

MSX-E173x soap api functions

Generated by Doxygen 1.7.1

Fri Sep 4 2015 16:30:07

Contents

1	Introduction	1
1.1	Introduction	1
1.2	Remark: SOAP functions prototypes	1
2	Module Documentation	3
2.1	MSX-E173x compatibility functions	3
2.2	MSX-E17xx functions	3
2.3	MSX-E17xx multifunction functions	3
2.4	Common functions	3
2.5	Common general functions	4
2.5.1	Function Documentation	5
2.5.1.1	MXCommon__GetModuleType	5
2.5.1.2	MXCommon__GetHostname	6
2.5.1.3	MXCommon__SetHostname	6
2.5.1.4	MXCommon__GetClientConnections	6
2.5.1.5	MXCommon__Sterror	7
2.5.1.6	MXCommon__Reboot	8
2.5.1.7	MXCommon__ResetAllIOFunctionalities	8
2.5.1.8	MXCommon__DataseverRestart	9
2.5.1.9	MXCommon__GetEthernetLinksStates	9
2.6	Common temperature functions	10
2.6.1	Detailed Description	10
2.6.2	Function Documentation	11
2.6.2.1	MXCommon__GetModuleTemperatureValueAndStatus	11
2.6.2.2	MXCommon__SetModuleTemperatureWarningLevels	11
2.7	Common hardware trigger functions	12
2.7.1	Function Documentation	12
2.7.1.1	MXCommon__SetHardwareTriggerFilterTime	12

2.7.1.2	MXCommon__GetHardwareTriggerFilterTime	13
2.7.1.3	MXCommon__GetHardwareTriggerState	13
2.8	Common security functions	14
2.8.1	Detailed Description	14
2.8.2	Function Documentation	15
2.8.2.1	MXCommon__SetCustomerKey	15
2.8.2.2	MXCommon__TestCustomerID	15
2.9	Common time functions	16
2.9.1	Detailed Description	16
2.9.2	Function Documentation	16
2.9.2.1	MXCommon__SetTime	16
2.9.2.2	MXCommon__SysToHardwareClock	17
2.9.2.3	MXCommon__HardwareClockToSys	17
2.9.2.4	MXCommon__GetTime	18
2.9.2.5	MXCommon__GetUpTime	18
2.10	Common I/O auto configuration functions	18
2.10.1	Detailed Description	19
2.10.2	Function Documentation	19
2.10.2.1	MXCommon__GetAutoConfigurationFile	19
2.10.2.2	MXCommon__SetAutoConfigurationFile	20
2.10.2.3	MXCommon__StartAutoConfiguration	20
2.11	Common synchronisation timer functions	20
2.11.1	Function Documentation	21
2.11.1.1	MXCommon__InitAndStartSynchroTimer	21
2.11.1.2	MXCommon__StopAndReleaseSynchroTimer	22
2.12	Set/Backup/Restore general system configuration	22
2.12.1	Detailed Description	23
2.12.2	Function Documentation	23
2.12.2.1	MXCommon__GetConfigurationBackupFile	23
2.12.2.2	MXCommon__ApplyConfigurationBackupFile	24
2.12.2.3	MXCommon__ChangePassword	24
2.13	System state management	25
2.13.1	Detailed Description	25
2.13.2	Function Documentation	25
2.13.2.1	MXCommon__GetSubSystemState	25
2.13.2.2	MXCommon__GetSubsystemIDFromName	26

2.13.2.3	MXCommon__GetStateIDFromName	26
2.13.2.4	MXCommon__GetSubsystemNameFromID	27
2.13.2.5	MXCommon__GetStateNameFromID	27
2.14	Customer option management	27
2.14.1	Function Documentation	28
2.14.1.1	MXCommon__GetOptionInformation	28
2.15	Synchronisation management	28
2.15.1	Function Documentation	28
2.15.1.1	MXCommon__SetToMaster	28
2.15.1.2	MXCommon__GetSynchronizationStatus	29
2.16	input filter Filter management	29
2.16.1	Function Documentation	30
2.16.1.1	MXCommon__SetFilterChannels	30
2.17	MSX-E173x digital I/O functions	30
2.17.1	Function Documentation	31
2.17.1.1	MSXE173x__DigitalIOGetNumber	31
2.17.1.2	MSXE173x__DigitalIOInitPortConfiguration	31
2.17.1.3	MSXE173x__DigitalIOReadChannelValue	32
2.17.1.4	MSXE173x__DigitalIOReadAllChannelsValue	32
2.17.1.5	MSXE173x__DigitalIOWriteChannelValue	32
2.17.1.6	MSXE173x__DigitalIOWriteAllChannelsValue	33
2.17.1.7	MSXE173x__DigitalIOReleasePortConfiguration	33
2.17.1.8	MSXE173x__DigitalIOTestShortCircuit	34
2.17.1.9	MSXE173x__DigitalIORearmShortCircuit	34
2.18	MSX-E173x IO watchdog functions	34
2.18.1	Function Documentation	35
2.18.1.1	MSXE173x__IOWatchdogInitAndStart	35
2.18.1.2	MSXE173x__IOWatchdogStopAndRelease	35
2.18.1.3	MSXE173x__IOWatchdogGetStatusAndValue	36
2.19	MSX-E173x multifunction common functions	36
2.19.1	Function Documentation	37
2.19.1.1	MSXE173x__MFCommonSetInputsFilter	37
2.19.1.2	MSXE173x__MFCommonReferenceVoltageActivation	38
2.20	MSX-E173x ENDAT functions	38
2.20.1	Function Documentation	41
2.20.1.1	MSXE173x__MFEndatInitSensor	41

