valid values:

- 0= disabled
- 1= active

Description: Trigger active/disabled.

14.4.4.2 Get_Version (VER)

Retrieve the sensor software version.

Parameter: String SA_Version SA_Version
Direction: Up
Description: Software version of the controller.

14.4.5 Parameter management

14.4.5.1 Save_Setup (SSU)

Save the current setup of controller to flash.

14.4.5.2 Read_Setup (RSU)

Read the setup from controller flash.

14.4.5.3 Factory_Defaults (FDE)

Restore the controller to factory defaults.

To save the default values call Save_Setup too. The new parameters are returned.

Parameter: int SA_SRIndex SA_SRIndex
Direction: Up
Valid values:
 0= 26 Hz
 1= 104 Hz
 2= 520 Hz
 3= 1040 Hz
Description: Samplerate index.

Parameter: int SA_AvrType SA_AvrType
Direction: Up
Valid values:
 0= off
 1= Moving average
Description: Averaging type at controller.

Parameter: int SA_AvrNbr SA_AvrNbr
Direction: Up
Valid values:
Minimum: 2
Maximum: 4
Description: Averaging number at controller.

Parameter: int SA_ChExist1..4 SA_ChExist1..4
Direction: Up
Valid values:
 0= Channel not available
 1= Measured channel
 2= Mathematic channel (only for channel 1 to 3)
Description: Channel 1 to 4 is available at controller.

Parameter: int SA_ChTransmit1..4 SA_ChTransmit1..4

Direction: Up

Valid values:

0= no

1= yes

Description: Channel 1 to 4 is transmitted.

Parameter: int SA_TrgMode SA_TrgMode

Direction: Up

Valid values:

0= disabled

1= active

Description: Trigger active/disabled.

14.4.6 Internal commands

14.4.6.1 Set_TempCoeff (STC)

Set temperature coefficient for a channel at controller. This is an internal command.
It should not be used by the customer.

Parameter: int SP_Chан SP_Chан

Direction: Down

Valid values:

Minimum: 1

Maximum: 2

Description: Channel to be set for.

Parameter: int SP_TempCoeffParam SP_TempCoeffParam

Direction: Down

Valid values:

Minimum: 0

Maximum: 16

Description: Temperature coefficient parameter.

Parameter: double SP_TemperatureCoeffizient SP_TemperatureCoeffizient

Direction: Down

Valid values:

Minimum: 0

Maximum: 16777215 (0xffffffff)

Description: Temperature coefficient.

14.4.6.2 Get_TempCoeff (GTC)

Get temperature coefficient for a channel at controller. This is an internal command.
It should not be used by the customer.

Parameter: int SP_Chан SP_Chан

Direction: Down

Valid values:

Minimum: 1

Maximum: 2

Description: Channel to be set for.

Parameter: int SP_TempCoeffParam	SP_TempCoeffParam
Direction: Down	
Valid values:	
Minimum: 0	
Maximum: 16	
Description: Temperature coefficient parameter.	
Parameter: double SA_TemperatureCoeffizient	SA_TemperatureCoeffizient
Direction: Up	
Valid values:	
Minimum: -16777215 (-0xffffffff)	
Maximum: 16777215 (0xffffffff)	
Description: Temperature coefficient.	

14.4.6.3 Get_UnlinEddyVal (GUE)

Get unlinearized eddy value from controller. This is an internal command. It should not be used by the customer.

Parameter: double SA_UnlinearizedEddyValue	SA_UnlinearizedEddyValue
Direction: Up	
Valid values:	
Minimum: -16777215 (-0xffffffff)	
Maximum: 16777215 (0xffffffff)	
Description: Unlinearized eddy value .	

14.4.6.4 Set_LinPoint (SLP)

Set a linearisation point for a channel at controller. This is an internal command. It should not be used by the customer.

Parameter: int SP_Chан	SP_Chан
Direction: Down	
Valid values:	
Minimum: 1	
Maximum: 2	
Description: Channel to be set for.	

Parameter: int SP_LinPos	SP_LinPos
Direction: Down	
Valid values:	
0= at 0%	
1= at 5%	
2= at 10%	
3= at 15%	
4= at 20%	
5= at 25%	
6= at 30%	
7= at 35%	
8= at 40%	
9= at 45%	
10= at 50%	

14.4. Commands for KSS6380

11= at 55%
 12= at 60%
 13= at 65%
 14= at 70%
 15= at 75%
 16= at 80%
 17= at 85%
 18= at 90%
 19= at 95%
 20= at 100%

Description: Linearisation position to be set for.

14.4.6.5 Get_LinPoint (GLP)

Get a linearisation point for a channel at controller. This is an internal command. It should not be used by the customer.

Parameter: int SP_Chан

SP_Chан

Direction: Down

Valid values:

Minimum: 1
 Maximum: 2

Description: Channel to be get for.

Parameter: int SP_LinPos

SP_LinPos

Direction: Down

Valid values:

0= at 0%
 1= at 5%
 2= at 10%
 3= at 15%
 4= at 20%
 5= at 25%
 6= at 30%
 7= at 35%
 8= at 40%
 9= at 45%
 10= at 50%
 11= at 55%
 12= at 60%
 13= at 65%
 14= at 70%
 15= at 75%
 16= at 80%
 17= at 85%
 18= at 90%
 19= at 95%
 20= at 100%

Description: Linearisation position to be get for.

Parameter: double SA_LinPoint SA_LinPoint
Direction: Up
Valid values:
Minimum: 0
Maximum: 16777215 (0xffffffff)
Description: Linearisation point.

14.4.7 Special commands

14.4.7.1 Use_Defaults

This command tells the driver to use default values to operate with sensor data. If some parameters are not specified they are not changed. The sensor is not affected by this command.

Parameter: double IP_Range1..4 IP_Range1..4
Direction: Down
Valid values:
Minimum: -1.79769e+308 (-DBL_MAX)
Maximum: 1.79769e+308 (DBL_MAX)
Unit: μm or mm
Description: Tells the driver the range of sensor for channel 1 to 4. It is used to scale the raw sensor values into μm or mm. If it is zero, no scaling is done.

14.4.7.2 Get_DrvSetting

Returns the current settings of the driver used for operating with sensor data. It is the opposite of Use_Defaults. The sensor is not affected by this command.

Parameter: double IA_Range1..4 IA_Range1..4
Direction: Up
Valid values:
Minimum: -1.79769e+308 (-DBL_MAX)
Maximum: 1.79769e+308 (DBL_MAX)
Unit: μm or mm
Description: The range of sensor for channel 1 to 4, used by driver to scale values into μm or mm.

14.5 Commands for DT6500

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [TCP/IP](#) (native).

14.5. Commands for DT6500

If first bit of **IP_AutomaticMode** is set (1), MEDAQLib calls automatically sensor command [Get_Status](#), [Get_SampleTime](#), [Get_RawDataRanges](#) and [Get_ChannelInfos](#) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib to calculate datarate and to assign values.

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor, from 0 to 16777215.
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command [Get_RawDataRanges](#) and [Get_ChannelInfos](#) for DT6530 or [Use_Defaults](#) parameter **IP_Range1..8** for DT6500).

The values of selected channels are filled in the arrays one after another. Each array always starts with first selected channel.

14.5.1 User Level

14.5.1.1 Logout (LGO)

Change user level to user at web interface. This command is only available for new controller DT6530.

14.5.1.2 Login (LGI)

Change user level to setup at web interface. This command is only available for new controller DT6530.

Parameter: String SP_Password SP_Password
Direction: Down
Description: Valid password to login.

14.5.1.3 Set_Password (PWD)

Change the password for login. This command is only available for new controller DT6530.

Parameter: String SP_OldPassword SP_OldPassword
Direction: Down
Description: Old password.

Parameter: String SP_NewPassword SP_NewPassword
Direction: Down
Description: New password.

14.5.2 Measurement

14.5.2.1 Set_SRIndex (SRA)

Set the samplerate for data acquisition.

Parameter: int SP_SRIndex

SP_SRIndex

Direction: Down

Valid values:

0= 2.60 Hz
 1= 5.21 Hz
 2= 10.42 Hz
 3= 15.63 Hz
 4= 26.04 Hz
 5= 31.25 Hz
 6= 52.08 Hz
 7= 62.50 Hz
 8= 104.17 Hz
 9= 520.83 Hz
 10= 1041.67 Hz
 11= 2083.33 Hz
 12= 3906.25 Hz
 13= 7812.50 Hz

Description: Samplerate index

14.5.2.2 Get_SRIndex (SRA?)

Get the current samplerate.

Parameter: int SA_SRIndex

SA_SRIndex

Direction: Up

Valid values:

0= 2.60 Hz
 1= 5.21 Hz
 2= 10.42 Hz
 3= 15.63 Hz
 4= 26.04 Hz
 5= 31.25 Hz
 6= 52.08 Hz
 7= 62.50 Hz
 8= 104.17 Hz
 9= 520.83 Hz
 10= 1041.67 Hz
 11= 2083.33 Hz
 12= 3906.25 Hz
 13= 7812.50 Hz

Description: Samplerate index.

14.5.2.3 Set_SampleTime (STI)

Set the sample time for data acquisition. This command is only available for new controller DT6530. It replaces the obsolete sensor command Set_SRIndex.

Parameter: double SP_SampleTime SP_SampleTime

Direction: Down

Valid values:

Minimum: 0.0

Maximum: 999999999.9999988

Unit: μ s

Description: Desired sample time

Parameter: double SA_SampleTime SA_SampleTime

Direction: Up

Valid values:

Minimum: 0.0

Maximum: 1000000000.0

Unit: μ s

Description: Real sample time at controller

14.5.2.4 Get_SampleTime (STI?)

Get the current sample time. This command is only available for new controller DT6530 It replaces the obsolete sensor command Get_SRIndex.

Parameter: double SA_SampleTime SA_SampleTime

Direction: Up

Valid values:

Minimum: 0.0

Maximum: 1000000000.0

Unit: μ s

Description: Real sample time at controller

14.5.2.5 Set_Trigger (TRG)

Activate/disable the trigger at controller.

Parameter: int SP_TrMode SP_TrMode

Direction: Down

Valid values:

 0= Off

 1= Rising edge

 2= High level

 3= Gate at rising edge

Description: Trigger active/disabled.

14.5.2.6 Get_Trigger (TRG?)

Retrieve the trigger mode at controller.

Parameter: int SA_TrMode SA_TrMode
Direction: Up
Valid values:
 0= Off
 1= Rising edge
 2= High level
 3= Gate at rising edge
Description: Trigger active/disabled.

14.5.2.7 Get_Measure (GMD)

Retrieve one value from controller, even if trigger is active. To get the values from MEDAQLib, use the functions [Poll](#) or [TransferData](#).

14.5.2.8 Set_AvrType (AVT)

Set the averaging type at controller.

Parameter: int SP_AvrType SP_AvrType
Direction: Down
Valid values:
 0= off
 1= Moving average
 2= Mean (arithmetic)
 3= Median
 4= Dynamic noise rejection
Description: Averaging type.

14.5.2.9 Get_AvrType (AVT?)

Retrieve the averaging type at controller.

Parameter: int SA_AvrType SA_AvrType
Direction: Up
Valid values:
 0= off
 1= Moving average
 2= Mean (arithmetic)
 3= Median
 4= Dynamic noise rejection
Description: Averaging type at controller.

14.5.2.10 Set_AvrNbr (AVN)

Set the averaging number at controller.

Parameter: int SP_AvrNbr SP_AvrNbr
Direction: Down
Valid values:
Minimum: 2
Maximum: 8
Description: Averaging number.

14.5.2.11 Get_AvrNbr (AVN?)

Retrieve the averaging number at controller.

Parameter: int SA_AvrNbr SA_AvrNbr
Direction: Up
Valid values:
Minimum: 2
Maximum: 8
Description: Averaging number at controller.

14.5.3 Data output

14.5.3.1 ChannelStatus (CHS)

Retrieve the available channels at controller.

Parameter: int SA_ChExist1..8 SA_ChExist1..8
Direction: Up
Valid values:
 0= Channel not available
 1= Measured channel
 2= Mathematic channel
Description: Channel 1 to 8 is available at controller.

14.5.3.2 Set_Channel (CHT)

Set the channels to transmit from controller.

Parameter: int SP_ChTransmit1..8 SP_ChTransmit1..8
Direction: Down
Valid values:
 0= no
 1= yes
Description: Channel 1 to 8 is transmitted.

14.5.3.3 Get_Channel (CHT?)

Get the channels transmitted from controller.

Parameter: int SA_ChTransmit1..8 SA_ChTransmit1..8
Direction: Up
Valid values:
 0= no
 1= yes
Description: Channel 1 to 8 is transmitted.

14.5.3.4 Set_Range (MRA)

Write the measurement range for a channel to controller. This command is only available for new controller DT6530.

Parameter: int SP_Chан SP_Chан
Direction: Down
Valid values:
Minimum: 1
Maximum: 8
Description: Sensor channel to set it's range.

Parameter: double SP_Range SP_Range
Direction: Down
Valid values:
Minimum: -999999999.99999988
Maximum: 999999999.99999988
Unit: μm
Description: Range of sensor channel.

14.5.3.5 Get_RawDataRange (MDF)

Read the raw data range for a channel from controller. The range is used by MEDAQLib for scaling raw data. This command is only available for new controller DT6530.

Parameter: int SP_Chан SP_Chан
Direction: Down
Valid values:
Minimum: 1
Maximum: 8
Description: Channel to read the raw data range for.

Parameter: double SA_RawRangeMin SA_RawRangeMin
Direction: Up
Valid values:
Minimum: -2147483648.0 (INT_MIN)
Maximum: 2147483647.0 (INT_MAX)
Description: Minimum raw data range of channel.

Parameter: double SA_RawRangeMax SA_RawRangeMax
Direction: Up
Valid values:
 Minimum: -2147483648.0 (INT_MIN)
 Maximum: 2147483647.0 (INT_MAX)
Description: Maximum raw data range of channel.

14.5.3.6 Get_RawDataRanges

Calls the sensor command Get_RawDataRange for any requested channel.

Parameter: int SP_Complete SP_Complete
Direction: Down
Valid values:
 0= FALSE
 1= TRUE
Description: Specifies if any possible channel should be requested or only known channels (from former call to [ChannelStatus](#) or [Get_Status](#)).

Parameter: double SA_RawRangeMin1..8 SA_RawRangeMin1..8
Direction: Up
Valid values:
 Minimum: -2147483648.0 (INT_MIN)
 Maximum: 2147483647.0 (INT_MAX)
Description: Minimum raw data range of channel 1 to 8.

Parameter: double SA_RawRangeMax1..8 SA_RawRangeMax1..8
Direction: Up
Valid values:
 Minimum: -2147483648.0 (INT_MIN)
 Maximum: 2147483647.0 (INT_MAX)
Description: Maximum raw data range of channel 1 to 8.

14.5.4 Display

14.5.4.1 Set_Display (DIS)

Set the display settings at controller.

Parameter: int SP_ShowChannels SP_ShowChannels
Direction: Down
Valid values:
 0= none
 1= all
 2= selected
Description: Channels to be displayed.

Parameter: int SP_ShowLinearized SP_ShowLinearized
Direction: Down
Valid values:
 0= off
 1= on
Description: Channels should be displayed linearized.

14.5.4.2 Get_Display (DIS?)

Retrieve the display settings at controller.

Parameter: int SA_ShowChannels

SA_ShowChannels

Direction: Up

Valid values:

0= none

1= all

2= selected

Description: Channels to be displayed.

Parameter: int SA_ShowLinearized

SA_ShowLinearized

Direction: Up

Valid values:

0= off

1= on

Description: Channels should be displayed linearized.

14.5.5 Math

14.5.5.1 Set_MathFunction (SMF)

Set mathematic function at controller.

The result of the math function is transmitted like normal sensor data at a specific channel.

Because of the mathematic operations (multiplication, addition) the raw values can exceed the range of 24 bit. To avoid this, the controller does automatically divide the results by eight (Result= Result/8). MEDAQLib does output the raw values if it comes from sensor and multiply the scaled values by eight to compensate the dividing at controller.

Parameter: int SP_Chан

SP_Chан

Direction: Down

Valid values:

Minimum: 1

Maximum: 8

Description: Channels which returns the result.

Parameter: double SP_Offset

SP_Offset

Direction: Down

Valid values:

Minimum: -16777215 (-0xffffffff)

Maximum: 16777215 (0xffffffff)

Description: Offset to be added to result.

Parameter: double SP_FactorCh1..8

SP_FactorCh1..8

Direction: Down

Valid values:

Minimum: -9.9

Maximum: 9.9

Description: Multiplication factor for channel 1 to 8.

14.5.5.2 Get_MathFunction (GMF)

Get mathematic function from controller.

Parameter: int SP_Chан SP_Chан

Direction: Down

Valid values:

Minimum: 1

Maximum: 8

Description: Channels for which the function should be read.

Parameter: double SA_Offset SA_Offset

Direction: Up

Valid values:

Minimum: -16777215 (-0xfffffff)

Maximum: 16777215 (0xfffffff)

Description: Offset to be added to result.

Parameter: double SA_FactorCh1..8 SA_FactorCh1..8

Direction: Up

Valid values:

Minimum: -9.9

Maximum: 9.9

Description: Multiplication factor for channel 1 to 8.

14.5.5.3 Clr_MathFunction (CMF)

Clears mathematic function at controller.

Parameter: int SP_Chан SP_Chан

Direction: Down

Valid values:

Minimum: 1

Maximum: 8

Description: Channels for which the function should be cleared.

14.5.6 Interfaces

14.5.6.1 Set_DataPort (SDP)

Set the TCP/IP data port at controller. This command is only available for new controller DT6530.

Parameter: int SP_DataPort SP_DataPort

Direction: Down

Valid values:

Minimum: 1024

Maximum: 65535

Description: TCP/IP data port at controller.

14.5.6.2 Get_DataPort (GDP)

Retrieve the TCP/IP data port from controller. This command is only available for new controller DT6530.

Parameter: int SA_DataPort SA_DataPort
Direction: Up
Valid values:
 Minimum: 1024
 Maximum: 65535
Description: TCP/IP data port at controller.

14.5.6.3 Set_IPConfiguration (IPS)

Set the IP configuration at controller. This command is only available for new controller DT6530.

Parameter: int SP_DHCPEnabled SP_DHCPEnabled
Direction: Down
Valid values:
 0= FALSE
 1= TRUE
Description: Specify if controller should use a static IP address or ask for IP at DHCP server (dynamic IP address).

Parameter: String SP_Address SP_Address
Direction: Down
Valid values:
 Valid IP address in form of xxx.xxx.xxx.xxx
Description: IP address of the controller. This parameter is only evaluated on static IP assignment.

Parameter: String SP_SubnetMask SP_SubnetMask
Direction: Down
Valid values:
 Valid network mask (e.g. 255.255.255.0 for a Class C network)
Description: Network mask of the controller. This parameter is only evaluated on static IP assignment.

Parameter: String SP_Gateway SP_Gateway
Direction: Down
Valid values:
 Valid IP address of default gateway in form of xxx.xxx.xxx.xxx
Description: The default gateway must be specified if the controller should communicate with peers in foreign subnets. This parameter is only evaluated on static IP assignment.

14.5.6.4 Get_IPConfiguration (IPS?)

Get the IP configuration at controller. This command is only available for new controller DT6530.

Parameter: int SA_DHCPEnabled SA_DHCPEnabled

Direction: Up

Valid values:

0= FALSE

1= TRUE

Description: Get settings if controller should use a static IP address ask for IP at DHCP server (dynamic IP address).

Parameter: String SA_Address SA_Address

Direction: Up

Valid values:

Valid IP address in form of xxx.xxx.xxx.xxx

Description: IP address of the controller. If DHCP is enabled it returns the currently assigned IP address.

Parameter: String SA_SubnetMask SA_SubnetMask

Direction: Up

Valid values:

Valid network mask (e.g. 255.255.255.0 for a Class C network)

Description: Network mask of the controller. If DHCP is enabled it returns the currently assigned network mask.

Parameter: String SA_Gateway SA_Gateway

Direction: Up

Valid values:

Valid IP address of default gateway in form of xxx.xxx.xxx.xxx

Description: Address of the default gateway. If DHCP is enabled it returns the currently assigned default gateway.

14.5.6.5 Set_EthernetMode (IFC)

Switches ethernet mode between Ethernet and Ethercat. This command is only available for new controller DT6530. The controller must be rebooted to apply this setting.

Parameter: int SP_EthernetMode SP_EthernetMode

Direction: Down

Valid values:

0= Ethernet

1= Ethercat

Description: Ethernet mode.

14.5.6.6 Get_EthernetMode (IFC?)

Get ethernet mode of controller. This command is only available for new controller DT6530.

Parameter: int SA_EthernetMode

SA_EthernetMode

Direction: Up

Valid values:

0= Ethernet

1= Ethercat

Description: Ethernet mode.

14.5.6.7 Set_AppLanguage (LNG)

Set language of web interface at controller. This command is only available for new controller DT6530.

Parameter: int SP_ApplicationLanguage

SP_ApplicationLanguage

Direction: Down

Valid values:

0= System

1= English

2= German

Description: Language of web interface.

14.5.6.8 Get_AppLanguage (LNG?)

Get language of web interface from controller. This command is only available for new controller DT6530.

Parameter: int SA_ApplicationLanguage

SA_ApplicationLanguage

Direction: Up

Valid values:

0= System

1= English

2= German

Description: Language of web interface.

14.5.7 Linearization

14.5.7.1 Set_LinMode (LIN)

Set the linearisation mode for a channel at controller.

Parameter: int SP_Chан

SP_Chан

Direction: Down

Valid values:

Minimum: 1

Maximum: 8

Description: Channels to be linearized.

14.5. Commands for DT6500

Parameter: int SP_LinMode SP_LinMode

Direction: Down

Valid values:

- 0= off
- 1= offset correction
- 2= 2 point linearization
- 3= 3 point linearization
- 4= 5 point linearization
- 5= 10 point linearization

Description: Linearisation mode.

14.5.7.2 Get_LinMode (LIN?)

Retrieve the linearisation mode for all channels at controller.

Parameter: int SA_LinMode1..8 SA_LinMode1..8

Direction: Up

Valid values:

- 0= off
- 1= offset correction
- 2= 2 point linearization
- 3= 3 point linearization
- 4= 5 point linearization
- 5= 10 point linearization

Description: Linearisation mode for channel 1 to 8.

14.5.7.3 Set_LinPoint (SLP)

Set a linearisation point for a channel at controller.

Parameter: int SP_Chан SP_Chан

Direction: Down

Valid values:

- Minimum:** 1
Maximum: 8

Description: Channel to be set for.

Parameter: int SP_LinPos SP_LinPos

Direction: Down

Valid values:

- 0= at 0%
- 1= at 10%
- 2= at 20%
- 3= at 30%
- 4= at 40%
- 5= at 50%
- 6= at 60%
- 7= at 70%
- 8= at 80%
- 9= at 90%
- 10= at 100%

Description: Linearisation position to be set for.

14.5.7.4 Set_LinPointOffline (SLP)

Set a linearisation point for a channel at controller (specified at command). This command is only available for new controller DT6530.

Parameter: int SP_Chан SP_Chан

Direction: Down

Valid values:

Minimum: 1

Maximum: 8

Description: Channel to be set for.

Parameter: int SP_LinPos SP_LinPos

Direction: Down

Valid values:

0= at 0%

1= at 10%

2= at 20%

3= at 30%

4= at 40%

5= at 50%

6= at 60%

7= at 70%

8= at 80%

9= at 90%

10= at 100%

Description: Linearisation position to be set for.

Parameter: double SP_LinPoint SP_LinPoint

Direction: Down

Valid values:

Minimum: -16777215 (-0xffffffff)

Maximum: 16777215 (0xffffffff)

Description: Linearisation point.

14.5.7.5 Get_LinPoint (GLP)

Get a linearisation point for a channel at controller.

Parameter: int SP_Chан SP_Chан

Direction: Down

Valid values:

Minimum: 1

Maximum: 8

Description: Channel to be get for.

Parameter: int SP_LinPos SP_LinPos

Direction: Down

Valid values:

- 0= at 0%
- 1= at 10%
- 2= at 20%
- 3= at 30%
- 4= at 40%
- 5= at 50%
- 6= at 60%
- 7= at 70%
- 8= at 80%
- 9= at 90%
- 10= at 100%

Description: Linearisation position to be get for.

Parameter: double SA_LinPoint SA_LinPoint

Direction: Up

Valid values:

- Minimum:** -16777215 (-0xffffffff)
- Maximum:** 16777215 (0xffffffff)

Description: Linearisation point.

14.5.8 Information

14.5.8.1 Get_Status (STS)

Retrieve detailed information about the controller.

Parameter: int SA_SRIndex SA_SRIndex

Direction: Up

Valid values:

- 0= 2.60 Hz
- 1= 5.21 Hz
- 2= 10.42 Hz
- 3= 15.63 Hz
- 4= 26.04 Hz
- 5= 31.25 Hz
- 6= 52.08 Hz
- 7= 62.50 Hz
- 8= 104.17 Hz
- 9= 520.83 Hz
- 10= 1041.67 Hz
- 11= 2083.33 Hz
- 12= 3906.25 Hz
- 13= 7812.50 Hz

Description: Samplerate index.

Parameter: int SA_AvrType SA_AvrType

Direction: Up

Valid values:

- 0= off
- 1= Moving average
- 2= Mean (arithmetic)
- 3= Median
- 4= Dynamic noise rejection

Description: Averaging type at controller.

Parameter: int SA_AvrNbr SA_AvrNbr

Direction: Up

Valid values:

Minimum: 2
Maximum: 8

Description: Averaging number at controller.

Parameter: int SA_ChExist1..8 SA_ChExist1..8

Direction: Up

Valid values:

- 0= Channel not available
- 1= Measured channel
- 2= Mathematic channel

Description: Channel 1 to 8 is available at controller.

Parameter: int SA_ChTransmit1..8 SA_ChTransmit1..8

Direction: Up

Valid values:

- 0= no
- 1= yes

Description: Channel 1 to 8 is transmitted.

Parameter: int SA_TrgMode SA_TrgMode

Direction: Up

Valid values:

- 0= Off
- 1= Rising edge
- 2= High level
- 3= Gate at rising edge

Description: Trigger active/disabled.

Parameter: int SA_LinMode1..8 SA_LinMode1..8

Direction: Up

Valid values:

- 0= off
- 1= offset correction
- 2= 2 point linearization
- 3= 3 point linearization
- 4= 5 point linearization
- 5= 10 point linearization

Description: Linearisation mode for channel 1 to 8.

Parameter: int SA_ShowChannels SA_ShowChannels

Direction: Up

Valid values:

0= none

1= all

2= selected

Description: Channels to be displayed.

Parameter: int SA_ShowLinearized SA_ShowLinearized

Direction: Up

Valid values:

0= off

1= on

Description: Channels should be displayed linearized.

14.5.8.2 Get_Version (VER)

Retrieve the sensor software version.

Parameter: String SA_Version SA_Version

Direction: Up

Description: Software version of the controller.

14.5.8.3 Get_ChannelInfo (CHI)

Retrieve information about a sensor channel. This command is only available for new controller DT6530.

Parameter: int SP_Chан SP_Chан

Direction: Down

Valid values:

Minimum: 1

Maximum: 8

Description: Channels to get information for.

Parameter: String SA_ArticleNumber SA_ArticleNumber

Direction: Up

Valid values:

Numeric value

Description: Article number of the sensor channel.

Parameter: String SA_SensorName SA_SensorName

Direction: Up

Description: Name of the sensor channel.

Parameter: String SA_SerialNumber SA_SerialNumber

Direction: Up

Valid values:

Numeric value

Description: Serial number of the sensor channel.

Parameter: double SA_Offset	SA_Offset
Direction: Up	
Valid values:	
Minimum: -1000000000.0	
Maximum: 1000000000.0	
Unit: See parameter SA_Unit	
Description: Offset of the sensor channel.	
Parameter: double SA_Range	SA_Range
Direction: Up	
Valid values:	
Minimum: -1000000000.0	
Maximum: 1000000000.0	
Unit: See parameter SA_Unit	
Description: Range of sensor channel.	
Parameter: String SA_Unit	SA_Unit
Direction: Up	
Description: Unit of parameter SA_Offset and SA_Range, e.g. μm or mm.	

14.5.8.4 Get_ChannelInfos

Calls the sensor command Get_ChannelInfo for any requested channel.

Parameter: int SP_Complete	SP_Complete
Direction: Down	
Valid values:	
0= FALSE	
1= TRUE	
Description: Specifies if any possible channel should be requested or only known channels (from former call to ChannelStatus or Get_Status).	
Parameter: String SA_ArticleNumber1..8	SA_ArticleNumber1..8
Direction: Up	
Valid values:	
Numeric value	
Description: Article number of the sensor channel 1 to 8.	
Parameter: String SA_SensorName1..8	SA_SensorName1..8
Direction: Up	
Description: Name of the sensor channel 1 to 8.	
Parameter: String SA_SerialNumber1..8	SA_SerialNumber1..8
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the sensor channel 1 to 8.	
Parameter: double SA_Offset1..8	SA_Offset1..8
Direction: Up	
Valid values:	
Minimum: -1000000000.0	
Maximum: 1000000000.0	
Unit: See parameter SA_Unit	
Description: Offset of the sensor channel 1 to 8.	

14.5. Commands for DT6500

Parameter: double SA_Range1..8	SA_Range1..8
Direction: Up	
Valid values:	
Minimum: -1000000000.0	
Maximum: 1000000000.0	
Unit: See parameter SA_Unit	
Description: Range of sensor channel 1 to 8.	
Parameter: String SA_Unit1..8	SA_Unit1..8
Direction: Up	
Description: Unit of parameter SA_Offset and SA_Range, e.g. μm or mm.	

14.5.8.5 Get_ControllerInfo (COI)

Retrieve information about the controller. This command is only available for new controller DT6530.

Parameter: String SA_ArticleNumber	SA_ArticleNumber
Direction: Up	
Description: Article number of the controller.	
Parameter: String SA_ControllerName	SA_ControllerName
Direction: Up	
Description: Name of the controller.	
Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the controller.	
Parameter: String SA_Option	SA_Option
Direction: Up	
Valid values:	
Numeric value	
Description: Option of the controller.	
Parameter: String SA_Softwareversion	SA_Softwareversion
Direction: Up	
Description: Software version of firmware in the controller.	

14.5.9 Parameter management

14.5.9.1 Save_Setup (SSU)

Save the current setup of controller to flash. This command is only available for old controller DT6500, not DT6530.

14.5.9.2 Read_Setup (RSU)

Read the setup from controller flash. This command is only available for old controller DT6500, not DT6530.

14.5.9.3 Factory_Defaults (FDE)

Restore the controller to factory defaults.

To save the default values call Save_Setup too. The new parameters are returned.

Parameter: int SA_SRIndex

SA_SRIndex

Direction: Up

Valid values:

- 0= 2.60 Hz
- 1= 5.21 Hz
- 2= 10.42 Hz
- 3= 15.63 Hz
- 4= 26.04 Hz
- 5= 31.25 Hz
- 6= 52.08 Hz
- 7= 62.50 Hz
- 8= 104.17 Hz
- 9= 520.83 Hz
- 10= 1041.67 Hz
- 11= 2083.33 Hz
- 12= 3906.25 Hz
- 13= 7812.50 Hz

Description: Samplerate index.

Parameter: int SA_AvrType

SA_AvrType

Direction: Up

Valid values:

- 0= off
- 1= Moving average
- 2= Mean (arithmetic)
- 3= Median
- 4= Dynamic noise rejection

Description: Averaging type at controller.

Parameter: int SA_AvrNbr

SA_AvrNbr

Direction: Up

Valid values:

- Minimum: 2
- Maximum: 8

Description: Averaging number at controller.

Parameter: int SA_ChExist1..8

SA_ChExist1..8

Direction: Up

Valid values:

- 0= Channel not available
- 1= Measured channel
- 2= Mathematic channel

Description: Channel 1 to 8 is available at controller.

Parameter: int SA_ChTransmit1..8

SA_ChTransmit1..8

Direction: Up

Valid values:

- 0= no
- 1= yes

Description: Channel 1 to 8 is transmitted.

Parameter: int SA_TrMode SA_TrMode

Direction: Up

Valid values:

- 0= Off
- 1= Rising edge
- 2= High level
- 3= Gate at rising edge

Description: Trigger active/disabled.

Parameter: int SA_LinMode1..8 SA_LinMode1..8

Direction: Up

Valid values:

- 0= off
- 1= offset correction
- 2= 2 point linearization
- 3= 3 point linearization
- 4= 5 point linearization
- 5= 10 point linearization

Description: Linearisation mode for channel 1 to 8.

Parameter: int SA_ShowChannels SA_ShowChannels

Direction: Up

Valid values:

- 0= none
- 1= all
- 2= selected

Description: Channels to be displayed.

Parameter: int SA_ShowLinearized SA_ShowLinearized

Direction: Up

Valid values:

- 0= off
- 1= on

Description: Channels should be displayed linearized.

14.5.10 Internal commands

14.5.10.1 Update_Firmware

Update firmware version at controller.

This command is only available for new controller DT6530.

Attention! This function can take approx. 1 minute. This is an internal command. It should not be used by the customer.

Parameter: Binary data SP_FirmwareFile SP_FirmwareFile

Direction: Down

Valid values:

Firmware file as binary data.

Description: Firmware version, read from file.

Parameter: int SA_Result SA_Result
Direction: Up
Valid values:
 0= Failed
 1= Success
Description: Result of firmware update.

14.5.11 Special commands

14.5.11.1 Use_Defaults

This command tells the driver to use default values to operate with sensor data. If some parameters are not specified they are not changed. The sensor is not affected by this command.

Parameter: double IP_Range1..8 IP_Range1..8
Direction: Down
Valid values:
Minimum: -1.79769e+308 (-DBL_MAX)
Maximum: 1.79769e+308 (DBL_MAX)
Unit: μm or mm
Description: Tells the driver the range of sensor for channel 1 to 8. It is used to scale the raw sensor values into μm or mm. If it is zero, no scaling is done.

14.5.11.2 Get_DrvSetting

Returns the current settings of the driver used for operating with sensor data. It is the opposite of Use_Defaults. The sensor is not affected by this command.

Parameter: double IA_Range1..8 IA_Range1..8
Direction: Up
Valid values:
Minimum: -1.79769e+308 (-DBL_MAX)
Maximum: 1.79769e+308 (DBL_MAX)
Unit: μm or mm
Description: The range of sensor for channel 1 to 8, used by driver to scale values into μm or mm.

15 Commands for Color (ACS) sensors

15.1 Commands for ACS7000

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [RS232](#) (additional, e.g. IF2001_USB (RS422) and RS232 high level interface).
- [IF2004](#) (native).
- [TCP/IP](#) (native).
- [IF2004_USB](#) (native).
- [IF2008](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_AllParameters](#) (SP_Additional= 1) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib timeout checking, to calculate datarate, to interpret and scale data and to assign values. If second bit of [IP_AutomaticMode](#) is set (2), MEDAQLib calls optionally sensor command [Set_IPDataTransferMode](#) and [Set_DataOutInterface](#) at [OpenSensor](#).

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor.
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command [Get_AllParameters](#) (SP_Additional= 1)).

The values of one data frame are filled in the arrays one after another. Each array always starts with a new data frame.

15.1.1 General commands

15.1.1.1 General

15.1.1.1.1 Get_Help (HELP)

Retrieve a help text from controller for a specific command.

Parameter: String SP_Command

SP_Command

Direction: Down

Valid values:

"" (empty string, means general help)
or any command name

Description: Name of the command.

Parameter: String SA_HelpText SA_HelpText
Direction: Up
Description: Help text to the command.

15.1.1.1.2 Get_Info (GETINFO)

Retrieve information about the controller.

Parameter: String SA_Sensor SA_Sensor
Direction: Up
Description: Name of the controller.

Parameter: String SA_SerialNumber SA_SerialNumber
Direction: Up
Valid values:
 Numeric value
Description: Serial number of the controller.

Parameter: String SA_Option SA_Option
Direction: Up
Valid values:
 Numeric value
Description: Option of the controller.

Parameter: String SA_ArticleNumber SA_ArticleNumber
Direction: Up
Valid values:
 Numeric value
Description: Article number of the controller.

Parameter: String SA_MacAddress SA_MacAddress
Direction: Up
Valid values:
 Valid MAC address in form of xx-xx-xx-xx-xx-xx
Description: MAC address (low level ethernet address) of the controller.

Parameter: String SA_Softwareversion SA_Softwareversion
Direction: Up
Description: Software version of firmware in the controller.

Parameter: String SA_Imagetype SA_Imagetype
Direction: Up
Description: Firmware image type used by the controller.

Parameter: String SA_Webstatic SA_Webstatic
Direction: Up
Valid values:
 Numeric value
Description: Version of webstatic section.

15.1.1.1.3 Set_Echo (ECHO)

Set echo for sensor commands.

Parameter: int SP_Echo SP_Echo
Direction: Down
Valid values:
 0= Off
 1= On
Description: Echo mode.

15.1.1.1.4 Get_Echo (ECHO)

Get the echo mode.

Parameter: int SA_Echo SA_Echo
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= Off
 1= On
Description: Echo mode.

15.1.1.1.5 Get_AllParameters (PRINT)

Get all parameters from controller.

Parameter: int SP_Additional SP_Additional
Direction: Down
Valid values:
 0= No
 1= Yes
Description: If set, additional information about controller and color table is output.

Parameter: int SA_AveragingType SA_AveragingType
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= None
 1= Moving average (MOVING)
 2= Recursive averaging (RECURSIVE)
 3= Median
Description: Averaging type.

Parameter: int SA_MovingCount SA_MovingCount

Direction: Up

Valid values:

- 2
- 4
- 8
- 16
- 32
- 64
- 128
- 256
- 512
- 1024

Description: Number of value for the averaging window. This parameter is only available at moving average.

Parameter: int SA_RecursiveCount SA_RecursiveCount

Direction: Up

Valid values:

- Minimum: 2
- Maximum: 32768

Description: Number of values for recursive averaging. This parameter is only available at recursive average.

Parameter: int SA_MedianCount SA_MedianCount

Direction: Up

Valid values:

- 3
- 5
- 7
- 9

Description: Number of values to build median. This parameter is only available at median.

Parameter: int SA_SensorBaudrate SA_SensorBaudrate

Direction: Up

Valid values:

- 4000000
- 3500000
- 2000000
- 1500000
- 921600
- 691200
- 460800
- 230400
- 115200
- 9600

Unit: Baud

Description: Baudrate of controller.

Parameter: int SA_DigitalOutBinFormat SA_DigitalOutBinFormat

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = MSB
- 1 = LSB

Description: Pin assignent at digital outputs.

Parameter: int SA_DigitalOutColorFormat	SA_DigitalOutColorFormat
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = None	
1 = Binary	
2 = Channel	
3 = Lab-Check	
Description: Color format at digital outputs.	
Parameter: int SA_ColorSpace	SA_ColorSpace
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = XYZ	
1 = LAB	
Description: Actually used colorspace	
Parameter: double SA_DeltaKL	SA_DeltaKL
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 3.0	
Description: Weighting factor	
Parameter: double SA_DeltaKC	SA_DeltaKC
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 3.0	
Description: Weighting factor	
Parameter: double SA_DeltaKH	SA_DeltaKH
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 3.0	
Description: Weighting factor	
Parameter: int SA_DeltaMode	SA_DeltaMode
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Euclidean distance (EUKLID)	
1 = Distance according to DIN99	
2 = Distance according to CIE94	
3 = Distance according to CMC	
4 = Distance according to CIEDE2000	
5 = Cylindrical distance model (CYLINDER)	
6 = Box distance model (BOX)	
Description: Delta mode	

Parameter: int SA_DistanceMode	SA_DistanceMode
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Best hit (BESTHIT)	
1 = Selection	
Description: Distance mode.	
Parameter: int SA_Echo	SA_Echo
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Off	
1 = On	
Description: Echo mode.	
Parameter: int SA_EthernetMode	SA_EthernetMode
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Ethernet	
1 = Ethercat	
Description: Ethernet mode.	
Parameter: int SA_UserLevel	SA_UserLevel
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = User	
1 = Professional	
Description: Actual user level.	
Parameter: int SA_DHCPEnabled	SA_DHCPEnabled
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = FALSE	
1 = TRUE	
Description: Get settings if controller should use a static IP address ask for IP at DHCP server (dynamic IP address).	
Parameter: String SA_Address	SA_Address
Direction: Up	
Valid values:	
Valid IP address in form of xxx.xxx.xxx.xxx	
Description: IP adress of the controller. If DHCP is enabled it returns the currently assigned IP address.	
Parameter: String SA_SubnetMask	SA_SubnetMask
Direction: Up	
Valid values:	
Valid network mask (e.g. 255.255.255.0 for a Class C network)	
Description: Network mask of the controller. If DHCP is enabled it returns the currently assigned network mask.	

Parameter: String SA_Gateway	SA_Gateway
Direction: Up	
Valid values:	
Valid IP address of default gateway in form of xxx.xxx.xxx.xxx	
Description: Address of the default gateway. If DHCP is enabled it returns the currently assigned default gateway.	
Parameter: int SA_Keylock	SA_Keylock
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Inactive	
1 = Active	
2 = Automatic (AUTO)	
Description: Keylock.	
Parameter: int SA_KeylockTime	SA_KeylockTime
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 60	
Unit: Minutes	
Description: Keylock time (only available at automatic keylock).	
Parameter: int SA_KeylockState	SA_KeylockState
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Inactive (IS_INACTIVE)	
1 = Active (IS_ACTIVE)	
Description: Actual keylock state (only available at automatic keylock).	
Parameter: int SA_LEDControl	SA_LEDControl
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Max	
1 = Min	
2 = Manual	
3 = Automatic (AUTO)	
4 = Off	
Description: LED control	
Parameter: int SA_LEDIntensityColdWhite	SA_LEDIntensityColdWhite
Direction: Up	
Valid values:	
Minimum: 50	
Maximum: 1023	
Description: LED intensity of cold white quadrant	
Parameter: int SA_LEDIntensityGreen	SA_LEDIntensityGreen
Direction: Up	
Valid values:	
Minimum: 50	
Maximum: 1023	
Description: LED intensity of green quadrant	

Parameter: int SA_LEDIntensityWarmWhite	SA_LEDIntensityWarmWhite
Direction: Up	
Valid values:	
Minimum: 50	
Maximum: 1023	
Description: LED intensity of warm white quadrant	
Parameter: int SA_LEDIntensityViolet	SA_LEDIntensityViolet
Direction: Up	
Valid values:	
Minimum: 50	
Maximum: 1023	
Description: LED intensity of violet quadrant	
Parameter: int SA_LightSource	SA_LightSource
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = D65	
1 = D50	
2 = D75	
3 = A	
4 = C	
5 = E	
6 = F4	
7 = F7	
8 = F11	
Description: Light source (illuminant)	
Parameter: int SA_MeasureMode	SA_MeasureMode
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Color measurement (COLORMEASURE)	
1 = Color detection (COLORDETECTION)	
2 = Video (VIDEOSPECTRUM)	
Description: Measure mode.	
Parameter: double SA_Measrate	SA_Measrate
Direction: Up	
Valid values:	
Minimum: 20	
Maximum: 2000	
Unit: Hz	
Description: Samplerate of measurement.	
Parameter: int SA_Protocol	SA_Protocol
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = TCP server (SERVER/TCP)	
1 = TCP client (CLIENT/TCP)	
2 = UDP sender (CLIENT/UDP)	
3 = None	
Description: Specifies if data should be send using TCP or UDP.	

Parameter: String SA_RemoteAddress	SA_RemoteAddress
Direction: Up	
Valid values:	
Valid IP address of receiver of data	
Description: Address of remote computer to send data to.	
Parameter: int SA_Port	SA_Port
Direction: Up	
Valid values:	
Minimum: 1024	
Maximum: 65535	
Description: Port to send data to or to listen for incoming requests.	
Parameter: int SA_Observer	SA_Observer
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Two degree (TWO_DEGREE)	
1 = Ten degree (TEN_DEGREE)	
Description: Viewing angle of observer.	
Parameter: int SA_OutputVideoRaw_ETH	SA_OutputVideoRaw_ETH
Direction: Up	
Valid values:	
0 = no	
1 = yes	
Description: Specify if raw video signal is transmitted.	
Parameter: int SA_OutputVideoDark_ETH	SA_OutputVideoDark_ETH
Direction: Up	
Valid values:	
0 = no	
1 = yes	
Description: Specify if dark corrected video signal is transmitted.	
Parameter: int SA_OutputVideoLinearized_ETH	SA_OutputVideoLinearized_ETH
Direction: Up	
Valid values:	
0 = no	
1 = yes	
Description: Specify if linearized video signal is transmitted.	
Parameter: int SA_OutputVideoLightSpectrum_ETH	SA_OutputVideoLightSpectrum_ETH
Direction: Up	
Valid values:	
0 = no	
1 = yes	
Description: Specify if light referenced signal is transmitted.	
Parameter: int SA_OutputStatusFramerate_ETH	SA_OutputStatusFramerate_ETH
Direction: Up	
Valid values:	
0 = no	
1 = yes	
Description: Specify if framerate is transmitted.	