2.20.1.2	MSXE173x__MFEndatGetPosition	42
2.20.1.3	MSXE173x__MFEndatGetSensorProperties	43
2.20.1.4	MSXE173x__MFEndatGetErrorSources	45
2.20.1.5	MSXE173x__MFEndatResetErrorBits	46
2.20.1.6	MSXE173x__MFEndatSensorReceiveReset	46
2.20.1.7	MSXE173x__MFEndatSelectMemoryArea	47
2.20.1.8	MSXE173x__MFEndatSensorSendParameter	48
2.20.1.9	MSXE173x__MFEndatSensorReceiveParameter	49
2.20.1.10	MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea	50
2.20.1.11	MSXE173x__MFEndatSelectAdditionalData	52
2.20.1.12	MSXE173x__MFEndatGetPositionWithAddData	54
2.20.1.13	MSXE173x__MFEndatGetSelectedAdditionalData	55
2.20.1.14	MSXE173x__MFEndatInitAndEnableLatchPositionValues	55
2.20.1.15	MSXE173x__MFEndatGetCurrentLatchConfiguration	58
2.20.1.16	MSXE173x__MFEndatDisableAndReleaseLatchPositionValues	60
2.21	MSX-E17xx digital I/O functions	60
2.21.1	Function Documentation	61
2.21.1.1	MSXE17xx__DigitalIOGetNumber	61
2.21.1.2	MSXE17xx__DigitalIOInitPortConfiguration	62
2.21.1.3	MSXE17xx__DigitalIOReadChannelValue	62
2.21.1.4	MSXE17xx__DigitalIOReadAllChannelsValue	63
2.21.1.5	MSXE17xx__DigitalIOWriteChannelValue	63
2.21.1.6	MSXE17xx__DigitalIOWriteAllChannelsValue	63
2.21.1.7	MSXE17xx__DigitalIOReleasePortConfiguration	64
2.21.1.8	MSXE17xx__DigitalIOTestShortCircuit	64
2.21.1.9	MSXE17xx__DigitalIORearmShortCircuit	65
2.22	MSX-E17xx IO watchdog functions	65
2.22.1	Function Documentation	65
2.22.1.1	MSXE17xx__IOWatchdogInitAndStart	65
2.22.1.2	MSXE17xx__IOWatchdogStopAndRelease	66
2.22.1.3	MSXE17xx__IOWatchdogGetStatusAndValue	66
2.23	MSX-E17xx multifunction common functions	67
2.23.1	Function Documentation	67
2.23.1.1	MSXE17xx__MFCommonGetSubModuleFunctionality	67
2.23.1.2	MSXE17xx__MFCommonSetInputsFilter	68
2.23.1.3	MSXE17xx__MFCommonReferenceVoltageActivation	69

2.23.1.4	MSXE17xx__MFCommonSetFIFO0Level	70
2.24	MSX-E17xx ENDAT functions	70
2.24.1	Function Documentation	73
2.24.1.1	MSXE17xx__MFEndatInitSensor	73
2.24.1.2	MSXE17xx__MFEndatGetPosition	74
2.24.1.3	MSXE17xx__MFEndatGetSensorProperties	75
2.24.1.4	MSXE17xx__MFEndatGetErrorSources	77
2.24.1.5	MSXE17xx__MFEndatResetErrorBits	78
2.24.1.6	MSXE17xx__MFEndatSensorReceiveReset	78
2.24.1.7	MSXE17xx__MFEndatSelectMemoryArea	79
2.24.1.8	MSXE17xx__MFEndatSensorSendParameter	80
2.24.1.9	MSXE17xx__MFEndatSensorReceiveParameter	81
2.24.1.10	MSXE17xx__MFEndatSensorSendPosAndRecvSelMemArea	82
2.24.1.11	MSXE17xx__MFEndatSelectAdditionalData	84
2.24.1.12	MSXE17xx__MFEndatGetPositionWithAddData	86
2.24.1.13	MSXE17xx__MFEndatGetSelectedAdditionalData	87
2.24.1.14	MSXE17xx__MFEndatInitAndEnableLatchPositionValues	87
2.24.1.15	MSXE17xx__MFEndatGetCurrentLatchConfiguration	90
2.24.1.16	MSXE17xx__MFEndatDisableAndReleaseLatchPositionValues	92
2.25	Software hints	92
2.26	SOAP function calls in C/C++ language	92
2.26.1	C/C++ language SOAP function prototypes	93
2.26.2	Rules and use of gSOAP calls in C/C++	93
2.26.3	SOAP calls sample	95
2.27	MSX-E systems servers	99
2.27.1	SOAP server	99
2.27.2	Event server	99
2.27.3	Modbus server	99
3	Data Structure Documentation	101
3.1	ByteArray Struct Reference	101
3.1.1	Field Documentation	101
3.1.1.1	__ptr	101
3.1.1.2	__size	101
3.1.1.3	__offset	101
3.2	DefaultResponse Struct Reference	101
3.2.1	Field Documentation	102

3.2.1.1	iReturnValue	102
3.2.1.2	syserrno	102
3.3	MSXE173x__DigitalIOGetNumberResponse Struct Reference	102
3.3.1	Detailed Description	102
3.3.2	Field Documentation	103
3.3.2.1	sResponse	103
3.3.2.2	ulNumberOfDigitalIO	103
3.4	MSXE173x__IOWatchdogGetStatusAndValueResponse Struct Reference	103
3.4.1	Field Documentation	103
3.4.1.1	sResponse	103
3.4.1.2	ulStatus	103
3.4.1.3	ulValue	103
3.4.1.4	ulInfo	103
3.5	MSXE173x__MFEndatGetCurrentLatchConfigurationResponse Struct Reference	103
3.5.1	Field Documentation	105
3.5.1.1	sResponse	105
3.5.1.2	ulRunning	105
3.5.1.3	ulLatchSource	105
3.5.1.4	ulTriggerEdgeCount	105
3.5.1.5	ulDataFormat	105
3.5.1.6	ulModel	105
3.5.1.7	ulPositionSize	105
3.5.1.8	ulSignalPeriod	105
3.5.1.9	ulStepPerRevolution	105
3.5.1.10	ulNumberOfRevolution	105
3.5.1.11	ulScalingFactor	105
3.6	MSXE173x__MFEndatGetErrorSourcesResponse Struct Reference	105
3.6.1	Field Documentation	106
3.6.1.1	sResponse	106
3.6.1.2	ulErrorSrc	106
3.7	MSXE173x__MFEndatGetPositionResponse Struct Reference	106
3.7.1	Field Documentation	106
3.7.1.1	sResponse	106
3.7.1.2	ulPositionLow	106
3.7.1.3	ulPositionHigh	106
3.8	MSXE173x__MFEndatGetPositionWithAddDataResponse Struct Reference	106

3.8.1	Field Documentation	107
3.8.1.1	sResponse	107
3.8.1.2	ulPositionLow	107
3.8.1.3	ulPositionHigh	107
3.8.1.4	ulAddData1	107
3.8.1.5	ulAddData2	107
3.9	MSXE173x__MFEndatGetSelectedAdditionalDataResponse Struct Reference	107
3.9.1	Field Documentation	108
3.9.1.1	sResponse	108
3.9.1.2	ulAdCount	108
3.9.1.3	ulMrsCodeAD1	108
3.9.1.4	ulMrsCodeAD2	108
3.10	MSXE173x__MFEndatGetSensorPropertiesResponse Struct Reference	108
3.10.1	Field Documentation	110
3.10.1.1	sResponse	110
3.10.1.2	ulIDNumberLsb	110
3.10.1.3	ulIDNumberMsb	110
3.10.1.4	ulSerialNumberLsb	110
3.10.1.5	ulSerialNumberMsb	110
3.10.1.6	ulModel	110
3.10.1.7	ulMode	110
3.10.1.8	ulPositionSize	110
3.10.1.9	ulSignalPeriod	110
3.10.1.10	ulStepPerRevolution	110
3.10.1.11	ulNumberOfRevolution	110
3.10.1.12	ulScalingFactor	110
3.10.1.13	ulAdditionalData	110
3.11	MSXE173x__MFEndatSensorSendParameterResponse Struct Reference	110
3.11.1	Field Documentation	111
3.11.1.1	sResponse	111
3.11.1.2	ulParam	111
3.12	MSXE173x__MFEndatSensorSendPosAndRecvSelMemAreaResponse Struct Reference	111
3.12.1	Field Documentation	111
3.12.1.1	sResponse	111
3.12.1.2	ulPositionLow	111
3.12.1.3	ulPositionHigh	111

3.13 MSXE173x__Response Struct Reference	111
3.13.1 Field Documentation	111
3.13.1.1 iReturnValue	111
3.13.1.2 syserrno	112
3.14 MSXE173x__unsignedLongResponse Struct Reference	112
3.14.1 Field Documentation	112
3.14.1.1 sResponse	112
3.14.1.2 ulValue	112
3.15 MSXE173x__unsignedLongTimeStampResponse Struct Reference	112
3.15.1 Field Documentation	113
3.15.1.1 sResponse	113
3.15.1.2 ulValue	113
3.15.1.3 ulTimeStampLow	113
3.15.1.4 ulTimeStampHigh	113
3.16 MSXE17xx__DigitalIOGetNumberResponse Struct Reference	113
3.16.1 Field Documentation	113
3.16.1.1 sResponse	113
3.16.1.2 ulNumberOfDigitalIO	113
3.17 MSXE17xx__IOWatchdogGetStatusAndValueResponse Struct Reference	113
3.17.1 Field Documentation	114
3.17.1.1 sResponse	114
3.17.1.2 ulStatus	114
3.17.1.3 ulValue	114
3.17.1.4 ulInfo	114
3.18 MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse Struct Reference	114
3.18.1 Field Documentation	115
3.18.1.1 sResponse	115
3.18.1.2 ulRunning	115
3.18.1.3 ulLatchSource	115
3.18.1.4 ulTriggerEdgeCount	115
3.18.1.5 ulDataFormat	115
3.18.1.6 ulModel	115
3.18.1.7 ulPositionSize	115
3.18.1.8 ulSignalPeriod	115
3.18.1.9 ulStepPerRevolution	115
3.18.1.10 ulNumberOfRevolution	115