Parameter: int SA_OutputStatusShutter_ETH	SA_OutputStatusShutter_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	
Parameter: int SA_OutputStatusTempDetector_ETH	SA_OutputStatusTempDetector_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if temperature of detector is transmitted.	
Parameter: int SA_OutputStatusTempLightSrc_ETH	SA_OutputStatusTempLight-Src_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if temperature of light source is transmitted.	
Parameter: int SA_OutputLightSensorRed_ETH	SA_OutputLightSensorRed_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if red part of light sensor is transmitted.	
Parameter: int SA_OutputLightSensorGreen_ETH	SA_OutputLightSensor-Green_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if green part of light sensor is transmitted.	
Parameter: int SA_OutputLightSensorBlue_ETH	SA_OutputLightSensor-Blue_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if blue part of light sensor is transmitted.	
Parameter: int SA_OutputLightSensorBright_ETH	SA_OutputLightSensor-Bright_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if brightness of light sensor is transmitted.	
Parameter: int SA_OutputStatusCounter_ETH	SA_OutputStatusCounter_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	

Parameter: int SA_OutputStatusTimestamp_ETH	SA_OutputStatusTimestamp_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SA_OutputStatusError_ETH	SA_OutputStatusError_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if error flags are transmitted.	
Parameter: int SA_OutputColorXYZ_ETH	SA_OutputColorXYZ_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in XYZ is transmitted.	
Parameter: int SA_OutputColorRGB_ETH	SA_OutputColorRGB_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in RGB is transmitted.	
Parameter: int SA_OutputColorLAB_ETH	SA_OutputColorLAB_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in L*a*b* is transmitted.	
Parameter: int SA_OutputColorLUV_ETH	SA_OutputColorLUV_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in L*u*v* is transmitted.	
Parameter: int SA_OutputColorLCH_ETH	SA_OutputColorLCH_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in L*c*h* is transmitted.	
Parameter: int SA_OutputColorLAB99_ETH	SA_OutputColorLAB99_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in L*a*b*99 is transmitted.	

Parameter: int SA_OutputColorLCH99_ETH	SA_OutputColorLCH99_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in L*c*h*99 is transmitted.	
Parameter: int SA_OutputDistDistance01_ETH	SA_OutputDistDistance01_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 1 is transmitted.	
Parameter: int SA_OutputDistDistance02_ETH	SA_OutputDistDistance02_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 2 is transmitted.	
Parameter: int SA_OutputDistDistance03_ETH	SA_OutputDistDistance03_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 3 is transmitted.	
Parameter: int SA_OutputDistDistance04_ETH	SA_OutputDistDistance04_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 4 is transmitted.	
Parameter: int SA_OutputDistDistance05_ETH	SA_OutputDistDistance05_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 5 is transmitted.	
Parameter: int SA_OutputDistDistance06_ETH	SA_OutputDistDistance06_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 6 is transmitted.	
Parameter: int SA_OutputDistDistance07_ETH	SA_OutputDistDistance07_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 7 is transmitted.	

Parameter: int SA_OutputDistDistance08_ETH	SA_OutputDistDistance08_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 8 is transmitted.	
Parameter: int SA_OutputDistDistance09_ETH	SA_OutputDistDistance09_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 9 is transmitted.	
Parameter: int SA_OutputDistDistance10_ETH	SA_OutputDistDistance10_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 10 is transmitted.	
Parameter: int SA_OutputDistDistance11_ETH	SA_OutputDistDistance11_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 11 is transmitted.	
Parameter: int SA_OutputDistDistance12_ETH	SA_OutputDistDistance12_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 12 is transmitted.	
Parameter: int SA_OutputDistDistance13_ETH	SA_OutputDistDistance13_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 13 is transmitted.	
Parameter: int SA_OutputDistDistance14_ETH	SA_OutputDistDistance14_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 14 is transmitted.	
Parameter: int SA_OutputDistDistance15_ETH	SA_OutputDistDistance15_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 15 is transmitted.	

Parameter: int SA_OutputDistDistance16_ETH	SA_OutputDistDistance16_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 16 is transmitted.	
Parameter: int SA_OutputDistMinDistance_ETH	SA_OutputDistMinDis- tance_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if minimum color distance is transmitted.	
Parameter: int SA_OutputDistDetectedID_ETH	SA_OutputDistDetectedID_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if no. of detected color is transmitted.	
Parameter: int SA_OutputDistMinDistID_ETH	SA_OutputDistMinDistID_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if no. of nearest color is transmitted.	
Parameter: int SA_OutputStatisticMin_ETH	SA_OutputStatisticMin_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic minimum value is transmitted.	
Parameter: int SA_OutputStatisticMax_ETH	SA_OutputStatisticMax_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic maximum value is transmitted.	
Parameter: int SA_OutputStatisticPeak2Peak_ETH	SA_OutputStatistic- Peak2Peak_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic peak to peak value is transmitted.	
Parameter: int SA_OutputStatusFramerate_RS422	SA_OutputStatusFramer- ate_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if framerate is transmitted.	

Parameter: int SA_OutputStatusShutter_RS422	SA_OutputStatusShutter_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	
Parameter: int SA_OutputStatusTempDetector_RS422	SA_OutputStatusTempDetec- tor_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if temperature of detector is transmitted.	
Parameter: int SA_OutputStatusTempLightSrc_RS422	SA_OutputStatusTempLight- Src_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if temperature of light source is transmitted.	
Parameter: int SA_OutputLightSensorRed_RS422	SA_OutputLightSensorRed_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if red part of light sensor is transmitted.	
Parameter: int SA_OutputLightSensorGreen_RS422	SA_OutputLightSensor- Green_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if green part of light sensor is transmitted.	
Parameter: int SA_OutputLightSensorBlue_RS422	SA_OutputLightSensor- Blue_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if blue part of light sensor is transmitted.	
Parameter: int SA_OutputLightSensorBright_RS422	SA_OutputLightSensor- Bright_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if brightness of light sensor is transmitted.	
Parameter: int SA_OutputStatusCounter_RS422	SA_OutputStatusCounter_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	

Parameter: int SA_OutputStatusTimestamp_RS422	SA_OutputStatusTimestamp_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SA_OutputStatusError_RS422	SA_OutputStatusError_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if error flags are transmitted.	
Parameter: int SA_OutputColorXYZ_RS422	SA_OutputColorXYZ_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in XYZ is transmitted.	
Parameter: int SA_OutputColorRGB_RS422	SA_OutputColorRGB_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in RGB is transmitted.	
Parameter: int SA_OutputColorLAB_RS422	SA_OutputColorLAB_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in L*a*b* is transmitted.	
Parameter: int SA_OutputColorLUV_RS422	SA_OutputColorLUV_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in L*u*v* is transmitted.	
Parameter: int SA_OutputColorLCH_RS422	SA_OutputColorLCH_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in L*c*h* is transmitted.	
Parameter: int SA_OutputColorLAB99_RS422	SA_OutputColorLAB99_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in L*a*b*99 is transmitted.	

Parameter: int SA_OutputColorLCH99_RS422	SA_OutputColorLCH99_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in L*c*h*99 is transmitted.	
Parameter: int SA_OutputDistMinDistance_RS422	SA_OutputDistMinDistance_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if minimum color distance is transmitted.	
Parameter: int SA_OutputDistDetectedID_RS422	SA_OutputDistDetectedID_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if no. of detected color is transmitted.	
Parameter: int SA_OutputDistMinDistID_RS422	SA_OutputDistMinDistID_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if no. of nearest color is transmitted.	
Parameter: int SA_OutputStatisticMin_RS422	SA_OutputStatisticMin_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic minimum value is transmitted.	
Parameter: int SA_OutputStatisticMax_RS422	SA_OutputStatisticMax_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic maximum value is transmitted.	
Parameter: int SA_OutputStatisticPeak2Peak_RS422	SA_OutputStatisticPeak2Peak_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic peak to peak value is transmitted.	
Parameter: int SA_DigitalOutDetectedID	SA_DigitalOutDetectedID
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if no. of detected color is output.	

Parameter: int SA_HoldLastValid	SA_HoldLastValid
Direction: Up	
Valid values:	
Minimum: -1	
Maximum: 1024	
Description: Values to replace by last valid value. -1 means no value to hold, 0 means never output an error value (always hold last valid value).	
Parameter: int SA_DataOutInterface	SA_DataOutInterface
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= RS422	
2= Ethernet	
3= HTTP	
4= Ethercat	
Description: Active interface for data output.	
Parameter: int SA_Resampling	SA_Resampling
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 2000	
Description: Resampling value.	
Parameter: int SA_ResampleRS422	SA_ResampleRS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: RS422 output is resampled.	
Parameter: int SA_ResampleEthernet	SA_ResampleEthernet
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Output over ethernet is resampled.	
Parameter: int SA_ShutterMode	SA_ShutterMode
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Automatic samplerate and shutter time (SEARCH)	
1= Fixed samplerate, automatic shutter time (MEAS)	
2= Fixed samplerate and shutter time (MANUAL)	
Description: Shutter mode.	
Parameter: int SA_StatisticDepth	SA_StatisticDepth
Direction: Up	
Valid values:	
Minimum: 2	
Maximum: 2147483647 (INT_MAX)	
Description: Window size for floating statistic calculation. The value must be power of two (2, 4, 8, ..., 16384). Value greater as 16384 calculates statistic over all values.	

Parameter: int SA_StatisticSignal SA_StatisticSignal

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= Color XYZ (XYZ)
- 2= Color RGB (RGB)
- 3= Color L*a*b* (LAB)
- 4= Color L*u*v* (LUV)
- 5= Color L*c*h* (LCH)
- 6= Color L*a*b*99 (LAB99)
- 7= Color L*c*h*99 (LCH99)
- 8= Distance 1 (DIST01)
- 9= Distance 2 (DIST02)
- 10= Distance 3 (DIST03)
- 11= Distance 4 (DIST04)
- 12= Distance 5 (DIST05)
- 13= Distance 6 (DIST06)
- 14= Distance 7 (DIST07)
- 15= Distance 8 (DIST08)
- 16= Distance 9 (DIST09)
- 17= Distance 10 (DIST10)
- 18= Distance 11 (DIST11)
- 19= Distance 12 (DIST12)
- 20= Distance 13 (DIST13)
- 21= Distance 14 (DIST14)
- 22= Distance 15 (DIST15)
- 23= Distance 16 (DIST16)
- 24= Minimum color distance (MINDIST)
- 25= No. of detected color (DETECTID)
- 26= No. of nearest color (MINDISTID)

Description: Value which is used for statistic calculation.

Parameter: int SA_DefaultUser SA_DefaultUser

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= User
- 1= Professional

Description: Default user level.

Parameter: int SA_SyncMode SA_SyncMode

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= Master
- 2= Slave

Description: Synchronisation mode.

Parameter: int SA_TriggerMode SA_TriggerMode

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor

0= None
 1= Edge
 2= Level (PULSE)
 3= Software

Description: Trigger mode.

Parameter: int SA_TriggerLevel

SA_TriggerLevel

Direction: Up

Valid values:

-1= Unknown parameter value from sensor
 0= High (Rising edge / High-active)
 1= Low (Falling edge / Low-active)

Description: Trigger level.

Parameter: int SA_TriggerCount

SA_TriggerCount

Direction: Up

Valid values:

Minimum: 0
Maximum: 16383

Description: Number of values to measure. 0 means no trigger, 16383 means endless measurement.

Parameter: int SA_VideoAverage

SA_VideoAverage

Direction: Up

Valid values:

-1= Unknown parameter value from sensor
 0= None
 1= Recursive over 2 lines (REC2)
 2= Recursive over 4 lines (REC4)
 3= Recursive over 8 lines (REC8)
 4= Recursive over 16 lines (REC16)
 5= Recursive over 32 lines (REC32)
 6= Recursive over 64 lines (REC64)
 7= Recursive over 128 lines (REC128)

Description: Averaging mode.

Parameter: String SA_ColorTable

SA_ColorTable

Direction: Up

Description: Whole table in one string, separated by new lines.

Parameter: int SA_ColorTableCount

SA_ColorTableCount

Direction: Up

Valid values:

Minimum: 0
Maximum: 16

Description: Number of entries in the table. All following parameters exists from 1 to this number, e.g. SA_Pos1, SA_Pos2, ...

Parameter: int SA_Pos1..x

SA_Pos1..x

Direction: Up

Valid values:

Minimum: 1
Maximum: 16

Description: Index of the color in the table.

Parameter: String SA_ColorName1..x	SA_ColorName1..x
Direction: Up	
Description: Name of the color in the table.	
Parameter: int SA_Observer1..x	SA_Observer1..x
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Two degree (2 deg)	
1= Ten degree (10 deg)	
Description: Viewing angle of observer.	
Parameter: int SA_LightSource1..x	SA_LightSource1..x
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= D65	
1= D50	
2= D75	
3= A	
4= C	
5= E	
6= F4	
7= F7	
8= F11	
Description: Light source (illuminant)	
Parameter: double SA_L*1..x	SA_L*1..x
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 255.0	
Description: L*, only availabe if colorspace is LAB	
Parameter: double SA_a*1..x	SA_a*1..x
Direction: Up	
Valid values:	
Minimum: -256.0	
Maximum: 255.0	
Description: a*, only availabe if colorspace is LAB	
Parameter: double SA_b*1..x	SA_b*1..x
Direction: Up	
Valid values:	
Minimum: -256.0	
Maximum: 255.0	
Description: b*, only availabe if colorspace is LAB	
Parameter: double SA_X1..x	SA_X1..x
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 255.0	
Description: X, only availabe if colorspace is XYZ	

Parameter: double SA_Y1..x	SA_Y1..x
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 255.0	
Description: Y, only available if colorspace is XYZ	
Parameter: double SA_Z1..x	SA_Z1..x
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 255.0	
Description: Z, only available if colorspace is XYZ	
Parameter: int SA_Spectrum1..x	SA_Spectrum1..x
Direction: Up	
Valid values:	
0= not available	
1= available	
Description: Spectrum	
Parameter: String SA_Sensor	SA_Sensor
Direction: Up	
Description: Name of the controller.	
Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the controller.	
Parameter: String SA_Option	SA_Option
Direction: Up	
Valid values:	
Numeric value	
Description: Option of the controller.	
Parameter: String SA_ArticleNumber	SA_ArticleNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Article number of the controller.	
Parameter: String SA_MacAddress	SA_MacAddress
Direction: Up	
Valid values:	
Valid MAC address in form of xx-xx-xx-xx-xx-xx	
Description: MAC address (low level ethernet address) of the controller.	
Parameter: String SA_Softwareversion	SA_Softwareversion
Direction: Up	
Description: Software version of firmware in the controller.	
Parameter: String SA_ImageType	SA_ImageType
Direction: Up	
Description: Firmware image type used by the controller.	

Parameter: String SA_Webstatic SA_Webstatic
Direction: Up
Valid values:
 Numeric value
Description: Version of webstatic section.

15.1.1.1.6 Set_SyncMode (SYNC)

Set the synchronisation mode.

Parameter: int SP_SyncMode SP_SyncMode
Direction: Down
Valid values:
 0= None
 1= Master
 2= Slave
Description: Synchronisation mode.

15.1.1.1.7 Get_SyncMode (SYNC)

Get the synchronisation mode.

Parameter: int SA_SyncMode SA_SyncMode
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= None
 1= Master
 2= Slave
Description: Synchronisation mode.

15.1.1.1.8 Reset_Boot (RESET)

Resets the sensor.

15.1.1.1.9 Set_Keylock (KEYLOCK)

Set key lock for sensor.

Parameter: int SP_Keylock SP_Keylock
Direction: Down
Valid values:
 0= Inactive
 1= Active
 2= Automatic (AUTO)
Description: Keylock.

Parameter: int SP_KeylockTime SP_KeylockTime
Direction: Down
Valid values:
Minimum: 0
Maximum: 60
Unit: Minutes
Description: Keylock time (only used at automatic keylock).

15.1.1.1.10 Get_Keylock (KEYLOCK)

Get key lock for sensor.

Parameter: int SA_Keylock SA_Keylock
Direction: Up
Valid values:
-1 = Unknown parameter value from sensor
0 = Inactive
1 = Active
2 = Automatic (AUTO)
Description: Keylock.

Parameter: int SA_KeylockTime SA_KeylockTime
Direction: Up
Valid values:
Minimum: 0
Maximum: 60
Unit: Minutes
Description: Keylock time (only available at automatic keylock).

Parameter: int SA_KeylockState SA_KeylockState
Direction: Up
Valid values:
-1 = Unknown parameter value from sensor
0 = Inactive (IS_INACTIVE)
1 = Active (IS_ACTIVE)
Description: Actual keylock state (only available at automatic keylock).

15.1.1.2 User Level

15.1.1.2.1 Logout (LOGOUT)

Change user level to user.

15.1.1.2.2 Login (LOGIN)

Change user level to professional.

Parameter: String SP_Password SP_Password
Direction: Down
Description: Valid password to login.

15.1.1.2.3 Get_UserLevel (GETUSERLEVEL)

Retrieve actual user level.

Parameter: int SA_UserLevel SA_UserLevel
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = User
 1 = Professional
Description: Actual user level.

15.1.1.2.4 Set_DefaultUser (STDUSER)

Set the default user level after booting the system.

Parameter: int SP_DefaultUser SP_DefaultUser
Direction: Down
Valid values:
 0 = User
 1 = Professional
Description: Default user level.

15.1.1.2.5 Get_DefaultUser (STDUSER)

Get the default user level after booting the system.

Parameter: int SA_DefaultUser SA_DefaultUser
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = User
 1 = Professional
Description: Default user level.

15.1.1.2.6 Set_Password (PASSWD)

Change the password for login.

Parameter: String SP_OldPassword SP_OldPassword
Direction: Down
Description: Old password.

Parameter: String SP_NewPassword SP_NewPassword
Direction: Down
Description: New password.

15.1.1.3 Sensor

15.1.1.3.1 Set_Observer (OBSERVER)

Set the viewing angle.

Parameter: int SP_Observer

SP_Observer

Direction: Down

Valid values:

0= Two degree (TWO_DEGREE)

1= Ten degree (TEN_DEGREE)

Description: Viewing angle of observer.

15.1.1.3.2 Get_Observer (OBSERVER)

Get the viewing angle.

Parameter: int SA_Observer

SA_Observer

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= Two degree (TWO_DEGREE)

1= Ten degree (TEN_DEGREE)

Description: Viewing angle of observer.

15.1.1.3.3 Set_LightSource (LQSRC)

Set the light source.

Parameter: int SP_LightSource

SP_LightSource

Direction: Down

Valid values:

0= D65

1= D50

2= D75

3= A

4= C

5= E

6= F4

7= F7

8= F11

Description: Light source (illuminant)

15.1.1.3.4 Get_LightSource (LQSRC)

Get the light source.

Parameter: int SA_LightSource

SA_LightSource

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= D65
- 1= D50
- 2= D75
- 3= A
- 4= C
- 5= E
- 6= F4
- 7= F7
- 8= F11

Description: Light source (illuminant)

15.1.1.3.5 Set_LEDControl (LEDCTRL)

Control of the illumination LED.

Parameter: int SP_LEDControl

SP_LEDControl

Direction: Down

Valid values:

- 0= Max
- 1= Min
- 2= Manual
- 3= Automatic (AUTO)
- 4= Off

Description: LED control

15.1.1.3.6 Get_LEDControl (LEDCTRL)

Control of the illumination LED.

Parameter: int SA_LEDControl

SA_LEDControl

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Max
- 1= Min
- 2= Manual
- 3= Automatic (AUTO)
- 4= Off

Description: LED control

15.1.1.3.7 Set_LEDIntensityColdWhite (LEDKW)

Set LED intensity of cold white quadrant.

Parameter: int SP_LEDIntensityColdWhite

SP_LEDIntensityColdWhite

Direction: Down

Valid values:

Minimum: 50

Maximum: 1023

Description: LED intensity of cold white quadrant

15.1.1.3.8 Get_LEDIntensityColdWhite (LEDKW)

Get LED intensity of cold white quadrant.

Parameter: int SA_LEDIntensityColdWhite

SA_LEDIntensityColdWhite

Direction: Up

Valid values:

Minimum: 50

Maximum: 1023

Description: LED intensity of cold white quadrant

15.1.1.3.9 Set_LEDIntensityGreen (LEDGR)

Set LED intensity of green quadrant.

Parameter: int SP_LEDIntensityGreen

SP_LEDIntensityGreen

Direction: Down

Valid values:

Minimum: 50

Maximum: 1023

Description: LED intensity of green quadrant

15.1.1.3.10 Get_LEDIntensityGreen (LEDGR)

Get LED intensity of green quadrant.

Parameter: int SA_LEDIntensityGreen

SA_LEDIntensityGreen

Direction: Up

Valid values:

Minimum: 50

Maximum: 1023

Description: LED intensity of green quadrant

15.1.1.3.11 Set_LEDIntensityWarmWhite (LEDWW)

Set LED intensity of warm white quadrant.

Parameter: int SP_LEDIntensityWarmWhite

SP_LEDIntensityWarmWhite

Direction: Down

Valid values:

Minimum: 50

Maximum: 1023

Description: LED intensity of warm white quadrant

15.1.1.3.12 Get_LEDIntensityWarmWhite (LEDWW)

Get LED intensity of warm white quadrant.

Parameter: int SA_LEDIntensityWarmWhite

SA_LEDIntensityWarmWhite

Direction: Up

Valid values:

Minimum: 50

Maximum: 1023

Description: LED intensity of warm white quadrant

15.1.1.3.13 Set_LEDIntensityViolet (LEDUV)

Set LED intensity of violet quadrant.

Parameter: int SP_LEDIntensityViolet

SP_LEDIntensityViolet

Direction: Down

Valid values:

Minimum: 50

Maximum: 1023

Description: LED intensity of violet quadrant

15.1.1.3.14 Get_LEDIntensityViolet (LEDUV)

Get LED intensity of violet quadrant.

Parameter: int SA_LEDIntensityViolet

SA_LEDIntensityViolet

Direction: Up

Valid values:

Minimum: 50

Maximum: 1023

Description: LED intensity of violet quadrant

15.1.1.3.15 AutoLEDAdjustment (AUTOLEDADJ)

Brightness of the illumination LED is adjusted once, using the optimum range for the selected manual measuring frequency.

15.1.1.3.16 DarkCorr (DARKCORR)

Make a dark correction.

15.1.1.3.17 LightCorr (LIGHTCORR)

Make a light correction.

15.1.1.4 Triggering

15.1.1.4.1 Set_TriggerMode (TRIGGER)

Set the trigger mode.

Parameter: int SP_TriggerMode

SP_TriggerMode

Direction: Down

Valid values:

- 0= None
- 1= Edge
- 2= Level (PULSE)
- 3= Software

Description: Trigger mode.

15.1.1.4.2 Get_TriggerMode (TRIGGER)

Get the active trigger mode.

Parameter: int SA_TriggerMode

SA_TriggerMode

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= Edge
- 2= Level (PULSE)
- 3= Software

Description: Trigger mode.

15.1.1.4.3 Set_TriggerLevel (TRIGGERLEVEL)

Set the trigger level.

Parameter: int SP_TriggerLevel

SP_TriggerLevel

Direction: Down

Valid values:

- 0= High (Rising edge / High-active)
- 1= Low (Falling edge / Low-active)

Description: Trigger level.

15.1.1.4.4 Get_TriggerLevel (TRIGGERLEVEL)

Get the active trigger level.

Parameter: int SA_TriggerLevel

SA_TriggerLevel

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= High (Rising edge / High-active)
- 1= Low (Falling edge / Low-active)

Description: Trigger level.

15.1.1.4.5 Set_TriggerCount (TRIGGERCOUNT)

Set the number of values to measure at trigger.

Parameter: int SP_TriggerCount SP_TriggerCount
Direction: Down
Valid values:
Minimum: 0
Maximum: 16383
Description: Number of values to measure. 0 means no trigger, 16383 means endless measurement.

15.1.1.4.6 Get_TriggerCount (TRIGGERCOUNT)

Get the number of values to measure at trigger.

Parameter: int SA_TriggerCount SA_TriggerCount
Direction: Up
Valid values:
Minimum: 0
Maximum: 16383
Description: Number of values to measure. 0 means no trigger, 16383 means endless measurement.

15.1.1.4.7 Software_Trigger (TRIGGERSW)

Execute a software trigger.

15.1.1.5 Interfaces

15.1.1.5.1 Set_IPConfiguration (IPCONFIG)

Set the IP configuration at controller.

Parameter: int SP_DHCPEnabled SP_DHCPEnabled
Direction: Down
Valid values:
 0 = FALSE
 1 = TRUE
Description: Specify if controller should use a static IP address or ask for IP at DHCP server (dynamic IP address).

Parameter: String SP_Address SP_Address
Direction: Down
Valid values:
 Valid IP address in form of xxx.xxx.xxx.xxx
Description: IP address of the controller. This parameter is only evaluated on static IP assignment.

Parameter: String SP_SubnetMask SP_SubnetMask

Direction: Down

Valid values:

Valid network mask (e.g. 255.255.255.0 for a Class C network)

Description: Network mask of the controller. This parameter is only evaluated on static IP assignment.

Parameter: String SP_Gateway SP_Gateway

Direction: Down

Valid values:

Valid IP address of default gateway in form of xxx.xxx.xxx.xxx

Description: The default gateway must be specified if the controller should communicate with peers in foreign subnets. This parameter is only evaluated on static IP assignment.

15.1.1.5.2 Get_IPConfiguration (IPCONFIG)

Get the IP configuration at controller.

Parameter: int SA_DHCPEnabled SA_DHCPEnabled

Direction: Up

Valid values:

-1 = Unknown parameter value from sensor

0 = FALSE

1 = TRUE

Description: Get settings if controller should use a static IP address ask for IP at DHCP server (dynamic IP address).

Parameter: String SA_Address SA_Address

Direction: Up

Valid values:

Valid IP address in form of xxx.xxx.xxx.xxx

Description: IP address of the controller. If DHCP is enabled it returns the currently assigned IP address.

Parameter: String SA_SubnetMask SA_SubnetMask

Direction: Up

Valid values:

Valid network mask (e.g. 255.255.255.0 for a Class C network)

Description: Network mask of the controller. If DHCP is enabled it returns the currently assigned network mask.

Parameter: String SA_Gateway SA_Gateway

Direction: Up

Valid values:

Valid IP address of default gateway in form of xxx.xxx.xxx.xxx

Description: Address of the default gateway. If DHCP is enabled it returns the currently assigned default gateway.

15.1.1.5.3 Set_IPDataTransferMode (MEATRANSFER)

Set IP protocol at controller.

Parameter: int SP_Protocol

SP_Protocol

Direction: Down

Valid values:

- 0= TCP server (SERVER/TCP)
- 1= TCP client (CLIENT/TCP)
- 2= UDP sender (CLIENT/UDP)
- 3= None

Description: Specifies if data should be send using TCP or UDP.

Parameter: String SP_RemoteAddress

SP_RemoteAddress

Direction: Down

Valid values:

Valid IP address of receiver of data

Description: Address of remote computer to send data to. On TCP server this parameter is ignored.

Parameter: int SP_Port

SP_Port

Direction: Down

Valid values:

- Minimum:** 1024
- Maximum:** 65535

Description: Port to send data to or to listen for incoming requests.

15.1.1.5.4 Get_IPDataTransferMode (MEATRANSFER)

Get IP protocol at controller.

Parameter: int SA_Protocol

SA_Protocol

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= TCP server (SERVER/TCP)
- 1= TCP client (CLIENT/TCP)
- 2= UDP sender (CLIENT/UDP)
- 3= None

Description: Specifies if data should be send using TCP or UDP.

Parameter: String SA_RemoteAddress

SA_RemoteAddress

Direction: Up

Valid values:

Valid IP address of receiver of data

Description: Address of remote computer to send data to.

Parameter: int SA_Port

SA_Port

Direction: Up

Valid values:

- Minimum:** 1024
- Maximum:** 65535

Description: Port to send data to or to listen for incoming requests.

15.1.1.5.5 Set_EthernetMode (ETHERMODE)

Switches ethernet mode between Ethernet and Ethercat.

Parameter: int SP_EthernetMode

SP_EthernetMode

Direction: Down

Valid values:

0= Ethernet

1= Ethercat

Description: Ethernet mode.

15.1.1.5.6 Get_EthernetMode (ETHERMODE)

Get ethernet mode of controller.

Parameter: int SA_EthernetMode

SA_EthernetMode

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= Ethernet

1= Ethercat

Description: Ethernet mode.

15.1.1.5.7 Set_Baudrate (BAUDRATE)

Set baudrate of controller for serial RS422 communication. After receiving the answer of the command, the baudrate of the local serial interface is changed, too.

Parameter: int SP_SensorBaudrate

SP_SensorBaudrate

Direction: Down

Valid values:

4000000

3500000

2000000

1500000

921600

691200

460800

230400

115200

9600

Unit: Baud

Description: Baudrate of controller.

15.1.1.5.8 Get_Baudrate (BAUDRATE)

Get baudrate of controller for serial RS422 communication.

Parameter: int SA_SensorBaudrate

SA_SensorBaudrate

Direction: Up

Valid values:

- 4000000
- 3500000
- 2000000
- 1500000
- 921600
- 691200
- 460800
- 230400
- 115200
- 9600

Unit: Baud

Description: Baudrate of controller.

15.1.1.5.9 Set_DigitalOutColorFormat (COLOROUT_FORMAT)

Set the color format at digital outputs.

Parameter: int SP_DigitalOutColorFormat

SP_DigitalOutColorFormat

Direction: Down

Valid values:

- 0= None
- 1= Binary
- 2= Channel
- 3= Lab-Check

Description: Color format at digital outputs.

15.1.1.5.10 Get_DigitalOutColorFormat (COLOROUT_FORMAT)

Get the color format at digital outputs.

Parameter: int SA_DigitalOutColorFormat

SA_DigitalOutColorFormat

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= Binary
- 2= Channel
- 3= Lab-Check

Description: Color format at digital outputs.

15.1.1.5.11 Set_DigitalOutBinFormat (BIN_FORMAT)

Set the pin assignment at digital outputs for Set_DigitalOutColorFormat (Binary).

Parameter: int SP_DigitalOutBinFormat

SP_DigitalOutBinFormat

Direction: Down

Valid values:

0= MSB

1= LSB

Description: Pin assignent at digital outputs.

15.1.1.5.12 Get_DigitalOutBinFormat (BIN_FORMAT)

Get the pin assignment at digital outputs for Set_DigitalOutColorFormat (Binary).

Parameter: int SA_DigitalOutBinFormat

SA_DigitalOutBinFormat

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= MSB

1= LSB

Description: Pin assignent at digital outputs.

15.1.1.6 Parameter management

15.1.1.6.1 Save_Parameters (STORE)

Save actual parameters at controller. There can be saved several settings on different locations. So it is easy to switch to another setting.

Parameter: int SP_ParameterSet

SP_ParameterSet

Direction: Down

Valid values:

Minimum: 1

Maximum: 8

Description: Location to save the settings.

15.1.1.6.2 Load_Parameters (READ)

Load actual parameters into controller RAM. There can be loaded several settings from different locations. So it is easy to switch to another setting.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_ParameterType

SP_ParameterType

Direction: Down

Valid values:

0= All settings (ALL)

1= Device settings (DEVICE)

2= Measurement settings (MEAS)

Description: Specifies which settings should be loaded.

Parameter: int SP_ParameterSet SP_ParameterSet
Direction: Down
Valid values:
 Minimum: 1
 Maximum: 8
Description: Location from where the settings should be loaded.

15.1.1.6.3 Set_Default (SETDEFAULT)

Reset the controller to default settings.
 If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_DefaultType SP_DefaultType
Direction: Down
Valid values:
 0= All settings (ALL)
 1= Just the current setting (CURRENT)
 2= Only color table (COLOR)
Description: Specifies which settings should be resetted.

Parameter: int SP_KeepDevice SP_KeepDevice
Direction: Down
Valid values:
 0= no
 1= yes
Description: Specifies if device settings should be kept temporary.

Parameter: int SP_SaveTemporary SP_SaveTemporary
Direction: Down
Valid values:
 0= no
 1= yes
Description: Specifies if temporary settings should be stored in a setup.

15.1.2 Measurement

15.1.2.1 General

15.1.2.1.1 Set_MeasureMode (MEASMODE)

Set the measure mode.
 If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_MeasureMode SP_MeasureMode
Direction: Down
Valid values:
 0= Color measurement (COLORMEASURE)
 1= Color detection (COLORDETECTION)
 2= Video (VIDEOSPECTRUM)
Description: Measure mode.

15.1.2.1.2 Get_MeasureMode (MEASMODE)

Get the measure mode.

Parameter: int SA_MeasureMode SA_MeasureMode
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = Color measurement (COLORMEASURE)
 1 = Color detection (COLORDETECTION)
 2 = Video (VIDEOSPECTRUM)
Description: Measure mode.

15.1.2.1.3 Set_ShutterMode (SHUTTERMODE)

Set the shutter mode.

Parameter: int SP_ShutterMode SP_ShutterMode
Direction: Down
Valid values:
 0 = Automatic samplerate and shutter time (SEARCH)
 1 = Fixed samplerate, automatic shutter time (MEAS)
 2 = Fixed samplerate and shutter time (MANUAL)
Description: Shutter mode.

15.1.2.1.4 Get_ShutterMode (SHUTTERMODE)

Get the shutter mode.

Parameter: int SA_ShutterMode SA_ShutterMode
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = Automatic samplerate and shutter time (SEARCH)
 1 = Fixed samplerate, automatic shutter time (MEAS)
 2 = Fixed samplerate and shutter time (MANUAL)
Description: Shutter mode.

15.1.2.1.5 Get_Video (GETVIDEO)

Get recent video signals from sensor.

Parameter: Binary data SA_VideoRaw SA_VideoRaw
Direction: Up
Valid values:
 256 words (each 2 byte), each word is an intensity value.
Description: Raw video signal

Parameter: Binary data SA_VideoDark	SA_VideoDark
Direction: Up	
Valid values:	256 words (each 2 byte), each word is an dark corrected value.
Description:	Dark corrected video signal
Parameter: Binary data SA_VideoLin	SA_VideoLin
Direction: Up	
Valid values:	256 words (each 2 byte), each word is an linearized value.
Description:	Linearized video signal
Parameter: Binary data SA_VideoLight	SA_VideoLight
Direction: Up	
Valid values:	256 words (each 2 byte), each word is an light referenced value.
Description:	Light referenced video signal
Parameter: double SA_VideoTimestamp	SA_VideoTimestamp
Direction: Up	
Valid values:	Minimum: 0 Maximum: 1.79769e+308 (DBL_MAX)
Unit: ms	
Description:	Timestamp of the video signal. It starts from 1970 Jan 01 at 01:00. It is generated when the video has arrived at TCP/IP socket.

Example how to read a video signal from sensor:

```

/* Do not forget to handle potential error after each call to MEDAQLib! */
/* Create sensor instance, open sensor via TCP/IP, set output to ethernet */
/* and than switch to video mode: */
err= SetIntExecSCmd (instance, "Set_MeasureMode", "SP_MeasureMode", 3 /*Video*/);

/* Select the desired video signal: */
err= SetParameterInt (instance, "SP_OutputVideoRaw_ETH", 1);
err= SetParameterInt (instance, "SP_OutputVideoDark_ETH", 1);
err= ExecSCmd (instance, "Set_OutputVideo_ETH");

/* Aquire video signals: */
err= ExecSCmd (instance, "Get_Video");
WORD videoRaw[256], videoDark[256];
DWORD maxLen= sizeof (videoRaw);
err= GetParameterBinary (instance, "SA_VideoRaw", (char *)videoRaw, &maxLen);
assert (maxlen==sizeof (videoRaw)); // additinal validity check
maxLen= sizeof (videoDark);
err= GetParameterBinary (instance, "SA_VideoDark", (char *)videoDark, &maxLen);
assert (maxlen==sizeof (videoDark)); // additinal validity check

/* Do anything with the received video signals */

```

15.1.2.1.6 Set_Samplerate (MEASRATE)

Set the samplerate.

Parameter: double SP_Measrate	SP_Measrate
Direction: Down	
Valid values:	Minimum: 20 Maximum: 2000
Unit: Hz	
Description:	Samplerate of measurement.

15.1.2.1.7 Get_Samplerate (MEASRATE)

Get the samplerate.

Parameter: double SA_Measrate

SA_Measrate

Direction: Up

Valid values:

Minimum: 20

Maximum: 2000

Unit: Hz

Description: Samplerate of measurement.

15.1.2.1.8 Set_DeltaMode (DELTAMODE)

Set the method of color difference calculation.

Parameter: int SP_DeltaMode

SP_DeltaMode

Direction: Down

Valid values:

0= Euclidean distance (EUKLID)

1= Distance according to DIN99

2= Distance according to CIE94

3= Distance according to CMC

4= Distance according to CIEDE2000

5= Cylindrical distance model (CYLINDER)

6= Box distance model (BOX)

Description: Delta mode

15.1.2.1.9 Get_DeltaMode (DELTAMODE)

Get the method of color difference calculation.

Parameter: int SA_DeltaMode

SA_DeltaMode

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= Euclidean distance (EUKLID)

1= Distance according to DIN99

2= Distance according to CIE94

3= Distance according to CMC

4= Distance according to CIEDE2000

5= Cylindrical distance model (CYLINDER)

6= Box distance model (BOX)

Description: Delta mode

15.1.2.1.10 Set_DeltaKL (DELTA_KL)

Set weighting factor for delta mode CIE94, CMC and CIEDE2000.

Parameter: double SP_DeltaKL

SP_DeltaKL

Direction: Down

Valid values:

Minimum: 0.0

Maximum: 3.0

Description: Weighting factor

15.1.2.1.11 Get_DeltaKL (DELTA_KL)

Get weighting factor for delta mode CIE94, CMC and CIEDE2000.

Parameter: double SA_DeltaKL SA_DeltaKL
Direction: Up
Valid values:
Minimum: 0.0
Maximum: 3.0
Description: Weighting factor

15.1.2.1.12 Set_DeltaKC (DELTA_KC)

Set weighting factor for delta mode CIE94, CMC and CIEDE2000.

Parameter: double SP_DeltaKC SP_DeltaKC
Direction: Down
Valid values:
Minimum: 0.0
Maximum: 3.0
Description: Weighting factor

15.1.2.1.13 Get_DeltaKC (DELTA_KC)

Get weighting factor for delta mode CIE94, CMC and CIEDE2000.

Parameter: double SA_DeltaKC SA_DeltaKC
Direction: Up
Valid values:
Minimum: 0.0
Maximum: 3.0
Description: Weighting factor

15.1.2.1.14 Set_DeltaKH (DELTA_KH)

Set weighting factor for delta mode CIE94, CMC and CIEDE2000.

Parameter: double SP_DeltaKH SP_DeltaKH
Direction: Down
Valid values:
Minimum: 0.0
Maximum: 3.0
Description: Weighting factor

15.1.2.1.15 Get_DeltaKH (DELTA_KH)

Get weighting factor for delta mode CIE94, CMC and CIEDE2000.

Parameter: double SA_DeltaKH SA_DeltaKH
Direction: Up
Valid values:
Minimum: 0.0
Maximum: 3.0
Description: Weighting factor

15.1.2.2 Color database

15.1.2.2.1 Get_ColorTable (COLORTABLE)

Get a list of all colors in database.

Parameter: int SA_ColorSpace

SA_ColorSpace

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= XYZ
- 1= LAB

Description: Actually used colorspace

Parameter: String SA_ColorTable

SA_ColorTable

Direction: Up

Description: Whole table in one string, separated by new lines.

Parameter: int SA_ColorTableCount

SA_ColorTableCount

Direction: Up

Valid values:

- Minimum:** 0
- Maximum:** 16

Description: Number of entries in the table. All following parameters exists from 1 to this number, e.g. SA_Pos1, SA_Pos2, ...

Parameter: int SA_Pos1..x

SA_Pos1..x

Direction: Up

Valid values:

- Minimum:** 1
- Maximum:** 16

Description: Index of the color in the table.

Parameter: String SA_ColourName1..x

SA_ColourName1..x

Direction: Up

Description: Name of the color in the table.

Parameter: int SA_Observer1..x

SA_Observer1..x

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Two degree (2 deg)
- 1= Ten degree (10 deg)

Description: Viewing angle of observer.

Parameter: int SA_LightSource1..x

SA_LightSource1..x

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= D65
- 1= D50
- 2= D75
- 3= A
- 4= C
- 5= E
- 6= F4
- 7= F7
- 8= F11

Description: Light source (illuminant)

Parameter: double SA_L*1..x	SA_L*1..x
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 255.0	
Description: L*, only available if colorspace is LAB	
Parameter: double SA_a*1..x	SA_a*1..x
Direction: Up	
Valid values:	
Minimum: -256.0	
Maximum: 255.0	
Description: a*, only available if colorspace is LAB	
Parameter: double SA_b*1..x	SA_b*1..x
Direction: Up	
Valid values:	
Minimum: -256.0	
Maximum: 255.0	
Description: b*, only available if colorspace is LAB	
Parameter: double SA_X1..x	SA_X1..x
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 255.0	
Description: X, only available if colorspace is XYZ	
Parameter: double SA_Y1..x	SA_Y1..x
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 255.0	
Description: Y, only available if colorspace is XYZ	
Parameter: double SA_Z1..x	SA_Z1..x
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 255.0	
Description: Z, only available if colorspace is XYZ	
Parameter: int SA_Spectrum1..x	SA_Spectrum1..x
Direction: Up	
Valid values:	
0 = not available	
1 = available	
Description: Spectrum	

15.1.2.2.2 Edit_Color (COLORNEW)

Add a new color or edit an existing color.

Parameter: int SP_Pos	SP_Pos
Direction: Down	
Valid values:	
Minimum: 1	
Maximum: 16	
Description: Index of the color to add or edit.	
Parameter: String SP_ColourName	SP_ColourName
Direction: Down	
Description: Name of the color.	
Parameter: int SP_UsedColorSpace	SP_UsedColorSpace
Direction: Down	
Valid values:	
0= XYZ	
1= LAB	
2= Spectrum	
Description: Color space to use	
Parameter: int SP_Observer	SP_Observer
Direction: Down	
Valid values:	
0= Two degree (2)	
1= Ten degree (10)	
Description: Viewing angle of observer (not used at color space spectrum).	
Parameter: int SP_LightSource	SP_LightSource
Direction: Down	
Valid values:	
0= D65	
1= D50	
2= D75	
3= A	
4= C	
5= E	
6= F4	
7= F7	
8= F11	
Description: Light source (illuminant) (not used at color space spectrum).	
Parameter: double SP_L*	SP_L*
Direction: Down	
Valid values:	
Minimum: 0.0	
Maximum: 255.0	
Description: L*, if colorspace is LAB	

Parameter: double SP_a*	SP_a*
Direction: Down	
Valid values:	
Minimum: -256.0	
Maximum: 255.0	
Description: a*, if colorspace is LAB	
Parameter: double SP_b*	SP_b*
Direction: Down	
Valid values:	
Minimum: -256.0	
Maximum: 255.0	
Description: b*, if colorspace is LAB	
Parameter: double SP_X	SP_X
Direction: Down	
Valid values:	
Minimum: 0.0	
Maximum: 255.0	
Description: X, if colorspace is XYZ	
Parameter: double SP_Y	SP_Y
Direction: Down	
Valid values:	
Minimum: 0.0	
Maximum: 255.0	
Description: Y, if colorspace is XYZ	
Parameter: double SP_Z	SP_Z
Direction: Down	
Valid values:	
Minimum: 0.0	
Maximum: 255.0	
Description: Z, if colorspace is XYZ	

15.1.2.2.3 Set_ColorDescription (COLORDESCR)

Set description of a color in color table.

Parameter: String SP_ColorName	SP_ColorName
Direction: Down	
Description: Name of the color.	
Parameter: String SP_Description	SP_Description
Direction: Down	
Description: Description of the color.	

15.1.2.2.4 Get_ColorDescription (COLORDESCR)

Get description of a color in color table.

Parameter: String SP_ColorName	SP_ColorName
Direction: Down	
Description: Name of the color.	

Parameter: String SA_Description SA_Description
Direction: Up
Description: Description of the color.

15.1.2.2.5 Set_ColorThresholds (THRESHOLDS)

Set thresholds for a color.

Parameter: String SP_ColorName SP_ColorName

Direction: Down
Description: Name of the color.

Parameter: double SP_Delta_E_L SP_Delta_E_L

Direction: Down
Valid values:
Minimum: 0.0
Maximum: 64.0
Description: Delta value for L*.

Parameter: double SP_Delta_A_AB SP_Delta_A_AB

Direction: Down
Valid values:
Minimum: 0.0
Maximum: 64.0
Description: Delta value for a*b* resp. a*.

Parameter: double SP_Delta_B SP_Delta_B

Direction: Down
Valid values:
Minimum: 0.0
Maximum: 64.0
Description: Delta value for b*.

15.1.2.2.6 Get_ColorThresholds (THRESHOLDS)

Get thresholds for a color.

Parameter: String SP_ColorName SP_ColorName
Direction: Down
Description: Name of the color.

Parameter: double SA_Delta_E_L SA_Delta_E_L

Direction: Up
Valid values:
Minimum: 0.0
Maximum: 64.0
Description: Delta value for L*.

Parameter: double SA_Delta_A_AB SA_Delta_A_AB

Direction: Up
Valid values:
Minimum: 0.0
Maximum: 64.0
Description: Delta value for a*b* resp. a*.

Parameter: double SA_Delta_B SA_Delta_B
Direction: Up
Valid values:
Minimum: 0.0
Maximum: 64.0
Description: Delta value for b*.

15.1.2.2.7 Set_ColorSpace (COLORSPACE)

Set the Colorspace used at color table.

Parameter: int SP_ColorSpace SP_ColorSpace
Direction: Down
Valid values:
 0 = XYZ
 1 = LAB
Description: Colorspace to use

15.1.2.2.8 Get_ColorSpace (COLORSPACE)

Get the Colorspace used by color table.

Parameter: int SA_ColorSpace SA_ColorSpace
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = XYZ
 1 = LAB
Description: Actually used colorspace

15.1.2.2.9 Move_Color (MOVECOLOR)

Move a color from one to another position in color table.

Parameter: int SP_ActualPos SP_ActualPos
Direction: Down
Valid values:
Minimum: 1
Maximum: 16
Description: Actual color index.

Parameter: int SP_NewPos SP_NewPos
Direction: Down
Valid values:
Minimum: 1
Maximum: 16
Description: New color index.

15.1.2.2.10 Reset_ColorMapping (RESETMAPPING)

Resets all color positions in color table to original positions.

15.1.2.2.11 Delete_Color (COLORDELETE)

Deletes a color from color table.

Parameter: String SP_ColorName

SP_ColorName

Direction: Down

Description: Name of the color.

15.1.2.2.12 Clear_ColorTable

Clear the whole color table.

15.1.2.3 Measurement value processing

15.1.2.3.1 Set_VideoAverage (VSAVERAGE)

Set video averaging (before processing).

Parameter: int SP_VideoAverage

SP_VideoAverage

Direction: Down

Valid values:

- 0= None
- 1= Recursive over 2 lines (REC2)
- 2= Recursive over 4 lines (REC4)
- 3= Recursive over 8 lines (REC8)
- 4= Recursive over 16 lines (REC16)
- 5= Recursive over 32 lines (REC32)
- 6= Recursive over 64 lines (REC64)
- 7= Recursive over 128 lines (REC128)

Description: Averaging mode.

15.1.2.3.2 Get_VideoAverage (VSAVERAGE)

Get video averaging (before processing).

Parameter: int SA_VideoAverage

SA_VideoAverage

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= Recursive over 2 lines (REC2)
- 2= Recursive over 4 lines (REC4)
- 3= Recursive over 8 lines (REC8)
- 4= Recursive over 16 lines (REC16)
- 5= Recursive over 32 lines (REC32)
- 6= Recursive over 64 lines (REC64)
- 7= Recursive over 128 lines (REC128)

Description: Averaging mode.

15.1.2.3.3 Set_Averaging (AVERAGE)

Set data averaging at controller.

Parameter: int SP_AveragingType SP_AveragingType

Direction: Down

Valid values:

- 0= None
- 1= Moving average (MOVING)
- 2= Recursive averaging (RECURSIVE)
- 3= Median

Description: Averaging type.

Parameter: int SP_MovingCount SP_MovingCount

Direction: Down

Valid values:

- 2
- 4
- 8
- 16
- 32
- 64
- 128
- 256
- 512
- 1024

Description: Number of value for the averaging window. This parameter is only used at moving average.

Parameter: int SP_RecursiveCount SP_RecursiveCount

Direction: Down

Valid values:

- Minimum: 2
- Maximum: 32768

Description: Number of values for recursive averaging. This parameter is only used at recursive average.

Parameter: int SP_MedianCount SP_MedianCount

Direction: Down

Valid values:

- 3
- 5
- 7
- 9

Description: Number of values to build median. This parameter is only used at median.

15.1.2.3.4 Get_Averaging (AVERAGE)

Get data averaging at controller.

Parameter: int SA_AveragingType SA_AveragingType

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= Moving average (MOVING)
- 2= Recursive averaging (RECURSIVE)
- 3= Median

Description: Averaging type.

Parameter: int SA_MovingCount SA_MovingCount

Direction: Up

Valid values:

- 2
- 4
- 8
- 16
- 32
- 64
- 128
- 256
- 512
- 1024

Description: Number of value for the averaging window. This parameter is only available at moving average.

Parameter: int SA_RecursiveCount SA_RecursiveCount

Direction: Up

Valid values:

- Minimum:** 2
- Maximum:** 32768

Description: Number of values for recursive averaging. This parameter is only available at recursive average.

Parameter: int SA_MedianCount SA_MedianCount

Direction: Up

Valid values:

- 3
- 5
- 7
- 9

Description: Number of values to build median. This parameter is only available at median.

15.1.2.3.5 Set_StatisticSignal (STATISTICSIGNAL)

Set the measured value which is used for statistic calculation.

Parameter: int SP_StatisticSignal

SP_StatisticSignal

Direction: Down

Valid values:

- 0= None
- 1= Color XYZ (XYZ)
- 2= Color RGB (RGB)
- 3= Color L*a*b* (LAB)
- 4= Color L*u*v* (LUV)
- 5= Color L*c*h* (LCH)
- 6= Color L*a*b*99 (LAB99)
- 7= Color L*c*h*99 (LCH99)
- 8= Distance 1 (DIST01)
- 9= Distance 2 (DIST02)
- 10= Distance 3 (DIST03)
- 11= Distance 4 (DIST04)
- 12= Distance 5 (DIST05)
- 13= Distance 6 (DIST06)
- 14= Distance 7 (DIST07)
- 15= Distance 8 (DIST08)
- 16= Distance 9 (DIST09)
- 17= Distance 10 (DIST10)
- 18= Distance 11 (DIST11)
- 19= Distance 12 (DIST12)
- 20= Distance 13 (DIST13)
- 21= Distance 14 (DIST14)
- 22= Distance 15 (DIST15)
- 23= Distance 16 (DIST16)
- 24= Minimum color distance (MINDIST)
- 25= No. of detected color (DETECTID)
- 26= No. of nearest color (MINDISTID)

Description: Value which is used for statistic calculation.

15.1.2.3.6 Get_StatisticSignal (STATISTICSIGNAL)

Get the measured value which is used for statistic calculation.

Parameter: int SA_StatisticSignal

SA_StatisticSignal

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 1= Color XYZ (XYZ)
- 2= Color RGB (RGB)
- 3= Color L*a*b* (LAB)
- 4= Color L*u*v* (LUV)
- 5= Color L*c*h* (LCH)
- 6= Color L*a*b*99 (LAB99)
- 7= Color L*c*h*99 (LCH99)
- 8= Distance 1 (DIST01)

9= Distance 2 (DIST02)
 10= Distance 3 (DIST03)
 11= Distance 4 (DIST04)
 12= Distance 5 (DIST05)
 13= Distance 6 (DIST06)
 14= Distance 7 (DIST07)
 15= Distance 8 (DIST08)
 16= Distance 9 (DIST09)
 17= Distance 10 (DIST10)
 18= Distance 11 (DIST11)
 19= Distance 12 (DIST12)
 20= Distance 13 (DIST13)
 21= Distance 14 (DIST14)
 22= Distance 15 (DIST15)
 23= Distance 16 (DIST16)
 24= Minimum color distance (MINDIST)
 25= No. of detected color (DETECTID)
 26= No. of nearest color (MINDISTID)

Description: Value which is used for statistic calculation.

15.1.2.3.7 Set_StatisticDepth (STATISTICDEPTH)

Set the window size for floating statistic calculation.

Parameter: int SP_StatisticDepth

SP_StatisticDepth

Direction: Down

Valid values:

Minimum: 2

Maximum: 2147483647 (INT_MAX)

Description: Window size for floating statistic calculation. The value must be power of two (2, 4, 8, ..., 16384). Value greater as 16384 calculates statistic over all values.

15.1.2.3.8 Get_StatisticDepth (STATISTICDEPTH)

Get the window size for floating statistic calculation.

Parameter: int SA_StatisticDepth

SA_StatisticDepth

Direction: Up

Valid values:

Minimum: 2

Maximum: 2147483647 (INT_MAX)

Description: Window size for floating statistic calculation. The value must be power of two (2, 4, 8, ..., 16384). Value greater as 16384 calculates statistic over all values.

15.1.2.3.9 Reset_Statistic (RESETSTATISTIC)

Reset the statistic (min and max values).

15.1.3 Data output

15.1.3.1 General

15.1.3.1.1 Set_DataOutInterface (OUTPUT)

Set the active interface for data output.

If first bit of **IP_AutomaticMode** is set (1), **Get_AllParameters** (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_DataOutInterface

SP_DataOutInterface

Direction: Down

Valid values:

- 0= None
- 1= RS422
- 2= Ethernet
- 3= HTTP
- 4= Ethercat

Description: Active interface for data output.

15.1.3.1.2 Get_DataOutInterface (OUTPUT)

Get the active interface for data output.

Parameter: int SA_DataOutInterface

SA_DataOutInterface

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= RS422
- 2= Ethernet
- 3= HTTP
- 4= Ethercat

Description: Active interface for data output.

15.1.3.1.3 Set_Resampling (OUTREDUCE)

Set resampling to reduce output data.

Parameter: int SP_Resampling

SP_Resampling

Direction: Down

Valid values:

- Minimum:** 1
- Maximum:** 2000

Description: Resampling value.

Parameter: int SP_ResampleRS422

SP_ResampleRS422

Direction: Down

Valid values:

- 0= no
- 1= yes

Description: Specify if RS422 output should be resampled.

Parameter: int SP_ResampleEthernet SP_ResampleEthernet
Direction: Down
Valid values:
 0= no
 1= yes
Description: Specify if output over ethernet should be resampled.

15.1.3.1.4 Get_Resampling (OUTREDUCE)

Get resampling for reducing output data.

Parameter: int SA_Resampling SA_Resampling
Direction: Up
Valid values:
Minimum: 1
Maximum: 2000
Description: Resampling value.

Parameter: int SA_ResampleRS422 SA_ResampleRS422
Direction: Up
Valid values:
 0= no
 1= yes
Description: RS422 output is resampled.

Parameter: int SA_ResampleEthernet SA_ResampleEthernet
Direction: Up
Valid values:
 0= no
 1= yes
Description: Output over ethernet is resampled.

15.1.3.1.5 Set_HoldLastValid (OUTHOLD)

Set the number of values to be replaced by last valid value instead of error values.

Parameter: int SP_HoldLastValid SP_HoldLastValid
Direction: Down
Valid values:
Minimum: -1
Maximum: 1024
Description: Values to replace by last valid value. -1 means no value to hold, 0 means never output an error value (always hold last valid value).

15.1.3.1.6 Get_HoldLastValid (OUTHOLD)

Get the number of values to be replaced by last valid value instead of error values.

Parameter: int SA_HoldLastValid SA_HoldLastValid
Direction: Up
Valid values:
Minimum: -1
Maximum: 1024
Description: Values to replace by last valid value. -1 means no value to hold, 0 means never output an error value (always hold last valid value).

15.1.3.2 Selected measurement values

15.1.3.2.1 Set_OutputVideo_ETH (OUTVIDEO)

Set the video signal to be output at ethernet interface.

Parameter: int SP_OutputVideoRaw_ETH

SP_OutputVideoRaw_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if raw video signal is transmitted.

Parameter: int SP_OutputVideoDark_ETH

SP_OutputVideoDark_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if dark corrected video signal is transmitted.

Parameter: int SP_OutputVideoLinearized_ETH

SP_OutputVideoLinearized_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if linearized video signal is transmitted.

Parameter: int SP_OutputVideoLightSpectrum_ETH

SP_OutputVideoLightSpectrum_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if light referenced video signal is transmitted.

15.1.3.2.2 Get_OutputVideo_ETH (OUTVIDEO)

Get the video signal to be output at ethernet interface.

Parameter: int SA_OutputVideoRaw_ETH

SA_OutputVideoRaw_ETH

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if raw video signal is transmitted.

Parameter: int SA_OutputVideoDark_ETH

SA_OutputVideoDark_ETH

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if dark corrected video signal is transmitted.

Parameter: int SA_OutputVideoLinearized_ETH	SA_OutputVideoLinearized_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if linearized video signal is transmitted.	
Parameter: int SA_OutputVideoLightSpectrum_ETH	SA_OutputVideoLightSpectrum_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if light referenced signal is transmitted.	
15.1.3.2.3 Set_OutputColor_ETH (OUTCOLOR_ETH)	
Set color data to be output at ETH interface.	
Parameter: int SP_OutputColorXYZ_ETH	SP_OutputColorXYZ_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in XYZ is transmitted.	
Parameter: int SP_OutputColorRGB_ETH	SP_OutputColorRGB_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in RGB is transmitted.	
Parameter: int SP_OutputColorLAB_ETH	SP_OutputColorLAB_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in L*a*b* is transmitted.	
Parameter: int SP_OutputColorLUV_ETH	SP_OutputColorLUV_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in L*u*v* is transmitted.	
Parameter: int SP_OutputColorLCH_ETH	SP_OutputColorLCH_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in L*c*h* is transmitted.	

Parameter: int SP_OutputColorLAB99_ETH SP_OutputColorLAB99_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if measured data in L*a*b*99 is transmitted.

Parameter: int SP_OutputColorLCH99_ETH SP_OutputColorLCH99_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if measured data in L*c*h*99 is transmitted.

15.1.3.2.4 Get_OutputColor_ETH (OUTCOLOR_ETH)

Get color data to be output at ETH interface.

Parameter: int SA_OutputColorXYZ_ETH SA_OutputColorXYZ_ETH

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if measured data in XYZ is transmitted.

Parameter: int SA_OutputColorRGB_ETH SA_OutputColorRGB_ETH

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if measured data in RGB is transmitted.

Parameter: int SA_OutputColorLAB_ETH SA_OutputColorLAB_ETH

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if measured data in L*a*b* is transmitted.

Parameter: int SA_OutputColorLUV_ETH SA_OutputColorLUV_ETH

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if measured data in L*u*v* is transmitted.

Parameter: int SA_OutputColorLCH_ETH SA_OutputColorLCH_ETH

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if measured data in L*c*h* is transmitted.

Parameter: int SA_OutputColorLAB99_ETH SA_OutputColorLAB99_ETH

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if measured data in L*a*b*99 is transmitted.

Parameter: int SA_OutputColorLCH99_ETH SA_OutputColorLCH99_ETH

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if measured data in L*c*h*99 is transmitted.

15.1.3.2.5 Set_OutputColor_RS422 (OUTCOLOR_RS422)

Set color data to be output at RS422 interface.

Parameter: int SP_OutputColorXYZ_RS422 SP_OutputColorXYZ_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if measured data in XYZ is transmitted.

Parameter: int SP_OutputColorRGB_RS422 SP_OutputColorRGB_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if measured data in RGB is transmitted.

Parameter: int SP_OutputColorLAB_RS422 SP_OutputColorLAB_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if measured data in L*a*b* is transmitted.

Parameter: int SP_OutputColorLUV_RS422 SP_OutputColorLUV_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if measured data in L*u*v* is transmitted.

Parameter: int SP_OutputColorLCH_RS422 SP_OutputColorLCH_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if measured data in L*c*h* is transmitted.

Parameter: int SP_OutputColorLAB99_RS422 SP_OutputColorLAB99_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if measured data in L*a*b*99 is transmitted.

Parameter: int SP_OutputColorLCH99_RS422 SP_OutputColorLCH99_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if measured data in L*c*h*99 is transmitted.

15.1.3.2.6 Get_OutputColor_RS422 (OUTCOLOR_RS422)

Get color data to be output at RS422 interface.

Parameter: int SA_OutputColorXYZ_RS422 SA_OutputColorXYZ_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if measured data in XYZ is transmitted.

Parameter: int SA_OutputColorRGB_RS422 SA_OutputColorRGB_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if measured data in RGB is transmitted.

Parameter: int SA_OutputColorLAB_RS422 SA_OutputColorLAB_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if measured data in L*a*b* is transmitted.

Parameter: int SA_OutputColorLUV_RS422 SA_OutputColorLUV_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if measured data in L*u*v* is transmitted.

Parameter: int SA_OutputColorLCH_RS422 SA_OutputColorLCH_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if measured data in L*c*h* is transmitted.

15.1. Commands for ACS7000

Parameter: int SA_OutputColorLAB99_RS422	SA_OutputColorLAB99_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in L*a*b*99 is transmitted.	
Parameter: int SA_OutputColorLCH99_RS422	SA_OutputColorLCH99_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if measured data in L*c*h*99 is transmitted.	

15.1.3.2.7 Set_DistanceMode (DISTANCEMODE)

Set the distance mode.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_DistanceMode	SP_DistanceMode
Direction: Down	
Valid values:	
0= Best hit (BESTHIT)	
1= Selection	
Description: Distance mode.	

15.1.3.2.8 Get_DistanceMode (DISTANCEMODE)

Get the distance mode.

Parameter: int SA_DistanceMode	SA_DistanceMode
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Best hit (BESTHIT)	
1= Selection	
Description: Distance mode.	

15.1.3.2.9 Set_OutputDistance_ETH (OUTDIST_ETH)

Set the color distance data to be output at ethernet interface.

Parameter: int SP_OutputDistDistance01_ETH	SP_OutputDistDistance01_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 1 is transmitted.	

Parameter: int SP_OutputDistDistance02_ETH	SP_OutputDistDistance02_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 2 is transmitted.	
Parameter: int SP_OutputDistDistance03_ETH	SP_OutputDistDistance03_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 3 is transmitted.	
Parameter: int SP_OutputDistDistance04_ETH	SP_OutputDistDistance04_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 4 is transmitted.	
Parameter: int SP_OutputDistDistance05_ETH	SP_OutputDistDistance05_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 5 is transmitted.	
Parameter: int SP_OutputDistDistance06_ETH	SP_OutputDistDistance06_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 6 is transmitted.	
Parameter: int SP_OutputDistDistance07_ETH	SP_OutputDistDistance07_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 7 is transmitted.	
Parameter: int SP_OutputDistDistance08_ETH	SP_OutputDistDistance08_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 8 is transmitted.	
Parameter: int SP_OutputDistDistance09_ETH	SP_OutputDistDistance09_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 9 is transmitted.	

Parameter: int SP_OutputDistDistance10_ETH	SP_OutputDistDistance10_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 10 is transmitted.	
Parameter: int SP_OutputDistDistance11_ETH	SP_OutputDistDistance11_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 11 is transmitted.	
Parameter: int SP_OutputDistDistance12_ETH	SP_OutputDistDistance12_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 12 is transmitted.	
Parameter: int SP_OutputDistDistance13_ETH	SP_OutputDistDistance13_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 13 is transmitted.	
Parameter: int SP_OutputDistDistance14_ETH	SP_OutputDistDistance14_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 14 is transmitted.	
Parameter: int SP_OutputDistDistance15_ETH	SP_OutputDistDistance15_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 15 is transmitted.	
Parameter: int SP_OutputDistDistance16_ETH	SP_OutputDistDistance16_- ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 16 is transmitted.	
Parameter: int SP_OutputDistMinDistance_ETH	SP_OutputDistMinDis- tance_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if minimum color distance is transmitted.	

Parameter: int SP_OutputDistDetectedID_ETH
Direction: Down
Valid values:
 0= no
 1= yes
Description: Specify if no. of detected color is transmitted.

Parameter: int SP_OutputDistMinDistID_ETH
Direction: Down
Valid values:
 0= no
 1= yes
Description: Specify if no. of nearest color is transmitted.

15.1.3.2.10 Get_OutputDistance_ETH (OUTDIST_ETH)

Get the color distance data to be output at ethernet interface.

Parameter: int SA_OutputDistDistance01_ETH	SA_OutputDistDistance01_-
Direction: Up	ETH
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 1 is transmitted.	
Parameter: int SA_OutputDistDistance02_ETH	SA_OutputDistDistance02_-
Direction: Up	ETH
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 2 is transmitted.	
Parameter: int SA_OutputDistDistance03_ETH	SA_OutputDistDistance03_-
Direction: Up	ETH
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 3 is transmitted.	
Parameter: int SA_OutputDistDistance04_ETH	SA_OutputDistDistance04_-
Direction: Up	ETH
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 4 is transmitted.	
Parameter: int SA_OutputDistDistance05_ETH	SA_OutputDistDistance05_-
Direction: Up	ETH
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 5 is transmitted.	

Parameter: int SA_OutputDistDistance06_ETH	SA_OutputDistDistance06_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 6 is transmitted.	
Parameter: int SA_OutputDistDistance07_ETH	SA_OutputDistDistance07_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 7 is transmitted.	
Parameter: int SA_OutputDistDistance08_ETH	SA_OutputDistDistance08_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 8 is transmitted.	
Parameter: int SA_OutputDistDistance09_ETH	SA_OutputDistDistance09_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 9 is transmitted.	
Parameter: int SA_OutputDistDistance10_ETH	SA_OutputDistDistance10_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 10 is transmitted.	
Parameter: int SA_OutputDistDistance11_ETH	SA_OutputDistDistance11_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 11 is transmitted.	
Parameter: int SA_OutputDistDistance12_ETH	SA_OutputDistDistance12_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 12 is transmitted.	
Parameter: int SA_OutputDistDistance13_ETH	SA_OutputDistDistance13_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 13 is transmitted.	

Parameter: int SA_OutputDistDistance14_ETH	SA_OutputDistDistance14_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 14 is transmitted.	
Parameter: int SA_OutputDistDistance15_ETH	SA_OutputDistDistance15_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 15 is transmitted.	
Parameter: int SA_OutputDistDistance16_ETH	SA_OutputDistDistance16_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if color distance 16 is transmitted.	
Parameter: int SA_OutputDistMinDistance_ETH	SA_OutputDistMinDis- tance_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if minimum color distance is transmitted.	
Parameter: int SA_OutputDistDetectedID_ETH	SA_OutputDistDetectedID_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if no. of detected color is transmitted.	
Parameter: int SA_OutputDistMinDistID_ETH	SA_OutputDistMinDistID_- ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if no. of nearest color is transmitted.	

15.1.3.2.11 Set_OutputDistance_RS422 (OUTDIST_RS422)

Set the color distance data to be output at RS422 interface.

Parameter: int SP_OutputDistMinDistance_RS422	SP_OutputDistMinDis- tance_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if minimum color distance is transmitted.	

15.1. Commands for ACS7000

Parameter: int SP_OutputDistDetectedID_RS422	SP_OutputDistDetectedID_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if no. of detected color is transmitted.	
Parameter: int SP_OutputDistMinDistID_RS422	SP_OutputDistMinDistID_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if no. of nearest color is transmitted.	

15.1.3.2.12 Get_OutputDistance_RS422 (OUTDIST_RS422)

Get the color distance data to be output at RS422 interface.

Parameter: int SA_OutputDistMinDistance_RS422	SA_OutputDistMinDis- tance_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if minimum color distance is transmitted.	
Parameter: int SA_OutputDistDetectedID_RS422	SA_OutputDistDetectedID_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if no. of detected color is transmitted.	
Parameter: int SA_OutputDistMinDistID_RS422	SA_OutputDistMinDistID_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if no. of nearest color is transmitted.	

15.1.3.2.13 Set_OutputStatistic_ETH (OUTSTATISTIC_ETH)

Set statistic data to be output at ETH interface.

Parameter: int SP_OutputStatisticMin_ETH	SP_OutputStatisticMin_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic minimum value is transmitted.	

15.1. Commands for ACS7000

Parameter: int SP_OutputStatisticMax_ETH	SP_OutputStatisticMax_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic maximum value is transmitted.	
Parameter: int SP_OutputStatisticPeak2Peak_ETH	SP_OutputStatisticPeak2Peak_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic peak to peak value is transmitted.	

15.1.3.2.14 Get_OutputStatistic_ETH (OUTSTATISTIC_ETH)

Get statistic data to be output at ETH interface.

Parameter: int SA_OutputStatisticMin_ETH	SA_OutputStatisticMin_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic minimum value is transmitted.	

Parameter: int SA_OutputStatisticMax_ETH	SA_OutputStatisticMax_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic maximum value is transmitted.	

Parameter: int SA_OutputStatisticPeak2Peak_ETH	SA_OutputStatisticPeak2Peak_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic peak to peak value is transmitted.	

15.1.3.2.15 Set_OutputStatistic_RS422 (OUTSTATISTIC_RS422)

Set statistic data to be output at RS422 interface.

Parameter: int SP_OutputStatisticMin_RS422	SP_OutputStatisticMin_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic minimum value is transmitted.	

15.1. Commands for ACS7000

Parameter: int SP_OutputStatisticMax_RS422	SP_OutputStatisticMax_- RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic maximum value is transmitted.	
Parameter: int SP_OutputStatisticPeak2Peak_RS422	SP_OutputStatistic- Peak2Peak_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic peak to peak value is transmitted.	

15.1.3.2.16 Get_OutputStatistic_RS422 (OUTSTATISTIC_RS422)

Get statistic data to be output at RS422 interface.

Parameter: int SA_OutputStatisticMin_RS422	SA_OutputStatisticMin_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic minimum value is transmitted.	
Parameter: int SA_OutputStatisticMax_RS422	SA_OutputStatisticMax_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic maximum value is transmitted.	
Parameter: int SA_OutputStatisticPeak2Peak_RS422	SA_OutputStatistic- Peak2Peak_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if statistic peak to peak value is transmitted.	

15.1.3.2.17 Set_DigitalOutDistance (OUTDIST_COLOROUT)

Set the color distance data at digital outputs.

Parameter: int SP_DigitalOutDetectedID	SP_DigitalOutDetectedID
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if no. of detected color is output.	

15.1.3.2.18 Get_DigitalOutDistance (OUTDIST_COLOROUT)

Get the color distance data at digital outputs.

Parameter: int SA_DigitalOutDetectedID

SA_DigitalOutDetectedID

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if no. of detected color is output.

15.1.3.2.19 Set_OutputStatus_ETH (OUTSTATUS_ETH)

Set status data to be output at ETH interface.

Parameter: int SP_OutputStatusFramerate_ETH

SP_OutputStatusFramerate_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if framerate is transmitted.

Parameter: int SP_OutputStatusShutter_ETH

SP_OutputStatusShutter_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if shutter time is transmitted.

Parameter: int SP_OutputStatusTempDetector_ETH

SP_OutputStatusTempDetector_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if temperature of detector is transmitted.

Parameter: int SP_OutputStatusTempLightSrc_ETH

SP_OutputStatusTempLightSrc_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if temperature of light source is transmitted.

Parameter: int SP_OutputLightSensorRed_ETH

SP_OutputLightSensorRed_ETH

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if red part of light sensor is transmitted.

Parameter: int SP_OutputLightSensorGreen_ETH	SP_OutputLightSensor-Green_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if green part of light sensor is transmitted.	
Parameter: int SP_OutputLightSensorBlue_ETH	SP_OutputLightSensor-Blue_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if blue part of light sensor is transmitted.	
Parameter: int SP_OutputLightSensorBright_ETH	SP_OutputLightSensor-Bright_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if brightness of light sensor is transmitted.	
Parameter: int SP_OutputStatusCounter_ETH	SP_OutputStatusCounter-ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SP_OutputStatusTimestamp_ETH	SP_OutputStatusTimestamp_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SP_OutputStatusError_ETH	SP_OutputStatusError_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if error flags are transmitted.	

15.1.3.2.20 Get_OutputStatus_ETH (OUTSTATUS_ETH)

Get status data to be output at ETH interface.

Parameter: int SA_OutputStatusFramerate_ETH	SA_OutputStatusFramerate_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if framerate is transmitted.	

Parameter: int SA_OutputStatusShutter_ETH	SA_OutputStatusShutter_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	
Parameter: int SA_OutputStatusTempDetector_ETH	SA_OutputStatusTempDetector_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if temperature of detector is transmitted.	
Parameter: int SA_OutputStatusTempLightSrc_ETH	SA_OutputStatusTempLight-Src_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if temperature of light source is transmitted.	
Parameter: int SA_OutputLightSensorRed_ETH	SA_OutputLightSensorRed_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if red part of light sensor is transmitted.	
Parameter: int SA_OutputLightSensorGreen_ETH	SA_OutputLightSensor-Green_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if green part of light sensor is transmitted.	
Parameter: int SA_OutputLightSensorBlue_ETH	SA_OutputLightSensor-Blue_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if blue part of light sensor is transmitted.	
Parameter: int SA_OutputLightSensorBright_ETH	SA_OutputLightSensor-Bright_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if brightness of light sensor is transmitted.	
Parameter: int SA_OutputStatusCounter_ETH	SA_OutputStatusCounter_-ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	

Parameter: int SA_OutputStatusTimestamp_ETH	SA_OutputStatusTimestamp_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SA_OutputStatusError_ETH	SA_OutputStatusError_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if error flags are transmitted.	

15.1.3.2.21 Set_OutputStatus_RS422 (OUTSTATUS_RS422)

Set status data to be output at RS422 interface.

Parameter: int SP_OutputStatusFramerate_RS422	SP_OutputStatusFramerate_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if framerate is transmitted.	
Parameter: int SP_OutputStatusShutter_RS422	SP_OutputStatusShutter_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	
Parameter: int SP_OutputStatusTempDetector_RS422	SP_OutputStatusTempDetector_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if temperature of detector is transmitted.	
Parameter: int SP_OutputStatusTempLightSrc_RS422	SP_OutputStatusTempLightSrc_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if temperature of light source is transmitted.	
Parameter: int SP_OutputLightSensorRed_RS422	SP_OutputLightSensorRed_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if red part of light sensor is transmitted.	

Parameter: int SP_OutputLightSensorGreen_RS422	SP_OutputLightSensor-Green_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if green part of light sensor is transmitted.	
Parameter: int SP_OutputLightSensorBlue_RS422	SP_OutputLightSensor-Blue_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if blue part of light sensor is transmitted.	
Parameter: int SP_OutputLightSensorBright_RS422	SP_OutputLightSensor-Bright_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if brightness of light sensor is transmitted.	
Parameter: int SP_OutputStatusCounter_RS422	SP_OutputStatusCounter_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	
Parameter: int SP_OutputStatusTimestamp_RS422	SP_OutputStatusTimestamp_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if timestamp is transmitted.	
Parameter: int SP_OutputStatusError_RS422	SP_OutputStatusError_RS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if error flags are transmitted.	

15.1.3.2.22 Get_OutputStatus_RS422 (OUTSTATUS_RS422)

Get status data to be output at RS422 interface.

Parameter: int SA_OutputStatusFramerate_RS422	SA_OutputStatusFramerate_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if framerate is transmitted.	

Parameter: int SA_OutputStatusShutter_RS422	SA_OutputStatusShutter_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if shutter time is transmitted.	
Parameter: int SA_OutputStatusTempDetector_RS422	SA_OutputStatusTempDetec- tor_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if temperature of detector is transmitted.	
Parameter: int SA_OutputStatusTempLightSrc_RS422	SA_OutputStatusTempLight- Src_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if temperature of light source is transmitted.	
Parameter: int SA_OutputLightSensorRed_RS422	SA_OutputLightSensorRed_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if red part of light sensor is transmitted.	
Parameter: int SA_OutputLightSensorGreen_RS422	SA_OutputLightSensor- Green_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if green part of light sensor is transmitted.	
Parameter: int SA_OutputLightSensorBlue_RS422	SA_OutputLightSensor- Blue_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if blue part of light sensor is transmitted.	
Parameter: int SA_OutputLightSensorBright_RS422	SA_OutputLightSensor- Bright_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if brightness of light sensor is transmitted.	
Parameter: int SA_OutputStatusCounter_RS422	SA_OutputStatusCounter_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if counter is transmitted.	

Parameter: int SA_OutputStatusTimestamp_RS422	SA_OutputStatusTimestamp_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description:	Specify if timestamp is transmitted.
Parameter: int SA_OutputStatusError_RS422	SA_OutputStatusError_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description:	Specify if error flags are transmitted.

16 Commands for Interfaces

16.1 Commands for ETH_IF1032

See interface manual for detailed description of interface commands.

This interface supports following interfaces:

- [TCP/IP](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_Status](#), [Get_RawDataRanges](#) and [Get_ChannelInfos](#) after [OpenSensor](#). Otherwise, you have to call it manually to allow MEDAQLib to calculate datarate and to assign values.

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from interface, either from -2147483648 (INT_MIN) to 2147483647 (INT_MAX) or from 0 to 4294967295 (UINT_MAX) or from -3.402823466e+38 (-FLT_MAX) to 3.402823466e+38 (FLT_MAX).
- Scaled values are scaled using sensor range (if known by MEDAQLib, use interface command [Get_RawDataRanges](#) and [Get_ChannelInfos](#)).

The values of selected channels are filled in the arrays one after another. Each array always starts with first selected channel.

16.1.1 User Level

16.1.1.1 Logout (LGO)

Change user level to user at web interface.

16.1.1.2 Login (LGI)

Change user level to setup at web interface.

Parameter: String SP_Password

SP_Password

Direction: Down

Description: Valid password to login.

16.1.1.3 Set_Password (PWD)

Change the password for login.

Parameter: String SP_OldPassword

SP_OldPassword

Direction: Down

Description: Old password.

Parameter: String SP_NewPassword SP_NewPassword
Direction: Down
Description: New password.

16.1.2 Measurement

16.1.2.1 Set_SampleTime (STI)

Set the sample time for data acquisition.

Parameter: double SP_SampleTime SP_SampleTime
Direction: Down
Valid values:
 Minimum: 0.0
 Maximum: 999999999.9999988
Unit: μ s
Description: Desired sample time

Parameter: double SA_SampleTime SA_SampleTime
Direction: Up
Valid values:
 Minimum: 0.0
 Maximum: 1000000000.0
Unit: μ s
Description: Real sample time at interface

16.1.2.2 Get_SampleTime (STI?)

Get the current sample time.

Parameter: double SA_SampleTime SA_SampleTime
Direction: Up
Valid values:
 Minimum: 0.0
 Maximum: 1000000000.0
Unit: μ s
Description: Real sample time at interface

16.1.2.3 Set_Trigger (TRG)

Activate/disable the trigger at interface.

Parameter: int SP_TrMode SP_TrMode
Direction: Down
Valid values:
 0= Off
 1= Rising edge
 2= High level
 3= Gate at rising edge
Description: Trigger active/disabled.

16.1.2.4 Get_Trigger (TRG?)

Retrieve the trigger mode at interface.

Parameter: int SA_TrgMode SA_TrgMode
Direction: Up
Valid values:
 0= Off
 1= Rising edge
 2= High level
 3= Gate at rising edge
Description: Trigger active/disabled.

16.1.2.5 Set_AvrType (AVT)

Set the averaging type at interface.

Parameter: int SP_AvrType SP_AvrType
Direction: Down
Valid values:
 0= off
 1= Moving average
 2= Mean (arithmetic)
 3= Median
 4= Dynamic noise rejection
Description: Averaging type.

16.1.2.6 Get_AvrType (AVT?)

Retrieve the averaging type at interface.

Parameter: int SA_AvrType SA_AvrType
Direction: Up
Valid values:
 0= off
 1= Moving average
 2= Mean (arithmetic)
 3= Median
 4= Dynamic noise rejection
Description: Averaging type at interface.

16.1.2.7 Set_AvrNbr (AVN)

Set the averaging number at interface.

Parameter: int SP_AvrNbr SP_AvrNbr
Direction: Down
Valid values:
Minimum: 2
Maximum: 8
Description: Averaging number.

16.1.2.8 Get_AvrNbr (AVN?)

Retrieve the averaging number at interface.

Parameter: int SA_AvrNbr SA_AvrNbr
Direction: Up
Valid values:
Minimum: 2
Maximum: 8
Description: Averaging number at interface.

16.1.3 Data output

16.1.3.1 ChannelStatus (CHS)

Retrieve the available channels at interface.

Parameter: int SA_ChExist1..32 SA_ChExist1..32
Direction: Up
Valid values:
 0= Channel not available
 1= Measured channel
Description: Channel 1 to 32 is available at interface.

16.1.3.2 Get_RawDataRange (MDF)

Read the raw data range for a channel from interface. The range is used by MEDAQLib for scaling raw data.

Parameter: int SP_Chан SP_Chан
Direction: Down
Valid values:
Minimum: 1
Maximum: 32
Description: Channel to read the raw data range for.

Parameter: double SA_RawRangeMin SA_RawRangeMin
Direction: Up
Valid values:
Minimum: -2147483648.0 (INT_MIN)
Maximum: 2147483647.0 (INT_MAX)
Description: Minimum raw data range of channel.

Parameter: double SA_RawRangeMax SA_RawRangeMax
Direction: Up
Valid values:
Minimum: -2147483648.0 (INT_MIN)
Maximum: 2147483647.0 (INT_MAX)
Description: Maximum raw data range of channel.

16.1.3.3 Get_RawDataRanges

Calls the sensor command Get_RawDataRange for any requested channel.

Parameter: int SP_Complete

SP_Complete

Direction: Down

Valid values:

0 = FALSE

1 = TRUE

Description: Specifies if any possible channel should be requested or only known channels (from former call to [ChannelStatus](#) or [Get_Status](#)).

Parameter: double SA_RawRangeMin1..32

SA_RawRangeMin1..32

Direction: Up

Valid values:

Minimum: -2147483648.0 (INT_MIN)

Maximum: 2147483647.0 (INT_MAX)

Description: Minimum raw data range of channel 1 to 32.

Parameter: double SA_RawRangeMax1..32

SA_RawRangeMax1..32

Direction: Up

Valid values:

Minimum: -2147483648.0 (INT_MIN)

Maximum: 2147483647.0 (INT_MAX)

Description: Maximum raw data range of channel 1 to 32.

16.1.4 Interfaces

16.1.4.1 Set_DataPort (SDP)

Set the TCP/IP data port at interface.

Parameter: int SP_DataPort

SP_DataPort

Direction: Down

Valid values:

Minimum: 1024

Maximum: 65535

Description: TCP/IP data port at interface.

16.1.4.2 Get_DataPort (GDP)

Retrieve the TCP/IP data port from interface.

Parameter: int SA_DataPort

SA_DataPort

Direction: Up

Valid values:

Minimum: 1024

Maximum: 65535

Description: TCP/IP data port at interface.

16.1.4.3 Set_IPConfiguration (IPS)

Set the IP configuration at interface.

Parameter: int SP_DHCPEnabled SP_DHCPEnabled

Direction: Down

Valid values:

0= FALSE

1= TRUE

Description: Specify if interface should use a static IP address or ask for IP at DHCP server (dynamic IP address).

Parameter: String SP_Address SP_Address

Direction: Down

Valid values:

Valid IP address in form of xxx.xxx.xxx.xxx

Description: IP address of the interface. This parameter is only evaluated on static IP assignment.

Parameter: String SP_SubnetMask SP_SubnetMask

Direction: Down

Valid values:

Valid network mask (e.g. 255.255.255.0 for a Class C network)

Description: Network mask of the interface. This parameter is only evaluated on static IP assignment.

Parameter: String SP_Gateway SP_Gateway

Direction: Down

Valid values:

Valid IP address of default gateway in form of xxx.xxx.xxx.xxx

Description: The default gateway must be specified if the interface should communicate with peers in foreign subnets. This parameter is only evaluated on static IP assignment.

16.1.4.4 Get_IPConfiguration (IPS?)

Get the IP configuration at interface.

Parameter: int SA_DHCPEnabled SA_DHCPEnabled

Direction: Up

Valid values:

0= FALSE

1= TRUE

Description: Get settings if interface should use a static IP address ask for IP at DHCP server (dynamic IP address).

Parameter: String SA_Address SA_Address

Direction: Up

Valid values:

Valid IP address in form of xxx.xxx.xxx.xxx

Description: IP address of the interface. If DHCP is enabled it returns the currently assigned IP address.

16.1. Commands for ETH_IF1032

Parameter: String SA_SubnetMask	SA_SubnetMask
Direction: Up	
Valid values:	
Valid network mask (e.g. 255.255.255.0 for a Class C network)	
Description:	Network mask of the interface. If DHCP is enabled it returns the currently assigned network mask.
Parameter: String SA_Gateway	SA_Gateway
Direction: Up	
Valid values:	
Valid IP address of default gateway in form of xxx.xxx.xxx.xxx	
Description:	Address of the default gateway. If DHCP is enabled it returns the currently assigned default gateway.

16.1.4.5 Set_EthernetMode (IFC)

Switches ethernet mode between Ethernet and Ethercat. The interface must be rebooted to apply this setting.

Parameter: int SP_EthernetMode	SP_EthernetMode
Direction: Down	
Valid values:	
0= Ethernet	
1= Ethercat	
Description:	Ethernet mode.

16.1.4.6 Get_EthernetMode (IFC?)

Get ethernet mode of interface.

Parameter: int SA_EthernetMode	SA_EthernetMode
Direction: Up	
Valid values:	
0= Ethernet	
1= Ethercat	
Description:	Ethernet mode.

16.1.4.7 Set_AppLangauge (LNG)

Set language of web interface at interface.

Parameter: int SP_ApplicationLanguage	SP_ApplicationLanguage
Direction: Down	
Valid values:	
0= System	
1= English	
2= German	
Description:	Language of web interface.

16.1.4.8 Get_AppLanguage (LNG?)

Get language of web interface from interface.

Parameter: int SA_ApplicationLanguage

SA_ApplicationLanguage

Direction: Up

Valid values:

0= System

1= English

2= German

Description: Language of web interface.

16.1.5 Sensor interface

16.1.5.1 Set_SensorInterface (SIF)

Selects the sensor interface of the module.

Parameter: int SP_SensorInterface

SP_SensorInterface

Direction: Down

Valid values:

0= Analog

1= not assigned currently

2= RS232 (not available in EthIF1032)

3= RS485

Description: Sensor interface of the module

16.1.5.2 Get_SensorInterface (SIF?)

Get the sensor interface of the module.

Parameter: int SA_SensorInterface

SA_SensorInterface

Direction: Up

Valid values:

0= Analog

1= not assigned currently

2= RS232 (not available in the IF1032/ETH)

3= RS485

Description: Sensor interface of the module

16.1.5.3 Set_SensorBaudrate (SBR)

Changes the Baudrate of the RS485 interface.

Parameter: int SP_SensorBaudrate

SP_SensorBaudrate

Direction: Down

Valid values:

Minimum: 9200

Maximum: 6250000

Description: Baudrate of the RS485 interface

16.1. Commands for ETH_IF1032

Parameter: int SA_SensorBaudrate SA_SensorBaudrate
Direction: Up
Valid values:
Minimum: 9200
Maximum: 6250000
Description: Real baudrate of the RS485 interface

16.1.5.4 Get_SensorBaudrate (SBR?)

Get the Baudrate of the RS485 interface.

Parameter: int SA_SensorBaudrate SA_SensorBaudrate
Direction: Up
Valid values:
Minimum: 9200
Maximum: 6250000
Description: Baudrate of the RS485 interface

16.1.5.5 Set_SensorAddress (SAD)

Changes the address used to communicate with the RS485.