3.18.1.11	ulScalingFactor	115
3.19	MSXE17xx__MFEndatGetErrorSourcesResponse Struct Reference	115
3.19.1	Field Documentation	116
3.19.1.1	sResponse	116
3.19.1.2	ulErrorSrc	116
3.20	MSXE17xx__MFEndatGetPositionResponse Struct Reference	116
3.20.1	Field Documentation	116
3.20.1.1	sResponse	116
3.20.1.2	ulPositionLow	116
3.20.1.3	ulPositionHigh	116
3.21	MSXE17xx__MFEndatGetPositionWithAddDataResponse Struct Reference	116
3.21.1	Field Documentation	117
3.21.1.1	sResponse	117
3.21.1.2	ulPositionLow	117
3.21.1.3	ulPositionHigh	117
3.21.1.4	ulAddData1	117
3.21.1.5	ulAddData2	117
3.22	MSXE17xx__MFEndatGetSelectedAdditionalDataResponse Struct Reference	117
3.22.1	Field Documentation	118
3.22.1.1	sResponse	118
3.22.1.2	ulAdCount	118
3.22.1.3	ulMrsCodeAD1	118
3.22.1.4	ulMrsCodeAD2	118
3.23	MSXE17xx__MFEndatGetSensorPropertiesResponse Struct Reference	118
3.23.1	Field Documentation	120
3.23.1.1	sResponse	120
3.23.1.2	ulIDNumberLsb	120
3.23.1.3	ulIDNumberMsb	120
3.23.1.4	ulSerialNumberLsb	120
3.23.1.5	ulSerialNumberMsb	120
3.23.1.6	ulModel	120
3.23.1.7	ulMode	120
3.23.1.8	ulPositionSize	120
3.23.1.9	ulSignalPeriod	120
3.23.1.10	ulStepPerRevolution	120
3.23.1.11	ulNumberOfRevolution	120

3.23.1.12	ulScalingFactor	120
3.23.1.13	ulAdditionalData	120
3.24	MSXE17xx__MFEndatSensorSendParameterResponse Struct Reference	120
3.24.1	Field Documentation	121
3.24.1.1	sResponse	121
3.24.1.2	ulParam	121
3.25	MSXE17xx__MFEndatSensorSendPosAndRecvSelMemAreaResponse Struct Reference	121
3.25.1	Field Documentation	121
3.25.1.1	sResponse	121
3.25.1.2	ulPositionLow	121
3.25.1.3	ulPositionHigh	121
3.26	MSXE17xx__Response Struct Reference	121
3.26.1	Field Documentation	121
3.26.1.1	iReturnValue	121
3.26.1.2	syserrno	122
3.27	MSXE17xx__unsignedLongResponse Struct Reference	122
3.27.1	Field Documentation	122
3.27.1.1	sResponse	122
3.27.1.2	ulValue	122
3.28	MSXE17xx__unsignedLongTimeStampResponse Struct Reference	122
3.28.1	Field Documentation	123
3.28.1.1	sResponse	123
3.28.1.2	ulValue	123
3.28.1.3	ulTimeStampLow	123
3.28.1.4	ulTimeStampHigh	123
3.29	MXCommon__ByteArrayResponse Struct Reference	123
3.29.1	Field Documentation	123
3.29.1.1	sResponse	123
3.29.1.2	sArray	123
3.30	MXCommon__FileResponse Struct Reference	123
3.30.1	Field Documentation	124
3.30.1.1	sResponse	124
3.30.1.2	sArray	124
3.30.1.3	ulEOF	124
3.31	MXCommon__GetAutoConfigurationFileResponse Struct Reference	124
3.31.1	Field Documentation	124

3.31.1.1	sResponse	124
3.31.1.2	bArray	124
3.31.1.3	ulEOF	124
3.32	MXCommon__GetEthernetLinksStatesResponse Struct Reference	124
3.32.1	Field Documentation	125
3.32.1.1	sResponse	125
3.32.1.2	sPort0	125
3.32.1.3	sPort1	125
3.33	MXCommon__GetHardwareTriggerFilterTimeResponse Struct Reference	125
3.33.1	Field Documentation	125
3.33.1.1	sResponse	125
3.33.1.2	ulFilterTime	125
3.33.1.3	ulInfo01	125
3.33.1.4	ulInfo02	125
3.34	MXCommon__GetHardwareTriggerStateResponse Struct Reference	125
3.34.1	Field Documentation	126
3.34.1.1	sResponse	126
3.34.1.2	ulState	126
3.34.1.3	ulInfo01	126
3.34.1.4	ulInfo02	126
3.35	MXCommon__GetModuleTemperatureValueAndStatusResponse Struct Reference	126
3.35.1	Field Documentation	127
3.35.1.1	sResponse	127
3.35.1.2	dTemperatureValue	127
3.35.1.3	ulTemperatureStatus	127
3.35.1.4	ulInfo	127
3.36	MXCommon__GetTimeResponse Struct Reference	127
3.36.1	Field Documentation	127
3.36.1.1	sResponse	127
3.36.1.2	ulLowTime	127
3.36.1.3	ulHighTime	127
3.37	MXCommon__GetUpTimeResponse Struct Reference	127
3.37.1	Field Documentation	128
3.37.1.1	sResponse	128
3.37.1.2	ulUpTime	128
3.38	MXCommon__Response Struct Reference	128