Parameter: int SP_DeviceInstance SP_DeviceInstance
Direction: Down
Valid values:
Minimum: 1
Maximum: 1
Description: Device instance of the RS485 device

Parameter: int SP_SensorAddress SP_SensorAddress
Direction: Down
Valid values:
Minimum: 2
Maximum: 126
Description: Address used to communicate with the RS485

16.1.5.6 Get_SensorAddress (SAD?)

Get the address used to communicate with the RS485.

Parameter: int SP_DeviceInstance SP_DeviceInstance
Direction: Down
Valid values:
Minimum: 1
Maximum: 1
Description: Device instance of the RS485 device

Parameter: int SA_SensorAddress SA_SensorAddress
Direction: Up
Valid values:
Minimum: 2
Maximum: 126
Description: Address used to communicate with the RS485

16.1.5.7 Set_AnalogRange (ARA)

Set the measuring range of the analog inputs. This changes only the scaling, not the actual input range.

Parameter: int SP_AnalogChannel SP_AnalogChannel

Direction: Down

Valid values:

Minimum: 1

Maximum: 4

Description: Analog channel number

Parameter: double SP_AnalogRange SP_AnalogRange

Direction: Down

Valid values:

Minimum: -999999999.9999988

Maximum: 999999999.9999988

Description: Analog range value

16.1.5.8 Get_AnalogRange (ARA?)

Get the measuring range of the analog inputs.

Parameter: int SP_AnalogChannel SP_AnalogChannel

Direction: Down

Valid values:

Minimum: 1

Maximum: 4

Description: Analog channel number

Parameter: double SA_AnalogRange SA_AnalogRange

Direction: Up

Valid values:

Minimum: -1000000000.0

Maximum: 1000000000.0

Description: Analog range value

16.1.5.9 Set_AnalogOffset (AOF)

Set the measuring range offset of the analog inputs. This only changes the scaling, not the actual input range.

Parameter: int SP_AnalogChannel SP_AnalogChannel

Direction: Down

Valid values:

Minimum: 1

Maximum: 4

Description: Analog channel number

16.1. Commands for ETH_IF1032

Parameter: double SP_AnalogOffset SP_AnalogOffset
Direction: Down
Valid values:
Minimum: -999999999.9999988
Maximum: 999999999.9999988
Description: Analog offset value

16.1.5.10 Get_AnalogOffset (AOF?)

Get the measuring range offset of the analog inputs.

Parameter: int SP_AnalogChannel SP_AnalogChannel
Direction: Down
Valid values:
Minimum: 1
Maximum: 4
Description: Analog channel number

Parameter: double SA_AnalogOffset SA_AnalogOffset
Direction: Up
Valid values:
Minimum: -1000000000.0
Maximum: 1000000000.0
Description: Analog offset value

16.1.5.11 Set_AnalogUnit (AUN)

Set the unit of the analog inputs. This only changes the scaling, not the actual input range.

Parameter: int SP_AnalogChannel SP_AnalogChannel
Direction: Down
Valid values:
Minimum: 1
Maximum: 4
Description: Analog channel number

Parameter: int SP_AnalogUnit SP_AnalogUnit
Direction: Down
Valid values:
0= m
1= mm
2= μ m
3= V
4= digit
5= mA
6= °C
7= %
8= Hz
9= m/s
10= m/s^2
11= g (gravity)

16.1. Commands for ETH_IF1032

12= °
 13= °/s
 14= rad
 15= rad/s
 16= -
 17= °F
 18= inch
 19= mil
 20= s
 21= ms
 22= μ s

Description: Analog unit

16.1.5.12 Get_AnalogUnit (AUN?)

Get the unit of the analog inputs.

Parameter: int SP_AnalogChannel

SP_AnalogChannel

Direction: Down

Valid values:

Minimum: 1
 Maximum: 4

Description: Analog channel number

Parameter: int SA_AnalogUnit

SA_AnalogUnit

Direction: Up

Valid values:

0= m
 1= mm
 2= μ m
 3= V
 4= digit
 5= mA
 6= °C
 7= %
 8= Hz
 9= m/s
 10= m/s^2
 11= g (gravity)
 12= °
 13= °/s
 14= rad
 15= rad/s
 16= -
 17= °F
 18= inch
 19= mil
 20= s
 21= ms
 22= μ s

Description: Analog unit

16.1.5.13 Set_AnalogMathFunction (AMF)

Set mathematic function for analog inputs at controller.

The result of the math function is transmitted like normal sensor data at a channel 4.

Parameter: int SP_AnalogMathFunctionEnable

SP_AnalogMathFunctionEnable

Direction: Down

Valid values:

0= Disabled

1= Enabled

Description: Specify if analog math function is enabled.

Parameter: double SP_AnalogOffset

SP_AnalogOffset

Direction: Down

Valid values:

Minimum: -999999999.9999998

Maximum: 999999999.9999998

Description: Offset to be added to result.

Parameter: double SP_AnalogFactor1..3

SP_AnalogFactor1..3

Direction: Down

Valid values:

Minimum: -999999999.9999998

Maximum: 999999999.9999998

Description: Multiplication factor for analog channel 1 to 3.

16.1.5.14 Get_AnalogMathFunction (AMF?)

Get mathematic function for analog inputs at controller.

Parameter: int SA_AnalogMathFunctionEnable

SA_AnalogMathFunctionEnable

Direction: Up

Valid values:

0= Disabled

1= Enabled

Description: Specify if analog math function is enabled.

Parameter: double SA_AnalogOffset

SA_AnalogOffset

Direction: Up

Valid values:

Minimum: -1000000000.0

Maximum: 1000000000.0

Description: Offset to be added to result.

Parameter: double SA_AnalogFactor1..3

SA_AnalogFactor1..3

Direction: Up

Valid values:

Minimum: -1000000000.0

Maximum: 1000000000.0

Description: Multiplication factor for analog channel 1 to 3.

16.1.6 Information

16.1.6.1 Get_Status (STS)

Retrieve detailed information about the interface.

Parameter: double SA_SampleTime	SA_SampleTime
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 1000000000.0	
Unit: μ s	
Description: Real sample time at interface	
Parameter: int SA_AvrType	SA_AvrType
Direction: Up	
Valid values:	
0= off	
1= Moving average	
2= Mean (arithmetic)	
3= Median	
4= Dynamic noise rejection	
Description: Averaging type at interface.	
Parameter: int SA_AvrNbr	SA_AvrNbr
Direction: Up	
Valid values:	
Minimum: 2	
Maximum: 8	
Description: Averaging number at interface.	
Parameter: int SA_ChExist1..32	SA_ChExist1..32
Direction: Up	
Valid values:	
0= Channel not available	
1= Measured channel	
Description: Channel 1 to 32 is available at interface.	
Parameter: int SA_TrgMode	SA_TrgMode
Direction: Up	
Valid values:	
0= Off	
1= Rising edge	
2= High level	
3= Gate at rising edge	
Description: Trigger active/disabled.	

16.1.6.2 Get_Version (VER)

Retrieve the sensor software version.

Parameter: String SA_Version	SA_Version
Direction: Up	
Description: Software version of the interface.	

16.1.6.3 Get_ChannelInfo (CHI)

Retrieve information about a sensor channel.

Parameter: int SP_Chан	SP_Chан
Direction: Down	
Valid values:	
Minimum: 1	
Maximum: 32	
Description: Channels to get information for.	
Parameter: String SA_ArticleNumber	SA_ArticleNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Article number of the sensor channel.	
Parameter: String SA_SensorName	SA_SensorName
Direction: Up	
Description: Name of the sensor channel.	
Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the sensor channel.	
Parameter: double SA_Offset	SA_Offset
Direction: Up	
Valid values:	
Minimum: -1000000000.0	
Maximum: 1000000000.0	
Unit: See parameter SA_Unit	
Description: Offset of the sensor channel.	
Parameter: double SA_Range	SA_Range
Direction: Up	
Valid values:	
Minimum: -1000000000.0	
Maximum: 1000000000.0	
Unit: See parameter SA_Unit	
Description: Range of sensor channel.	
Parameter: String SA_Unit	SA_Unit
Direction: Up	
Description: Unit of parameter SA_Offset and SA_Range, e.g. μm or mm.	

16.1.6.4 Get_ChannelInfos

Calls the sensor command Get_ChannelInfo for any requested channel.

Parameter: int SP_Complete	SP_Complete
Direction: Down	
Valid values:	
0 = FALSE	
1 = TRUE	
Description: Specifies if any possible channel should be requested or only known channels (from former call to ChannelStatus or Get_Status).	

16.1. Commands for ETH_IF1032

Parameter: String SA_ArticleNumber1..32	SA_ArticleNumber1..32
Direction: Up	
Valid values:	
Numeric value	
Description: Article number of the sensor channel 1 to 32.	
Parameter: String SA_SensorName1..32	SA_SensorName1..32
Direction: Up	
Description: Name of the sensor channel 1 to 32.	
Parameter: String SA_SerialNumber1..32	SA_SerialNumber1..32
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the sensor channel 1 to 32.	
Parameter: double SA_Offset1..32	SA_Offset1..32
Direction: Up	
Valid values:	
Minimum: -1000000000.0	
Maximum: 1000000000.0	
Unit: See parameter SA_Unit	
Description: Offset of the sensor channel 1 to 32.	
Parameter: double SA_Range1..32	SA_Range1..32
Direction: Up	
Valid values:	
Minimum: -1000000000.0	
Maximum: 1000000000.0	
Unit: See parameter SA_Unit	
Description: Range of sensor channel 1 to 32.	
Parameter: String SA_Unit1..32	SA_Unit1..32
Direction: Up	
Description: Unit of parameter SA_Offset and SA_Range, e.g. μm or mm.	

16.1.6.5 Get_ControllerInfo (COI)

Retrieve information about the interface.

Parameter: String SA_ArticleNumber	SA_ArticleNumber
Direction: Up	
Description: Article number of the interface.	
Parameter: String SA_ControllerName	SA_ControllerName
Direction: Up	
Description: Name of the interface.	
Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the interface.	

Parameter: String SA_Option SA_Option

Direction: Up

Valid values:

Numeric value

Description: Option of the interface.

Parameter: String SA_Softwareversion SA_Softwareversion

Direction: Up

Description: Software version of firmware in the interface.

16.1.7 Internal commands

16.1.7.1 Update_Firmware

Update firmware version at interface.

Attention! This function can take approx. 1 minute. This is an internal command. It should not be used by the customer.

Parameter: Binary data SP_FirmwareFile SP_FirmwareFile

Direction: Down

Valid values:

Firmware file as binary data.

Description: Firmware version, read from file.

Parameter: int SA_Result SA_Result

Direction: Up

Valid values:

0= Failed

1= Success

Description: Result of firmware update.

16.2 Commands for USBAdapter_IF2004

See USBAdapter_IF2004 manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [IF2004_USB](#) (native).

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from IF2004_USB adapter, from 0 to 255 (ExtTriggerIn and RxD).
- Scaled values are same as raw values.

16.2.1 Special commands

16.2.1.1 Set_SyncMaster

Set a synchronisation master for a specified sensor channel while interface is open.

Parameter: int SP_SensorChannel

SP_SensorChannel

Direction: Down

Valid values:

- 0= Sensor channel 1 (Connector 1/2)
- 1= Sensor channel 2 (Connector 1/2)
- 2= Sensor channel 3 (Connector 3/4)
- 3= Sensor channel 4 (Connector 3/4)

Description: Sensor channel to synchronize.

Parameter: int SP_SyncMasterChannel

SP_SyncMasterChannel

Direction: Down

Valid values:

- 1= No synchronisation master
- 0= Sensor channel 1 (Connector 1/2)
- 1= Sensor channel 2 (Connector 1/2)
- 2= Sensor channel 3 (Connector 3/4)
- 3= Sensor channel 4 (Connector 3/4)
- 4= Digital IN

Description: Channel number to synchronise with. See for further information.

16.2.1.2 Set_Algorithm

Set the data processing algorithm at comparator mode.

This command is only available at special option 1.

Parameter: int SP_Algorithm

SP_Algorithm

Direction: Down

Valid values:

- 0= Minimum value of selected sensor
- 1= Maximum value of selected sensor

Description: Data processing algorithm for comparator.

16.2.1.3 Get_Algorithm

Get the data processing algorithm at comparator mode.

This command is only available at special option 1.

Parameter: int SA_Algorithm

SA_Algorithm

Direction: Up

Valid values:

- 0= Minimum value of selected sensor
- 1= Maximum value of selected sensor

Description: Data processing algorithm for comparator.

16.2.1.4 Set_SelectedSensors

Set the selected sensors for comparator mode.

This command is only available at special option 1.

Parameter: int SP_SelectedSensor1..4

SP_SelectedSensor1..4

Direction: Down

Valid values:

- 1 = Leave unchanged
- 0 = Off
- 1 = On

Description: Selected sensors for comparator mode.

16.2.1.5 Get_SelectedSensors

Get the selected sensors for comparator mode.

This command is only available at special option 1.

Parameter: int SA_SelectedSensor1..4

SA_SelectedSensor1..4

Direction: Up

Valid values:

- 0 = Off
- 1 = On

Description: Selected sensors for comparator mode.

16.2.1.6 Set_SensorChannelBaudrate

Set the baudrate of a sensor channel.

This command is normally not needed, because sensor instance set the baudrate itself.

Parameter: int SP_SensorChannel

SP_SensorChannel

Direction: Down

Valid values:

- 0 = Sensor channel 1 (Connector 1/2)
- 1 = Sensor channel 2 (Connector 1/2)
- 2 = Sensor channel 3 (Connector 3/4)
- 3 = Sensor channel 4 (Connector 3/4)

Description: Sensor channel to set baudrate.

Parameter: int SP_SensorChannelBaudrate

SP_SensorChannelBaudrate

Direction: Down

Valid values:

- Minimum:** 733
- Maximum:** 8000000

Unit: Baud

Description: Desired baudrate of sensor channel.

Parameter: int SA_SensorChannelRealBaudrate
Direction: Up
Valid values:
 Minimum: 733
 Maximum: 8000000
Unit: Baud
Description: Real baudrate of sensor channel.

SA_SensorChannelRealBaudrate

16.2.1.7 Get_SensorChannelBaudrate

Get the baudrate of a sensor channel.

Parameter: int SA_SensorChannelBaudrate1..4
Direction: Up
Valid values:
 Minimum: 733
 Maximum: 8000000
Unit: Baud
Description: Nominal baudrate of sensor channel 1 to 4.

SA_SensorChannelBaudrate1..4

Parameter: int SA_SensorChannelRealBaudrate1..4
Direction: Up
Valid values:
 Minimum: 733
 Maximum: 8000000
Unit: Baud
Description: Real baudrate of sensor channel 1 to 4.

SA_SensorChannelRealBaudrate1..4

16.2.2 Parameter management

16.2.2.1 Save_Parameters

Save actual parameters at controller.

This command is only available from hardware revision 1.

16.2.2.2 Load_Parameters

Load actual parameters into controller RAM. This command is only available from hardware revision 1.

16.2.3 Information

16.2.3.1 Get_FPGAVersion

Get the version of the FPGA of the adapter.

Parameter: int SA_FPGAVersion
Direction: Up
Valid values:
 Minimum: 0
 Maximum: 255
Description: Version of the FPGA of the adapter.

SA_FPGAVersion

16.2.3.2 Get_HWRevision

Get the hardware revision of the adapter.

Parameter: int SA_HWRevision SA_HWRevision
Direction: Up
Valid values:
 Minimum: 0
 Maximum: 255
Description: Hardware revision of the adapter.

16.2.3.3 Get_Option

Get the option of the adapter.

Parameter: int SA_Option SA_Option
Direction: Up
Valid values:
 Minimum: 0
 Maximum: 15
Description: Option of the adapter.

16.2.4 Timer

16.2.4.1 Set_TimerFrequency

Set the frequency and pulse width for the timers on the adapter.

Parameter: int SP_TimerNumber SP_TimerNumber
Direction: Down
Valid values:
 1 = Timer 1
 2 = Timer 2
Description: Number of the timer to parametrize.

Parameter: double SP_TimerFrequency SP_TimerFrequency
Direction: Down
Valid values:
 Minimum: 0.0
 Maximum: 12000000.0
Unit: Hz
Description: Frequency of the timer.
 0.0 means the timer is off.

Parameter: double SP_TimerRatio SP_TimerRatio
Direction: Down
Valid values:
 Minimum: 0.0
 Maximum: 1.0
Description: Ratio of the high and low contingent of one period.
 If you are unsure, specify a value of 0.5.
 If SP_TimerFrequency is 0.0, a value of 0.0 means the timer is low,
 otherwise the timer is high.

Parameter: double SA_RealFrequency SA_RealFrequency

Direction: Up

Valid values:

Minimum: 0.0

Maximum: 12000000.0

Unit: Hz

Description: The real frequency applied to the timer. Because of internal limitations not each value can be set.

Parameter: double SA_RealRatio SA_RealRatio

Direction: Up

Valid values:

Minimum: 0.0

Maximum: 1.0

Description: The real ratio applied to the timer. Because of internal limitations not each value can be set.

16.2.4.2 Get_TimerFrequency

Get the frequency and pulse width for the timers on the adapter.

Parameter: int SP_TimerNumber SP_TimerNumber

Direction: Down

Valid values:

1= Timer 1

2= Timer 2

Description: Number of the timer to parametrize.

Parameter: double SA_RealFrequency SA_RealFrequency

Direction: Up

Valid values:

Minimum: 0.0

Maximum: 12000000.0

Unit: Hz

Description: The frequency applied to the timer.

Parameter: double SA_RealRatio SA_RealRatio

Direction: Up

Valid values:

Minimum: 0.0

Maximum: 1.0

Description: The ratio applied to the timer.

16.2.5 RS422 (Sensor)

16.2.5.1 Inputs

16.2.5.1.1 Use_Gate

The external trigger inputs 1 to 3 of the adapter (5V TTL signal) can be used to lock or free the FIFO for data from specific channels.

Parameter: int SP_GateChannel SP_GateChannel

Direction: Down

Valid values:

- 0= External trigger input 1 for Sensor channel 1+2 (Connector 1/2)
- 1= External trigger input 1 for Sensor channel 1+2 (Connector 1/2)
- 2= External trigger input 2 for Sensor channel 3+4 (Connector 3/4)
- 3= External trigger input 2 for Sensor channel 3+4 (Connector 3/4)
- 4= External trigger input 3 for external trigger input and RxD

Description: Number of the channel to lock by external trigger input. Because multiple channels are affected by one external trigger input, the numbers 0..1 or 2..3 always affects the same external trigger input.

Parameter: int SP_ActivateGate SP_ActivateGate

Direction: Down

Valid values:

- 0= on
- 1= off

Description: Specifies if the gate function is enabled or disabled for the specific digital input.

16.2.5.1.2 Get_RxDValue

Get the RxD values.

Parameter: int SA_RxDValue1..4 SA_RxDValue1..4

Direction: Up

Valid values:

- 0
- 1

Description: Value of the RxD line.

16.2.5.2 Outputs

16.2.5.2.1 Set_TxDSource

Set the source which is used for TxD line.

Parameter: int SP_TxDChannel1..4 SP_TxDChannel1..4

Direction: Down

Valid values:

- 1= Leave unchanged
- 0= Sensor output (transmitter)
- 1= Continuously low
- 2= Continuously high
- 3= Any sensor output (1-4)

Description: Source which is used for TxD line.

16.2.5.2.2 Get_TxDSource

Get the source which is used for TxD line.

Parameter: int SA_TxDChannel1..4

SA_TxDChannel1..4

Direction: Up

Valid values:

- 1= Leave unchanged
- 0= sensor output (transmitter)
- 1= Continuously low
- 2= Continuously high
- 3= Any sensor output (1-4)

Description: Source which is used for TxD line.

16.2.5.3 Trigger

16.2.5.3.1 Set_TrgetSource

Set the source which is used for Trg line.

Parameter: int SP_TrGChannel1..4

SP_TrGChannel1..4

Direction: Down

Valid values:

- 1= Leave unchanged
- 0= Trigger input 1
- 1= Trigger input 2
- 2= Trigger input 3
- 3= Trigger input 4
- 4= Timer 1 pulse width
- 5= Timer 2 pulse width
- 6= Continuously low
- 7= Continuously high

Description: Source which is used for Trg line.

16.2.5.3.2 Get_TrgetSource

Get the source which is used for Trg line.

Parameter: int SA_TrGChannel1..4

SA_TrGChannel1..4

Direction: Up

Valid values:

- 1= Leave unchanged
- 0= Trigger input 1
- 1= Trigger input 2
- 2= Trigger input 3
- 3= Trigger input 4
- 4= Timer 1 pulse width
- 5= Timer 2 pulse width
- 6= Continuously low
- 7= Continuously high

Description: Source which is used for Trg line.

16.2.6 RS422 (Computer)

16.2.6.1 Set_DataMode

Set the data output format at RS422 and USB interface.
 This command is only available from hardware revision 1.

Parameter: int SP_DataMode

SP_DataMode

Direction: Down

Valid values:

- 0= RS422 input for Sensor 1
- 1= Comparator at RS422 (three byte data format, like ILD2300, tuple at USB). Only valid at special option 1.
- 2= Off (no data at RS422, only at USB)
- 3= Tuple at RS422 (two bytes, address and data per sensor byte, no data at USB)

Description: Data output format at RS422 and USB interface.

If tuple mode is active, data is no longer output at USB interface.

If comparator mode is active, only sensor data is output at RS422, Digital IN is already output at USB. This mode does only work if all selected sensors supports three byte data format and outputs one value per frame.

16.2.6.2 Get_DataMode

Get the data output format at RS422 and USB interface.
 This command is only available from hardware revision 1.

Parameter: int SA_DataMode

SA_DataMode

Direction: Up

Valid values:

- 0= RS422 input for Sensor 1
- 1= Comparator at RS422 (three byte data format, like ILD2300, tuple at USB)
- 2= Off (no data at RS422, only at USB)
- 3= Tuple at RS422 (two bytes, address and data per sensor byte, no data at USB)

Description: Data output format at RS422 and USB interface.

16.2.6.3 Set_RS422OutSource

Set the source which is used for RS422 output at comparator mode.
 This command is only available at special option 1.

Parameter: int SP_RS422OutSource

SP_RS422OutSource

Direction: Down

Valid values:

- 0= Timer 1 pulse width
- 1= Timer 2 pulse width
- 2= Sensor channel 1
- 3= Sensor channel 3
- 4= External trigger input 1
- 5= External trigger input 2
- 6= External trigger input 3
- 7= External trigger input 4

Description: Source which is used for RS422 output at comparator mode.

16.2.6.4 Get_RS422OutSource

Get the source which is used for RS422 output at comparator mode.
 This command is only available at special option 1.

Parameter: int SA_RS422OutSource	SA_RS422OutSource
Direction: Up	
Valid values:	
0= Timer 1 pulse width	
1= Timer 2 pulse width	
2= Sensor channel 1	
3= Sensor channel 3	
4= External trigger input 1	
5= External trigger input 2	
6= External trigger input 3	
7= External trigger input 4	
Description: Source which is used for RS422 output at comparator mode.	

16.2.6.5 Set_RS422Baudrate

Set the baudrate of RS422 output.
 This command is only available from hardware revision 1.

Parameter: int SP_RS422Baudrate	SP_RS422Baudrate
Direction: Down	
Valid values:	
Minimum: 733	
Maximum: 8000000	
Description: Desired baudrate of RS422 output.	
If the device should be found by SensorFinder commands over RS422, only following baudrates are allowed: 8000000, 4000000, 3500000, 3000000, 2500000, 2000000, 1500000, 921600, 691200, 460800, 230400, 115200, 9600.	
Parameter: int SA_RS422RealBaudrate	SA_RS422RealBaudrate
Direction: Up	
Valid values:	
Minimum: 733	
Maximum: 8000000	
Description: Real baudrate of RS422 output.	

16.2.6.6 Get_RS422Baudrate

Get the baudrate of RS422 output.
 This command is only available from hardware revision 1.

Parameter: int SA_RS422Baudrate	SA_RS422Baudrate
Direction: Up	
Valid values:	
Minimum: 733	
Maximum: 8000000	
Description: Nominal baudrate of RS422 output.	

Parameter: int SA_RS422RealBaudrate SA_RS422RealBaudrate
Direction: Up
Valid values:
Minimum: 733
Maximum: 8000000
Description: Real baudrate of RS422 output.

16.2.7 Switching inputs

16.2.7.1 Set_ExtTriggerInSource

Set the source which is used for external trigger inputs.

Parameter: int SP_ExtTriggerInSource SP_ExtTriggerInSource
Direction: Down
Valid values:
 0= Sensor channel 1
 1= Sensor channel 3
 2= Timer 1 pulse width
 3= Timer 2 pulse width
Description: Source which is used for external trigger inputs.

16.2.7.2 Get_ExtTriggerInSource

Get the source which is used for external trigger inputs.

Parameter: int SA_ExtTriggerInSource SA_ExtTriggerInSource
Direction: Up
Valid values:
 0= Sensor channel 1
 1= Sensor channel 3
 2= Timer 1 pulse width
 3= Timer 2 pulse width
Description: Source which is used for external trigger inputs.

16.2.7.3 Set_ExtTriggerInDirection

Set the external trigger input direction.

Parameter: int SP_ExtTriggerInDirection1..4 SP_ExtTriggerInDirection1..4
Direction: Down
Valid values:
 -1= Leave unchanged
 0= Normal
 1= Inverted
Description: External trigger input direction.

16.2.7.4 Get_ExtTriggerInDirection

Get the external trigger input direction.

Parameter: int SA_ExtTriggerInDirection1..4

Direction: Up

Valid values:

-1= Leave unchanged

0= Normal

1= Inverted

Description: External trigger input direction.

SA_ExtTriggerInDirection1..4

16.2.7.5 Get_ExtTriggerInValue

Get the external trigger input values.

Parameter: int SA_ExtTriggerInValue1..4

SA_ExtTriggerInValue1..4

Direction: Up

Valid values:

0

1

Description: External trigger input value.

16.2.8 Switching outputs

16.2.8.1 Set_ExtTriggerOutSource

Set the source which is used for external trigger outputs.

Parameter: int SP_ExtTriggerOutSource1..2

SP_ExtTriggerOutSource1..2

Direction: Down

Valid values:

-1= Leave unchanged

0= Sensor channel 1

1= Sensor channel 2

2= Sensor channel 3

3= Sensor channel 4

4= Timer 1 pulse width

5= Timer 2 pulse width

6= Continuously low

7= Continuously high

Description: Source which is used for external trigger outputs.

16.2.8.2 Get_ExtTriggerOutSource

Get the source which is used for external trigger outputs.

Parameter: int SA_ExtTriggerOutSource1..2

SA_ExtTriggerOut-
Source1..2

Direction: Up

Valid values:

- 1= Leave unchanged
- 0= Sensor channel 1
- 1= Sensor channel 2
- 2= Sensor channel 3
- 3= Sensor channel 4
- 4= Timer 1 pulse width
- 5= Timer 2 pulse width
- 6= Continuously low
- 7= Continuously high

Description: Source which is used for external trigger outputs.

16.2.9 LED

16.2.9.1 Set_LEDMode

Set the mode of the LED's at device.

Parameter: int SP_LEDMode1

SP_LEDMode1

Direction: Down

Valid values:

- 1= Leave unchanged
- 0= Light with USB ready
- 1= Light with trigger input 1
- 2= Light with RxD 1
- 3= Light with TxD 1

Description: Mode of LED 1.

Parameter: int SP_LEDMode2

SP_LEDMode2

Direction: Down

Valid values:

- 1= Leave unchanged
- 0= Light with extern Power +24V
- 1= Light with trigger input 2
- 2= Light with RxD 2
- 3= Light with TxD 2

Description: Mode of LED 2.

Parameter: int SP_LEDMode3

SP_LEDMode3

Direction: Down

Valid values:

- 1= Leave unchanged
- 0= Light if FIFO has data
- 1= Light with trigger input 3
- 2= Light with RxD 3
- 3= Light with TxD 3

Description: Mode of LED 3.

Parameter: int SP_LEDMode4 SP_LEDMode4
Direction: Down
Valid values:
 -1= Leave unchanged
 0= Light with any TxD (1-4)
 1= Light with trigger input 4
 2= Light with RxD 4
 3= Light with TxD 4
Description: Mode of LED 4.

16.2.9.2 Get_LEDMode

Get the mode of the LED's at device.

Parameter: int SA_LEDMode1 SA_LEDMode1
Direction: Up
Valid values:
 -1= Leave unchanged
 0= Light with USB ready
 1= Light with trigger input 1
 2= Light with RxD 1
 3= Light with TxD 1
Description: Mode of LED 1.

Parameter: int SA_LEDMode2 SA_LEDMode2
Direction: Up
Valid values:
 -1= Leave unchanged
 0= Light with extern Power +24V
 1= Light with trigger input 2
 2= Light with RxD 2
 3= Light with TxD 2
Description: Mode of LED 2.

Parameter: int SA_LEDMode3 SA_LEDMode3
Direction: Up
Valid values:
 -1= Leave unchanged
 0= Light if FIFO has data
 1= Light with trigger input 3
 2= Light with RxD 3
 3= Light with TxD 3
Description: Mode of LED 3.

Parameter: int SA_LEDMode4 SA_LEDMode4
Direction: Up
Valid values:
 -1= Leave unchanged
 0= Light with any TxD (1-4)
 1= Light with trigger input 4
 2= Light with RxD 4
 3= Light with TxD 4
Description: Mode of LED 4.

16.3 Commands for PCICardIF2004

See IF2004 manual for detailed description of sensor commands.

Attention! The command [Use_Defaults](#) must be called for correct functionality of encoder. Please see the example at command [Use_Defaults](#).

This sensor supports following interfaces:

- [IF2004](#) (native).

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from encoder on IF2004, from 0 to 65536.
- Scaled values are scaled using encoder distance per count (if known by MEDAQLib, use sensor command [Use_Defaults](#) with parameter [IP_Range](#)).

16.3.1 Special commands

16.3.1.1 [Use_Defaults](#)

This command parametrizes the IF2004 PCI card. If same parameters are not specified they are not changed.

Parameter: double [IP_Range](#)

[IP_Range](#)

Direction: Down

Valid values:

Minimum: -1.79769e+308 (-DBL_MAX)

Maximum: 1.79769e+308 (DBL_MAX)

Unit: μm or mm

Description: It is the distance per count of the encoder used by the driver for scaling data.

If it is zero, no scaling is done.

Parameter: int [IP_EncCountMode](#)

[IP_EncCountMode](#)

Valid values:

0= Counter without phase discriminator (Trace A is direction, Trace B is pulse, Trace C is load or latch signal).

1= Counter with phase discriminator and 1-fold discriminator.

2= Counter with phase discriminator and 2-fold discriminator.

3= Counter with phase discriminator and 4-fold discriminator.

Direction: Down

Description: The counting mode of the encoder

Parameter: int [IP_EncSwapTraceA_B](#)

[IP_EncSwapTraceA_B](#)

Direction: Down

Valid values:

0= no swap

1= swap Trace A and B

Description: Trace A and B are swapped which negates the count direction

Parameter: int IP_EncInvertTraceA_B	IP_EncInvertTraceA_B
Direction: Down	
Valid values:	
0= no invert	
1= invert Trace A and B	
Description: Trace A and B are inverted, for Encoders where Trace C is 1 when Trace A and B are 0	
Parameter: int IP_EncLatchSrc	IP_EncLatchSrc
Direction: Down	
Valid values:	
0= Never	
1..4= Latch with start bit on channel 1 to 4.	
5= Latch with next reference on Trace C (if unlocked).	
6= Latch with second reference on Trace C (if unlocked).	
7= Latch with every reference on Trace C (always).	
Description: Specifies when the encoder value should be stored to FIFO (only useful when IP_ChannelNumber is 3)	
Parameter: int IP_EncLoadOnRef	IP_EncLoadOnRef
Direction: Down	
Valid values:	
0= never	
1= Load with next reference on Trace C (if unlocked).	
2= Load with every reference on Trace C (always).	
3= Clear with every reference on Trace C (always) and load when count is -1.	
Description: Specifies when the encoder value should be changed by IF2004 card.	

Example how to parametrize the encoder:

```

DWORD instance= CreateSensorInstance (PCI_CARD_IF2004);
DWORD err= OpenSensorIF2004 (instance, 0, 3);
/* error handling, if err!=ERR_NOERROR */

SetParameterString (instance, "S_Command", "Use_Defaults");
/* Set encoder to increment or decrement each full period */
SetParameterInt (instance, "IP_EncCountMode", 1);
/* Each time when a value arrives at first sensor channel, put the encoder value into
FIFO */
SetParameterInt (instance, "IP_EncLatchSrc", 1);
err= SensorCommand (instance);
/* error handling, if err!=ERR_NOERROR*/

```

16.3.1.2 Get_DrvSetting

Returns the current settings of the driver used for operating with sensor data. It is the opposite of Use_Defaults.

Parameter: double IA_Range	IA_Range
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 1.79769e+308 (DBL_MAX)	
Unit: μm or mm	
Description: It is the distance per count of the encoder used by the driver for scaling data.	

16.3.1.3 Set_SyncMaster

Set a synchronisation master for a specified sensor channel while interface is open.

Parameter: int SP_SensorChannel	SP_SensorChannel
Direction: Down	
Valid values:	
0= Sensor channel 1 (Base Board, Connector 1/2)	
1= Sensor channel 2 (Base Board, Connector 1/2)	
2= Sensor channel 3 (Base Board, Connector 3/4)	
3= Sensor channel 4 (Base Board, Connector 3/4)	
Description: Sensor channel to synchronize.	
Parameter: int SP_SyncMasterChannel	SP_SyncMasterChannel
Direction: Down	
Valid values:	
-1= No synchronisation master	
0= Sensor channel 1 (Base Board, Connector 1/2)	
1= Sensor channel 2 (Base Board, Connector 1/2)	
2= Sensor channel 3 (Base Board, Connector 3/4)	
3= Sensor channel 4 (Base Board, Connector 3/4)	
Description: Channel number to synchronise with. See for further information.	

16.3.2 General

16.3.2.1 Get_FPGAVersion

Get the version of the FPGA on the card.

Parameter: int SA_FPGAVersion	SA_FPGAVersion
Direction: Up	
Valid values:	
0= V1.1	
1= V1.2 (115.2 KBaud, new data format for SENSOR_ILD1700)	
2= V1.3 (1.25 MBaud, support for two sensor values per data frame)	
Description: Version of the FPGA on the card.	

16.3.2.2 IF2004_SystemReset

Make a system reset: Clear FIFO and send a pulse ($100 \mu s$) reset line at both sensor connectors.

16.3.2.3 IF2004_SensorReset12

Set the reset line of sensor connector 0+1 to specified value.

Parameter: int SP_SensorReset12	SP_SensorReset12
Direction: Down	
Valid values:	
0= FALSE	
1= TRUE	
Description: Value for reset line.	

16.3.2.4 IF2004_SensorReset34

Set the reset line of sensor connector 2+3 to specified value.

Parameter: int SP_SensorReset34

SP_SensorReset34

Direction: Down

Valid values:

0 = FALSE

1 = TRUE

Description: Value for reset line.

16.3.3 Encoder

16.3.3.1 Actions

16.3.3.1.1 Enc_ClearCounter

Clears the counter (set to 0).

16.3.3.1.2 Enc_SetLoadReg

Set the load register for counter.

Parameter: int SP_LoadReg

SP_LoadReg

Direction: Down

Valid values:

Minimum: 0

Maximum: 65535

Description: Counter value.

16.3.3.1.3 Enc_LoadCounter

Load the counter from the load register.

16.3.3.1.4 Enc_LatchCounter

Get the recent counter value to the latch register.

16.3.3.1.5 Enc_GetLatchReg

Get the counter value from the latch register.

Parameter: int SA_LatchReg

SA_LatchReg

Direction: Up

Valid values:

Minimum: 0

Maximum: 65535

Description: Counter value.

16.3.3.1.6 Enc_UnlockTraceC

Unlocks Trace C.

16.3.3.1.7 Enc_CheckRef

Checks the actual state of trace C.

Parameter: int SA_Ref

SA_Ref

Direction: Up

Valid values:

0= FALSE

1= TRUE

Description: True if reference (trace C) is set.

16.3.3.1.8 Enc_IsFirstRef

Checks if Ref (Trace C) was reached since last Unlock.

Parameter: int SA_IsFirstRef

SA_IsFirstRef

Direction: Up

Valid values:

0= FALSE

1= TRUE

Description: True, if it is first reference.

16.3.3.1.9 Enc_IsSecondRef

Checks if Ref (Trace C) was reached twice since last Unlock.

Parameter: int SA_IsSecondRef

SA_IsSecondRef

Direction: Up

Valid values:

0= FALSE

1= TRUE

Description: True, if it is second reference.

16.3.4 RS422

16.3.4.1 Inputs

16.3.4.1.1 IF2004_CheckGate

Checks the actual state of gate.

Parameter: int SA_Gate

SA_Gate

Direction: Up

Valid values:

0= FALSE

1= TRUE

Description: True if gate is set.

16.4 Commands for PCICard_IF2008

See PCICard_IF2008 manual for detailed description of sensor commands.

Attention! The command [Use_Defaults](#) must be called if encoder or ADC values should be scaled.

This sensor supports following interfaces:

- [IF2008](#) (native).

To allow MEDAQLib to scale data, it is recommended to call sensor command [Use_Defaults](#) after [OpenSensor](#).

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from IF2008 card, from 0 to 4294967295 (Encoder) or 65535 (ADC) or 255 (Digital in).
- Scaled values are scaled using count factors or ranges (if known by MEDAQLib, use sensor command [Use_Defaults](#)).

16.4.1 Special commands

16.4.1.1 Use_Defaults

This command parametrizes the IF2008 PCI card. If same parameters are not specified they are not changed.

Parameter: double IP_Encoder1CountFactor

IP_Encoder1CountFactor

Direction: Down

Valid for sensor:

PCI_CARD_IF2008

Valid values:

Minimum: -1.79769e+308 (-DBL_MAX)

Maximum: 1.79769e+308 (DBL_MAX)

Default: 1.0

Description: It is the distance per count of the encoder 1 used by the driver for scaling data. If it is zero, no scaling is done.

Parameter: double IP_Encoder2CountFactor

IP_Encoder2CountFactor

Direction: Down

Valid for sensor:

PCI_CARD_IF2008

Valid values:

Minimum: -1.79769e+308 (-DBL_MAX)

Maximum: 1.79769e+308 (DBL_MAX)

Default: 1.0

Description: It is the distance per count of the encoder 2 used by the driver for scaling data. If it is zero, no scaling is done.

16.4. Commands for PCICard_IF2008

Parameter: int IP_ADC1Range	IP_ADC1Range
Direction: Down	
Valid for sensor:	
PCI_CARD_IF2008	
Valid values:	
0= 0..5V	
1= 0..10V	
2= +-5V	
3= +-10V	
Default: 3	
Description: It is range of ADC 1 used by the driver for scaling data.	
Parameter: int IP_ADC2Range	IP_ADC2Range
Direction: Down	
Valid for sensor:	
PCI_CARD_IF2008	
Valid values:	
0= 0..5V	
1= 0..10V	
2= +-5V	
3= +-10V	
Default: 3	
Description: It is range of ADC 2 used by the driver for scaling data.	

16.4.1.2 Get_DrvSetting

Returns the current settings of the driver used for operating. It is the opposite of Use_Defaults.

Parameter: double IA_Encoder1CountFactor	IA_Encoder1CountFactor
Direction: Up	
Valid for sensor:	
PCI_CARD_IF2008	
Valid values:	
Minimum: -1.79769e+308 (-DBL_MAX)	
Maximum: 1.79769e+308 (DBL_MAX)	
Description: It is the distance per count of the encoder 1 used by the driver for scaling data. If it is zero, no scaling is done.	
Parameter: double IA_Encoder2CountFactor	IA_Encoder2CountFactor
Direction: Up	
Valid for sensor:	
PCI_CARD_IF2008	
Valid values:	
Minimum: -1.79769e+308 (-DBL_MAX)	
Maximum: 1.79769e+308 (DBL_MAX)	
Description: It is the distance per count of the encoder 2 used by the driver for scaling data. If it is zero, no scaling is done.	

Parameter: int IA_ADC1Range IA_ADC1Range

Direction: Up

Valid for sensor:

PCI_CARD_IF2008

Valid values:

0= 0..5V

1= 0..10V

2= +-5V

3= +-10V

Description: It is range of ADC 1 used by the driver for scaling data.

Parameter: int IA_ADC2Range IA_ADC2Range

Direction: Up

Valid for sensor:

PCI_CARD_IF2008

Valid values:

0= 0..5V

1= 0..10V

2= +-5V

3= +-10V

Description: It is range of ADC 2 used by the driver for scaling data.

16.4.1.3 Set_SyncMaster

Set a synchronisation master for a specified sensor channel while interface is open.

Parameter: int SP_SensorChannel SP_SensorChannel

Direction: Down

Valid values:

0= Sensor channel 1 (Base Board, Connector X1)

1= Sensor channel 2 (Base Board, Connector X1)

2= Sensor channel 3 (Base Board, Connector X2)

3= Sensor channel 4 (Base Board, Connector X2)

4= Sensor channel 5 (Extension Board, Connector X1)

5= Sensor channel 6 (Extension Board, Connector X1)

Description: Sensor channel to synchronize.

Parameter: int SP_SyncMasterChannel SP_SyncMasterChannel

Direction: Down

Valid values:

-1= No synchronisation master

0= Sensor channel 1 (Base Board, Connector X1)

1= Sensor channel 2 (Base Board, Connector X1)

2= Sensor channel 3 (Base Board, Connector X2)

3= Sensor channel 4 (Base Board, Connector X2)

4= Sensor channel 5 (Extension Board, Connector X1)

5= Sensor channel 6 (Extension Board, Connector X1)

6= Encoder 1

7= Encoder 2

8= Digital IN

9= RxD

10= ADC 1

11= ADC 2

Description: Channel number to synchronise with. See for further information.

16.4.1.4 Set_PowerSwitch

Switches the sensor power on or off.

Parameter: int SP_DisablePower SP_DisablePower
Direction: Down
Valid values:
 0= FALSE
 1= TRUE
Description: Sensor power disabled (for all channels)

16.4.2 Information

16.4.2.1 Get_FPGAVersion

Get the version of the FPGA on the card.

Parameter: int SA_FPGAVersion SA_FPGAVersion
Direction: Up
Valid values:
Minimum: 0
Maximum: 63
Description: Version of the FPGA on the card.

16.4.2.2 Get_HWRevision

Get the hardware revision of the IF2008 card.

Parameter: int SA_HWRevision SA_HWRevision
Direction: Up
Valid values:
Minimum: 0
Maximum: 3
Description: Hardware revision of the IF2008 card.

16.4.2.3 Is_Channel56Available

Check if additional sensor channels at IF2008E extension card are available.

Parameter: int SA_Channel56Available SA_Channel56Available
Direction: Up
Valid values:
Minimum: 0= no
Maximum: 1= yes
Description: Availability of sensor channels at IF2008E extension card.

16.4.2.4 Is_ADCAvailable

Check if Analog/Digital converter (ADC) at IF2008E extension card are available.

Parameter: int SA_ADCAvailable

SA_ADCAvailable

Direction: Up

Valid values:

Minimum: 0 = no

Maximum: 1 = yes

Description: Availability of Analog/Digital converter (ADC) at IF2008E extension card.

16.4.2.5 Is_DigitalIOAvailable

Check if digital IO at IF2008E extension card or IF2008IO extension slot is available.

Parameter: int SA_DigitalIOAvailable

SA_DigitalIOAvailable

Direction: Up

Valid values:

Minimum: 0 = no

Maximum: 1 = yes

Description: Availability of digital IO at IF2008E extension card or IF2008IO extension slot.

16.4.2.6 Get_PowerError

Detect if sensor power is overloaded or short-circuited.

Parameter: int SA_PowerError

SA_PowerError

Direction: Up

Valid values:

Minimum: 0 = OK

Maximum: 1 = overloaded

Description: State of sensor power supply.

16.4.3 Timer

16.4.3.1 Set_TimerFrequency

Set the frequency and pulse width for the timers on the card.

Parameter: int SP_TimerNumber

SP_TimerNumber

Direction: Down

Valid values:

1 = Timer 1

2 = Timer 2

3 = Timer 3

Description: Number of the timer to parametrize.