3.38.1	Field Documentation	128
3.38.1.1	iReturnValue	128
3.38.1.2	syserrno	128
3.39	MXCommon__TestCustomerIDResponse Struct Reference	128
3.39.1	Field Documentation	129
3.39.1.1	sResponse	129
3.39.1.2	bValueArray	129
3.39.1.3	bCryptedValueArray	129
3.40	MXCommon__unsignedLongResponse Struct Reference	129
3.40.1	Field Documentation	129
3.40.1.1	sResponse	129
3.40.1.2	ulValue	129
3.41	sGetEthernetLinksStatesPort Struct Reference	129
3.41.1	Field Documentation	130
3.41.1.1	ulState	130
3.41.1.2	ulSpeed	130
3.41.1.3	ulDuplex	130
3.41.1.4	ulInfo1	130
3.41.1.5	ulInfo2	130
3.42	UnsignedLongArray Struct Reference	130
3.42.1	Field Documentation	130
3.42.1.1	__ptr	130
3.42.1.2	__size	130
3.42.1.3	__offset	130
3.43	UnsignedShortArray Struct Reference	130
3.43.1	Field Documentation	131
3.43.1.1	__ptr	131
3.43.1.2	__size	131
3.43.1.3	__offset	131
3.44	xsd__base64Binary Struct Reference	131
3.44.1	Field Documentation	131
3.44.1.1	__ptr	131
3.44.1.2	__size	131
4	File Documentation	133
4.1	MSXE173x_public_doc.h File Reference	133
4.1.1	Define Documentation	146

4.1.1.1	MSXE170X_COUNTER_QUADRUPLE_MODE	146
4.1.1.2	MSXE170X_COUNTER_DOUBLE_MODE	146
4.1.1.3	MSXE170X_COUNTER_SIMPLE_MODE	146
4.1.1.4	MSXE170X_COUNTER_DIRECT_MODE	146
4.1.1.5	MSXE170X_COUNTER_HYSTERESIS_ON	146
4.1.1.6	MSXE170X_COUNTER_HYSTERESIS_OFF	147
4.1.1.7	MSXE170X_COUNTER_INCREMENT	147
4.1.1.8	MSXE170X_COUNTER_DECREMENT	147
4.1.1.9	MSXE170X_COUNTER_LOW_EDGE_LATCH_AND_CLEAR_- COUNTER	147
4.1.1.10	MSXE170X_COUNTER_HIGH_EDGE_LATCH_AND_CLEAR_- COUNTER	147
4.1.1.11	MSXE170X_COUNTER_LOW_EDGE_LATCH_COUNTER	147
4.1.1.12	MSXE170X_COUNTER_HIGH_EDGE_LATCH_COUNTER	147
4.1.2	Typedef Documentation	147
4.1.2.1	xsd_string	147
4.1.2.2	xsd_char	147
4.1.2.3	xsd_float	147
4.1.2.4	xsd_double	147
4.1.2.5	xsd_int	147
4.1.2.6	xsd_long	147
4.1.2.7	xsd_unsignedByte	147
4.1.2.8	xsd_unsignedInt	147
4.1.2.9	xsd_unsignedShort	147
4.1.2.10	xsd_unsignedLong	147
4.1.3	Function Documentation	147
4.1.3.1	MXCommon__GetModuleType	147
4.1.3.2	MXCommon__GetHostname	148
4.1.3.3	MXCommon__SetHostname	148
4.1.3.4	MXCommon__GetClientConnections	148
4.1.3.5	MXCommon__Sterror	149
4.1.3.6	MXCommon__Reboot	150
4.1.3.7	MXCommon__ResetAllIOFunctionalities	150
4.1.3.8	MXCommon__DataseverRestart	151
4.1.3.9	MXCommon__GetEthernetLinksStates	151
4.1.3.10	MXCommon__GetModuleTemperatureValueAndStatus	152
4.1.3.11	MXCommon__SetModuleTemperatureWarningLevels	153

4.1.3.12	MXCommon__SetHardwareTriggerFilterTime	153
4.1.3.13	MXCommon__GetHardwareTriggerFilterTime	154
4.1.3.14	MXCommon__GetHardwareTriggerState	154
4.1.3.15	MXCommon__SetCustomerKey	155
4.1.3.16	MXCommon__TestCustomerID	155
4.1.3.17	MXCommon__SetTime	155
4.1.3.18	MXCommon__SysToHardwareClock	156
4.1.3.19	MXCommon__HardwareClockToSys	156
4.1.3.20	MXCommon__GetTime	157
4.1.3.21	MXCommon__GetUpTime	157
4.1.3.22	MXCommon__GetAutoConfigurationFile	157
4.1.3.23	MXCommon__SetAutoConfigurationFile	158
4.1.3.24	MXCommon__StartAutoConfiguration	158
4.1.3.25	MXCommon__InitAndStartSynchroTimer	158
4.1.3.26	MXCommon__StopAndReleaseSynchroTimer	159
4.1.3.27	MXCommon__GetConfigurationBackupFile	160
4.1.3.28	MXCommon__ApplyConfigurationBackupFile	161
4.1.3.29	MXCommon__ChangePassword	161
4.1.3.30	MXCommon__GetSubSystemState	162
4.1.3.31	MXCommon__GetSubsystemIDFromName	162
4.1.3.32	MXCommon__GetStateIDFromName	162
4.1.3.33	MXCommon__GetSubsystemNameFromID	163
4.1.3.34	MXCommon__GetStateNameFromID	163
4.1.3.35	MXCommon__GetOptionInformation	164
4.1.3.36	MXCommon__SetToMaster	164
4.1.3.37	MXCommon__GetSynchronizationStatus	164
4.1.3.38	MXCommon__SetFilterChannels	165
4.1.3.39	MSXE173x__DigitalIOGetNumber	165
4.1.3.40	MSXE173x__DigitalIOInitPortConfiguration	165
4.1.3.41	MSXE173x__DigitalIOReadChannelValue	166
4.1.3.42	MSXE173x__DigitalIOReadAllChannelsValue	166
4.1.3.43	MSXE173x__DigitalIOWriteChannelValue	167
4.1.3.44	MSXE173x__DigitalIOWriteAllChannelsValue	167
4.1.3.45	MSXE173x__DigitalIOReleasePortConfiguration	167
4.1.3.46	MSXE173x__DigitalIOTestShortCircuit	168
4.1.3.47	MSXE173x__DigitalIORearmShortCircuit	168