Parameter: double SP_TimerFrequency	SP_TimerFrequency
Direction: Down	
Valid values:	
Minimum: 0.0	
Maximum: 20000000.0	
Unit: Hz	
Description: Frequency of the timer.	
0.0 means the timer is off.	
Parameter: double SP_TimerRatio	SP_TimerRatio
Direction: Down	
Valid values:	
Minimum: 0.0	
Maximum: 1.0	
Description: Ratio of the high and low contingent of one period.	
If you are unsure, specify a value of 0.5.	
If SP_TimerFrequency is 0.0, a value of 0.0 means the timer is low, otherwise the timer is high.	
Parameter: double SA_RealFrequency	SA_RealFrequency
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 20000000.0	
Unit: Hz	
Description: The real frequency applied to the timer. Because of internal limitations not each value can be set.	
Parameter: double SA_RealRatio	SA_RealRatio
Direction: Up	
Valid values:	
Minimum: 0.0	
Maximum: 1.0	
Description: The real ratio applied to the timer. Because of internal limitations not each value can be set.	

16.4.4 Encoder

16.4.4.1 Settings

16.4.4.1.1 Set_EncoderInterpolation

Set the interpolation of the encoder.

Parameter: int SP_EncoderNumber	SP_EncoderNumber
Direction: Down	
Valid values:	
6= Encoder 1	
7= Encoder 2	
Description: Number of the encoder to parametrize.	

16.4. Commands for PCICard_IF2008

Parameter: int SP_EncoderInterpolation

SP_EncoderInterpolation

Direction: Down

Valid values:

- 0= 1 fold (TTL & 1Vss)
- 1= 2 fold (TTL & 1Vss)
- 2= 3 fold (only 1Vss)
- 3= 4 fold (TTL & 1Vss)
- 4= 5 fold (only 1Vss)
- 5= 6 fold (only 1Vss)
- 6= 8 fold (only 1Vss)
- 7= 10 fold (only 1Vss)
- 8= 12 fold (only 1Vss)
- 9= 16 fold (only 1Vss)
- 10= 20 fold (only 1Vss)
- 11= 24 fold (only 1Vss)
- 12= 32 fold (only 1Vss)
- 13= 40 fold (only 1Vss)
- 14= 48 fold (only 1Vss)
- 15= 64 fold (only 1Vss)

Description: Interpolation mode of the encoder.

Attention! If a encoder with 5V TTL signal is used, only mode 0, 1 and 3 does work. If another mode is selected, the result is unpredictable.

16.4.4.1.2 Get_EncoderInterpolation

Get the interpolation of the encoder.

Parameter: int SP_EncoderNumber

SP_EncoderNumber

Direction: Down

Valid values:

- 6= Encoder 1
- 7= Encoder 2

Description: Number of the encoder to retrieve information.

Parameter: int SA_EncoderInterpolation

SA_EncoderInterpolation

Direction: Up

Valid values:

- 0= 1 fold (TTL & 1Vss)
- 1= 2 fold (TTL & 1Vss)
- 2= 3 fold (only 1Vss)
- 3= 4 fold (TTL & 1Vss)
- 4= 5 fold (only 1Vss)
- 5= 6 fold (only 1Vss)
- 6= 8 fold (only 1Vss)
- 7= 10 fold (only 1Vss)
- 8= 12 fold (only 1Vss)
- 9= 16 fold (only 1Vss)
- 10= 20 fold (only 1Vss)
- 11= 24 fold (only 1Vss)
- 12= 32 fold (only 1Vss)
- 13= 40 fold (only 1Vss)
- 14= 48 fold (only 1Vss)
- 15= 64 fold (only 1Vss)

Description: Interpolation mode of the encoder.

16.4.4.1.3 Set_EncoderDirection

Set the direction of the encoder.

Parameter: int SP_EncoderNumber	SP_EncoderNumber
Direction: Down	
Valid values:	
6= Encoder 1	
7= Encoder 2	
Description:	Number of the encoder to parametrize.
Parameter: int SP_EncoderDirection	SP_EncoderDirection
Direction: Down	
Valid values:	
0= normal	
1= reverse	
Description:	Count direction of the encoder.

16.4.4.1.4 Get_EncoderDirection

Get the direction of the encoder.

Parameter: int SP_EncoderNumber	SP_EncoderNumber
Direction: Down	
Valid values:	
6= Encoder 1	
7= Encoder 2	
Description:	Number of the encoder to parametrize.
Parameter: int SA_EncoderDirection	SA_EncoderDirection
Direction: Up	
Valid values:	
0= normal	
1= reverse	
Description:	Count direction of the encoder.

16.4.4.1.5 Set_EncoderMode

Set the behaviour of the encoder when a reference is reached.

Parameter: int SP_EncoderNumber	SP_EncoderNumber
Direction: Down	
Valid values:	
6= Encoder 1	
7= Encoder 2	
Description:	Number of the encoder to parametrize.

Parameter: int SP_EncoderMode SP_EncoderMode

Direction: Down

Valid values:

- 0= no function
- 1= load counter with next reference mark
- 2= load counter with all reference marks
- 3= clear counter with all reference marks, load when -1 is reached
- 4= counter without phase discriminator (pulse counter)

Description: Behaviour when a reference is reached.

If counter is set to pulse counter, the encoder direction specifies the function of the input lines. On normal direction, A is for count direction and B is for pulse counting, on reverse direction it is reversed.

A reference mark only can be detected, if A, B and C are high at the same time. On encoders this is ensured, on other pulse generators it must be ensured by the hardware developer.

16.4.4.1.6 Get_EncoderMode

Get the behaviour of the encoder when a reference is reached.

Parameter: int SP_EncoderNumber SP_EncoderNumber

Direction: Down

Valid values:

- 6= Encoder 1
- 7= Encoder 2

Description: Number of the encoder to parametrize.

Parameter: int SA_EncoderMode SA_EncoderMode

Direction: Up

Valid values:

- 0= no function
- 1= load counter with next reference mark
- 2= load counter with all reference marks
- 3= clear counter with all reference marks, load when -1 is reached
- 4= counter without phase discriminator (pulse counter)

Description: Behaviour when a reference is reached.

16.4.4.1.7 Set_EncoderLatchSource

Set the latch source which triggers aquiring one value.

Parameter: int SP_EncoderNumber SP_EncoderNumber

Direction: Down

Valid values:

- 6= Encoder 1
- 7= Encoder 2

Description: Number of the encoder to parametrize.

Parameter: int SP_EncoderLatchSource SP_EncoderLatchSource

Direction: Down

Valid values:

- 0= never (locked)
- 1= Timer 1
- 2= Timer 2
- 3= Timer 3
- 4= Sensor channel 1
- 5= Sensor channel 2
- 6= Sensor channel 3
- 7= Sensor channel 4
- 8= Sensor channel 5
- 9= Sensor channel 6
- 10= Digital In 1
- 11= Digital In 2
- 12= Digital In 3
- 13= Digital In 4
- 14= second reference mark
- 15= all reference marks

Description: Latch source which triggers aquiring one value.

16.4.4.1.8 Get_EncoderLatchSource

Get the latch source which triggers aquiring one value.

Parameter: int SP_EncoderNumber

SP_EncoderNumber

Direction: Down

Valid values:

- 6= Encoder 1
- 7= Encoder 2

Description: Number of the encoder to parametrize.

Parameter: int SA_EncoderLatchSource

SA_EncoderLatchSource

Direction: Up

Valid values:

- 0= never (locked)
- 1= Timer 1
- 2= Timer 2
- 3= Timer 3
- 4= Sensor channel 1
- 5= Sensor channel 2
- 6= Sensor channel 3
- 7= Sensor channel 4
- 8= Sensor channel 5
- 9= Sensor channel 6
- 10= Digital In 1
- 11= Digital In 2
- 12= Digital In 3
- 13= Digital In 4
- 14= second reference mark
- 15= all reference marks

Description: Latch source which triggers aquiring one value.

16.4.4.2 Actions

16.4.4.2.1 Clear_Encoder

Clears the encoder value (set it to zero).

Parameter: int SP_EncoderNumber

SP_EncoderNumber

Direction: Down

Valid values:

6= Encoder 1

7= Encoder 2

Description: Number of the encoder to clear.

16.4.4.2.2 Set_EncoderPreload

Set the preload value which is used when loading the encoder.

Parameter: int SP_EncoderNumber

SP_EncoderNumber

Direction: Down

Valid values:

6= Encoder 1

7= Encoder 2

Description: Number of the encoder to parametrize.

Parameter: int SP_EncoderPreloadValue

SP_EncoderPreloadValue

Direction: Down

Valid values:

Minimum: -2147483648 (INT_MIN)

Maximum: 2147483647 (INT_MAX)

Description: Preload value which is used when loading the encoder.

16.4.4.2.3 Load_Encoder

Loads the encoder value with the value set by function Set_EncoderPreload.

Parameter: int SP_EncoderNumber

SP_EncoderNumber

Direction: Down

Valid values:

6= Encoder 1

7= Encoder 2

Description: Number of the encoder to load.

16.4.4.2.4 Latch_Encoder

Latch the encoder (get latest value into encoder register).

The value can be retrieved using function Get_EncoderValue.

Parameter: int SP_EncoderNumber

SP_EncoderNumber

Direction: Down

Valid values:

6= Encoder 1

7= Encoder 2

Description: Number of the encoder to latch.

16.4.4.2.5 Get_EncoderValue

Get the value of encoder register.

This function is useful in combination with function Latch_Encoder.

Parameter: int SP_EncoderNumber	SP_EncoderNumber
Direction: Down	
Valid values:	
6= Encoder 1	
7= Encoder 2	
Description:	Number of the encoder to latch.
Parameter: int SA_EncoderRawValue	SA_EncoderRawValue
Direction: Up	
Valid values:	
Minimum: -2147483648 (INT_MIN)	
Maximum: 2147483647 (INT_MAX)	
Description:	Raw value of encoder register.
Parameter: double SA_EncoderScaledValue	SA_EncoderScaledValue
Direction: Up	
Valid values:	
Minimum: -1.79769e+308 (-DBL_MAX)	
Maximum: 1.79769e+308 (DBL_MAX)	
Description:	Scaled value of encoder register.
	If IP_Encoder1CountFactor or IP_Encoder2CountFactor (at command Use_Defaults) is not set, the scaled value is equal to the raw value.

16.4.4.2.6 EnableRef_Encoder

Enable loading encoder value at next reference (set by function Set_EncoderMode with parameter next reference).

Parameter: int SP_EncoderNumber	SP_EncoderNumber
Direction: Down	
Valid values:	
6= Encoder 1	
7= Encoder 2	
Description:	Number of the encoder for enabling reference.

16.4.4.2.7 Get_EncoderReference

Get status if reference was reached first or second times.

Parameter: int SP_EncoderNumber	SP_EncoderNumber
Direction: Down	
Valid values:	
6= Encoder 1	
7= Encoder 2	
Description:	Number of the encoder to latch.

Parameter: int SA_FirstReference	SA_FirstReference
Direction: Up	
Valid values:	
Minimum: 0= no	
Maximum: 1= yes	
Description: Reference was reached first time.	
Parameter: int SA_SecondReference	SA_SecondReference
Direction: Up	
Valid values:	
Minimum: 0= no	
Maximum: 1= yes	
Description: Reference was reached second time.	

16.4.5 RS422

16.4.5.1 Inputs

16.4.5.1.1 Use_Gate

The digital inputs In 1 to In 4 of the card (5V TTL signal) can be used to lock or free the FIFO for data from specific channels.

Parameter: int SP_GateChannel	SP_GateChannel
Direction: Down	
Valid values:	
0= Digital In 1 for Sensor channel 1+2 (Base Board, Connector X1)	
1= Digital In 1 for Sensor channel 1+2 (Base Board, Connector X1)	
2= Digital In 2 for Sensor channel 3-6 (Base Board, Connector X2 + Extension Board, Connector X1)	
3= Digital In 2 for Sensor channel 3-6 (Base Board, Connector X2 + Extension Board, Connector X1)	
4= Digital In 2 for Sensor channel 3-6 (Base Board, Connector X2 + Extension Board, Connector X1)	
5= Digital In 2 for Sensor channel 3-6 (Base Board, Connector X2 + Extension Board, Connector X1)	
6= Digital In 3 for Encoder 1+2	
7= Digital In 3 for Encoder 1+2	
8= Digital In 4 for Digital In+RxD, ADC 1+2	
9= Digital In 4 for Digital In+RxD, ADC 1+2	
10= Digital In 4 for Digital In+RxD, ADC 1+2	
11= Digital In 4 for Digital In+RxD, ADC 1+2	
Description: Number of the channel to lock by digital in. Because multiple channels are affected by one digital input, the numbers 0..1, 2..5, 6..7 or 8..11 always affects the same digital input.	

Parameter: int SP_ActivateGate	SP_ActivateGate
Direction: Down	
Valid values:	
0= off	
1= on	
Description: Specifies if the gate function is enabled or disabled for the specific digital input.	

16.4.5.1.2 Get_RxDValue

Get the RxD values.

Parameter: int SA_RxDValue1..6

SA_RxDValue1..6

Direction: Up

Valid values:

0

1

Description: Value of the RxD line.

16.4.5.2 Outputs

16.4.5.2.1 Set_TxDSource

Set the source which is used for TxD line.

Parameter: int SP_TxDChannel1..6

SP_TxDChannel1..6

Direction: Down

Valid values:

-1 = Leave unchanged

0 = Sensor output (transmitter)

1 = User (output function)

Description: Source which is used for TxD line.

-1 means the channel will not be changed. 0 means the value is set by sensor output (if a sensor command is send to the sensor), 1 means the value is set by function Set_TxDValue.

16.4.5.2.2 Get_TxDSource

Get the source which is used for TxD line.

Parameter: int SA_TxDChannel1..6

SA_TxDChannel1..6

Direction: Up

Valid values:

0 = sensor output (transmitter)

1 = user (output function)

Description: Source which is used for TxD line.

16.4.5.2.3 Set_TxDValue

Set the values of TxD lines.

This function does only work when Set_TxDSource is set to 1 (output function)

Parameter: int SP_TxDValue1..6

SP_TxDValue1..6

Direction: Down

Valid values:

-1 = Leave unchanged

0

1

Description: TxD value.

-1 means leave the value unchanged.

16.4.5.2.4 Get_TxDValue

Get the values of TxD lines.

Parameter: int SA_TxDValue1..6

SA_TxDValue1..6

Direction: Up

Valid values:

0

1

Description: TxD value.

16.4.5.3 Trigger

16.4.5.3.1 Set_TrgSource

Set the source which is used for Trg line.

Parameter: int SP_TrgChannel1..6

SP_TrgChannel1..6

Direction: Down

Valid values:

-1= Leave unchanged

0= User (output function)

1= Timer 1 pulse width

2= Timer 2 pulse width

3= Timer 3 pulse width

4= Digital input 1

5= Digital input 2

6= Digital input 3

7= Digital input 4

Description: Source which is used for Trg line.

-1 means the value is not set. 0 means the value is set by function

Set_TrgValue, Timer 1..3 means the value is equal to the timer value.

Digital input is only available from FPGA version 8.

16.4.5.3.2 Get_TrgSource

Get the source which is used for Trg line.

Parameter: int SA_TrgChannel1..6

SA_TrgChannel1..6

Direction: Up

Valid values:

-1= Leave unchanged

0= User (output function)

1= Timer 1 pulse width

2= Timer 2 pulse width

3= Timer 3 pulse width

4= Digital input 1

5= Digital input 2

6= Digital input 3

7= Digital input 4

Description: Source which is used for Trg line.

16.4.5.3.3 Set_TrgValue

Set the values of Trg lines.

This function does only work when Set_TrgSource is set to 0 (output function)

Parameter: int SP_TrKeyValue1..6

SP_TrKeyValue1..6

Direction: Down

Valid values:

- 1 = Leave unchanged
- 0
- 1

Description: TxD value.

-1 means leave the value unchanged.

16.4.5.3.4 Get_TrgValue

Get the values of Trg lines.

Parameter: int SA_TrKeyValue1..6

SA_TrKeyValue1..6

Direction: Up

Valid values:

- 0
- 1

Description: Trg value.

16.4.6 Switching inputs

16.4.6.1 Set_DigitalInLatchSource

Set the latch source which triggers acquiring one value.

Attention! This command is only allowed if IF2008E extension card or IF2008IO extension slot is installed

Parameter: int SP_DigitalInLatchSource

SP_DigitalInLatchSource

Direction: Down

Valid values:

- 0 = never (locked)
- 1 = Timer 1 pulse width
- 2 = Timer 2 pulse width
- 3 = Timer 3 pulse width
- 4 = Sensor channel 1
- 5 = Sensor channel 2
- 6 = Sensor channel 3
- 7 = Sensor channel 4

Description: Latch source which triggers acquiring one value.

16.4.6.2 Get_DigitalInLatchSource

Get the latch source which triggers aquiring one value.

Attention! This command is only allowed if IF2008E extension card or IF2008IO extension slot is installed

Parameter: int SA_DigitalInLatchSource

SA_DigitalInLatchSource

Direction: Up

Valid values:

- 0= never (locked)
- 1= Timer 1 pulse width
- 2= Timer 2 pulse width
- 3= Timer 3 pulse width
- 4= Sensor channel 1
- 5= Sensor channel 2
- 6= Sensor channel 3
- 7= Sensor channel 4

Description: Latch source which triggers aquiring one value.

16.4.6.3 Get_DigitalInValue

Get the digital In values.

Attention! This command is only allowed if IF2008E extension card or IF2008IO extension slot is installed

Parameter: int SA_DigitalInValue1..4

SA_DigitalInValue1..4

Direction: Up

Valid values:

- 0
- 1

Description: Digital In value.

16.4.7 Switching outputs

16.4.7.1 Set_DigitalOutSource

Set the source which is used to output one value.

Attention! This command is only allowed if IF2008E extension card or IF2008IO extension slot is installed

Parameter: int SP_DigitalOut1..4

SP_DigitalOut1..4

Direction: Down

Valid values:

- 1= Leave unchanged
- 0= User (output function)
- 1= Timer 1 pulse width
- 2= Timer 2 pulse width
- 3= Timer 3 pulse width

Description: Source which is used to output one value.

-1 means the value is not set. 0 means the value is set by function Set_DigitalOutValue, Timer 1..3 means the value is equal to the timer value.

16.4.7.2 Get_DigitalOutSource

Get the source which is used to output one value.

Attention! This command is only allowed if IF2008E extension card or IF2008IO extension slot is installed

Parameter: int SA_DigitalOut1..4

SA_DigitalOut1..4

Direction: Up

Valid values:

- 0= user (output function)
- 1= Timer 1 pulse width
- 2= Timer 2 pulse width
- 3= Timer 3 pulse width

Description: Source which is used to output one value.

16.4.7.3 Set_DigitalOutValue

Set the digital Out values.

This function does only work when Set_DigitalOutSource is set to 0 (output function)

Attention! This command is only allowed if IF2008E extension card or IF2008IO extension slot is installed

Parameter: int SP_DigitalOutValue1..4

SP_DigitalOutValue1..4

Direction: Down

Valid values:

- 1= Leave unchanged
- 0
- 1

Description: Digital Out value. -1 means leave the value unchanged.

16.4.7.4 Get_DigitalOutValue

Get the digital Out values.

Attention! This command is only allowed if IF2008E extension card or IF2008IO extension slot is installed

Parameter: int SA_DigitalOutValue1..4

SA_DigitalOutValue1..4

Direction: Up

Valid values:

- 0
- 1

Description: Digital Out value.

16.4.8 Analog inputs

16.4.8.1 Set_ADCLatchSource

Set the latch source which triggers aquiring one value.

Attention! This command is only allowed if IF2008E extension card is installed

Parameter: int SP_ADCNumber

SP_ADCNumber

Direction: Down

Valid values:

- 10= ADC 1
- 11= ADC 2

Description: Number of the ADC (Analog/Digital converter) to parametrize.

Parameter: int SP_ADCLatchSource

SP_ADCLatchSource

Direction: Down

Valid values:

- 0= never (locked)
- 1= Timer 1
- 2= Timer 2
- 3= Timer 3
- 4= Sensor channel 1
- 5= Sensor channel 2
- 6= Sensor channel 3
- 7= Sensor channel 4
- 8= Sensor channel 5
- 9= Sensor channel 6
- 10= Digital In 1
- 11= Digital In 2
- 12= Digital In 3
- 13= Digital In 4

Description: Latch source which triggers aquiring one value.

16.4.8.2 Get_ADCLatchSource

Get the latch source which triggers aquiring one value.

Attention! This command is only allowed if IF2008E extension card is installed

Parameter: int SP_ADCNumber

SP_ADCNumber

Direction: Down

Valid values:

- 10= ADC 1
- 11= ADC 2

Description: Number of the ADC (Analog/Digital converter) to parametrize.

16.4. Commands for PCICard_IF2008

Parameter: int SA_ADCLatchSource SA_ADCLatchSource
Direction: Up
Valid values:
 0= never (locked)
 1= Timer 1
 2= Timer 2
 3= Timer 3
 4= Sensor channel 1
 5= Sensor channel 2
 6= Sensor channel 3
 7= Sensor channel 4
 8= Sensor channel 5
 9= Sensor channel 6
 10= Digital In 1
 11= Digital In 2
 12= Digital In 3
 13= Digital In 4
Description: Latch source which triggers acquiring one value.

16.4.8.3 Get_ADCValue

Get the value of the ADC.

Attention! This command is only allowed if IF2008E extension card is installed

Parameter: int SP_ADCNumber SP_ADCNumber
Direction: Down
Valid values:
 10= ADC 1
 11= ADC 2
Description: Number of the ADC to return value.

Parameter: int SA_ADCRawValue SA_ADCRawValue
Direction: Up
Valid values:
Minimum: 0
Maximum: 65535
Description: Raw value of ADC.

Parameter: double SA_ADCScaledValue SA_ADCScaledValue
Direction: Up
Valid values:
Minimum: 0, -5 or -10, depending on analog range
Maximum: 5 or 10, depending on analog range
Unit: V
Description: Scaled value of ADC.
 The range depends on IP_ADC1Range or IP_ADC2Range (at command Use_Defaults).

16.5 Commands for CSP2008

See sensor manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [RS232](#) (additional, e.g. IF2001_USB (RS422) and RS232 high level interface).
- [IF2004](#) (native).
- [TCP/IP](#) (native).
- [IF2004_USB](#) (native).
- [IF2008](#) (native).

You have to call [Use_Defaults](#) manually to allow MEDAQLib timeout checking, to calculate datarate and to interpret and scale data.

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from sensor.
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command [Use_Defaults](#)).

The values of one data frame are filled in the arrays one after another. Each array always starts with a new data frame.

The CSP2008 does not act as an interface in this case. You do not have direct access to the sensors connected to CSP2008.

16.5.1 Parameter management

16.5.1.1 Load_Parameters (LOADSETUP)

Load actual parameters into controller RAM. There can be loaded several settings from different locations. So it is easy to switch to another setting.

Parameter: int SP_ParameterSet

SP_ParameterSet

Direction: Down

Valid values:

Minimum: 1

Maximum: 20

Description: Location from where the settings should be loaded.

16.5.1.2 Save_Parameters (STORESETUP)

Save actual parameters at controller. There can be saved several settings on different locations. So it is easy to switch to another setting.

Parameter: int SP_ParameterSet

SP_ParameterSet

Direction: Down

Valid values:

Minimum: 1

Maximum: 20

Description: Location to save the settings.

16.5.2 Mastering

16.5.2.1 Set_MasterValue (SWMASTER)

Set the master value.

16.5.2.2 Reset_MasterValue (SWREMASTER)

Reset the master value.

16.5.3 Software trigger

16.5.3.1 Set_SoftwareTriggerEnable (SWTRIGGERENABLE)

Enable/Disable the software trigger mode.

Parameter: int SP_SoftwareTriggerEnable

SP_SoftwareTriggerEnable

Direction: Down

Valid values:

0= Off

1= On

Description: Software trigger mode.

16.5.3.2 Get_SoftwareTriggerEnable (SWTRIGGERENABLE)

Get the software trigger mode.

Parameter: int SA_SoftwareTriggerEnable

SA_SoftwareTriggerEnable

Direction: Up

Valid values:

0= Off

1= On

Description: Software trigger mode.

16.5.3.3 Set_SoftwareTrigger (SWTRIGGER)

Set a software trigger.

Parameter: int SP_SoftwareTrigger

SP_SoftwareTrigger

Direction: Down

Valid values:

0= Off

1= On

Description: Software trigger.

16.5.3.4 Get_SoftwareTrigger (SWTRIGGER)

Get the software trigger.

Parameter: int SA_SoftwareTrigger SA_SoftwareTrigger
Direction: Up
Valid values:
 0= Off
 1= On
Description: Software trigger.

16.5.4 Special commands

16.5.4.1 Use_Defaults

This command tells the driver to use default values to operate with sensor data. If some parameters are not specified they are not changed. The sensor is not affected by this command.

Parameter: double IP_Samplerate IP_Samplerate
Direction: Down
Valid values:
Minimum: 0
Maximum: 1.79769e+308 (DBL_MAX)
Unit: Hz
Description: Tells the driver the sampling rate of the controller.

Parameter: double IP_Datarate IP_Datarate
Direction: Down
Valid values:
Minimum: 0
Maximum: 1.79769e+308 (DBL_MAX)
Unit: Hz
Description: Tells the driver the datarate (output rate) of the controller. It is used for timeout checking. If it is zero, no timeout check is performed.

Parameter: double IP_RangeMin IP_RangeMin
Direction: Down
Valid values:
Minimum: -1.79769e+308 (-DBL_MAX)
Maximum: 1.79769e+308 (DBL_MAX)
Unit: Either mm or μm or any other
Description: Tells the driver the minimum range of sensor. It is used to scale the raw sensor values into mm or μm . If IP_RangeMin and IP_RangeMax is zero, no scaling is done.

Parameter: double IP_RangeMax IP_RangeMax
Direction: Down
Valid values:
Minimum: -1.79769e+308 (-DBL_MAX)
Maximum: 1.79769e+308 (DBL_MAX)
Unit: Either mm or μm or any other
Description: Tells the driver the maximum range of sensor. It is used to scale the raw sensor values into mm or μm . If IP_RangeMin and IP_RangeMax is zero, no scaling is done.

16.5.4.2 Get_DrvSetting

Returns the current settings of the driver used for operating with sensor data. It is the opposite of Use_Defaults. The sensor is not affected by this command.

Parameter: double IA_Samplerate	IA_Samplerate
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 1.79769e+308 (DBL_MAX)	
Unit: Hz	
Description: The sampling rate of the controller used by driver. 0 means the samplerate is unknown.	
Parameter: double IA_Datarate	IA_Datarate
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 1.79769e+308 (DBL_MAX)	
Unit: Hz	
Description: The datarate (output rate) of the controller used by driver to check timeout from sensor. 0 means the datarate is unknown.	
Parameter: double IA_RangeMin	IA_RangeMin
Direction: Up	
Valid values:	
Minimum: -1.79769e+308 (-DBL_MAX)	
Maximum: 1.79769e+308 (DBL_MAX)	
Unit: Either mm or μm or any other	
Description: Minimum range of sensor assumed by the driver	
Parameter: double IA_RangeMax	IA_RangeMax
Direction: Up	
Valid values:	
Minimum: -1.79769e+308 (-DBL_MAX)	
Maximum: 1.79769e+308 (DBL_MAX)	
Unit: Either mm or μm or any other	
Description: Maximum range of sensor assumed by the driver	

16.6 Commands for C-Box

See C-Box manual for detailed description of sensor commands.

This sensor supports following interfaces:

- [RS232](#) (additional, e.g. IF2001_USB (RS422) and RS232 high level interface).
- [IF2004](#) (native).
- [TCP/IP](#) (native).
- [IF2004_USB](#) (native).
- [IF2008](#) (native).

- [WinUSB](#) (native).

If first bit of [IP_AutomaticMode](#) is set (1), MEDAQLib calls automatically sensor command [Get_AllParameters](#) (SP_Additional= 1) after [OpenSensor](#).

Otherwise, you have to call it manually to allow MEDAQLib timeout checking, to calculate datarate, to interpret and scale data and to assign values. If second bit of [IP_AutomaticMode](#) is set (2), MEDAQLib calls optionally sensor command [Set_DataOutInterface](#), [Get_LaserPower1/Get_LaserPower2](#) and optionally [Set_LaserPower1/Set_LaserPower2](#) for attached sensors at [OpenSensor](#).

At open, MEDAQLib calls [Get_ScanStatus](#) periodically up to 20 seconds until C-Box has finished scanning.

Meaning of raw and scaled values (function [Poll](#) and [TransferData](#)):

- Raw values are as it comes directly from C-Box and the sensors behind.
- Scaled values are scaled using sensor range (if known by MEDAQLib, use sensor command [Get_AllParameters](#) (SP_Additional= 1)).

The values of one data frame are filled in the arrays one after another. Each array always starts with a new data frame.

16.6.1 General commands

16.6.1.1 General

16.6.1.1.1 Get_Info (GETINFO)

Retrieve information about the controller.

Parameter: String SA_Sensor SA_Sensor
Direction: Up
Description: Name of the controller.

Parameter: String SA_SerialNumber SA_SerialNumber
Direction: Up
Valid values:
 Numeric value
Description: Serial number of the controller.

Parameter: String SA_Option SA_Option
Direction: Up
Valid values:
 Numeric value
Description: Option of the controller.

Parameter: String SA_ArticleNumber SA_ArticleNumber
Direction: Up
Valid values:
 Numeric value
Description: Article number of the controller.

16.6. Commands for C-Box

Parameter: String SA_MacAddress	SA_MacAddress
Direction: Up	
Valid values:	
Valid MAC address in form of xx-xx-xx-xx-xx-xx	
Description: MAC address (low level ethernet address) of the controller.	
Parameter: String SA_Softwareversion	SA_Softwareversion
Direction: Up	
Description: Software version of firmware in the controller.	
Parameter: String SA_FPGAVersion	SA_FPGAVersion
Direction: Up	
Description: FPGA version in the controller.	
Parameter: String SA_WebpageVersion	SA_WebpageVersion
Direction: Up	
Description: Version of the Web pages in the controller.	

16.6.1.1.2 Set_SyncMode (SYNC)

Set the synchronisation mode.

Parameter: int SP_SyncMode	SP_SyncMode
Direction: Down	
Valid values:	
0= None	
1= Internal	
2= External	
Description: Synchronisation mode.	
Parameter: int SP_SyncLogic	SP_SyncLogic
Direction: Down	
Valid values:	
0= Low level logic (LLL)	
1= High level logic (HLL)	
Description: Logic level of external input.	

16.6.1.1.3 Get_SyncMode (SYNC)

Get the synchronisation mode.

Parameter: int SA_SyncMode	SA_SyncMode
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Internal	
2= External	
Description: Synchronisation mode.	

16.6. Commands for C-Box

Parameter: int SA_SyncLogic SA_SyncLogic
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = Low level logic (LLL)
 1 = High level logic (HLL)
Description: Logic level of external input.

16.6.1.1.4 Reset_Boot (RESET)

Resets the controller.

Parameter: int SP_AllDevices SP_AllDevices
Direction: Down
Valid values:
 0 = Only C-Box
 1 = C-Box and plugged sensors (ALL)
Description: Reset only C-Box or plugged sensors additionally.

16.6.1.1.5 Get_ScanStatus (SCANSTATUS)

Ask the controller if scanning for sensors is active.

Parameter: int SA_ScanStatus SA_ScanStatus
Direction: Up
Valid values:
 0 = No scan in progress (READY)
 1 = Active
Description: Scan state

16.6.1.1.6 Get_AllParameters (PRINT)

Get all parameters from controller.

If SP_Additional is 1 there also may be contained the answer parameters of [Get_Info1](#) and [Get_Info2](#).

Parameter: int SP_Additional SP_Additional
Direction: Down
Valid values:
 0 = No
 1 = Yes
Description: If set, additional information about controller and sensors are output.

Parameter: int SA_SensorBaudrate	SA_SensorBaudrate
Direction: Up	
Valid values:	
8000000	
4000000	
3500000	
3000000	
2500000	
2000000	
1500000	
921600	
691200	
460800	
230400	
115200	
9600	
Unit: Baud	
Description: Baudrate of controller.	
Parameter: int SA_DHCPEnabled	SA_DHCPEnabled
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = FALSE	
1 = TRUE	
Description: Get settings if controller should use a static IP address ask for IP at DHCP server (dynamic IP address).	
Parameter: String SA_Address	SA_Address
Direction: Up	
Valid values:	
Valid IP address in form of xxx.xxx.xxx.xxx	
Description: IP address of the controller. If DHCP is enabled it returns the currently assigned IP address.	
Parameter: String SA_SubnetMask	SA_SubnetMask
Direction: Up	
Valid values:	
Valid network mask (e.g. 255.255.255.0 for a Class C network)	
Description: Network mask of the controller. If DHCP is enabled it returns the currently assigned network mask.	
Parameter: String SA_Gateway	SA_Gateway
Direction: Up	
Valid values:	
Valid IP address of default gateway in form of xxx.xxx.xxx.xxx	
Description: Address of the default gateway. If DHCP is enabled it returns the currently assigned default gateway.	
Parameter: int SA_Protocol	SA_Protocol
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = TCP server (SERVER/TCP)	
Description: Only one protocol is supported.	

Parameter: int SA_Port	SA_Port
Direction: Up	
Valid values:	
Minimum: 1024	
Maximum: 65535	
Description: Port to send data to or to listen for incoming requests.	
Parameter: int SA_LaserPower1	SA_LaserPower1
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Off	
1 = On	
Description: Laser power.	
Parameter: int SA_LaserPower2	SA_LaserPower2
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Off	
1 = On	
Description: Laser power.	
Parameter: int SA_MeasureMode	SA_MeasureMode
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Value of sensor 1 (SENSOR1VALUE)	
1 = Thickness between sensor 1 and sensor 2 (SENSOR12THICK)	
2 = Step from sensor 1 to sensor 2 (SENSOR12STEP)	
Description: Measure mode.	
Parameter: double SA_Measrate	SA_Measrate
Direction: Up	
Valid values:	
Set_SyncMode is None: 0.4 ... 80.0	
Set_SyncMode is Internal: Depending on connected sensor.	
Unit: kHz	
Description: Samplerate of measurement.	
Parameter: int SA_SyncMode	SA_SyncMode
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = None	
1 = Internal	
2 = External	
Description: Synchronisation mode.	
Parameter: int SA_SyncLogic	SA_SyncLogic
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Low level logic (LLL)	
1 = High level logic (HLL)	
Description: Logic level of external input.	

Parameter: int SA_TriggerMode	SA_TriggerMode
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Edge	
2= Level (PULSE)	
3= Software	
Description: Trigger mode.	
Parameter: int SA_TriggerLevel	SA_TriggerLevel
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= High	
1= Low	
Description: Trigger level.	
Parameter: int SA_TriggerLogic	SA_TriggerLogic
Direction: Up	
Valid values:	
0= Low level logic (LLL)	
1= High level logic (HLL)	
Description: Logic level of external input.	
Parameter: int SA_TriggerCount	SA_TriggerCount
Direction: Up	
Valid values:	
Minimum: 0	
Maximum: 16383	
Description: Number of values to measure. 0 means no trigger, 16383 means endless measurement.	
Parameter: int SA_AveragingType	SA_AveragingType
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Moving average (MOVING)	
2= Recursive averaging (RECURSIVE)	
3= Median	
Description: Averaging type.	
Parameter: int SA_MovingCount	SA_MovingCount
Direction: Up	
Valid values:	
2	
4	
8	
16	
32	
64	
128	
256	
512	
Description: Number of value for the averaging window. This parameter is only available at moving average.	

Parameter: int SA_RecursiveCount	SA_RecursiveCount
Direction: Up	
Valid values:	
Minimum: 2, 4, 8, ...,	
Maximum: 32768	
Description: Number of values for recursive averaging. This parameter is only available at recursive average.	
Parameter: int SA_MedianCount	SA_MedianCount
Direction: Up	
Valid values:	
3	
5	
7	
9	
Description: Number of values to build median. This parameter is only available at median.	
Parameter: int SA_AveragingType1	SA_AveragingType1
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = None	
1 = Moving average (MOVING)	
2 = Recursive averaging (RECURSIVE)	
3 = Median	
Description: Averaging type.	
Parameter: int SA_MovingCount1	SA_MovingCount1
Direction: Up	
Valid values:	
Depending on the connected sensor	
Description: Number of value for the averaging window. This parameter is only available at moving average.	
Parameter: int SA_RecursiveCount1	SA_RecursiveCount1
Direction: Up	
Valid values:	
Depending on the connected sensor	
Description: Number of values for recursive averaging. This parameter is only available at recursive average.	
Parameter: int SA_MedianCount1	SA_MedianCount1
Direction: Up	
Valid values:	
Depending on the connected sensor	
Description: Number of values to build median. This parameter is only available at median.	
Parameter: int SA_AveragingType2	SA_AveragingType2
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = None	
1 = Moving average (MOVING)	
2 = Recursive averaging (RECURSIVE)	
3 = Median	
Description: Averaging type.	

Parameter: int SA_MovingCount2	SA_MovingCount2
Direction: Up	
Valid values:	
Depending on the connected sensor	
Description:	Number of value for the averaging window. This parameter is only available at moving average.
Parameter: int SA_RecursiveCount2	SA_RecursiveCount2
Direction: Up	
Valid values:	
Depending on the connected sensor	
Description:	Number of values for recursive averaging. This parameter is only available at recursive average.
Parameter: int SA_MedianCount2	SA_MedianCount2
Direction: Up	
Valid values:	
Depending on the connected sensor	
Description:	Number of values to build median. This parameter is only available at median.
Parameter: int SA_Master	SA_Master
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= no (NONE)	
1= yes (MASTER=	
Description:	Specifies if mastering is active.
Parameter: double SA_MasterValue	SA_MasterValue
Direction: Up	
Valid values:	
Minimum: -1024.0	
Maximum: +1024.0	
Unit: mm	
Description:	Master value
Parameter: int SA_DataOutInterface	SA_DataOutInterface
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= RS422	
2= Ethernet	
3= HTTP	
5= USB	
Description:	Active interface for data output.
Parameter: int SA_Resampling	SA_Resampling
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 1000	
Description:	Resampling value.

16.6. Commands for C-Box

Parameter: int SA_ResampleAnalog	SA_ResampleAnalog
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Analog output is resampled.	
Parameter: int SA_ResampleRS422	SA_ResampleRS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: RS422 output is resampled.	
Parameter: int SA_ResampleUSB	SA_ResampleUSB
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Output on USB is resampled.	
Parameter: int SA_ResampleEthernet	SA_ResampleEthernet
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Output over ethernet is resampled.	
Parameter: int SA_OutputScaleMode	SA_OutputScaleMode
Direction: Up	
Valid values:	
0= Standard	
1= Two point (TWOPOINT)	
Description: Output scale mode.	
Parameter: double SA_OutputMinValue	SA_OutputMinValue
Direction: Up	
Valid values:	
Minimum: -1024.0	
Maximum: 1024.0	
Unit: mm	
Description: Lowest possible value to transmit over RS422 or USB.	
Parameter: double SA_OutputMaxValue	SA_OutputMaxValue
Direction: Up	
Valid values:	
Minimum: -1024.0	
Maximum: 1024.0	
Unit: mm	
Description: Highest possible value to transmit over RS422 or USB.	

Parameter: int SA_HoldLastValid	SA_HoldLastValid
Direction: Up	
Valid values:	
Minimum: -1	
Maximum: 1024	
Description: Values to replace by last valid value. -1 means no value to hold, 0 means never output an error value (always hold last valid value).	
Parameter: int SA_OutputAdditional	SA_OutputAdditional
Direction: Up	
Valid values:	
0= C-Box counter (C-BOXCOUNTER)	
1= C-Box timestamp (C-BOXTIMESTAMP)	
2= C-Box trigger input (TRG-IN)	
Description: Specify which additional value is transmitted.	
Parameter: int SA_OutputSensor1Value_RS422	SA_OutputSensor1Value_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 1 measure value is transmitted.	
Parameter: int SA_OutputSensor1Additional_RS422	SA_OutputSen- sor1Additional_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 1 additional value is transmitted.	
Parameter: int SA_OutputSensor2Value_RS422	SA_OutputSensor2Value_- RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 2 measure value is transmitted.	
Parameter: int SA_OutputSensor2Additional_RS422	SA_OutputSen- sor2Additional_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 2 additional value is transmitted.	
Parameter: int SA_OutputC-BoxValue_RS422	SA_OutputC-BoxValue_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if C-Box measure value is transmitted.	
Parameter: int SA_OutputC-BoxAdditional_RS422	SA_OutputC- BoxAdditional_RS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if C-Box additional value is transmitted.	

16.6. Commands for C-Box

Parameter: int SA_OutputSensor1Value_USB	SA_OutputSensor1Value_USB
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 1 measure value is transmitted.	
Parameter: int SA_OutputSensor1Additional_USB	SA_OutputSen-
Direction: Up	sor1Additional_USB
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 1 additional value is transmitted.	
Parameter: int SA_OutputSensor2Value_USB	SA_OutputSensor2Value_USB
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 2 measure value is transmitted.	
Parameter: int SA_OutputSensor2Additional_USB	SA_OutputSen-
Direction: Up	sor2Additional_USB
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 2 additional value is transmitted.	
Parameter: int SA_OutputC-BoxValue_USB	SA_OutputC-BoxValue_USB
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if C-Box measure value is transmitted.	
Parameter: int SA_OutputC-BoxAdditional_USB	SA_OutputC-
Direction: Up	BoxAdditional_USB
Valid values:	
0= no	
1= yes	
Description: Specify if C-Box additional value is transmitted.	
Parameter: int SA_OutputSensor1Value_ETH	SA_OutputSensor1Value_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 1 measure value is transmitted.	
Parameter: int SA_OutputSensor1Additional_ETH	SA_OutputSen-
Direction: Up	sor1Additional_ETH
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 1 additional value is transmitted.	

Parameter: int SA_OutputSensor2Value_ETH	SA_OutputSensor2Value_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 2 measure value is transmitted.	
Parameter: int SA_OutputSensor2Additional_ETH	SA_OutputSen-
Direction: Up	sor2Additional_ETH
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 2 additional value is transmitted.	
Parameter: int SA_OutputC-BoxValue_ETH	SA_OutputC-BoxValue_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if C-Box measure value is transmitted.	
Parameter: int SA_OutputC-BoxAdditional_ETH	SA_OutputC-
Direction: Up	BoxAdditional_ETH
Valid values:	
0= no	
1= yes	
Description: Specify if C-Box additional value is transmitted.	
Parameter: int SA_AnalogOutput	SA_AnalogOutput
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= Fixed	
1= Sensor 1 value (SENSOR1VALUE)	
2= Sensor 2 value (SENSOR2VALUE)	
3= C-Box value (C-BOXVALUE)	
Description: Data to be used for analog output.	
Parameter: double SA_AnalogValue	SA_AnalogValue
Direction: Up	
Valid values:	
Minimum: 0, -5, -10 or 4, depending on analog range	
Maximum: 5, 10 or 20, depending on analog range	
Unit: V or mA, depending on analog range	
Description: Fixed output value for analog output.	
Parameter: int SA_AnalogRange	SA_AnalogRange
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= 0 - 5V	
2= 0 - 10V	
3= -5 - 5V	
4= -10 - 10V	
5= 4 - 20mA	
Description: Analog output range.	

Parameter: int SA_AnalogScaleMode	SA_AnalogScaleMode
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = Standard	
1 = Two point (TWOPOINT)	
Description: Analog scale mode.	
Parameter: double SA_MinValue	SA_MinValue
Direction: Up	
Valid values:	
Minimum: -3.4028e+38 (-FLT_MAX)	
Maximum: 3.4028e+38 (FLT_MAX)	
Unit: mm	
Description: Value which represents lowest voltage/current (at two point scaling).	
Parameter: double SA_MaxValue	SA_MaxValue
Direction: Up	
Valid values:	
Minimum: -3.4028e+38 (-FLT_MAX)	
Maximum: 3.4028e+38 (FLT_MAX)	
Unit: mm	
Description: Value which represents highest voltage/current (at two point scaling).	
Parameter: int SA_ApplicationLanguage	SA_ApplicationLanguage
Direction: Up	
Valid values:	
0 = Browser	
1 = English	
2 = German	
Description: Language of web interface.	
Parameter: String SA_Sensor	SA_Sensor
Direction: Up	
Description: Name of the controller.	
Parameter: String SA_SerialNumber	SA_SerialNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Serial number of the controller.	
Parameter: String SA_Option	SA_Option
Direction: Up	
Valid values:	
Numeric value	
Description: Option of the controller.	
Parameter: String SA_ArticleNumber	SA_ArticleNumber
Direction: Up	
Valid values:	
Numeric value	
Description: Article number of the controller.	

16.6. Commands for C-Box

Parameter: String SA_MacAddress	SA_MacAddress
Direction: Up	
Valid values:	
Valid MAC address in form of xx-xx-xx-xx-xx-xx	
Description:	MAC address (low level ethernet address) of the controller.
Parameter: String SA_Softwareversion	SA_Softwareversion
Direction: Up	
Description:	Software version of firmware in the controller.
Parameter: String SA_FPGAVersion	SA_FPGAVersion
Direction: Up	
Description:	FPGA version in the controller.
Parameter: String SA_WebpageVersion	SA_WebpageVersion
Direction: Up	
Description:	Version of the Web pages in the controller.

16.6.1.2 Triggering

16.6.1.2.1 Set_TriggerMode (TRIGGER)

Set the trigger mode.

Parameter: int SP_TriggerMode	SP_TriggerMode
Direction: Down	
Valid values:	
0= None	
1= Edge	
2= Level (PULSE)	
3= Software	
Description:	Trigger mode.

16.6.1.2.2 Get_TriggerMode (TRIGGER)

Get the active trigger mode.

Parameter: int SA_TriggerMode	SA_TriggerMode
Direction: Up	
Valid values:	
-1= Unknown parameter value from sensor	
0= None	
1= Edge	
2= Level (PULSE)	
3= Software	
Description:	Trigger mode.

16.6.1.2.3 Set_TriggerLevel (TRIGGERLEVEL)

Set the trigger level.

Parameter: int SP_TriggerLevel SP_TriggerLevel

Direction: Down

Valid values:

0= High

1= Low

Description: Trigger level.

Parameter: int SP_TriggerLogic SP_TriggerLogic

Direction: Down

Valid values:

0= Low level logic (LLL)

1= High level logic (HLL)

Description: Logic level of external input.

16.6.1.2.4 Get_TriggerLevel (TRIGGERLEVEL)

Get the active trigger level.

Parameter: int SA_TriggerLevel SA_TriggerLevel

Direction: Up

Valid values:

-1= Unknown parameter value from sensor

0= High

1= Low

Description: Trigger level.

Parameter: int SA_TriggerLogic SA_TriggerLogic

Direction: Up

Valid values:

0= Low level logic (LLL)

1= High level logic (HLL)

Description: Logic level of external input.

16.6.1.2.5 Set_TriggerCount (TRIGGERCOUNT)

Set the number of values to measure at trigger.

Parameter: int SP_TriggerCount SP_TriggerCount

Direction: Down

Valid values:

Minimum: 0

Maximum: 16383

Description: Number of values to measure. 0 means no trigger, 16383 means endless measurement.

16.6.1.2.6 Get_TriggerCount (TRIGGERCOUNT)

Get the number of values to measure at trigger.

Parameter: int SA_TriggerCount SA_TriggerCount
Direction: Up
Valid values:
Minimum: 0
Maximum: 16383
Description: Number of values to measure. 0 means no trigger, 16383 means endless measurement.

16.6.1.2.7 Software_Trigger (TRIGGERSW)

Execute a software trigger.

16.6.1.3 Interfaces

16.6.1.3.1 Set_IPConfiguration (IPCONFIG)

Set the IP configuration at controller.

Parameter: int SP_DHCPEnabled SP_DHCPEnabled
Direction: Down
Valid values:
 0 = FALSE
 1 = TRUE
Description: Specify if controller should use a static IP address or ask for IP at DHCP server (dynamic IP address).

Parameter: String SP_Address SP_Address
Direction: Down
Valid values:
 Valid IP address in form of xxx.xxx.xxx.xxx
Description: IP address of the controller. This parameter is only evaluated on static IP assignment.

Parameter: String SP_SubnetMask SP_SubnetMask
Direction: Down
Valid values:
 Valid network mask (e.g. 255.255.255.0 for a Class C network)
Description: Network mask of the controller. This parameter is only evaluated on static IP assignment.

Parameter: String SP_Gateway SP_Gateway
Direction: Down
Valid values:
 Valid IP address of default gateway in form of xxx.xxx.xxx.xxx
Description: The default gateway must be specified if the controller should communicate with peers in foreign subnets. This parameter is only evaluated on static IP assignment.

16.6.1.3.2 Get_IPConfiguration (IPCONFIG)

Get the IP configuration at controller.

Parameter: int SA_DHCPEnabled SA_DHCPEnabled

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= FALSE
- 1= TRUE

Description: Get settings if controller should use a static IP address ask for IP at DHCP server (dynamic IP address).

Parameter: String SA_Address SA_Address

Direction: Up

Valid values:

Valid IP address in form of xxx.xxx.xxx.xxx

Description: IP address of the controller. If DHCP is enabled it returns the currently assigned IP address.

Parameter: String SA_SubnetMask SA_SubnetMask

Direction: Up

Valid values:

Valid network mask (e.g. 255.255.255.0 for a Class C network)

Description: Network mask of the controller. If DHCP is enabled it returns the currently assigned network mask.

Parameter: String SA_Gateway SA_Gateway

Direction: Up

Valid values:

Valid IP address of default gateway in form of xxx.xxx.xxx.xxx

Description: Address of the default gateway. If DHCP is enabled it returns the currently assigned default gateway.

16.6.1.3.3 Set_IPDataTransferMode (MEATRANSFER)

Set IP protocol at controller.

Parameter: int SP_Protocol SP_Protocol

Direction: Down

Valid values:

0= TCP server (SERVER/TCP)

Description: Only one protocol is supported.

Parameter: int SP_Port SP_Port

Direction: Down

Valid values:

Minimum: 1024

Maximum: 65535

Description: Port to send data to or to listen for incoming requests.

16.6.1.3.4 Get_IPDataTransferMode (MEASTRANSFER)

Get IP protocol at controller.

Parameter: int SA_Protocol

SA_Protocol

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = TCP server (SERVER/TCP)

Description: Only one protocol is supported.

Parameter: int SA_Port

SA_Port

Direction: Up

Valid values:

Minimum: 1024

Maximum: 65535

Description: Port to send data to or to listen for incoming requests.

16.6.1.3.5 Set_Baudrate (BAUDRATE)

Set baudrate of controller for serial RS422 communication. After receiving the answer of the command, the baudrate of the local serial interface is changed, too.

Parameter: int SP_SensorBaudrate

SP_SensorBaudrate

Direction: Down

Valid values:

8000000

4000000

3500000

3000000

2500000

2000000

1500000

921600

691200

460800

230400

115200

9600

Unit: Baud

Description: Baudrate of controller.

16.6.1.3.6 Get_Baudrate (BAUDRATE)

Get baudrate of controller for serial RS422 communication.

Parameter: int SA_SensorBaudrate

SA_SensorBaudrate

Direction: Up

Valid values:

8000000

4000000

16.6. Commands for C-Box

3500000
 3000000
 2500000
 2000000
 1500000
 921600
 691200
 460800
 230400
 115200
 9600

Unit: Baud

Description: Baudrate of controller.

16.6.1.3.7 Set_AppLanguage (LANGUAGE)

Set language of web interface.

Parameter: int SP_ApplicationLanguage

SP_ApplicationLanguage

Direction: Down

Valid values:

0= Browser
 1= English
 2= German

Description: Language of web interface.

16.6.1.3.8 Get_AppLanguage (LANGUAGE)

Get language of web interface.

Parameter: int SA_ApplicationLanguage

SA_ApplicationLanguage

Direction: Up

Valid values:

0= Browser
 1= English
 2= German

Description: Language of web interface.

16.6.1.4 Sensors

16.6.1.4.1 Scan_Sensor1 (SCAN1)

Scan for a sensor at connector Sensor 1. The answer parameters are the same as at command [Get_Info1](#).

16.6.1.4.2 Scan_Sensor2 (SCAN2)

Scan for a sensor at connector Sensor 2. The answer parameters are the same as at command [Get_Info2](#).

16.6.1.4.3 Get_Info1 (GETINFO1)

Retrieve information about the sensor behind C-Box at connector 1. The answer parameters are depending on the connected sensor.

See the sensor command Get_Info at the specific sensor section. Use this parameter names with postfix '1', e.g. SA_SerialNumber1 for serial number of [SENSOR_ILD2300](#).

16.6.1.4.4 Get_Info2 (GETINFO2)

Retrieve information about the sensor behind C-Box at connector 2. The answer parameters are depending on the connected sensor.

See the sensor command Get_Info at the specific sensor section. Use this parameter names with postfix '2', e.g. SA_SerialNumber2 for serial number of [SENSOR_ILD2300](#).

16.6.1.4.5 Set_LaserPower1 (LASERPOW1)

Set the laser state (if supported by connected sensor).

A special line between C-Box and sensor is used to switch the laser at sensor.

Parameter: int SP_LaserPower1

SP_LaserPower1

Direction: Down

Valid values:

- 0= Off
- 1= On

Description: Laser power.

16.6.1.4.6 Get_LaserPower1 (LASERPOW1)

Get the laser state.

A special line between C-Box and sensor is used to switch the laser at sensor.

Parameter: int SA_LaserPower1

SA_LaserPower1

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Off
- 1= On

Description: Laser power.

16.6.1.4.7 Set_LaserPower2 (LASERPOW2)

Set the laser state (if supported by connected sensor).

A special line between C-Box and sensor is used to switch the laser at sensor.

Parameter: int SP_LaserPower2

SP_LaserPower2

Direction: Down

Valid values:

- 0= Off
- 1= On

Description: Laser power.

16.6.1.4.8 Get_LaserPower2 (LASERPOW2)

Get the laser state.

A special line between C-Box and sensor is used to switch the laser at sensor.

Parameter: int SA_LaserPower2

SA_LaserPower2

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = Off
- 1 = On

Description: Laser power.

16.6.1.4.9 Set_Averaging1 (AVERAGE1)

Set data averaging of sensor 1 behind C-Box.

Parameter: int SP_AveragingType1

SP_AveragingType1

Direction: Down

Valid values:

- 0 = None
- 1 = Moving average (MOVING)
- 2 = Recursive averaging (RECURSIVE)
- 3 = Median

Description: Averaging type.

Parameter: int SP_MovingCount1

SP_MovingCount1

Direction: Down

Valid values:

Depending on the connected sensor

Description: Number of value for the averaging window. This parameter is only used at moving average.

Parameter: int SP_RecursiveCount1

SP_RecursiveCount1

Direction: Down

Valid values:

Depending on the connected sensor

Description: Number of values for recursive averaging. This parameter is only used at recursive average.

Parameter: int SP_MedianCount1

SP_MedianCount1

Direction: Down

Valid values:

Depending on the connected sensor

Description: Number of values to build median. This parameter is only used at median.

16.6.1.4.10 Get_Averaging1 (AVERAGE1)

Get data averaging of sensor 1 behind C-Box.

Parameter: int SA_AveragingType1 SA_AveragingType1

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= Moving average (MOVING)
- 2= Recursive averaging (RECURSIVE)
- 3= Median

Description: Averaging type.

Parameter: int SA_MovingCount1 SA_MovingCount1

Direction: Up

Valid values:

Depending on the connected sensor

Description: Number of value for the averaging window. This parameter is only available at moving average.

Parameter: int SA_RecursiveCount1 SA_RecursiveCount1

Direction: Up

Valid values:

Depending on the connected sensor

Description: Number of values for recursive averaging. This parameter is only available at recursive average.

Parameter: int SA_MedianCount1 SA_MedianCount1

Direction: Up

Valid values:

Depending on the connected sensor

Description: Number of values to build median. This parameter is only available at median.

16.6.1.4.11 Set_Averaging2 (AVERAGE2)

Set data averaging of sensor 2 behind C-Box.

Parameter: int SP_AveragingType2 SP_AveragingType2

Direction: Down

Valid values:

- 0= None
- 1= Moving average (MOVING)
- 2= Recursive averaging (RECURSIVE)
- 3= Median

Description: Averaging type.

Parameter: int SP_MovingCount2 SP_MovingCount2

Direction: Down

Valid values:

Depending on the connected sensor

Description: Number of value for the averaging window. This parameter is only used at moving average.

16.6. Commands for C-Box

Parameter: int SP_RecursiveCount2	SP_RecursiveCount2
Direction: Down	
Valid values:	
Depending on the connected sensor	
Description:	Number of values for recursive averaging. This parameter is only used at recursive average.
Parameter: int SP_MedianCount2	SP_MedianCount2
Direction: Down	
Valid values:	
Depending on the connected sensor	
Description:	Number of values to build median. This parameter is only used at median.

16.6.1.4.12 Get_Averaging2 (AVERAGE2)

Get data averaging of sensor 2 behind C-Box.

Parameter: int SA_AveragingType2	SA_AveragingType2
Direction: Up	
Valid values:	
-1 = Unknown parameter value from sensor	
0 = None	
1 = Moving average (MOVING)	
2 = Recursive averaging (RECURSIVE)	
3 = Median	
Description:	Averaging type.
Parameter: int SA_MovingCount2	SA_MovingCount2
Direction: Up	
Valid values:	
Depending on the connected sensor	
Description:	Number of value for the averaging window. This parameter is only available at moving average.
Parameter: int SA_RecursiveCount2	SA_RecursiveCount2
Direction: Up	
Valid values:	
Depending on the connected sensor	
Description:	Number of values for recursive averaging. This parameter is only available at recursive average.
Parameter: int SA_MedianCount2	SA_MedianCount2
Direction: Up	
Valid values:	
Depending on the connected sensor	
Description:	Number of values to build median. This parameter is only available at median.

16.6.1.4.13 SensorCommand1 (TUNNEL1)

Send a command directly to the attached sensor.

Parameter: String SP_Command1	SP_Command1
Direction: Down	
Description:	Generic sensor command.

Parameter: String SA_Answer1 SA_Answer1

Direction: Up

Description: Answer from sensor.

16.6.1.4.14 SensorCommand2 (TUNNEL2)

Send a command directly to the attached sensor.

Parameter: String SP_Command2 SP_Command2

Direction: Down

Description: Generic sensor command.

Parameter: String SA_Answer2 SA_Answer2

Direction: Up

Description: Answer from sensor.

16.6.1.4.15 Get_SensorInstance

This internal command returns the MEDAQLib instance handle to the sensors connected to C-Box.

The handle returned can be used for any calling sensor command. [OpenSensor](#) or [CloseSensor](#) using this handle is not allowed. After calling [CloseSensor](#) or [ReleaseSensorInstance](#) using C-Box handle, this handle is no longer valid.

Parameter: int SP_Sensor SP_Sensor

Direction: Down

Valid values:

1 = First sensor

2 = Second sensor

Description: Number of the sensor.

Parameter: int SA_InstanceHandle SA_InstanceHandle

Direction: Up

Valid values:

Minimum: 1

Maximum: 2147483647 (INT_MAX)

Description: Instance number which now can be used to access sensor directly.

16.6.1.5 Parameter management

16.6.1.5.1 Save_Parameters (STORE)

Save actual parameters at controller. There can be saved several settings on different locations. So it is easy to switch to another setting.

Parameter: int SP_ParameterSet SP_ParameterSet

Direction: Down

Valid values:

Minimum: 1

Maximum: 8

Description: Location to save the settings.

16.6.1.5.2 Load_Parameters (READ)

Load actual parameters into controller RAM. There can be loaded several settings from different locations. So it is easy to switch to another setting.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_ParameterType

SP_ParameterType

Direction: Down

Valid values:

- 0= All settings (ALL)
- 1= Device settings (DEVICE)
- 2= Measurement settings (MEAS)

Description: Specifies which settings should be loaded.

Parameter: int SP_ParameterSet

SP_ParameterSet

Direction: Down

Valid values:

- Minimum: 1
- Maximum: 8

Description: Location from where the settings should be loaded.

16.6.1.5.3 Set_Default (SETDEFAULT)

Reset the controller to default settings.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_JustActualSetup

SP_JustActualSetup

Direction: Down

Valid values:

- 0= Reset all settings (ALL)
- 1= Just reset actual setting

Description: Specifies which settings should be resetted.

Parameter: int SP_KeepDevice

SP_KeepDevice

Direction: Down

Valid values:

- 0= Reset actual interface parameters
- 1= Keep device settings temporary (NODEVICE)

Description: Specifies if parameters should be hold temporary.

16.6.2 Measurement

16.6.2.1 General

16.6.2.1.1 Set_MeasureMode (MEASMODE)

Set the measure mode.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_MeasureMode

SP_MeasureMode

Direction: Down

Valid values:

- 0= Value of sensor 1 (SENSOR1VALUE)
- 1= Thickness between sensor 1 and sensor 2 (SENSOR12THICK)
- 2= Step from sensor 1 to sensor 2 (SENSOR12STEP)

Description: Measure mode.

16.6.2.1.2 Get_MeasureMode (MEASMODE)

Get the measure mode.

Parameter: int SA_MeasureMode SA_MeasureMode
Direction: Up
Valid values:
 -1 = Unknown parameter value from sensor
 0 = Value of sensor 1 (SENSOR1VALUE)
 1 = Thickness between sensor 1 and sensor 2 (SENSOR12THICK)
 2 = Step from sensor 1 to sensor 2 (SENSOR12STEP)
Description: Measure mode.

16.6.2.1.3 Set_Samplerate (MEASRATE)

Set the samplerate.

Parameter: double SP_Measrate SP_Measrate
Direction: Down
Valid values:
[Set_SyncMode](#) is None: 0.4 ... 80.0
[Set_SyncMode](#) is Internal: Depending on connected sensor.
Unit: kHz
Description: Samplerate of measurement.

16.6.2.1.4 Get_Samplerate (MEASRATE)

Get the samplerate.

Parameter: double SA_Measrate SA_Measrate
Direction: Up
Valid values:
[Set_SyncMode](#) is None: 0.4 ... 80.0
[Set_SyncMode](#) is Internal: Depending on connected sensor.
Unit: kHz
Description: Samplerate of measurement.

16.6.2.2 Measurement value processing

16.6.2.2.1 Set_Averaging (AVERAGE)

Set data averaging at controller.

Parameter: int SP_AveragingType SP_AveragingType
Direction: Down
Valid values:
 0 = None
 1 = Moving average (MOVING)
 2 = Recursive averaging (RECURSIVE)
 3 = Median
Description: Averaging type.

Parameter: int SP_MovingCount SP_MovingCount

Direction: Down

Valid values:

- 2
- 4
- 8
- 16
- 32
- 64
- 128
- 256
- 512

Description: Number of value for the averaging window. This parameter is only used at moving average.

Parameter: int SP_RecursiveCount SP_RecursiveCount

Direction: Down

Valid values:

- Minimum: 2, 4, 8, ...,
- Maximum: 32768

Description: Number of values for recursive averaging. This parameter is only used at recursive average.

Parameter: int SP_MedianCount SP_MedianCount

Direction: Down

Valid values:

- 3
- 5
- 7
- 9

Description: Number of values to build median. This parameter is only used at median.

16.6.2.2.2 Get_Averaging (AVERAGE)

Get data averaging at controller.

Parameter: int SA_AveragingType SA_AveragingType

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= Moving average (MOVING)
- 2= Recursive averaging (RECURSIVE)
- 3= Median

Description: Averaging type.

Parameter: int SA_MovingCount SA_MovingCount

Direction: Up

Valid values:

- 2
- 4
- 8
- 16
- 32
- 64
- 128
- 256
- 512

Description: Number of value for the averaging window. This parameter is only available at moving average.

Parameter: int SA_RecursiveCount SA_RecursiveCount

Direction: Up

Valid values:

- Minimum:** 2, 4, 8, ...,
- Maximum:** 32768

Description: Number of values for recursive averaging. This parameter is only available at recursive average.

Parameter: int SA_MedianCount SA_MedianCount

Direction: Up

Valid values:

- 3
- 5
- 7
- 9

Description: Number of values to build median. This parameter is only available at median.

16.6.2.2.3 Set_MasterValue (MASTERMV)

Set the master value.

Parameter: int SP_Master SP_Master

Direction: Down

Valid values:

- 0= no (NONE)
- 1= yes (MASTER=)

Description: Specifies if mastering should be done or resetted.

Parameter: double SP_MasterValue SP_MasterValue

Direction: Down

Valid values:

- Minimum:** -1024.0
- Maximum:** +1024.0

Unit: mm

Description: Master value

16.6.2.2.4 Get_MasterValue (MASTERMV)

Get the master value.

Parameter: int SA_Master

SA_Master

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0 = no (NONE)
- 1 = yes (MASTER=

Description: Specifies if mastering is active.

Parameter: double SA_MasterValue

SA_MasterValue

Direction: Up

Valid values:

- Minimum: -1024.0
- Maximum: +1024.0

Unit: mm

Description: Master value

16.6.3 Data output

16.6.3.1 General

16.6.3.1.1 Set_DataOutInterface (OUTPUT)

Set the active interface for data output.

If first bit of IP_AutomaticMode is set (1), [Get_AllParameters](#) (SP_Additional= 1) is called automatically after this command. Otherwise, you have to call it manually.

Parameter: int SP_DataOutInterface

SP_DataOutInterface

Direction: Down

Valid values:

- 0= None
- 1= RS422
- 2= Ethernet
- 3= HTTP
- 5= USB

Description: Active interface for data output.

16.6.3.1.2 Get_DataOutInterface (OUTPUT)

Get the active interface for data output.

Parameter: int SA_DataOutInterface

SA_DataOutInterface

Direction: Up

Valid values:

- 1 = Unknown parameter value from sensor
- 0= None
- 1= RS422
- 2= Ethernet
- 3= HTTP
- 5= USB

Description: Active interface for data output.

16.6.3.1.3 Set_Resampling (OUTREDUCE)

Set resampling to reduce output data.

Parameter: int SP_Resampling	SP_Resampling
Direction: Down	
Valid values:	
Minimum: 1	
Maximum: 1000	
Description: Resampling value.	
Parameter: int SP_ResampleAnalog	SP_ResampleAnalog
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if analog output should be resampled.	
Parameter: int SP_ResampleRS422	SP_ResampleRS422
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if RS422 output should be resampled.	
Parameter: int SP_ResampleUSB	SP_ResampleUSB
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if output on USB should be resampled.	
Parameter: int SP_ResampleEthernet	SP_ResampleEthernet
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if output over ethernet should be resampled.	

16.6.3.1.4 Get_Resampling (OUTREDUCE)

Get resampling for reducing output data.

Parameter: int SA_Resampling	SA_Resampling
Direction: Up	
Valid values:	
Minimum: 1	
Maximum: 1000	
Description: Resampling value.	

16.6. Commands for C-Box

Parameter: int SA_ResampleAnalog	SA_ResampleAnalog
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Analog output is resampled.	
Parameter: int SA_ResampleRS422	SA_ResampleRS422
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: RS422 output is resampled.	
Parameter: int SA_ResampleUSB	SA_ResampleUSB
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Output on USB is resampled.	
Parameter: int SA_ResampleEthernet	SA_ResampleEthernet
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Output over ethernet is resampled.	

16.6.3.1.5 Set_HoldLastValid (OUTHOLD)

Set the number of values to be replaced by last valid value instead of error values.

Parameter: int SP_HoldLastValid	SP_HoldLastValid
Direction: Down	
Valid values:	
Minimum: -1	
Maximum: 1024	
Description: Values to replace by last valid value. -1 means no value to hold, 0 means never output an error value (always hold last valid value).	

16.6.3.1.6 Get_HoldLastValid (OUTHOLD)

Get the number of values to be replaced by last valid value instead of error values.

Parameter: int SA_HoldLastValid	SA_HoldLastValid
Direction: Up	
Valid values:	
Minimum: -1	
Maximum: 1024	
Description: Values to replace by last valid value. -1 means no value to hold, 0 means never output an error value (always hold last valid value).	

16.6.3.1.7 Set_OutputScale_RS422_USB (OUTSCALE_RS422_USB)

Set the scaling factor for RS422 and USB output.

Parameter: int SP_OutputScaleMode	SP_OutputScaleMode
Direction: Down	
Valid values:	
0= Standard	
1= Two point (TWOPOINT)	
Description: Output scale mode.	
Parameter: double SP_OutputMinValue	SP_OutputMinValue
Direction: Down	
Valid values:	
Minimum: -1024.0	
Maximum: 1024.0	
Unit: mm	
Description: Lowest possible value to transmit over RS422 or USB.	
Parameter: double SP_OutputMaxValue	SP_OutputMaxValue
Direction: Down	
Valid values:	
Minimum: -1024.0	
Maximum: 1024.0	
Unit: mm	
Description: Highest possible value to transmit over RS422 or USB.	

16.6.3.1.8 Get_OutputScale_RS422_USB (OUTSCALE_RS422_USB)

Get the scaling factor for RS422 and USB output.

Parameter: int SA_OutputScaleMode	SA_OutputScaleMode
Direction: Up	
Valid values:	
0= Standard	
1= Two point (TWOPOINT)	
Description: Output scale mode.	
Parameter: double SA_OutputMinValue	SA_OutputMinValue
Direction: Up	
Valid values:	
Minimum: -1024.0	
Maximum: 1024.0	
Unit: mm	
Description: Lowest possible value to transmit over RS422 or USB.	
Parameter: double SA_OutputMaxValue	SA_OutputMaxValue
Direction: Up	
Valid values:	
Minimum: -1024.0	
Maximum: 1024.0	
Unit: mm	
Description: Highest possible value to transmit over RS422 or USB.	

16.6.3.2 Selected measurement values

16.6.3.2.1 Set_Output_RS422 (OUT_RS422)

Set the data to be output at RS422 interface.

Parameter: int SP_OutputSensor1Value_RS422

SP_OutputSensor1Value_-
RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if sensor 1 measure value is transmitted.

Parameter: int SP_OutputSensor1Additional_RS422

SP_OutputSen-
sor1Additional_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if sensor 1 additional value is transmitted.

Parameter: int SP_OutputSensor2Value_RS422

SP_OutputSensor2Value_-
RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if sensor 2 measure value is transmitted.

Parameter: int SP_OutputSensor2Additional_RS422

SP_OutputSen-
sor2Additional_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if sensor 2 additional value is transmitted.

Parameter: int SP_OutputC-BoxValue_RS422

SP_OutputC-BoxValue_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if C-Box measure value is transmitted.

Parameter: int SP_OutputC-BoxAdditional_RS422

SP_OutputC-
BoxAdditional_RS422

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if C-Box additional value is transmitted.

16.6.3.2.2 Get_Output_RS422 (OUT_RS422)

Get the data to be output at RS422 interface.

Parameter: int SA_OutputSensor1Value_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if sensor 1 measure value is transmitted.

SA_OutputSensor1Value_-
RS422

Parameter: int SA_OutputSensor1Additional_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if sensor 1 additional value is transmitted.

SA_OutputSen-
sor1Additional_RS422

Parameter: int SA_OutputSensor2Value_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if sensor 2 measure value is transmitted.

SA_OutputSensor2Value_-
RS422

Parameter: int SA_OutputSensor2Additional_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if sensor 2 additional value is transmitted.

SA_OutputSen-
sor2Additional_RS422

Parameter: int SA_OutputC-BoxValue_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if C-Box measure value is transmitted.

SA_OutputC-BoxValue_RS422

Parameter: int SA_OutputC-BoxAdditional_RS422

Direction: Up

Valid values:

0= no

1= yes

Description: Specify if C-Box additional value is transmitted.

SA_OutputC-
BoxAdditional_RS422

16.6.3.2.3 Set_Output_USB (OUT_USB)

Set the data to be output at USB interface.

Parameter: int SP_OutputSensor1Value_USB

SP_OutputSensor1Value_USB

Direction: Down

Valid values:

0= no

1= yes

Description: Specify if sensor 1 measure value is transmitted.

16.6. Commands for C-Box

Parameter: int SP_OutputSensor1Additional_USB	SP_OutputSensor1Additional_USB
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 1 additional value is transmitted.	
Parameter: int SP_OutputSensor2Value_USB	SP_OutputSensor2Value_USB
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 2 measure value is transmitted.	
Parameter: int SP_OutputSensor2Additional_USB	SP_OutputSensor2Additional_USB
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 2 additional value is transmitted.	
Parameter: int SP_OutputC-BoxValue_USB	SP_OutputC-BoxValue_USB
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if C-Box measure value is transmitted.	
Parameter: int SP_OutputC-BoxAdditional_USB	SP_OutputC-BoxAdditional_USB
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if C-Box additional value is transmitted.	

16.6.3.2.4 Get_Output_USB (OUT_USB)

Get the data to be output at USB interface.

Parameter: int SA_OutputSensor1Value_USB	SA_OutputSensor1Value_USB
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 1 measure value is transmitted.	
Parameter: int SA_OutputSensor1Additional_USB	SA_OutputSensor1Additional_USB
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 1 additional value is transmitted.	

16.6. Commands for C-Box

Parameter: int SA_OutputSensor2Value_USB	SA_OutputSensor2Value_USB
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 2 measure value is transmitted.	
Parameter: int SA_OutputSensor2Additional_USB	SA_OutputSen-
Direction: Up	sor2Additional_USB
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 2 additional value is transmitted.	
Parameter: int SA_OutputC-BoxValue_USB	SA_OutputC-BoxValue_USB
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if C-Box measure value is transmitted.	
Parameter: int SA_OutputC-BoxAdditional_USB	SA_OutputC-
Direction: Up	BoxAdditional_USB
Valid values:	
0= no	
1= yes	
Description: Specify if C-Box additional value is transmitted.	

16.6.3.2.5 Set_Output_ETH (OUT_ETH)

Set the data to be output at ethernet interface.

Parameter: int SP_OutputSensor1Value_ETH	SP_OutputSensor1Value_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 1 measure value is transmitted.	
Parameter: int SP_OutputSensor1Additional_ETH	SP_OutputSen-
Direction: Down	sor1Additional_ETH
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 1 additional value is transmitted.	
Parameter: int SP_OutputSensor2Value_ETH	SP_OutputSensor2Value_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 2 measure value is transmitted.	

16.6. Commands for C-Box

Parameter: int SP_OutputSensor2Additional_ETH	SP_OutputSensor2Additional_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 2 additional value is transmitted.	
Parameter: int SP_OutputC-BoxValue_ETH	SP_OutputC-BoxValue_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if C-Box measure value is transmitted.	
Parameter: int SP_OutputC-BoxAdditional_ETH	SP_OutputC-BoxAdditional_ETH
Direction: Down	
Valid values:	
0= no	
1= yes	
Description: Specify if C-Box additional value is transmitted.	
16.6.3.2.6 Get_Output_ETH (OUT_ETH)	
Get the data to be output at ethernet interface.	
Parameter: int SA_OutputSensor1Value_ETH	SA_OutputSensor1Value_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 1 measure value is transmitted.	
Parameter: int SA_OutputSensor1Additional_ETH	SA_OutputSensor1Additional_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 1 additional value is transmitted.	
Parameter: int SA_OutputSensor2Value_ETH	SA_OutputSensor2Value_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 2 measure value is transmitted.	
Parameter: int SA_OutputSensor2Additional_ETH	SA_OutputSensor2Additional_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if sensor 2 additional value is transmitted.	

16.6. Commands for C-Box

Parameter: int SA_OutputC-BoxValue_ETH	SA_OutputC-BoxValue_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if C-Box measure value is transmitted.	
Parameter: int SA_OutputC-BoxAdditional_ETH	SA_OutputC-BoxAdditional_ETH
Direction: Up	
Valid values:	
0= no	
1= yes	
Description: Specify if C-Box additional value is transmitted.	

16.6.3.2.7 Set_OutputAdditional (OUT_ADDITIONAL)

Set the additional data to be output.

Parameter: int SP_OutputAdditional	SP_OutputAdditional
Direction: Down	
Valid values:	
0= C-Box counter (C-BOXCOUNTER)	
1= C-Box timestamp (C-BOXTIMESTAMP)	
2= C-Box trigger input (TRG-IN)	
Description: Specify which additional value should be transmitted.	

16.6.3.2.8 Get_OutputAdditional (OUT_ADDITIONAL)

Get the additional data to be output.

Parameter: int SA_OutputAdditional	SA_OutputAdditional
Direction: Up	
Valid values:	
0= C-Box counter (C-BOXCOUNTER)	
1= C-Box timestamp (C-BOXTIMESTAMP)	
2= C-Box trigger input (TRG-IN)	
Description: Specify which additional value is transmitted.	

16.6.3.3 Analog output

16.6.3.3.1 Set_AnalogOutput (ANALOGOUT)

Set the data to be used for analog output.

Parameter: int SP_AnalogOutput	SP_AnalogOutput
Direction: Down	
Valid values:	
0= Fixed	
1= Sensor 1 value (SENSOR1VALUE)	
2= Sensor 2 value (SENSOR2VALUE)	
3= C-Box value (C-BOXVALUE)	
Description: Data to be used for analog output.	

Parameter: double SP_AnalogValue SP_AnalogValue
Direction: Down
Valid values:
Minimum: 0, -5, -10 or 4, depending on analog range
Maximum: 5, 10 or 20, depending on analog range
Unit: V or mA, depending on analog range
Description: Fixed output value for analog output.

16.6.3.3.2 Get_AnalogOutput (ANALOGOUT)

Get the data to be used for analog output.

Parameter: int SA_AnalogOutput SA_AnalogOutput
Direction: Up
Valid values:
 -1= Unknown parameter value from sensor
 0= Fixed
 1= Sensor 1 value (SENSOR1VALUE)
 2= Sensor 2 value (SENSOR2VALUE)
 3= C-Box value (C-BOXVALUE)
Description: Data to be used for analog output.

Parameter: double SA_AnalogValue SA_AnalogValue
Direction: Up
Valid values:
Minimum: 0, -5, -10 or 4, depending on analog range
Maximum: 5, 10 or 20, depending on analog range
Unit: V or mA, depending on analog range
Description: Fixed output value for analog output.

16.6.3.3.3 Set_AnalogRange (ANALOG RANGE)

Set the analog output range.

Parameter: int SP_AnalogRange SP_AnalogRange
Direction: Down
Valid values:
 0= None
 1= 0 - 5V
 2= 0 - 10V
 3= -5 - 5V
 4= -10 - 10V
 5= 4 - 20mA
Description: Analog output range.

16.6. Commands for C-Box

16.6.3.3.4 Get_AnalogRange (ANALOG RANGE)

Get the analog output range.

Parameter: int SA_AnalogRange

SA_AnalogRange

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= None
- 1= 0 - 5V
- 2= 0 - 10V
- 3= -5 - 5V
- 4= -10 - 10V
- 5= 4 - 20mA

Description: Analog output range.

16.6.3.3.5 Set_AnalogScale (ANALOG SCALE)

Set the scaling factor for analog output. If both parameters are zero, the standard scaling is used.

Parameter: int SP_AnalogScaleMode

SP_AnalogScaleMode

Direction: Down

Valid values:

- 0= Standard
- 1= Two point (TWOPOINT)

Description: Analog scale mode.

Parameter: double SP_MinValue

SP_MinValue

Direction: Down

Valid values:

- Minimum:** -1024.0
- Maximum:** 1024.0

Unit: mm

Description: Value which represents lowest voltage/current (at two point scaling).

Parameter: double SP_MaxValue

SP_MaxValue

Direction: Down

Valid values:

- Minimum:** -1024.0
- Maximum:** 1024.0

Unit: mm

Description: Value which represents highest voltage/current (at two point scaling).

16.6.3.3.6 Get_AnalogScale (ANALOG SCALE)

Get the scaling factor for analog output. If both parameters are zero, the standard scaling is used.

Parameter: int SA_AnalogScaleMode

SA_AnalogScaleMode

Direction: Up

Valid values:

- 1= Unknown parameter value from sensor
- 0= Standard
- 1= Two point (TWOPOINT)

Description: Analog scale mode.

16.6. Commands for C-Box

Parameter: double SA_MinValue	SA_MinValue
Direction: Up	
Valid values:	
Minimum: -1024.0	
Maximum: 1024.0	
Unit: mm	
Description: Value which represents lowest voltage/current (at two point scaling).	
Parameter: double SA_MaxValue	SA_MaxValue
Direction: Up	
Valid values:	
Minimum: -1024.0	
Maximum: 1024.0	
Unit: mm	
Description: Value which represents highest voltage/current (at two point scaling).	

16.6.4 Internal commands

16.6.4.1 Get_FirmwareVersion

Retrieve firmware version from controller.
This is an internal command. It should not be used by the customer.

Parameter: String CA_FirmwareVersion	CA_FirmwareVersion
Direction: Up	
Description: Firmware version	

16.6.4.2 Prepare_UpdateFirmware

Prepares a firmware update at controller.
This is an internal command. It should not be used by the customer.

Parameter: Binary data XP_FirmwareFile	XP_FirmwareFile
Direction: Down	
Description: Firmware file	
Parameter: String CA_FileName	CA_FileName
Direction: Up	
Description: Internal name of firmware file	
Parameter: String CA_Date	CA_Date
Direction: Up	
Description: Date of firmware file	
Parameter: String CA_ArticleNumber	CA_ArticleNumber
Direction: Up	
Description: Article number of destination device	
Parameter: String CA_SerialNumber	CA_SerialNumber
Direction: Up	
Description: Serial number of destination device	

16.6.4.3 Start_UpdateFirmware

Start firmware update at controller.

This is an internal command. It should not be used by the customer.

16.6.4.4 Get_UpdateFirmwareProgress

Update firmware version at controller.

Attention! This function can takes up to 1 minute. This is an internal command. It should not be used by the customer.

Parameter: double CA_Progress CA_Progress

Direction: Up

Valid values:

Minimum: 0.0

Maximum: 1.0

Description: Progress of firmware update

Parameter: String CA_Description CA_Description

Direction: Up

Description: Current state of firmware update.

Parameter: int CA_Result CA_Result

Direction: Up

Valid values:

0= Failed

1= Success

Description: Result of firmware update (only available at end of update).

Parameter: int CA_Finished CA_Finished

Direction: Up

Valid values:

-1= Firmware update is not prepared, call Prepare_UpdateFirmware first

0= No

1= Yes

Description: Tell if firmware update is in progress.

16.6.4.5 Generate_Firmware

This is an internal command. It should not be used by the customer.