4.1.3.48	MSXE173x__IOWatchdogInitAndStart	169
4.1.3.49	MSXE173x__IOWatchdogStopAndRelease	169
4.1.3.50	MSXE173x__IOWatchdogGetStatusAndValue	169
4.1.3.51	MSXE173x__MFCommonSetInputsFilter	170
4.1.3.52	MSXE173x__MFCommonReferenceVoltageActivation	171
4.1.3.53	MSXE173x__MFEndatInitSensor	172
4.1.3.54	MSXE173x__MFEndatGetPosition	173
4.1.3.55	MSXE173x__MFEndatGetSensorProperties	174
4.1.3.56	MSXE173x__MFEndatGetErrorSources	176
4.1.3.57	MSXE173x__MFEndatResetErrorBits	177
4.1.3.58	MSXE173x__MFEndatSensorReceiveReset	177
4.1.3.59	MSXE173x__MFEndatSelectMemoryArea	178
4.1.3.60	MSXE173x__MFEndatSensorSendParameter	179
4.1.3.61	MSXE173x__MFEndatSensorReceiveParameter	180
4.1.3.62	MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea	181
4.1.3.63	MSXE173x__MFEndatSelectAdditionalData	183
4.1.3.64	MSXE173x__MFEndatGetPositionWithAddData	185
4.1.3.65	MSXE173x__MFEndatGetSelectedAdditionalData	186
4.1.3.66	MSXE173x__MFEndatInitAndEnableLatchPositionValues	186
4.1.3.67	MSXE173x__MFEndatGetCurrentLatchConfiguration	189
4.1.3.68	MSXE173x__MFEndatDisableAndReleaseLatchPositionValues	191
4.1.3.69	MSXE17xx__DigitalIOGetNumber	191
4.1.3.70	MSXE17xx__DigitalIOInitPortConfiguration	192
4.1.3.71	MSXE17xx__DigitalIOReadChannelValue	192
4.1.3.72	MSXE17xx__DigitalIOReadAllChannelsValue	193
4.1.3.73	MSXE17xx__DigitalIOWriteChannelValue	193
4.1.3.74	MSXE17xx__DigitalIOWriteAllChannelsValue	193
4.1.3.75	MSXE17xx__DigitalIOReleasePortConfiguration	194
4.1.3.76	MSXE17xx__DigitalIOTestShortCircuit	194
4.1.3.77	MSXE17xx__DigitalIORearmShortCircuit	195
4.1.3.78	MSXE17xx__IOWatchdogInitAndStart	195
4.1.3.79	MSXE17xx__IOWatchdogStopAndRelease	195
4.1.3.80	MSXE17xx__IOWatchdogGetStatusAndValue	196
4.1.3.81	MSXE17xx__MFCommonGetSubModuleFunctionality	196
4.1.3.82	MSXE17xx__MFCommonSetInputsFilter	197
4.1.3.83	MSXE17xx__MFCommonReferenceVoltageActivation	198

4.1.3.84	MSXE17xx__MFCommonSetFIFO0Level	199
4.1.3.85	MSXE17xx__MFEndatInitSensor	199
4.1.3.86	MSXE17xx__MFEndatGetPosition	200
4.1.3.87	MSXE17xx__MFEndatGetSensorProperties	201
4.1.3.88	MSXE17xx__MFEndatGetErrorSources	203
4.1.3.89	MSXE17xx__MFEndatResetErrorBits	204
4.1.3.90	MSXE17xx__MFEndatSensorReceiveReset	205
4.1.3.91	MSXE17xx__MFEndatSelectMemoryArea	206
4.1.3.92	MSXE17xx__MFEndatSensorSendParameter	207
4.1.3.93	MSXE17xx__MFEndatSensorReceiveParameter	208
4.1.3.94	MSXE17xx__MFEndatSensorSendPosAndRecvSelMemArea	209
4.1.3.95	MSXE17xx__MFEndatSelectAdditionalData	210
4.1.3.96	MSXE17xx__MFEndatGetPositionWithAddData	212
4.1.3.97	MSXE17xx__MFEndatGetSelectedAdditionalData	213
4.1.3.98	MSXE17xx__MFEndatInitAndEnableLatchPositionValues	214
4.1.3.99	MSXE17xx__MFEndatGetCurrentLatchConfiguration	217
4.1.3.100	MSXE17xx__MFEndatDisableAndReleaseLatchPositionValues	218

Chapter 1

Introduction

MainRevision:

1.1 Introduction

This documentation describes the SOAP functions and gives software hints to work with the MSX-E systems. Following documentations can be found under **Modules**.