Index

avail, 73
 CA_ArticleNumber, 923
 CA_Date, 923
 CA_Description, 924
 CA_FileName, 923
 CA_Finished, 924
 CA_FirmwareVersion, 923
 CA_Progress, 924
 CA_Result, 924
 CA_SerialNumber, 923
 cardInstance, 87, 89
 channelNumber, 87–90
 ClearAllParameters, 69
 CloseSensor, 70
 CP_AutomaticMode, 126
 CP_BaseLevelTimeout, 108
 CP_HTTPRetryCount, 253
 CP_InitAfterSensorCommand, 129
 CP_InterruptDataTransfer, 219, 278
 CP_SensorAnswerTimeout, 121, 129
 CP_ThreadPriority, 120
 CreateSensorInstance, 50
 CreateSensorInstByName, 51
 CreateSensorInstByNameU, 52

 DataAvail, 73
 deviceInstance, 87, 88, 91

 EnableLogging, 79
 enableLogging, 80, 82
 EnableLoggingU, 81
 errText, 76
 ExecSCmd, 92
 ExecSCmdGetDouble, 98
 ExecSCmdGetDoubleU, 99
 ExecSCmdGetInt, 97
 ExecSCmdGetIntU, 97
 ExecSCmdGetString, 99
 ExecSCmdGetStringU, 100
 ExecSCmdU, 92

 GetDLLVersion, 78
 GetDLLVersionU, 79
 GetError, 76
 GetErrorU, 76
 GetParameterBinary, 66
 GetParameterBinaryU, 67
 GetParameterDouble, 63
 GetParameterDoubleU, 64

 GetParameterDWORD_PTR, 62
 GetParameterDWORD_PTRU, 62
 GetParameterInt, 60
 GetParameterIntU, 61
 GetParameters, 67
 GetParameterString, 64
 GetParameterStringU, 65
 GetParametersU, 68

 IA_ADC1Range, 861
 IA_ADC2Range, 861
 IA_Baudrate, 114, 117
 IA_DataAvailEvent, 130
 IA_Datarate, 882
 IA_Encoder1CountFactor, 860
 IA_Encoder2CountFactor, 860
 IA_Found, 139
 IA_Index1..x, 126
 IA_MaxValuesPerFrame, 126
 IA_Progress, 139
 IA_Range, 694, 855
 IA_Range1..4, 726
 IA_Range1..8, 748
 IA_RangeMax, 882
 IA_RangeMin, 882
 IA_Raw_Datarate1..x, 127
 IA_Raw_Name1..x, 127
 IA_Raw_RangeMax1..x, 127
 IA_Raw_RangeMin1..x, 127
 IA_Raw_Unit1..x, 127
 IA_Samplerate, 882
 IA_Scaled_Datarate1..x, 127
 IA_Scaled_Name1..x, 127
 IA_Scaled_RangeMax1..x, 127
 IA_Scaled_RangeMin1..x, 127
 IA_Scaled_Unit1..x, 127
 IA_SettingsChanged, 128
 IA_Success, 151
 IA_ValuesPerFrame, 126
 instanceHandle, 53–74, 76, 79, 81, 84–
 100
 IP_ADC1Range, 860
 IP_ADC2Range, 860
 IP_AllowReserved, 150
 IP_AutoIPEnabled, 151
 IP_AutomaticMode, 102
 IP_Baudrate, 109, 111, 113, 116, 141–
 143, 147
 IP_BootPEnabled, 151
 IP_ByteSize, 110, 142

IP_CardInstance, 110, 115, 142, 147
IP_ChannelNumber, 111, 113, 115, 142, 143, 147
IP_ClearReceiveBuffer, 123
IP_ClearRingBuffer, 122
IP_ClearSendBuffer, 123
IP_DataPort, 119, 131
IP_Datarate, 881
IP_DeviceInstance, 112, 119, 143, 147
IP_DHCPEnabled, 151
IP_EnableLogging, 105, 124
IP_EncCountMode, 854
IP_EnclnvertTraceA_B, 855
IP_EnlatchSrc, 855
IP_EncLoadOnRef, 855
IP_Encoder1CountFactor, 859
IP_Encoder2CountFactor, 859
IP_EncSwapTraceA_B, 854
IP_EventOnAvailableValues, 129
IP_EventOnBufferFillsize, 130
IP_FindSimilarSensors, 137
IP_FixedErrorValue, 105
IP_Gateway, 150
IP_Index, 139
IP_Interface, 102, 141
IP_Interfaceldx, 141
IP_LogAppend, 106, 125
IP_LogFile, 106, 124
IP_LogFlush, 107, 125
IP_LogLevel, 106, 124
IP_LogSplitSize, 107, 125
IP_LogType, 105, 124
IP_MACAddress, 149
IP_MaxPacketSize, 107
IP_NewIPAddress, 150
IP_OldIPAddress, 149
IP_PacketDelay, 107
IP_Parity, 110, 114, 116, 142, 143, 147
IP_Password, 150
IP_Port, 108, 112, 141, 143
IP_PortDetection, 118, 131
IP_QuickScanRS485, 138
IP_Range, 693, 854
IP_Range1..4, 726
IP_Range1..8, 748
IP_RangeMax, 881
IP_RangeMin, 881
IP_RemoteAddr, 118, 144
IP_RemoteDataProtocol, 118, 130
IP_RemotePort, 118
IP_ResetFlag, 128
IP_RingBufferSize, 105
IP_Samplerate, 881
IP_ScaleErrorValues, 103
IP_ScanHWIF2004, 138
IP_ScanHWIF2004_USB, 138
IP_ScanHWIF2008, 138
IP_ScanHWRS232, 137
IP_ScanHWTCP/IP, 138
IP_ScanHWWinUSB, 138
IP_SensorAddress, 110
IP_SensorType, 136, 139, 148
IP_SensorTypeName, 140
IP_SerialNumber, 112, 143
IP_StaticIP, 150
IP_Stopbits, 109, 141
IP_SubnetMask, 150
IP_SyncMasterChannel, 111, 114, 116
IP_TimerResolution, 108, 359, 428, 631
IP_UsbReadBufCnt, 120
IP_UsbReadBufSize, 120, 359, 361
IP_UseGate, 111
IP_UUID, 149

len, 58, 59
location, 84, 85
logAppend, 81, 83
logFile, 80, 82
logFlush, 81, 83
logLevel, 80, 82, 84, 85
logSplitSize, 81, 83
LogToFile, 83
LogToFileU, 84
logType, 80, 82

maxLen, 65--68, 76, 79, 100, 101
maxValues, 72, 74, 75
message, 84, 85

OpenSensor, 69
OpenSensorIF2004, 86
OpenSensorIF2004_USB, 87
OpenSensorIF2004_USBU, 88
OpenSensorIF2008, 89
OpenSensorRS232, 85
OpenSensorRS232U, 86
OpenSensorTCP/IP, 90
OpenSensorTCP/IPU, 91
OpenSensorWinUSB, 91

parameterList, 59, 60, 68
paramName, 54--58, 61--67, 93--100
paramValue, 54--59, 61--67, 93--100
Poll, 71
port, 86, 88, 89

rawData, 72, 74, 75

read, 74, 75
 ReleaseSensorInstance, 53
 remoteAddr, 90, 91
 S_Command, 121
 SA_0_0V, 346, 372
 SA_0_10V, 346, 372
 SA_0_SOD, 346, 372
 SA_1_0V, 346, 372
 SA_1_10V, 346, 372
 SA_1_SOD, 346, 372
 SA_a*1..x, 769, 791
 SA_AbbeNumber_vd, 408, 439
 SA_AbbeNumber_vd1..x, 437
 SA_ActiveEdge, 350
 SA_ActiveMaterial, 303, 327, 386, 438
 SA_ActiveParameterSet, 301, 319
 SA_ActiveSensor, 379, 412
 SA_ActiveSerialIf, 515, 649
 SA_ADCAvailable, 863
 SA_ADCLatchSource, 878
 SA_ADCRawValue, 878
 SA_ADCScaledValue, 878
 SA_Address, 144, 300, 316, 383, 424,
 531, 567, 708, 737, 754, 780,
 829, 886, 899
 SA_AlarmHysteresis, 164, 171, 179
 SA_AlarmStart, 164, 171, 178, 179
 SA_AlarmWidth, 164, 171, 180
 SA_Algorithm, 841
 SA_AnalogEEPROMData, 682
 SA_AnalogFactor1..3, 836
 SA_AnalogGain, 523, 659
 SA_AnalogMathFunctionEnable, 836
 SA_AnalogOffset, 522, 659, 834, 836
 SA_AnalogOutput, 404, 483, 495, 506,
 557, 638, 894, 921
 SA_AnalogParamData, 683
 SA_AnalogRange, 405, 483, 558, 639,
 833, 894, 922
 SA_AnalogScaleMode, 240, 270, 405,
 484, 558, 640, 895, 922
 SA_AnalogUnit, 835
 SA_AnalogValue, 154, 894, 921
 SA_AnalogZoom, 499
 SA_Answer1, 906
 SA_Answer2, 906
 SA_ApplicationDisplayStyle, 674
 SA_ApplicationLanguage, 234, 246, 673,
 709, 738, 831, 895, 901
 SA_ApplicationScaleMode, 674
 SA_ApplicationYMax, 674
 SA_ApplicationYMin, 674
 SA_ArticleNumber, 209, 213, 227, 241,
 273, 291, 310, 377, 406, 528,
 559, 641, 698, 699, 713, 715,
 743, 745, 750, 770, 838, 839,
 883, 895
 SA_ArticleNumber1..32, 839
 SA_ArticleNumber1..4, 714
 SA_ArticleNumber1..8, 744
 SA_ASCII, 214, 217, 273, 275, 345
 SA_AssignLimits_ErrorOutput, 276
 SA_AutoAdaptLED, 366
 SA_AutoAdaptLEDThr, 367
 SA_AutoDark, 349
 SA_AutoLPEnabled, 145
 SA_Automatic, 242, 248
 SA_AutostartCommand, 162, 166, 173,
 182, 183, 188, 196
 SA_AutostartEdge, 165, 169, 173
 SA_AutostartTrigger, 165, 169, 173
 SA_AvailableTargets, 666
 SA_Average, 163, 170, 175, 185, 193,
 203
 SA_Average_for_reading, 524, 661
 SA_Averaging, 345, 368, 495, 506
 SA_AveragingType, 236, 258, 303, 330,
 387, 443, 535, 582, 696, 751,
 798, 888, 909
 SA_AveragingType1, 889, 904
 SA_AveragingType2, 889, 905
 SA_AvgRefractIndex, 364, 365
 SA_AvlIndex, 272, 274, 285, 286
 SA_AvrNbr, 667, 672, 677, 688, 691, 692,
 703, 712, 718, 721, 722, 731,
 742, 746, 827, 837
 SA_AvrType, 667, 672, 676, 688, 691,
 692, 703, 712, 718, 721, 722,
 730, 742, 746, 826, 837
 SA_AvType, 214, 216, 272, 275, 286
 SA_b*1..x, 769, 791
 SA_BarycenterOffset, 374
 SA_BarycenterScale, 374
 SA_Binning, 363
 SA_BootLoaderMode, 681
 SA_BootLoaderVer, 213, 273
 SA_BootLoaderVersion, 681
 SA_BootPEnabled, 145
 SA_BootVersion, 227, 241
 SA_BuildID, 227, 241
 SA_BuildTimestamp, 227, 241
 SA_ButtonState, 685
 SA_CableCnt, 666
 SA_CableHead, 227, 241
 SA_CableLength, 670, 671

SA_CableOption, 670, 671
 SA_CableProductCode, 670
 SA_CableRevisionIndex, 670
 SA_CableSerialNumber, 670
 SA_CalibProgress, 678
 SA_CalibrationTable, 291, 310
 SA_CalibState, 678
 SA_CalibTable, 375
 SA_CalibTarget, 667, 672, 678
 SA_CCD, 358
 SA_Channel56Available, 862
 SA_ChartType, 235, 252
 SA_ChExist1..32, 827, 837
 SA_ChExist1..4, 704, 713, 718, 721, 722
 SA_ChExist1..8, 731, 742, 746
 SA_ChTransmit1..4, 719, 721, 723
 SA_ChTransmit1..8, 732, 742, 746
 SA_CIMode, 220
 SA_Coefficient, 693
 SA_ColorName1..x, 769, 790
 SA_ColorSpace, 753, 790, 795
 SA_ColorTable, 768, 790
 SA_ColorTableCount, 768, 790
 SA_CompleteAnswer, 121, 129
 SA_ContinuousMode, 153
 SA_Contrast, 514, 649
 SA_ControllerName, 669, 699, 715, 745, 839
 SA_ControllerOption, 668
 SA_ControllerProductCode, 668
 SA_ControllerRevisionIndex, 668
 SA_ControllerSerialNumber, 668
 SA_ControllerSoftwareVersion, 668
 SA_ControllerTemperature, 677
 SA_CurrentName, 241, 247
 SA_CurrentOfMonitor, 222
 SA_DarkCorrAverage, 486
 SA_DarkCorrThreshold, 408, 413
 SA_DarkPixel, 500, 503
 SA_DarkSig, 359
 SA_DataMode, 848
 SA_DataOutInterface, 237, 260, 305, 333, 390, 450, 540, 598, 766, 801, 890, 911
 SA_DataPort, 707, 736, 828
 SA_DataScale, 373
 SA_DataSource, 239, 267, 403, 481, 555, 634
 SA_Date, 210, 213, 273, 490
 SA_DatOut, 276, 287
 SA_DefaultUser, 232, 243, 298, 311, 379, 410, 530, 562, 767, 773
 SA_Delta_A_AB, 794
 SA_Delta_B, 795
 SA_Delta_E_L, 794
 SA_DeltaKC, 753, 789
 SA_DeltaKH, 753, 789
 SA_DeltaKL, 753, 789
 SA_DeltaMode, 753, 788
 SA_Described_by, 408, 439
 SA_Description, 310, 328, 408, 438, 794
 SA_Description1..x, 327, 437
 SA_DHCPEnabled, 145, 300, 315, 383, 424, 531, 567, 708, 737, 754, 780, 829, 886, 899
 SA_DHCPName, 145
 SA_DHCPServer, 146
 SA_Diagnosis, 189
 SA_DigitalIndexData, 684
 SA_DigitalInLatchSource, 875
 SA_DigitalInValue1..4, 875
 SA_DigitalIOAvailable, 863
 SA_DigitalOut1..4, 876
 SA_DigitalOutBinFormat, 752, 784
 SA_DigitalOutColorFormat, 753, 783
 SA_DigitalOutDetectedID, 765, 817
 SA_DigitalOutValue1..4, 876
 SA_DisplayGain, 523, 659
 SA_DisplayOffset, 523, 659
 SA_DisplayUnit, 229, 232, 296, 298, 530, 561
 SA_DispMeasUnit, 514, 648
 SA_DistanceMode, 754, 808
 SA_DNSServer, 145
 SA_DoubleFrequency, 357
 SA_Echo, 231, 232, 297, 298, 378, 379, 529, 530, 751, 754
 SA_EchoValue, 490
 SA_Edge1_A, 533, 573
 SA_Edge1_B, 533, 573
 SA_Edge2_A, 533, 574
 SA_Edge2_B, 533, 574
 SA_Edge3_A, 533, 575
 SA_Edge3_B, 534, 575
 SA_Edge4_A, 534, 575
 SA_Edge4_B, 534, 575
 SA_Edge5_A, 534, 576
 SA_Edge5_B, 534, 576
 SA_Edge6_A, 534, 577
 SA_Edge6_B, 534, 577
 SA_Edge7_A, 534, 577
 SA_Edge7_B, 535, 577
 SA_Edge8_A, 535, 578
 SA_Edge8_B, 535, 578
 SA_EdgeCount, 502, 504
 SA_EdgeDetectThreshold, 649

SA_EdgeFilter1, 589
 SA_EdgeFilter1Signal, 592
 SA_EdgeFilter2, 590
 SA_EdgeFilter2Signal, 594
 SA_EnableFlash, 273, 277
 SA_EncoderDirection, 866
 SA_EncoderIncrements, 380, 416
 SA_EncoderInterpolation, 865
 SA_EncoderInterpolation1, 381, 417
 SA_EncoderInterpolation2, 381, 418
 SA_EncoderInterpolation3, 381, 418
 SA_EncoderLatchSource, 868
 SA_Encoder.MaxValue, 381, 417
 SA_Encoder.MaxValue1, 382, 422
 SA_Encoder.MaxValue2, 382, 423
 SA_Encoder.MaxValue3, 383, 423
 SA_Encoder.MinValue, 381, 416
 SA_EncoderMode, 867
 SA_EncoderMode1, 381, 419
 SA_EncoderMode2, 382, 419
 SA_EncoderMode3, 382, 420
 SA_EncoderNumber, 380, 416
 SA_EncoderPreload1, 382, 420
 SA_EncoderPreload2, 382, 421
 SA_EncoderPreload3, 382, 421
 SA_EncoderRawValue, 870
 SA_EncoderScaledValue, 870
 SA_Energy, 156
 SA_Error, 668
 SA_ErrorHandler, 210, 213, 216, 272,
 274, 286, 514, 648
 SA_ErrorHysteresis, 239, 268
 SA_ErrorLevel, 404, 482
 SA_ErrorLevelOut1, 240, 269, 556, 636
 SA_ErrorLevelOut2, 557, 637
 SA_ErrorMode, 163, 171, 178, 186, 194,
 201
 SA_ErrorNumber, 121
 SA_ErrorOutHoldTime, 240, 268
 SA_ErrorOutput, 271, 275
 SA_ErrorOutput1, 239, 267, 403, 479,
 555, 632
 SA_ErrorOutput2, 403, 480, 555, 633
 SA_ErrorStatus, 155, 156
 SA_ErrorText, 122
 SA_EthernetMode, 299, 317, 383, 426,
 531, 568, 709, 738, 754, 782,
 830
 SA_EvalBegin, 494, 505
 SA_EvalEnd, 494, 505
 SA_EvalMode, 494, 505
 SA_ExpectedEdges, 533, 573
 SA_ExportData, 249
 SA_Exposure, 356, 357
 SA_Ext_LaserSwitch, 514, 649
 SA_ExtInputMode, 215, 217
 SA_ExtTriggerInDirection1..4, 851
 SA_ExtTriggerInSource, 850
 SA_ExtTriggerInValue1..4, 851
 SA_ExtTriggerOutSource1..2, 852
 SA_FactorCapa, 720
 SA_FactorCh1..4, 706
 SA_FactorCh1..8, 735
 SA_FactorEddy, 720
 SA_FirmwareVersion, 681
 SA_FirstPeak, 355
 SA_FirstReference, 871
 SA_FPGAVersion, 843, 856, 862, 884,
 896
 SA_FramesPerPacket_ETH, 452
 SA_FreePara1, 502
 SA_FreeSR, 356
 SA_FrontEdge_Seg1, 524, 661
 SA_FrontEdge_Seg2, 524, 661
 SA_FrontEdge_Seg3, 662
 SA_FrontEdge_Seg4, 662
 SA_GainPoti, 680
 SA_Gate, 858
 SA_Gateway, 144, 300, 316, 383, 424,
 531, 567, 708, 737, 755, 780,
 830, 886, 899
 SA_HalfMinDist, 485
 SA_HardwareRevision, 227, 241, 378,
 406, 528, 559
 SA_HelpText, 226, 290, 377, 527, 750
 SA_HighFrequency, 358
 SA_HoldLastValid, 237, 261, 305, 335,
 391, 451, 541, 600, 766, 802,
 892, 913
 SA_HTTPBody, 253
 SA_HTTPHeader, 253
 SA_HWMode, 496, 507
 SA_HWRevision, 844, 862
 SA_HysteresisQ1, 153
 SA_HysteresisQ2, 154
 SA_IgnoreValues1, 590
 SA_IgnoreValues2, 591
 SA_Image, 528, 559
 SA_ImageType, 291, 310, 378, 406, 528,
 559, 750, 770
 SA_IndexData, 684
 SA_InitialChartType, 234, 251
 SA_InitialName, 242, 248
 SA_InstanceHandle, 906
 SA_InState, 503
 SA_InternalTriggerMode, 500

SA_InvalidValues, 165, 172, 177
 SA_IsFirstRef, 858
 SA_IsSecondRef, 858
 SA_KeyFunction, 231, 233
 SA_Keylock, 214, 217, 230, 232, 273, 276, 287, 755, 772
 SA_KeylockState, 230, 233, 755, 772
 SA_KeylockTime, 230, 233, 755, 772
 SA_L*1..x, 769, 791
 SA_LampTest, 352
 SA_LampTestThr, 353
 SA_Language, 514, 648
 SA_LaserDiodeError, 221
 SA_LaserIntensity, 514, 649
 SA_LaserPower, 235, 255, 302, 324
 SA_LaserPower1, 887, 902
 SA_LaserPower2, 887, 903
 SA_LaserState, 277, 287
 SA_LastValid, 369
 SA_LatchReg, 857
 SA_Ic0, 487
 SA_Ic1, 487
 SA_Ic2, 487
 SA_LED_Off, 367
 SA_LEDControl, 755, 775
 SA_LEDIntensity, 366
 SA_LEDIntensityColdWhite, 755, 776
 SA_LEDIntensityGreen, 755, 776
 SA_LEDIntensityViolet, 756, 777
 SA_LEDIntensityWarmWhite, 756, 777
 SA_LEDMode1, 853
 SA_LEDMode2, 853
 SA_LEDMode3, 853
 SA_LEDMode4, 853
 SA_LeftEdge, 501, 504
 SA_LightSource, 756, 775
 SA_LightSource1..x, 769, 790
 SA_LimitQ1-1, 153
 SA_LimitQ1-2, 153
 SA_LimitQ2-1, 154
 SA_LimitQ2-2, 154
 SA_LimitQA-1, 155
 SA_LimitQA-2, 155
 SA_LinMode, 689, 691, 692
 SA_LinMode1..4, 710
 SA_LinMode1..8, 739, 742, 747
 SA_LinPoint, 690, 712, 726, 741
 SA_LinPoti, 680
 SA_LocalAdapterDescription, 146
 SA_LocalDefaultGateway, 146
 SA_LocalDHCPEnabled, 146
 SA_LocalIPAddress, 146
 SA_LocalSubnetMask, 146
 SA_Lower_hysteresis, 276
 SA_Lower_limit, 275
 SA_LowerBound1, 589
 SA_LowerBound2, 591
 SA_LowerLimit, 404, 481, 523, 556, 635, 660
 SA_LowerWarning, 523, 660
 SA_LowFrequency, 358
 SA_LowPass, 704
 SA_MACAddress, 144
 SA_MacAddress, 291, 310, 377, 406, 528, 559, 750, 770, 884, 896
 SA_Manufacturer, 144
 SA_Master, 237, 259, 304, 333, 389, 449, 539, 597, 890, 911
 SA_Master_MidPoint_Setup, 276
 SA_Master_value, 276
 SA_MasterSignal, 389, 448, 539, 596
 SA_MasterValue, 237, 259, 305, 333, 390, 449, 525, 540, 598, 663, 890, 911
 SA_MaterialMultiPeak12, 386, 442
 SA_MaterialMultiPeak23, 386, 442
 SA_MaterialMultiPeak34, 386, 442
 SA_MaterialMultiPeak45, 386, 442
 SA_MaterialMultiPeak56, 387, 442
 SA_MaterialName, 310, 328, 408, 438
 SA_MaterialName1..x, 327, 437
 SA_MaterialTable, 326, 436
 SA_MaterialTableCount, 327, 437
 SA_MaxRaw, 525, 526, 663, 664
 SA_MaxRefractIndex, 364, 365
 SA_MaxScaled, 525, 526, 664
 SA_MaxValue, 240, 270, 405, 485, 503, 558, 640, 895, 923
 SA_MeanCount, 696
 SA_MeasDistance, 559, 563
 SA_MeasDistance1..x, 562
 SA_MeasDistIndex, 559, 563
 SA_MeasDistIndex1..x, 562
 SA_MeasDistTable, 559, 562
 SA_MeasDistTableCount, 562
 SA_MeasFrequency, 185, 193, 197
 SA_MeasMode, 660
 SA_MeasObject, 524, 661
 SA_MeasProgName, 522, 659
 SA_MeasProgNumber, 513, 522, 648, 659
 SA_Measrate, 235, 254, 302, 323, 385, 431, 695, 756, 788, 887, 908
 SA.MeasureCount, 667, 672, 675
 SA.MeasureDirection, 533, 572
 SA.MeasureMode, 301, 320, 345, 354,

384, 429, 532, 571, 667, 672, 676, 756, 786, 887, 908
SA_MeasurePeak, 235, 254, 301, 321, 389, 429
SA_MeasureTime, 163, 167, 171
SA_MeasureValueCnt, 305, 335
SA_MeasValue, 501, 503
SA_Median, 661
SA_Median_OnOff, 210
SA_MedianCount, 236, 258, 304, 330, 387, 444, 536, 582, 697, 752, 798, 889, 910
SA_MedianCount1, 889, 904
SA_MedianCount2, 890, 905
SA_MedianIndex, 214, 216
SA_MinGap, 486
SA_MinRaw, 525, 526, 663, 664
SA_MinRefractIndex, 364, 365
SA_MinScaled, 525, 526, 664
SA_MinSR, 357
SA_MinSRIIndex, 348
SA_MinValue, 240, 270, 405, 485, 503, 558, 640, 895, 923
SA_ModeQ1, 153
SA_ModeQ2, 154
SA_MovingCount, 214, 216, 236, 258, 303, 330, 387, 444, 536, 582, 696, 752, 798, 888, 910
SA_MovingCount1, 889, 904
SA_MovingCount2, 890, 905
SA_MultiFunctionInputLevel, 234, 251
SA_MultiFunctionInputMode, 234, 250
SA_MV-First-8, 502
SA_MV-Last-8, 502
SA_Name, 406, 412
SA_Name1..x, 407, 411
SA_NbrCorrectedValues, 304, 331, 388, 445, 537, 584
SA_NbrEvaluatedValues, 304, 331, 388, 445, 536, 583
SA_NormQ1, 154
SA_NormQ2, 154
SA_NormQA, 155
SA_NormSig, 375
SA_NullPoti, 680
SA_NumberOfPeaks, 384, 441
SA_NumberOfSegments, 661
SA_Observer, 757, 774
SA_Observer1..x, 769, 790
SA_Offset, 155, 166, 173--175, 186, 194, 198, 199, 699, 706, 714, 720, 735, 744, 838
SA_Offset1..32, 839
SA_Offset1..4, 714
SA_Offset1..8, 744
SA_OffsetValue, 286
SA_OperationMode, 495, 507
SA_Option, 209, 213, 227, 241, 273, 285, 291, 310, 369, 377, 406, 508, 528, 558, 642, 699, 715, 745, 750, 770, 840, 844, 883, 895
SA_OutMode, 500
SA_Output_Analog, 285
SA_Output_Digital, 285
SA_OutputAdditional, 892, 920
SA_OutputAdditionalCounter_ETH, 293, 308, 342, 401, 476, 554, 630
SA_OutputAdditionalCounter_RS422, 228, 238, 263, 265, 292, 307, 340, 400, 473, 553, 628
SA_OutputAdditionalDistanceRaw1_RS422, 228, 238
SA_OutputAdditionalDistanceRaw_RS422, 239, 263, 265
SA_OutputAdditionalEncoder1_ETH, 401, 475
SA_OutputAdditionalEncoder1_RS422, 400, 472
SA_OutputAdditionalEncoder2_ETH, 401, 475
SA_OutputAdditionalEncoder2_RS422, 400, 473
SA_OutputAdditionalEncoder3_ETH, 401, 475
SA_OutputAdditionalEncoder3_RS422, 400, 473
SA_OutputAdditionalIntensity1_ETH, 294, 308
SA_OutputAdditionalIntensity1_RS422, 228, 238, 292, 306
SA_OutputAdditionalIntensity2_ETH, 294, 308
SA_OutputAdditionalIntensity2_RS422, 292, 306
SA_OutputAdditionalIntensity_ETH, 309, 342, 402, 476
SA_OutputAdditionalIntensity_RS422, 238, 263, 265, 307, 340, 400, 473
SA_OutputAdditionalMeasrate_ETH, 402, 476
SA_OutputAdditionalMeasrate_RS422, 401, 473
SA_OutputAdditionalNbrEdges_ETH, 553, 630
SA_OutputAdditionalNbrEdges_RS422, 553, 628

SA_OutputAdditionalNbrGaps_ETH, 554, 630
 SA_OutputAdditionalNbrGaps_RS422, 553, 628
 SA_OutputAdditionalNbrPins_ETH, 554, 630
 SA_OutputAdditionalNbrPins_RS422, 553, 628
 SA_OutputAdditionalShutterTime_ETH, 293, 308, 341, 401, 475
 SA_OutputAdditionalShutterTime_RS422, 228, 238, 263, 265, 291, 307, 340, 400, 472
 SA_OutputAdditionalState_ETH, 294, 309, 342, 402, 476, 554, 630
 SA_OutputAdditionalState_RS422, 228, 238, 263, 265, 292, 308, 340, 401, 473, 553, 629
 SA_OutputAdditionalTemperature_ETH, 293, 309, 342
 SA_OutputAdditionalTemperature_RS422, 292, 308, 340
 SA_OutputAdditionalTimestamp_ETH, 293, 309, 342, 402, 476, 554, 630
 SA_OutputAdditionalTimestamp_RS422, 238, 263, 265, 292, 307, 340, 400, 473, 553, 628
 SA_OutputAdditionalTimestampHi_RS422, 228
 SA_OutputAdditionalTimestampLo_RS422, 228
 SA_OutputAdditionalTrgCounter_ETH, 294, 309, 342
 SA_OutputAdditionalTrgTimeDiff_ETH, 402, 476
 SA_OutputAdditionalTrgTimeDiff_RS422, 401, 473
 SA_OutputC-BoxAdditional_ETH, 894, 920
 SA_OutputC-BoxAdditional_RS422, 892, 916
 SA_OutputC-BoxAdditional_USB, 893, 918
 SA_OutputC-BoxValue_ETH, 894, 920
 SA_OutputC-BoxValue_RS422, 892, 916
 SA_OutputC-BoxValue_USB, 893, 918
 SA_OutputColorLAB99_ETH, 759, 806
 SA_OutputColorLAB99_RS422, 764, 808
 SA_OutputColorLAB_ETH, 759, 805
 SA_OutputColorLAB_RS422, 764, 807
 SA_OutputColorLCH99_ETH, 760, 806
 SA_OutputColorLCH99_RS422, 765, 808
 SA_OutputColorLCH_ETH, 759, 805
 SA_OutputColorLCH_RS422, 764, 807
 SA_OutputColorLUV_ETH, 759, 805
 SA_OutputColorLUV_RS422, 764, 807
 SA_OutputColorRGB_ETH, 759, 805
 SA_OutputColorRGB_RS422, 764, 807
 SA_OutputColorXYZ_ETH, 759, 805
 SA_OutputColorXYZ_RS422, 764, 807
 SA_OutputContent, 188, 195, 199, 200
 SA_OutputDiameterCenterAxis_ETH, 542, 604
 SA_OutputDiameterCenterAxis_RS422, 542, 603
 SA_OutputDiameterDifference_ETH, 542, 604
 SA_OutputDiameterDifference_RS422, 541, 603
 SA_OutputDiameterEdgeA_ETH, 542, 604
 SA_OutputDiameterEdgeA_RS422, 541, 603
 SA_OutputDiameterEdgeB_ETH, 542, 604
 SA_OutputDiameterEdgeB_RS422, 541, 603
 SA_OutputDistance1_ETH, 294, 308, 392, 456
 SA_OutputDistance1_RS422, 228, 238, 265, 292, 306, 336, 391, 453
 SA_OutputDistance2_ETH, 294, 308, 392, 456
 SA_OutputDistance2_RS422, 292, 306, 336, 391, 453
 SA_OutputDistance3_ETH, 392, 456
 SA_OutputDistance3_RS422, 391, 453
 SA_OutputDistance4_ETH, 392, 456
 SA_OutputDistance4_RS422, 391, 454
 SA_OutputDistance5_ETH, 393, 456
 SA_OutputDistance5_RS422, 391, 454
 SA_OutputDistance6_ETH, 393, 456
 SA_OutputDistance6_RS422, 392, 454
 SA_OutputDistDetectedID_ETH, 762, 813
 SA_OutputDistDetectedID_RS422, 765, 814
 SA_OutputDistDistance01_ETH, 760, 811
 SA_OutputDistDistance02_ETH, 760, 811
 SA_OutputDistDistance03_ETH, 760, 811
 SA_OutputDistDistance04_ETH, 760, 811
 SA_OutputDistDistance05_ETH, 760, 811
 SA_OutputDistDistance06_ETH, 760, 812
 SA_OutputDistDistance07_ETH, 760, 812
 SA_OutputDistDistance08_ETH, 761, 812
 SA_OutputDistDistance09_ETH, 761, 812
 SA_OutputDistDistance10_ETH, 761, 812
 SA_OutputDistDistance11_ETH, 761, 812
 SA_OutputDistDistance12_ETH, 761, 812
 SA_OutputDistDistance13_ETH, 761, 812
 SA_OutputDistDistance14_ETH, 761, 813
 SA_OutputDistDistance15_ETH, 761, 813

SA_OutputDistDistance16_ETH, 762, 813 SA_OutputSegment2CenterAxis_ETH, 548,
 SA_OutputDistMinDistance_ETH, 762, 813 620
 SA_OutputDistMinDistance_RS422, 765, 814 SA_OutputSegment2CenterAxis_RS422, 544,
 612
 SA_OutputDistMinDistID_ETH, 762, 813 SA_OutputSegment2Difference_ETH, 548,
 SA_OutputDistMinDistID_RS422, 765, 814 620
 SA_OutputEdgeDarkLight_ETH, 541, 602 SA_OutputSegment2Difference_RS422, 544,
 SA_OutputEdgeDarkLight_RS422, 541, 601 612
 SA_OutputEdgeLightDark_ETH, 541, 601 SA_OutputSegment2EdgeA_ETH, 548, 620
 SA_OutputEdgeLightDark_RS422, 541, 600SA_OutputSegment2EdgeA_RS422, 544,
 SA_OutputFormat, 155, 163, 171, 176, 612
 187, 195, 199, 200
 SA_OutputGapCenterAxis_ETH, 543, 607 SA_OutputSegment2EdgeB_ETH, 548, 620
 SA_OutputGapCenterAxis_RS422, 543, 606 612
 SA_OutputGapDifference_ETH, 543, 607 SA_OutputSegment3CenterAxis_ETH, 549,
 SA_OutputGapDifference_RS422, 542, 606 621
 SA_OutputGapEdgeA_ETH, 543, 606 SA_OutputSegment3CenterAxis_RS422, 545,
 SA_OutputGapEdgeA_RS422, 542, 605 613
 SA_OutputGapEdgeB_ETH, 543, 607 SA_OutputSegment3Difference_ETH, 548,
 SA_OutputGapEdgeB_RS422, 542, 605 621
 SA_OutputLightSensorBlue_ETH, 758, 819SA_OutputSegment3Difference_RS422, 544,
 SA_OutputLightSensorBlue_RS422, 763, 612
 822
 SA_OutputLightSensorBright_ETH, 758, SA_OutputSegment3EdgeA_ETH, 548, 620
 819 SA_OutputSegment3EdgeA_RS422, 544,
 612
 SA_OutputLightSensorBright_RS422, 763, SA_OutputSegment3EdgeB_ETH, 548, 621
 822 SA_OutputSegment3EdgeB_RS422, 544,
 SA_OutputLightSensorGreen_ETH, 758, 612
 819 SA_OutputSegment4CenterAxis_ETH, 549,
 SA_OutputLightSensorGreen_RS422, 763, 621 SA_OutputSegment4CenterAxis_RS422, 545,
 822 SA_OutputSegment4Difference_ETH, 549,
 SA_OutputLightSensorRed_ETH, 758, 819 613
 SA_OutputLightSensorRed_RS422, 763, SA_OutputSegment4Difference_RS422, 545,
 822 621
 SA_Output.MaxValue, 891, 914 SA_OutputSegment4Difference_RS422, 545,
 SA_Output.MinValue, 891, 914 613
 SA_Output.Mode, 214, 217 SA_OutputSegment4EdgeA_ETH, 549, 621
 SA_Output.ScaleMode, 891, 914 SA_OutputSegment4EdgeA_RS422, 545,
 SA_OutputSegment1CenterAxis_ETH, 548, 613
 620 SA_OutputSegment4EdgeB_ETH, 549, 621
 SA_OutputSegment1CenterAxis_RS422, 548SA_OutputSegment4EdgeB_RS422, 545,
 612 613
 SA_OutputSegment1Difference_ETH, 547, SA_OutputSegment5CenterAxis_ETH, 550,
 620 622
 SA_OutputSegment1Difference_RS422, 547SA_OutputSegment5CenterAxis_RS422, 546,
 611 614
 SA_OutputSegment1EdgeA_ETH, 547, 619SA_OutputSegment5Difference_ETH, 549,
 SA_OutputSegment1EdgeA_RS422, 543, 622
 611 SA_OutputSegment5Difference_RS422, 545,
 SA_OutputSegment1EdgeB_ETH, 547, 620 613
 SA_OutputSegment1EdgeB_RS422, 543, SA_OutputSegment5EdgeA_ETH, 549, 621
 611 SA_OutputSegment5EdgeA_RS422, 545,
 613

SA_OutputSegment5EdgeB_ETH, 549, 622 SA_OutputSensor1Value_USB, 893, 917
SA_OutputSegment5EdgeB_RS422, 545, SA_OutputSensor2Additional_ETH, 894,
613 919
SA_OutputSegment6CenterAxis_ETH, 550, SA_OutputSensor2Additional_RS422, 892,
622 916
SA_OutputSegment6CenterAxis_RS422, 548, SA_OutputSensor2Additional_USB, 893,
614 918
SA_OutputSegment6Difference_ETH, 550, SA_OutputSensor2Value_ETH, 894, 919
622 SA_OutputSensor2Value_RS422, 892, 916
SA_OutputSegment6Difference_RS422, 546, SA_OutputSensor2Value_USB, 893, 918
614 SA_OutputStatistic2Max_ETH, 552, 627
SA_OutputSegment6EdgeA_ETH, 550, 622 SA_OutputStatistic2Max_RS422, 552, 625
SA_OutputSegment6EdgeA_RS422, 546, SA_OutputStatistic2Min_ETH, 552, 627
614 SA_OutputStatistic2Min_RS422, 552, 625
SA_OutputSegment6EdgeB_ETH, 550, 622 SA_OutputStatistic2Peak2Peak_ETH, 553,
SA_OutputSegment6EdgeB_RS422, 546, 627
614 SA_OutputStatistic2Peak2Peak_RS422, 552,
SA_OutputSegment7CenterAxis_ETH, 551, 625
623 SA_OutputStatisticMax_ETH, 295, 307,
SA_OutputSegment7CenterAxis_RS422, 547, 338, 399, 471, 552, 626, 762,
615 815
SA_OutputSegment7Difference_ETH, 550, SA_OutputStatisticMax_RS422, 293, 306,
623 337, 399, 470, 551, 625, 765,
SA_OutputSegment7Difference_RS422, 546, 816
614 SA_OutputStatisticMin_ETH, 295, 307, 338,
SA_OutputSegment7EdgeA_ETH, 550, 622 399, 470, 552, 626, 762, 815
SA_OutputSegment7EdgeA_RS422, 546, SA_OutputStatisticMin_RS422, 293, 306,
614 337, 399, 469, 551, 624, 765,
SA_OutputSegment7EdgeB_ETH, 550, 623 816
SA_OutputSegment7EdgeB_RS422, 546, SA_OutputStatisticPeak2Peak_ETH, 295,
614 307, 339, 400, 471, 552, 626,
SA_OutputSegment8CenterAxis_ETH, 551, 762, 815
623 SA_OutputStatisticPeak2Peak_RS422, 293,
SA_OutputSegment8CenterAxis_RS422, 547, 306, 338, 399, 470, 551, 625,
615 765, 816
SA_OutputSegment8Difference_ETH, 551, SA_OutputStatusCounter_ETH, 758, 819
623 SA_OutputStatusCounter_RS422, 763, 822
SA_OutputSegment8Difference_RS422, 548, SA_OutputStatusError_ETH, 759, 820
615 SA_OutputStatusError_RS422, 764, 823
SA_OutputSegment8EdgeA_ETH, 551, 623 SA_OutputStatusFramerate_ETH, 757, 818
SA_OutputSegment8EdgeA_RS422, 547, SA_OutputStatusFramerate_RS422, 762,
615 821
SA_OutputSegment8EdgeB_ETH, 551, 623 SA_OutputStatusShutter_ETH, 758, 819
SA_OutputSegment8EdgeB_RS422, 547, SA_OutputStatusShutter_RS422, 763, 822
615 SA_OutputStatusTempDetector_ETH, 758,
SA_OutputSensor1Additional_ETH, 893, 819
919 SA_OutputStatusTempDetector_RS422, 763,
SA_OutputSensor1Additional_RS422, 892, 822
916 SA_OutputStatusTempLightSrc_ETH, 758,
SA_OutputSensor1Additional_USB, 893, 819
917 SA_OutputStatusTempLightSrc_RS422, 763,
SA_OutputSensor1Value_ETH, 893, 919 822
SA_OutputSensor1Value_RS422, 892, 916 SA_OutputStatusTimestamp_ETH, 759, 820

SA_OutputStatusTimestamp_RS422, 764, 823
 SA_OutputThickness12_ETH, 294, 307, 396, 466
 SA_OutputThickness12_RS422, 293, 306, 337, 393, 460
 SA_OutputThickness13_ETH, 396, 466
 SA_OutputThickness13_RS422, 393, 460
 SA_OutputThickness14_ETH, 396, 466
 SA_OutputThickness15_ETH, 397, 467
 SA_OutputThickness15_RS422, 394, 461
 SA_OutputThickness16_ETH, 397, 467
 SA_OutputThickness16_RS422, 394, 461
 SA_OutputThickness23_ETH, 397, 467
 SA_OutputThickness23_RS422, 394, 461
 SA_OutputThickness24_ETH, 397, 467
 SA_OutputThickness24_RS422, 394, 461
 SA_OutputThickness25_ETH, 397, 467
 SA_OutputThickness25_RS422, 394, 461
 SA_OutputThickness26_ETH, 398, 468
 SA_OutputThickness26_RS422, 395, 462
 SA_OutputThickness34_ETH, 398, 468
 SA_OutputThickness34_RS422, 395, 462
 SA_OutputThickness35_ETH, 398, 468
 SA_OutputThickness35_RS422, 395, 462
 SA_OutputThickness36_ETH, 398, 468
 SA_OutputThickness36_RS422, 395, 462
 SA_OutputThickness45_ETH, 398, 468
 SA_OutputThickness45_RS422, 395, 462
 SA_OutputThickness46_ETH, 399, 469
 SA_OutputThickness46_RS422, 396, 463
 SA_OutputThickness56_ETH, 399, 469
 SA_OutputThickness56_RS422, 396, 463
 SA_OutputTime, 217
 SA_OutputType, 210, 213, 215, 271, 274
 SA_OutputVideoCorr_ETH, 309, 343
 SA_OutputVideoDark_ETH, 402, 478, 757, 803
 SA_OutputVideoDarkTable_ETH, 403, 478
 SA_OutputVideoLight_ETH, 402, 478, 554, 632
 SA_OutputVideoLightSpectrum_ETH, 757, 804
 SA_OutputVideoLightTable_ETH, 403, 478, 554, 632
 SA_OutputVideoLinearized_ETH, 757, 804
 SA_OutputVideoRaw_ETH, 309, 343, 402, 478, 554, 631, 757, 803
 SA_OutputVideoRaw_RS422, 227, 239, 266
 SA_OutputVideoThreshold_ETH, 403, 478, 555, 632
 SA_Parameter10, 500
 SA_Parameter11, 500
 SA_Password, 155
 SA_PeakSearching, 215, 218
 SA_PilotLaser, 152, 184, 188, 196
 SA_Polarity, 494, 505
 SA_Port, 300, 317, 384, 425, 532, 568, 757, 781, 887, 900
 SA_Pos, 406, 412
 SA_Pos1..x, 327, 407, 411, 437, 768, 790
 SA_Power, 494, 505
 SA_PowerError, 863
 SA_PowerMode, 494, 505
 SA_PrecedingValues, 164, 172, 177
 SA_PresetMode, 242, 249
 SA_PresetNames, 241, 248
 SA_PreTreated, 359
 SA_Protocol, 300, 317, 383, 425, 532, 568, 756, 781, 886, 900
 SA_Q1Hysteresis, 186, 194, 204, 205
 SA_Q1Negation, 186, 194, 204, 205
 SA_Q1Start, 186, 194, 204
 SA_Q1Value, 153
 SA_Q1Width, 186, 194, 204, 205
 SA_Q2Hysteresis, 187, 195, 206
 SA_Q2Negation, 187, 195, 206, 207
 SA_Q2Start, 186, 194, 206
 SA_Q2Value, 154
 SA_Q2Width, 187, 194, 206
 SA_Range, 210, 213, 218, 227, 241, 273, 276, 285, 287, 291, 310, 346, 347, 407, 412, 489, 508, 642, 699, 714, 744, 838
 SA_Range1..32, 839
 SA_Range1..4, 715
 SA_Range1..8, 745
 SA_Range1..x, 407, 411
 SA_RangeBegin, 164, 172, 180, 187, 195, 207
 SA_RangeEnd, 164, 172, 181, 187, 195, 207, 208
 SA_Ranges, 347
 SA_RawRangeMax, 705, 733, 827
 SA_RawRangeMax1..32, 828
 SA_RawRangeMax1..4, 705
 SA_RawRangeMax1..8, 733
 SA_RawRangeMin, 705, 732, 827
 SA_RawRangeMin1..32, 828
 SA_RawRangeMin1..4, 705
 SA_RawRangeMin1..8, 733
 SA_RawValue, 685
 SA_RealFrequency, 845, 864
 SA_RealRatio, 845, 864

SA_RearEdge_Seg1, 524, 662
 SA_RearEdge_Seg2, 525, 662
 SA_RearEdge_Seg3, 662
 SA_RearEdge_Seg4, 662
 SA_RecursiveCount, 236, 258, 303, 330, 387, 444, 536, 582, 752, 798, 889, 910
 SA_RecursiveCount1, 889, 904
 SA_RecursiveCount2, 890, 905
 SA_Ref, 858
 SA_RefOffset, 500
 SA_RefractIndex, 364
 SA_RefractIndexFileIdx, 364, 365
 SA_RefractIndexFileName, 364, 365
 SA_RefractionCorrection, 384, 441
 SA_RefractiveIndex_nC, 408, 439
 SA_RefractiveIndex_nC1..x, 437
 SA_RefractiveIndex_nd, 408, 438
 SA_RefractiveIndex_nd1..x, 437
 SA_RefractiveIndex_nF, 310, 328, 408, 438
 SA_RefractiveIndex_nF1..x, 327, 437
 SA_RegHyst, 488
 SA_RemoteAddress, 300, 317, 384, 425, 532, 568, 757, 781
 SA_ResampleAnalog, 237, 260, 390, 451, 540, 599, 891, 913
 SA_ResampleEthernet, 305, 334, 390, 451, 540, 599, 766, 802, 891, 913
 SA_ResampleRS422, 237, 260, 305, 334, 390, 451, 540, 599, 766, 802, 891, 913
 SA_ResampleUSB, 891, 913
 SA_Resampling, 237, 261, 305, 334, 390, 450, 540, 599, 766, 802, 890, 912
 SA_Reserve_1, 508, 642
 SA_Reserve_10, 663
 SA_Reserve_2, 514, 649
 SA_Reserve_3, 515, 660
 SA_Reserve_4, 524, 661
 SA_Reserve_6, 662
 SA_Reserve_7, 662
 SA_Reserve_9, 663
 SA_Reserved1, 218
 SA_Reserved2, 218
 SA_Reserved3, 218
 SA_Reserved4, 218
 SA_Result, 681, 716, 748, 840
 SA_Reverse, 369
 SA_RightEdge, 501, 504
 SA_ROIEnd, 236, 256, 302, 325, 385, 435, 535, 580
 SA_ROIStart, 235, 256, 302, 325, 385, 435, 535, 580
 SA_RS232_Baudrate, 515, 650
 SA_RS232_Parity, 515, 650
 SA_RS232_StopBits, 515, 650
 SA_RS232_TimeoutRecv, 515, 650
 SA_RS232_TimeoutSend, 515, 650
 SA_RS232Baudrate, 499
 SA_RS232Mode, 499
 SA_RS422_Baudrate, 516, 650
 SA_RS422_Parity, 516, 650
 SA_RS422_StopBits, 516, 651
 SA_RS422_TimeoutRecv, 516, 651
 SA_RS422_TimeoutSend, 516, 651
 SA_RS422Baudrate, 849
 SA_RS422OutSource, 849
 SA_RS422RealBaudrate, 849, 850
 SA_RxDValue1..4, 846
 SA_RxDValue1..6, 872
 SA_Samplerate, 272, 285
 SA_SampleTime, 701, 712, 729, 825, 837
 SA_SaveSettingsMode, 215, 217
 SA_ScaledValue, 502, 504, 685
 SA_ScaleFactor, 163, 171, 174, 185, 193, 198
 SA_ScanStatus, 885
 SA_SearchDirection, 533, 572
 SA_SecondReference, 871
 SA_SelectedSensor1..4, 842
 SA_Sensor, 144, 212, 226, 240, 271, 284, 290, 309, 345, 347, 377, 406, 489, 528, 558, 750, 770, 883, 895
 SA_SensorAccessible, 147
 SA_SensorAddress, 832
 SA_SensorBaudrate, 152, 165, 170, 173, 187, 192, 195, 217, 234, 245, 275, 301, 318, 351, 384, 427, 532, 569, 698, 752, 783, 832, 886, 900
 SA_SensorCableLength, 669
 SA_SensorChanged, 671
 SA_SensorChannelBaudrate1..4, 843
 SA_SensorChannelRealBaudrate, 843
 SA_SensorChannelRealBaudrate1..4, 843
 SA_SensorData, 682
 SA_SensorDatabits, 153
 SA_SensorEndMeasRange, 669
 SA_SensorInterface, 831
 SA_SensorMidMeasRange, 669