SOAP means Simple Object Access Protocol. This protocol enables to use the MSX-E software functions over Ethernet. It is providing **Web Services** that can easily be consumed in many programming languages like C, C++, C#, VB.Net... With the SOAP functions, all functionalities of the MSX-E system can be managed / configured / monitored.

1.2 Remark: SOAP functions prototypes

In some programming languages, SOAP functions names and parameters could be different as those described in this documentation. Please see to [Software hints](#)

Chapter 2

Module Documentation

2.1 MSX-E173x compatibility functions

Modules

- [MSX-E173x digital I/O functions](#)
- [MSX-E173x IO watchdog functions](#)
- [MSX-E173x multifunction common functions](#)
- [MSX-E173x ENDAT functions](#)

2.2 MSX-E17xx functions

Modules

- [MSX-E17xx multifunction functions](#)
- [MSX-E17xx digital I/O functions](#)
- [MSX-E17xx IO watchdog functions](#)
- [MSX-E17xx ENDAT functions](#)

2.3 MSX-E17xx multifunction functions

Modules

- [MSX-E17xx multifunction common functions](#)

2.4 Common functions

Modules

- [Common general functions](#)

Various utility functions, mainly to identify a remote system.

- [Common temperature functions](#)

These functions deals with the internal temperature sub-system.

- [Common hardware trigger functions](#)

These functions allow to set and request the current value of the hardware trigger.

- [Common security functions](#)

The "customer key" feature may for instance be used by a customer to be sure that his application communicates only with certified MSX-E modules.

- [Common time functions](#)

A MSX-E module provides a "system clock" that may be in the simplest case set by the function [MXCommon__SetTime\(\)](#).

- [Common I/O auto configuration functions](#)

On the web site of some MSX-E module, there is the possibility to define an auto-configuration and auto start of the I/O.

- [Common synchronisation timer functions](#)

When modules are linked through a "synchronisation bus", the master can run a timer that generate a "synchro signal" on the slaves when overrun.

- [Set/Backup/Restore general system configuration](#)

Distinct of the I/O auto-configuration/auto-start functionality, these functions allows to manipulate the general system configuration.

- [System state management](#)

Every MSX-E modules are composed of several sub-systems that work together to provide the system functionalities.

- [Customer option management](#)

Enable to get informations about the options of the system.

- [Synchronisation management](#)

Manage the synchronisation state of the system.

- [input filter Filter management](#)

Manages the analog input filters in the system.

2.5 Common general functions

Various utility functions, mainly to identify a remote system.

Functions

- [int MXCommon__GetModuleType](#) (void *__, struct [MXCommon__ByteArrayResponse](#) *Response)

This function return the type of the MSX-E Module.

- int [MXCommon__GetHostname](#) (void *_ , struct [MXCommon__ByteArrayResponse](#) *Response)
This function return the hostname of the MSX-E Module.
- int [MXCommon__SetHostname](#) (struct [xsd__base64Binary](#) *bHostname, struct [MXCommon__Response](#) *Response)
This function allows to set the hostname of the MSX-E Module.
- int [MXCommon__GetClientConnections](#) (void *_ , struct [MXCommon__ByteArrayResponse](#) *Response)
This function return the client connection list.
- int [MXCommon__Strerror](#) (xsd__int errnum, struct [MXCommon__ByteArrayResponse](#) *Response)
Call the libc strerror() on the remote device (actually this is a call to strerror_r()).
- int [MXCommon__Reboot](#) (void *_ , struct [MXCommon__Response](#) *Response)
Ask the MSX-E module to reboot.
- int [MXCommon__ResetAllIOFunctionalities](#) (xsd__unsignedLong ulOption, struct [MXCommon__Response](#) *Response)
Reset the I/O functionalities of the MSX-E system.
- int [MXCommon__DataseverRestart](#) (xsd__unsignedLong ulAction, xsd__unsignedLong ulOption, struct [MXCommon__Response](#) *Response)
Restart the data-server service.
- int [MXCommon__GetEthernetLinksStates](#) (void *_ , struct [MXCommon__GetEthernetLinksStatesResponse](#) *Response)
Get MSX-E Ethernet links states.

2.5.1 Function Documentation

2.5.1.1 int [MXCommon__GetModuleType](#) (void * _ , struct [MXCommon__ByteArrayResponse](#) * *Response*)

Parameters

- [in] _ : no input parameter
- [out] *Response* • sArray : Module type string
• sResponse Composed of iReturnValue and syserrno

Return values

- SOAP_OK* SOAP call success
- otherwise* SOAP protocol error

2.5.1.2 int MXCommon__GetHostname (void * __, struct MXCommon__ByteArrayResponse * Response)

Parameters

- [in] *__* : no input parameter
- [out] **Response** • sArray : Hostname of the module
- iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
 - syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

2.5.1.3 int MXCommon__SetHostname (struct xsd__base64Binary * bHostname, struct MXCommon__Response * Response)

Parameters

- [in] **bHostname** : Hostname
- [out] **Response** • iReturnValue : Return value
- 0 : success
 - -1: system error (see syserrno)
 - syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

2.5.1.4 int MXCommon__GetClientConnections (void * __, struct MXCommon__ByteArrayResponse * Response)

Parameters

- [in] *__* : no input parameter
- [out] **Response** • sArray : string containing the list of connected clients.
- sResponse Composed of iReturnValue and syserrno

The sArray string is of the form IP-Address:first connection-second connection---- IP-Address:first connection-second connection----

Sample: 172.16.3.43:8989-5555 172.16.3.200:8989

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

2.5.1.5 int MXCommon__Strerror (xsd__int *errnum*, struct MXCommon__ByteArrayResponse * *Response*)

Usually SOAP functions return this value in a variable named syserror, which is meaningful only when the function return value, usually called iReturnValue, indicate an error (that is, have a value of -1 or -100, depending of the case).