SA_SensorName, 669, 699, 713, 743, 838
 SA_SensorName1..32, 839
 SA_SensorName1..4, 714
 SA_SensorName1..8, 744
 SA_SensorOption, 669
 SA_SensorParamData, 683
 SA_SensorProductCode, 669
 SA_SensorRevisionIndex, 669
 SA_SensorSerialNumber, 669
 SA_SensorStartMeasRange, 669
 SA_SensorStopbits, 153
 SA_SensorTable, 407, 411
 SA_SensorTableCount, 407, 411
 SA_SensorTemperature, 677
 SA_SensorType, 212, 271, 284
 SA_Serial, 407, 412
 SA_Serial1..x, 407, 411
 SA_SerialNumber, 146, 156, 210, 213, 226, 240, 273, 285, 291, 310, 377, 406, 508, 528, 558, 642, 699, 713, 715, 743, 745, 750, 770, 838, 839, 883, 895
 SA_SerialNumber1..32, 839
 SA_SerialNumber1..4, 714
 SA_SerialNumber1..8, 744
 SA_SerialOutFormat, 649
 SA_SettingNames, 241, 247
 SA_SetupRecord, 133
 SA_ShowChannels, 734, 743, 747
 SA_ShowLinearized, 734, 743, 747
 SA_ShutterFactor, 324
 SA_ShutterMode, 301, 322, 385, 430, 766, 786
 SA_ShutterTime1, 302, 323, 385, 431
 SA_ShutterTime2, 302, 324, 385, 432
 SA_SkippedSpectra, 360
 SA_SkippedVideo, 256, 434, 579
 SA_SlopeValue, 500
 SA_SmoothVideo, 499
 SA_SoftArtArm, 508, 642
 SA_SoftArtBoot, 508, 642
 SA_SoftArtDSP, 508, 642
 SA_SoftVerArm, 509, 642
 SA_SoftVerBoot, 509, 642
 SA_SoftVerDSP, 509, 642
 SA_SoftwareTrigger, 881
 SA_SoftwareTriggerEnable, 880
 SA_Softwareversion, 145, 210, 213, 227, 241, 271, 287, 291, 310, 378, 406, 489, 528, 559, 700, 715, 745, 750, 770, 840, 884, 896
 SA_SpectralAv, 363
 SA_Spectrum, 360
 SA_Spectrum1..x, 770, 791
 SA_Speed, 214, 216, 272, 274, 286
 SA_SpikeCorrection, 304, 331, 388, 445, 536, 583
 SA_SplitPixel, 486
 SA_SRIndex, 344, 355, 667, 672, 675, 687, 690, 692, 717, 721, 722, 728, 741, 746
 SA_SSIFormat, 188, 196, 208
 SA_StaticIP, 145
 SA_Statistic2Signal, 538, 587
 SA_StatisticDepth, 304, 332, 389, 447, 538, 588, 766, 800
 SA_StatisticSignal, 388, 446, 537, 585, 767, 799
 SA_SubnetMask, 144, 300, 316, 383, 424, 531, 567, 708, 737, 754, 780, 830, 886, 899
 SA_SWType, 273
 SA_Sync_TrgMode, 272, 274
 SA_SyncLogic, 885, 887
 SA_SyncMode, 295, 298, 379, 409, 529, 560, 767, 771, 884, 887
 SA_SyncTermination, 296, 298, 529, 560
 SA_Target_Distance, 524
 SA_TargetMode, 235, 252
 SA_TeachValue, 494, 502, 506
 SA_TeachValue1, 215, 216
 SA_TeachValue2, 215, 216
 SA_Temperature, 156, 166, 189, 378
 SA_TemperatureCoefficient, 724
 SA_TerminationChar, 188, 196, 200, 201
 SA_TextField, 668, 671, 673
 SA_Threshold, 215, 218, 303, 326, 361, 386, 436, 535, 581
 SA_Threshold1, 362
 SA_Threshold2, 362
 SA_Timestamp, 360
 SA_ToleranceHigh, 495, 506
 SA_ToleranceLow, 495, 506
 SA_ToleranceRange, 304, 331, 388, 445, 536, 584
 SA_ToleranceValue, 502
 SA_TransmitIntensity, 368
 SA_TrgChannel1..4, 847
 SA_TrgChannel1..6, 873
 SA_TrgMode, 687, 691, 692, 702, 713, 717, 721, 723, 730, 742, 747, 826, 837
 SA_TrgValue1..6, 874
 SA_TriggerCount, 233, 245, 299, 314, 380, 415, 531, 566, 768, 779,

888, 898
 SA_TriggerDelay, 165, 168, 172, 185, 191, 193
 SA_TriggerEdge, 165, 168, 172, 185, 191, 193
 SA_TriggerLevel, 299, 313, 380, 415, 531, 565, 768, 778, 888, 897
 SA_TriggerLogic, 888, 897
 SA_TriggerMode, 185, 191, 233, 244, 299, 312, 380, 414, 495, 506, 530, 564, 767, 778, 888, 896
 SA_TriggerMoment, 233, 244, 299, 313, 380, 414, 530, 565
 SA_TriggerOutput, 299, 314
 SA_TriggerTermination, 299, 313, 530, 564
 SA_TxDChannel1..4, 847
 SA_TxDChannel1..6, 872
 SA_TxDValue1..6, 873
 SA_Unit, 407, 412, 559, 563, 699, 714, 744, 838
 SA_Unit1..32, 839
 SA_Unit1..4, 715
 SA_Unit1..8, 745
 SA_Unit1..x, 407, 411, 563
 SA_UnlinearizedEddyValue, 724
 SA_UnlinearizedMode, 408, 477
 SA_Upper_hysteresis, 275
 SA_Upper_limit, 275
 SA_UpperBound1, 589
 SA_UpperBound2, 591
 SA_UpperLimit, 239, 267, 404, 481, 523, 556, 636, 660
 SA_UpperWarning, 523, 660
 SA_UserLevel, 232, 242, 298, 311, 379, 410, 530, 561, 754, 773
 SA_UUID, 144
 SA_ValidRange, 164, 172, 177
 SA_Version, 152, 157, 163, 184, 346, 691, 713, 722, 743, 837
 SA_VideoAverage, 303, 326, 386, 436, 768, 796
 SA_VideoCorr, 321
 SA_VideoDark, 432, 434, 787
 SA_VideoDarkTable, 432, 434
 SA_VideoLight, 432, 434, 579, 787
 SA_VideoLightTable, 432, 434, 579
 SA_VideoLin, 787
 SA_VideoMax, 503
 SA_VideoRaw, 255, 321, 432, 433, 579, 786
 SA_VideoSignal, 222, 280, 501
 SA_VideoThrAuto, 491, 499
 SA_VideoThreshold, 432, 434, 579
 SA_VideoThrFix, 498
 SA_VideoThrMode, 496, 507
 SA_VideoTimestamp, 255, 321, 432, 434, 579, 787
 SA_Watchdog, 353
 SA_WatchdogPeriod, 354
 SA_WaveLength, 311, 327, 328
 SA_WebpageVersion, 884, 896
 SA_Webstatic, 750, 771
 SA_WhiteRef, 375
 SA_WindowMax, 186, 193, 202
 SA_WindowMin, 185, 193, 202
 SA_X1..16, 345, 370
 SA_X1..x, 769, 791
 SA_Y1..x, 770, 791
 SA_Z1..x, 770, 791
 SA_ZeroPoint, 286
 scaledData, 72, 74, 75
 sensor, 50
 SensorCommand, 70
 sensorCommand, 92--100
 sensorName, 51, 52
 serialNumber, 87, 89
 SetDoubleExecSCmd, 94
 SetDoubleExecSCmdU, 95
 SetIntExecSCmd, 93
 SetIntExecSCmdU, 93
 SetParameterBinary, 58
 SetParameterBinaryU, 58
 SetParameterDouble, 56
 SetParameterDoubleU, 56
 SetParameterDWORD_PTR, 55
 SetParameterDWORD_PTRU, 55
 SetParameterInt, 53
 SetParameterIntU, 54
 SetParameters, 59
 SetParameterString, 57
 SetParameterStringU, 57
 SetParametersU, 60
 SetStringExecSCmd, 95
 SetStringExecSCmdU, 96
 SP_0V, 371
 SP_10V, 372
 SP_a*, 793
 SP_AbbeNumber_vd, 439
 SP_ActivateGate, 846, 871
 SP_Active, 504
 SP_ActiveEdge, 350
 SP_ActiveMaterial, 327, 438
 SP_ActiveSensor, 411
 SP_ActiveSerialIf, 512, 518, 646, 653
 SP_ActualPos, 795

SP_ADCLatchSource, 877
 SP_ADCNumber, 877, 878
 SP_Additional, 232, 297, 379, 529, 751, 885
 SP_Address, 315, 424, 566, 707, 736, 779, 829, 898
 SP_AlarmHysteresis, 179
 SP_AlarmStart, 178
 SP_AlarmWidth, 179
 SP_Algorithm, 841
 SP_AllDevices, 129, 885
 SP_AnalogChannel, 833–835
 SP_AnalogEEPROMData, 682
 SP_AnalogFactor1..3, 836
 SP_AnalogGain, 520, 655
 SP_AnalogMathFunctionEnable, 836
 SP_AnalogOffset, 520, 655, 834, 836
 SP_AnalogOutput, 482, 493, 637, 920
 SP_AnalogParamData, 683
 SP_AnalogRange, 483, 639, 833, 921
 SP_AnalogScaleMode, 269, 484, 639, 922
 SP_AnalogUnit, 834
 SP_AnalogValue, 921
 SP_AnalogZoom, 497
 SP_ApplicationDisplayMode, 674
 SP_ApplicationLanguage, 246, 673, 709, 738, 830, 901
 SP_ApplicationScaleMode, 673
 SP_ApplicationYMax, 674
 SP_ApplicationYMin, 673
 SP_ApplyImmediately, 250
 SP_ASCII, 224, 282
 SP_AutoAdaptLED, 366
 SP_AutoAdaptLEDTThr, 366
 SP_AutoDark, 348
 SP_Automatic, 248
 SP_AutostartCommand, 162, 182
 SP_AutostartEdge, 169
 SP_AutostartTrigger, 169
 SP_Average, 175, 203
 SP_Average_for_reading, 521, 656
 SP_Averaging, 367, 492
 SP_AveragingForDark, 348
 SP_AveragingType, 257, 329, 442, 581, 695, 797, 908
 SP_AveragingType1, 903
 SP_AveragingType2, 904
 SP_AvIndex, 281, 288
 SP_AvrNbr, 677, 688, 703, 718, 731, 826
 SP_AvrType, 676, 688, 702, 717, 730, 826
 SP_AvType, 223, 281, 288
 SP_b*, 793
 SP_BarycenterOffset, 374
 SP_BarycenterScale, 373
 SP_Binning, 363
 SP_BlockIndex, 501
 SP_BlockSize, 376
 SP_CalDefaultType, 488
 SP_CalibEnd, 679
 SP_CalibMid, 679
 SP_CalibStart, 679
 SP_CalibTable, 376, 488
 SP_CalibTarget, 678
 SP_Chан, 704–706, 709–711, 713, 719, 720, 723–725, 732, 734, 735, 738–740, 743, 827, 838
 SP_ChartType, 252
 SP_ChTransmit1..4, 719
 SP_ChTransmit1..8, 731
 SP_CmdStr, 128
 SP_Coefficient, 693
 SP_CoeffParam, 693
 SP_ColorName, 792–794, 796
 SP_ColorSpace, 795
 SP_Command, 226, 290, 377, 527, 749
 SP_Command1, 905
 SP_Command2, 906
 SP_Complete, 705, 714, 733, 744, 828, 838
 SP_ContinuousMode, 157
 SP_Contrast, 511, 518, 646, 652
 SP_DarkCorrAverage, 486
 SP_DarkCorrThreshold, 413
 SP_DarkPixel, 497
 SP_DataMode, 848
 SP_DataOutInterface, 259, 333, 449, 598, 801, 911
 SP_DataPort, 707, 735, 828
 SP_DataScale, 373
 SP_DataSource, 267, 480, 633
 SP_DefaultType, 249, 319, 428, 570, 785
 SP_DefaultUser, 243, 311, 410, 561, 773
 SP_Delta_A_AB, 794
 SP_Delta_B, 794
 SP_Delta_E_L, 794
 SP_DeltaKC, 789
 SP_DeltaKH, 789
 SP_DeltaKL, 788
 SP_DeltaMode, 788
 SP_Description, 328, 439, 440, 793
 SP_DeviceInstance, 832
 SP_DHCPEnabled, 315, 423, 566, 707, 736, 779, 829, 898
 SP_DigitalIndexData, 684

SP_DigitalInLatchSource, 874
 SP_DigitalOut1..4, 875
 SP_DigitalOutBinFormat, 784
 SP_DigitalOutColorFormat, 783
 SP_DigitalOutDetectedID, 816
 SP_DigitalOutValue1..4, 876
 SP_DisablePower, 862
 SP_DisplayGain, 520, 655
 SP_DisplayOffset, 520, 655
 SP_DisplayUnit, 229, 296, 560
 SP_DispMeasUnit, 511, 517, 645, 652
 SP_DistanceMode, 808
 SP_DoubleFrequency, 357
 SP_Echo, 231, 297, 378, 529, 751
 SP_Edge1_A, 573
 SP_Edge1_B, 573
 SP_Edge2_A, 574
 SP_Edge2_B, 574
 SP_Edge3_A, 574
 SP_Edge3_B, 574
 SP_Edge4_A, 575
 SP_Edge4_B, 575
 SP_Edge5_A, 576
 SP_Edge5_B, 576
 SP_Edge6_A, 576
 SP_Edge6_B, 576
 SP_Edge7_A, 577
 SP_Edge7_B, 577
 SP_Edge8_A, 578
 SP_Edge8_B, 578
 SP_EdgeDetectThreshold, 646, 652
 SP_EdgeFilter1, 588
 SP_EdgeFilter1Signal, 591
 SP_EdgeFilter2, 590
 SP_EdgeFilter2Signal, 593
 SP_EnableFlash, 278
 SP_Encoder1, 350
 SP_Encoder2, 351
 SP_Encoder3, 351
 SP_EncoderDirection, 866
 SP_EncoderIncrements, 416
 SP_EncoderInterpolation, 865
 SP_EncoderInterpolation1, 417
 SP_EncoderInterpolation2, 417
 SP_EncoderInterpolation3, 418
 SP_EncoderLatchSource, 868
 SP_Encoder.MaxValue, 416
 SP_Encoder.MaxValue1, 422
 SP_Encoder.MaxValue2, 422
 SP_Encoder.MaxValue3, 423
 SP_Encoder.MinValue, 416
 SP_EncoderMode, 867
 SP_EncoderMode1, 418
 SP_EncoderMode2, 419
 SP_EncoderMode3, 419
 SP_EncoderNumber, 415, 864--870
 SP_EncoderPreload1, 420
 SP_EncoderPreload2, 420
 SP_EncoderPreload3, 421
 SP_EncoderPreloadValue, 869
 SP_ErrorHandler, 211, 224, 282, 511, 517, 645, 652
 SP_ErrorHysteresis, 268
 SP_ErrorLevel, 482
 SP_ErrorLevelOut1, 269, 636
 SP_ErrorLevelOut2, 636
 SP_ErrorMode, 178, 201
 SP_ErrorOutHoldTime, 268
 SP_ErrorOutput, 279
 SP_ErrorOutput1, 266, 479, 632
 SP_ErrorOutput2, 479, 633
 SP_EthernetMode, 317, 426, 568, 708, 737, 782, 830
 SP_EvalBegin, 492
 SP_EvalEnd, 492
 SP_EvalMode, 491
 SP_ExpectedEdges, 572
 SP_ExportType, 249
 SP_Exposure, 356
 SP_Ext_LaserSwitch, 511, 517, 646, 652
 SP_ExtInputMode, 220
 SP_ExtTriggerInDirection1..4, 850
 SP_ExtTriggerInSource, 850
 SP_ExtTriggerOutSource1..2, 851
 SP_FactorCapa, 719
 SP_FactorCh1..4, 706
 SP_FactorCh1..8, 734
 SP_FactorEddy, 720
 SP_FirmwareFile, 681, 715, 747, 840
 SP_FirstPeak, 354
 SP_ForceOverwrite, 250
 SP_FramesPerPacket_ETH, 451
 SP_FreeSR, 356
 SP_FrontEdge_Seg1, 510, 521, 644, 657
 SP_FrontEdge_Seg2, 510, 521, 644, 657
 SP_FrontEdge_Seg3, 644, 657
 SP_FrontEdge_Seg4, 644, 657
 SP_FullRange, 376
 SP_GainPoti, 680
 SP_GateChannel, 846, 871
 SP_Gateway, 315, 424, 566, 708, 736, 780, 829, 898
 SP_HalfMinDist, 485
 SP_HighFrequency, 358
 SP_HoldLastValid, 261, 335, 451, 600, 802, 913

SP_HTPP.getTag, 253
 SP_HTPPMaxLength, 253
 SP_HTPPPath, 253
 SP_HWMode, 493
 SP_HysteresisQ1, 158
 SP_HysteresisQ2, 158
 SP_IgnoreValues1, 589
 SP_IgnoreValues2, 590
 SP_ImportData, 250
 SP_IncludeAdditionalParameters, 132
 SP_IncludeInterfaceParameters, 133
 SP_IncludeSavedSetups, 132
 SP_IndexData, 684
 SP_InitialChartType, 251
 SP_InitialName, 248
 SP_InternalTriggerMode, 497
 SP_InvalidValues, 176
 SP_JustActualSetup, 907
 SP_KeepDevice, 570, 785, 907
 SP_KeyFunction, 231
 SP_Keylock, 218, 230, 277, 287, 771
 SP_KeylockTime, 230, 772
 SP_L*, 792
 SP_LampTest, 352
 SP_LampTestThr, 352
 SP_Language, 511, 517, 645, 651
 SP_LaserDiode1, 221
 SP_LaserDiode2, 221
 SP_LaserIntensity, 511, 517, 646, 652
 SP_LaserPower, 254, 324
 SP_LaserPower1, 902
 SP_LaserPower2, 902
 SP_LastValid, 368
 SP_LED_Off, 367
 SP_LEDControl, 775
 SP_LEDIntensity, 365
 SP_LEDIntensityColdWhite, 776
 SP_LEDIntensityGreen, 776
 SP_LEDIntensityViolet, 777
 SP_LEDIntensityWarmWhite, 776
 SP_LEDMode1, 852
 SP_LEDMode2, 852
 SP_LEDMode3, 852
 SP_LEDMode4, 853
 SP_LightSource, 774, 792
 SP_LimitQ1-1, 159
 SP_LimitQ1-2, 159
 SP_LimitQ2-1, 159
 SP_LimitQ2-2, 159
 SP_LimitQA-1, 160
 SP_LimitQA-2, 161
 SP_LinMode, 689, 710, 739
 SP_LinPoint, 711, 740
 SP_LinPos, 689, 690, 710–712, 724, 725, 739–741
 SP_LinPoti, 680
 SP_LoadReg, 857
 SP_Location, 490, 491, 493, 496, 498
 SP_Lower_hysteresis, 283
 SP_Lower_limit, 283
 SP_LowerBound1, 589
 SP_LowerBound2, 590
 SP_LowerLimit, 480, 520, 634, 655
 SP_LowerWarning, 521, 656
 SP_LowFrequency, 358
 SP_LowPass, 703
 SP_Master, 258, 332, 449, 597, 910
 SP_Master_value, 283
 SP_MasterSignal, 447, 595
 SP_MasterValue, 259, 332, 449, 522, 597, 658, 910
 SP_MaterialIndex, 438
 SP_MaterialMultiPeak12, 441
 SP_MaterialMultiPeak23, 441
 SP_MaterialMultiPeak34, 441
 SP_MaterialMultiPeak45, 441
 SP_MaterialMultiPeak56, 442
 SP_MaterialName, 328, 439, 440
 SP_MaxValue, 270, 484, 640, 922
 SP_MeanCount, 696
 SP_MeasDistIndex, 563
 SP_MeasFrequency, 197
 SP_MeasMode, 656
 SP_MeasObject, 521, 657
 SP_MeasProgName, 520, 655
 SP_MeasProgNumber, 509, 510, 516, 643, 645, 651
 SP_Measrate, 254, 322, 430, 694, 787, 908
 SP_MeasureCount, 675
 SP_MeasureDirection, 572
 SP_MeasureMode, 279, 320, 354, 428, 571, 676, 785, 907
 SP_MeasurePeak, 253, 320, 429
 SP_MeasureTime, 167
 SP_MeasureValueCnt, 335
 SP_MeasValue, 281
 SP_Median, 656
 SP_Median_OnOff, 211
 SP_MedianCount, 257, 329, 443, 582, 696, 797, 909
 SP_MedianCount1, 903
 SP_MedianCount2, 905
 SP_MedianIndex, 223
 SP_MinGap, 485
 SP_MinValue, 269, 484, 640, 922

SP_ModeQ1, 159
 SP_ModeQ2, 160
 SP_MovingCount, 223, 257, 329, 443, 581, 696, 797, 909
 SP_MovingCount1, 903
 SP_MovingCount2, 904
 SP_MultiFunctionInputLevel, 251
 SP_MultiFunctionInputMode, 250
 SP_Multiplexer, 685
 SP_NbrCorrectedValues, 331, 445, 583
 SP_NbrEvaluatedValues, 331, 444, 583
 SP_NewName, 247
 SP_NewPassword, 243, 312, 410, 562, 701, 727, 773, 825
 SP_NewPos, 795
 SP_NormQ1, 160
 SP_NormQ2, 160
 SP_NormQA, 161
 SP_NullPoti, 680
 SP_NumberOfPeaks, 441
 SP_NumberOfPoints, 349
 SP_NumberOfSegments, 657
 SP_Observer, 774, 792
 SP_Offset, 158, 174, 198, 706, 719, 734
 SP_OldName, 247
 SP_OldPassword, 243, 312, 410, 562, 701, 727, 773, 824
 SP_OperationMode, 221, 493
 SP_Option, 369
 SP_OutMode, 498
 SP_OutNr, 371
 SP_OutputAdditional, 920
 SP_OutputAdditionalCounter_ETH, 341, 474, 629
 SP_OutputAdditionalCounter_RS422, 262, 264, 339, 471, 627
 SP_OutputAdditionalDistanceRaw_RS422, 262, 264
 SP_OutputAdditionalEncoder1_ETH, 474
 SP_OutputAdditionalEncoder1_RS422, 471
 SP_OutputAdditionalEncoder2_ETH, 474
 SP_OutputAdditionalEncoder2_RS422, 471
 SP_OutputAdditionalEncoder3_ETH, 474
 SP_OutputAdditionalEncoder3_RS422, 471
 SP_OutputAdditionalIntensity_ETH, 341, 474
 SP_OutputAdditionalIntensity_RS422, 262, 264, 339, 472
 SP_OutputAdditionalMeasrate_ETH, 475
 SP_OutputAdditionalMeasrate_RS422, 472
 SP_OutputAdditionalNbrEdges_ETH, 629
 SP_OutputAdditionalNbrEdges_RS422, 627
 SP_OutputAdditionalNbrGaps_ETH, 629
 SP_OutputAdditionalNbrGaps_RS422, 627
 SP_OutputAdditionalNbrPins_ETH, 629
 SP_OutputAdditionalNbrPins_RS422, 627
 SP_OutputAdditionalShutterTime_ETH, 340, 474
 SP_OutputAdditionalShutterTime_RS422, 262, 264, 339, 471
 SP_OutputAdditionalState_ETH, 341, 475, 629
 SP_OutputAdditionalState_RS422, 262, 264, 339, 472, 628
 SP_OutputAdditionalTemperature_ETH, 341
 SP_OutputAdditionalTemperature_RS422, 339
 SP_OutputAdditionalTimestamp_ETH, 341, 474, 629
 SP_OutputAdditionalTimestamp_RS422, 262, 264, 339, 472, 628
 SP_OutputAdditionalTrgCounter_ETH, 341
 SP_OutputAdditionalTrgTimeDiff_ETH, 475
 SP_OutputAdditionalTrgTimeDiff_RS422, 472
 SP_OutputC-BoxAdditional_ETH, 919
 SP_OutputC-BoxAdditional_RS422, 915
 SP_OutputC-BoxAdditional_USB, 917
 SP_OutputC-BoxValue_ETH, 919
 SP_OutputC-BoxValue_RS422, 915
 SP_OutputC-BoxValue_USB, 917
 SP_OutputColorLAB99_ETH, 805
 SP_OutputColorLAB99_RS422, 807
 SP_OutputColorLAB_ETH, 804
 SP_OutputColorLAB_RS422, 806
 SP_OutputColorLCH99_ETH, 805
 SP_OutputColorLCH99_RS422, 807
 SP_OutputColorLCH_ETH, 804
 SP_OutputColorLCH_RS422, 806
 SP_OutputColorLUV_ETH, 804
 SP_OutputColorLUV_RS422, 806
 SP_OutputColorRGB_ETH, 804
 SP_OutputColorRGB_RS422, 806
 SP_OutputColorXYZ_ETH, 804
 SP_OutputColorXYZ_RS422, 806
 SP_OutputContent, 199
 SP_OutputDiameterCenterAxis_ETH, 604
 SP_OutputDiameterCenterAxis_RS422, 603
 SP_OutputDiameterDifference_ETH, 604
 SP_OutputDiameterDifference_RS422, 602
 SP_OutputDiameterEdgeA_ETH, 603
 SP_OutputDiameterEdgeA_RS422, 602
 SP_OutputDiameterEdgeB_ETH, 604
 SP_OutputDiameterEdgeB_RS422, 602
 SP_OutputDistance1_ETH, 454

SP_OutputDistance1_RS422, 263, 336, 452
 SP_OutputDistance2_ETH, 454
 SP_OutputDistance2_RS422, 336, 452
 SP_OutputDistance3_ETH, 455
 SP_OutputDistance3_RS422, 452
 SP_OutputDistance4_ETH, 455
 SP_OutputDistance4_RS422, 452
 SP_OutputDistance5_ETH, 455
 SP_OutputDistance5_RS422, 453
 SP_OutputDistance6_ETH, 455
 SP_OutputDistance6_RS422, 453
 SP_OutputDistDetectedID_ETH, 811
 SP_OutputDistDetectedID_RS422, 814
 SP_OutputDistDistance01_ETH, 808
 SP_OutputDistDistance02_ETH, 809
 SP_OutputDistDistance03_ETH, 809
 SP_OutputDistDistance04_ETH, 809
 SP_OutputDistDistance05_ETH, 809
 SP_OutputDistDistance06_ETH, 809
 SP_OutputDistDistance07_ETH, 809
 SP_OutputDistDistance08_ETH, 809
 SP_OutputDistDistance09_ETH, 809
 SP_OutputDistDistance10_ETH, 810
 SP_OutputDistDistance11_ETH, 810
 SP_OutputDistDistance12_ETH, 810
 SP_OutputDistDistance13_ETH, 810
 SP_OutputDistDistance14_ETH, 810
 SP_OutputDistDistance15_ETH, 810
 SP_OutputDistDistance16_ETH, 810
 SP_OutputDistMinDistance_ETH, 810
 SP_OutputDistMinDistance_RS422, 813
 SP_OutputDistMinDistID_ETH, 811
 SP_OutputDistMinDistID_RS422, 814
 SP_OutputEdgeDarkLight_ETH, 602
 SP_OutputEdgeDarkLight_RS422, 601
 SP_OutputEdgeLightDark_ETH, 601
 SP_OutputEdgeLightDark_RS422, 600
 SP_OutputFormat, 175, 199
 SP_OutputGapCenterAxis_ETH, 606
 SP_OutputGapCenterAxis_RS422, 605
 SP_OutputGapDifference_ETH, 606
 SP_OutputGapDifference_RS422, 605
 SP_OutputGapEdgeA_ETH, 606
 SP_OutputGapEdgeA_RS422, 605
 SP_OutputGapEdgeB_ETH, 606
 SP_OutputGapEdgeB_RS422, 605
 SP_OutputLightSensorBlue_ETH, 818
 SP_OutputLightSensorBlue_RS422, 821
 SP_OutputLightSensorBright_ETH, 818
 SP_OutputLightSensorBright_RS422, 821
 SP_OutputLightSensorGreen_ETH, 818
 SP_OutputLightSensorGreen_RS422, 821
 SP_OutputLightSensorRed_ETH, 817
 SP_OutputLightSensorRed_RS422, 820
 SP_OutputMaxValue, 914
 SP_OutputMinValue, 914
 SP_OutputMode, 224
 SP_OutputScaleMode, 914
 SP_OutputSegment1CenterAxis_ETH, 616
 SP_OutputSegment1CenterAxis_RS422, 607
 SP_OutputSegment1Difference_ETH, 616
 SP_OutputSegment1Difference_RS422, 607
 SP_OutputSegment1EdgeA_ETH, 615
 SP_OutputSegment1EdgeA_RS422, 607
 SP_OutputSegment1EdgeB_ETH, 615
 SP_OutputSegment1EdgeB_RS422, 607
 SP_OutputSegment2CenterAxis_ETH, 616
 SP_OutputSegment2CenterAxis_RS422, 608
 SP_OutputSegment2Difference_ETH, 616
 SP_OutputSegment2Difference_RS422, 608
 SP_OutputSegment2EdgeA_ETH, 616
 SP_OutputSegment2EdgeA_RS422, 608
 SP_OutputSegment2EdgeB_ETH, 616
 SP_OutputSegment2EdgeB_RS422, 608
 SP_OutputSegment3CenterAxis_ETH, 617
 SP_OutputSegment3CenterAxis_RS422, 608
 SP_OutputSegment3Difference_ETH, 617
 SP_OutputSegment3Difference_RS422, 608
 SP_OutputSegment3EdgeA_ETH, 616
 SP_OutputSegment3EdgeA_RS422, 608
 SP_OutputSegment3EdgeB_ETH, 616
 SP_OutputSegment3EdgeB_RS422, 608
 SP_OutputSegment4CenterAxis_ETH, 617
 SP_OutputSegment4CenterAxis_RS422, 609
 SP_OutputSegment4Difference_ETH, 617
 SP_OutputSegment4Difference_RS422, 609
 SP_OutputSegment4EdgeA_ETH, 617
 SP_OutputSegment4EdgeA_RS422, 609
 SP_OutputSegment4EdgeB_ETH, 617
 SP_OutputSegment4EdgeB_RS422, 609
 SP_OutputSegment5CenterAxis_ETH, 618
 SP_OutputSegment5CenterAxis_RS422, 609
 SP_OutputSegment5Difference_ETH, 618
 SP_OutputSegment5Difference_RS422, 609
 SP_OutputSegment5EdgeA_ETH, 617
 SP_OutputSegment5EdgeA_RS422, 609
 SP_OutputSegment5EdgeB_ETH, 617
 SP_OutputSegment5EdgeB_RS422, 609
 SP_OutputSegment6CenterAxis_ETH, 618
 SP_OutputSegment6CenterAxis_RS422, 610
 SP_OutputSegment6Difference_ETH, 618
 SP_OutputSegment6Difference_RS422, 610
 SP_OutputSegment6EdgeA_ETH, 618
 SP_OutputSegment6EdgeA_RS422, 610
 SP_OutputSegment6EdgeB_ETH, 618

SP_OutputSegment6EdgeB_RS422, 610 SP_OutputStatusShutter_ETH, 817
 SP_OutputSegment7CenterAxis_ETH, 619 SP_OutputStatusShutter_RS422, 820
 SP_OutputSegment7CenterAxis_RS422, 619 SP_OutputStatusTempDetector_ETH, 817
 SP_OutputSegment7Difference_ETH, 619 SP_OutputStatusTempDetector_RS422, 820
 SP_OutputSegment7Difference_RS422, 619 SP_OutputStatusTempLightSrc_ETH, 817
 SP_OutputSegment7EdgeA_ETH, 618 SP_OutputStatusTempLightSrc_RS422, 820
 SP_OutputSegment7EdgeA_RS422, 610 SP_OutputStatusTimestamp_ETH, 818
 SP_OutputSegment7EdgeB_ETH, 618 SP_OutputStatusTimestamp_RS422, 821
 SP_OutputSegment7EdgeB_RS422, 610 SP_OutputThickness12_ETH, 463
 SP_OutputSegment8CenterAxis_ETH, 619 SP_OutputThickness12_RS422, 336, 457
 SP_OutputSegment8CenterAxis_RS422, 619 SP_OutputThickness13_ETH, 463
 SP_OutputSegment8Difference_ETH, 619 SP_OutputThickness13_RS422, 457
 SP_OutputSegment8Difference_RS422, 619 SP_OutputThickness14_ETH, 463
 SP_OutputSegment8EdgeA_ETH, 619 SP_OutputThickness14_RS422, 457
 SP_OutputSegment8EdgeA_RS422, 611 SP_OutputThickness15_ETH, 464
 SP_OutputSegment8EdgeB_ETH, 619 SP_OutputThickness15_RS422, 457
 SP_OutputSegment8EdgeB_RS422, 611 SP_OutputThickness16_ETH, 464
 SP_OutputSensor1Additional_ETH, 918 SP_OutputThickness16_RS422, 458
 SP_OutputSensor1Additional_RS422, 915 SP_OutputThickness23_ETH, 464
 SP_OutputSensor1Additional_USB, 917 SP_OutputThickness23_RS422, 458
 SP_OutputSensor1Value_ETH, 918 SP_OutputThickness24_ETH, 464
 SP_OutputSensor1Value_RS422, 915 SP_OutputThickness24_RS422, 458
 SP_OutputSensor1Value_USB, 916 SP_OutputThickness25_ETH, 464
 SP_OutputSensor2Additional_ETH, 919 SP_OutputThickness25_RS422, 458
 SP_OutputSensor2Additional_RS422, 915 SP_OutputThickness26_ETH, 465
 SP_OutputSensor2Additional_USB, 917 SP_OutputThickness26_RS422, 458
 SP_OutputSensor2Value_ETH, 918 SP_OutputThickness34_ETH, 465
 SP_OutputSensor2Value_RS422, 915 SP_OutputThickness34_RS422, 459
 SP_OutputSensor2Value_USB, 917 SP_OutputThickness35_ETH, 465
 SP_OutputStatistic2Max_ETH, 626 SP_OutputThickness35_RS422, 459
 SP_OutputStatistic2Max_RS422, 624 SP_OutputThickness36_ETH, 465
 SP_OutputStatistic2Min_ETH, 626 SP_OutputThickness36_RS422, 459
 SP_OutputStatistic2Min_RS422, 624 SP_OutputThickness45_ETH, 465
 SP_OutputStatistic2Peak2Peak_ETH, 626 SP_OutputThickness45_RS422, 459
 SP_OutputStatistic2Peak2Peak_RS422, 626 SP_OutputThickness46_ETH, 466
 SP_OutputStatisticMax_ETH, 338, 470, 625, 815 SP_OutputThickness46_RS422, 460
 SP_OutputStatisticMax_RS422, 337, 469, 624, 816 SP_OutputThickness56_ETH, 466
 SP_OutputStatisticMin_ETH, 338, 470, 625, 814 SP_OutputThickness56_RS422, 460
 SP_OutputStatisticMin_RS422, 337, 469, 624, 815 SP_OutputTime, 225
 SP_OutputStatisticPeak2Peak_ETH, 338, 470, 626, 815 SP_OutputType, 211, 224, 282
 SP_OutputStatisticPeak2Peak_RS422, 337, 469, 624, 816 SP_OutputVideoCorr_ETH, 343
 SP_OutputStatusCounter_ETH, 818 SP_OutputVideoDark_ETH, 477, 803
 SP_OutputStatusCounter_RS422, 821 SP_OutputVideoDarkTable_ETH, 477
 SP_OutputStatusError_ETH, 818 SP_OutputVideoLight_ETH, 477, 631
 SP_OutputStatusError_RS422, 821 SP_OutputVideoLightSpectrum_ETH, 803
 SP_OutputStatusFramerate_ETH, 817 SP_OutputVideoLightTable_ETH, 477, 631
 SP_OutputStatusFramerate_RS422, 820 SP_OutputVideoLinearized_ETH, 803
 SP_Parameter10, 498 SP_OutputVideoRaw_ETH, 342, 477, 630, 803
 SP_OutputVideoRaw_RS422, 264, 266 SP_OutputVideoThreshold_ETH, 478, 631
 SP_Overwrite, 247 SP_Parameter10, 498

SP_Parameter11, 498
 SP_ParameterSet, 319, 427, 428, 570, 784, 785, 879, 906, 907
 SP_ParameterType, 319, 427, 570, 784, 907
 SP_Password, 242, 311, 409, 561, 700, 727, 772, 824
 SP_PeakSearching, 222
 SP_PilotLaser, 157, 184
 SP_Polarity, 491
 SP_Port, 316, 425, 568, 781, 899
 SP_Pos, 792
 SP_Power, 491
 SP_PowerMode, 491
 SP_PrecedingValues, 176
 SP_PresetMode, 248
 SP_Protocol, 316, 425, 567, 781, 899
 SP_Q1Hysteresis, 204
 SP_Q1Negation, 204
 SP_Q1Start, 203
 SP_Q1Width, 203
 SP_Q2Hysteresis, 205
 SP_Q2Negation, 205
 SP_Q2Start, 205
 SP_Q2Width, 205
 SP_Range, 697, 704, 732
 SP_RangeBegin, 180, 207
 SP_RangeEnd, 181, 207
 SP_ReadMode, 255, 360, 433, 578
 SP_RearEdge_Seg1, 510, 522, 644, 658
 SP_RearEdge_Seg2, 510, 522, 644, 658
 SP_RearEdge_Seg3, 644, 658
 SP_RearEdge_Seg4, 645, 658
 SP_RecursiveCount, 257, 329, 443, 581, 797, 909
 SP_RecursiveCount1, 903
 SP_RecursiveCount2, 905
 SP_RefOffset, 498
 SP_RefractIndex, 363
 SP_RefractIndexFile, 375
 SP_RefractIndexFileIdx, 364, 374
 SP_RefractiveCorrection, 440
 SP_RefractiveIndex_nC, 440
 SP_RefractiveIndex_nd, 439, 440
 SP_RefractiveIndex_nF, 328, 440
 SP_RemoteAddress, 316, 425, 567, 781
 SP_ResampleAnalog, 260, 450, 599, 912
 SP_ResampleEthernet, 334, 450, 599, 802, 912
 SP_ResampleRS422, 260, 334, 450, 599, 801, 912
 SP_ResampleUSB, 912
 SP_Resampling, 261, 334, 450, 598, 801, 912
 SP_Reserve_10, 658
 SP_Reserve_2, 511, 517, 646, 653
 SP_Reserve_3, 512, 518, 656
 SP_Reserve_4, 521, 656
 SP_Reserve_6, 657
 SP_Reserve_7, 658
 SP_Reserve_9, 658
 SP_ResetMeasCounter, 229, 297
 SP_ResetTimestamp, 229, 296
 SP_ResetTriggerCounter, 297
 SP_Reverse, 369
 SP_ROIEnd, 256, 325, 435, 580
 SP_ROIStart, 256, 325, 434, 580
 SP_RS232_Baudrate, 512, 518, 646, 653
 SP_RS232_Parity, 512, 518, 647, 653
 SP_RS232_StopBits, 512, 518, 647, 653
 SP_RS232_TimeoutRecv, 512, 519, 647, 653
 SP_RS232_TimeoutSend, 512, 518, 647, 653
 SP_RS232Baudrate, 490, 497
 SP_RS232Mode, 496
 SP_RS422_Baudrate, 513, 519, 647, 654
 SP_RS422_Parity, 513, 519, 647, 654
 SP_RS422_StopBits, 513, 519, 647, 654
 SP_RS422_TimeoutRecv, 513, 519, 648, 654
 SP_RS422_TimeoutSend, 513, 519, 648, 654
 SP_RS422Baudrate, 849
 SP_RS422OutSource, 848
 SP_SampleTime, 701, 729, 825
 SP_SaveSettingsMode, 219
 SP_SaveTemporary, 785
 SP_ScaleFactor, 174, 197
 SP_SearchDirection, 571
 SP_SelectedSensor1..4, 842
 SP_Sensor, 347, 906
 SP_SensorAddress, 698, 832
 SP_SensorBaudrate, 170, 192, 219, 245, 277, 318, 351, 426, 569, 697, 782, 831, 900
 SP_SensorChannel, 841, 842, 856, 861
 SP_SensorChannelBaudrate, 842
 SP_SensorData, 682
 SP_SensorInterface, 831
 SP_SensorParamData, 683
 SP_SensorReset12, 856
 SP_SensorReset34, 857
 SP_SensorTable, 487
 SP_SerialOutFormat, 645, 652

SP_SettingName, 246, 247, 249
 SP_SetupRecord, 132
 SP_ShowChannels, 733
 SP_ShowLinearized, 733
 SP_ShutterFactor, 324
 SP_ShutterMode, 322, 430, 786
 SP_ShutterTime1, 323, 431
 SP_ShutterTime2, 323, 431
 SP_SlopeValue, 498
 SP_SmoothVideo, 497
 SP_SOD, 371
 SP_SoftwareTrigger, 880
 SP_SoftwareTriggerEnable, 880
 SP_SpectralAv, 362
 SP_Speed, 222, 280
 SP_SpikeCorrection, 330, 444, 583
 SP_SplitPixel, 486
 SP_SRIndex, 355, 675, 686, 716, 728
 SP_SSIFormat, 208
 SP_Stand-by, 157
 SP_Statistic2Signal, 586
 SP_StatisticDepth, 332, 447, 588, 800
 SP_StatisticSignal, 446, 584, 799
 SP_SubnetMask, 315, 424, 566, 707,
 736, 780, 829, 898
 SP_Sync_TrgMode, 279
 SP_SyncLogic, 884
 SP_SyncMasterChannel, 841, 856, 861
 SP_SyncMode, 295, 409, 560, 771, 884
 SP_SyncTermination, 295, 560
 SP_Target_Distance, 521
 SP_TargetMode, 252
 SP_TeachValue, 492
 SP_TeachValue1, 225
 SP_TeachValue2, 225
 SP_TempCoeffParam, 723, 724
 SP_TemperatureCoeffizient, 723
 SP_TerminationChar, 200
 SP_TextField, 671
 SP_Threshold, 223, 326, 361, 436, 580
 SP_Threshold1, 361
 SP_Threshold2, 362
 SP_TimerFrequency, 844, 864
 SP_TimerNumber, 844, 845, 863
 SP_TimerRatio, 844, 864
 SP_ToleranceHigh, 492
 SP_ToleranceLow, 492
 SP_ToleranceRange, 331, 445, 583
 SP_TransmitIntensity, 289, 368
 SP_TrgChannel1..4, 847
 SP_TrgChannel1..6, 873
 SP_TrgMode, 687, 702, 717, 729, 825
 SP_TrgValue1..6, 874
 SP_TriggerCount, 244, 314, 415, 565,
 779, 897
 SP_TriggerDelay, 168, 190
 SP_TriggerEdge, 168, 190
 SP_TriggerLevel, 313, 414, 565, 778, 897
 SP_TriggerLogic, 897
 SP_TriggerMode, 190, 243, 312, 413,
 493, 564, 778, 896
 SP_TriggerMode_Edge, 349
 SP_TriggerMode_State, 350
 SP_TriggerMoment, 244, 313, 414, 564
 SP_TriggerOutput, 314
 SP_TriggerTermination, 312, 564
 SP_TxDChannel1..4, 846
 SP_TxDChannel1..6, 872
 SP_TxValue1..6, 872
 SP_UnlinearizedMode, 476
 SP_Upper_hysteresis, 283
 SP_Upper_limit, 282
 SP_UpperBound1, 589
 SP_UpperBound2, 590
 SP_UpperLimit, 267, 481, 520, 634, 655
 SP_UpperWarning, 520, 655
 SP_UsedColorSpace, 792
 SP_UserMeasProgNumber, 519, 654
 SP_ValidRange, 176
 SP_VideoAverage, 325, 435, 796
 SP_VideoMode, 280
 SP_VideoThrAuto, 496
 SP_VideoThrFix, 496
 SP_VideoThrMode, 493
 SP_WaitSpectrumTimeout, 360
 SP_WaitVideoTimeout, 255, 433, 579
 SP_Watchdog, 353
 SP_WatchdogPeriod, 353
 SP_Weighting, 348
 SP_WindowMax, 202
 SP_WindowMin, 202
 SP_X, 793
 SP_X1..16, 370
 SP_Y, 793
 SP_Z, 793
 SP_Zero, 283, 373
 timestamp, 75
 TransferData, 73
 TransferDataTS, 74
 variable argument list, 84, 85
 versionStr, 78, 79
 XP_FirmwareFile, 923