Parameters

[in] **errnum** : Error number

[out] **Response** • sArray : See the description below.

- sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno).
- sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

STRERROR(3)
STRERROR(3)

Linux Programmer's Manual

NAME

strerror, strerror_r - return string describing error code

SYNOPSIS

```
#include <string.h>
```

```
char *strerror(int errnum);
```

```
#define _XOPEN_SOURCE 600
#include <string.h>
```

```
int strerror_r(int errnum, char *buf, size_t n);
```

DESCRIPTION

The `strerror()` function returns a string describing the error code passed in the argument `errnum`, possibly using the `LC_MESSAGES` part of the current locale to select the appropriate language.

This string must not be modified by the application, but may be modified by a subsequent call to `perror()` or `strerror()`. No library function will modify this string.

The `strerror_r()` function is similar to `strerror()`, but is thread safe. It returns the string in the user-supplied buffer `buf` of length `n`.

RETURN VALUE

The `strerror()` function returns the appropriate error description string, or an unknown error message if the error code is unknown.

The value of `errno` is not changed for a successful call, and is set to a non-zero value upon error.

The `strerror_r()` function returns 0 on success and -1 on failure, setting `errno`.

ERRORS

EINVAL The value of `errnum` is not a valid error number.

ERANGE Insufficient storage was supplied to contain the error description string.

CONFORMING TO

SVID 3, POSIX, 4.3BSD, ISO/IEC 9899:1990 (C89).

`strerror_r()` with prototype as given above is specified by SUSv3, and was in use under Digital Unix and HP Unix. An incompatible function, with prototype

```
char *strerror_r(int errnum, char *buf, size_t n);
```

is a GNU extension used by glibc (since 2.0), and must be regarded as obsolete in view of SUSv3.
 The GNU version may, but need not, use the user-supplied buffer.
 If it does, the result may be truncated in case the supplied buffer is too small.
 The result is always NUL-terminated.

SEE ALSO
 errno(3), perror(3), strsignal(3)

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

2.5.1.6 int MXCommon__Reboot (void * _, struct MXCommon__Response * *Response*)

Parameters

[in] **_** : no input parameter
 [out] **Response** • **iReturnValue** : Return value
 – 0 : success
 – -1 : system error (see syserrno)
 • **syserrno** : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

2.5.1.7 int MXCommon__ResetAllIOFunctionalities (xsd__unsignedLong *ulOption*, struct MXCommon__Response * *Response*)

The behavior of the function depends on the MSX-E system that is used.

On MSX-E3511: Stop the watchdogs and stop the generators
 On MSX-E3601: Stop the sequence acquisition and stop the calibration
 On MSX-E3701: Stop the acquisition

Parameters

[in] **ulOption** Reserved. Set to 0
 [out] **Response** **iReturnValue**
 • **0** The remote function performed OK
 • **-1** Internal system error occurred. See value of syserrno
 • **-100** Function not supported by the system
 syserrno system error code (the value of the libc "errno" code)

Return values

0 SOAP_OK
Others See SOAP error

2.5.1.8 int MXCommon__DataserverRestart (xsd__unsignedLong *ulAction*, xsd__unsignedLong *ulOption*, struct MXCommon__Response * *Response*)

Parameters

- [in] *ulAction* : action
- 0: normal restart
 - 1: with cache file reset
 - 2: with cache file deletion
- [in] *ulOption* : Reserved
- [out] *Response* • iReturnValue : Return value
- 0 : success
 - -1: system error (see syserrno)
 - syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

Note

(revision>6386) Depending on the system type, can be used to restart the data-recv service as well. In this case, parameter action is ignored.

2.5.1.9 int MXCommon__GetEthernetLinksStates (void * _, struct MXCommon__GetEthernetLinksStatesResponse * *Response*)

Parameters

- [in] _ : no input parameter
- [out] *Response* Structure that contains the MSX-E Ethernet links states and errors:
- sResponse.iReturnValue*
- **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** Fail to get Ethernet links states
 - **-100** Internal system error occurred. See value of syserrno
- sResponse.syserrno* system error code (the value of the libc "errno" code)
- sPort0: Fisrt port informations*
- **ulState**
 - **0** Link down
 - **1** Link up
 - **ulSpeed**
 - **10** 10 Mb/s
 - **100** 100 Mb/s
 - **ulDuplex**
 - **0** Half duplex
 - **1** Full duplex

- **ulInfo1** Reserved
- **ulInfo2** Reserved

sPort1: Second port informations

- **ulState**
 - **0** Link down
 - **1** Link up
- **ulSpeed**
 - **10** 10 Mb/s
 - **100** 100 Mb/s
- **ulDuplex**
 - **0** Half duplex
 - **1** Full duplex
- **ulInfo1** Reserved
- **ulInfo2** Reserved

Return values

0 SOAP_OK

Others See SOAP error

2.6 Common temperature functions

These functions deals with the internal temperature sub-system.

Data Structures

- struct [MXCommon__GetModuleTemperatureValueAndStatusResponse](#)

Functions

- int [MXCommon__GetModuleTemperatureValueAndStatus](#) (xsd__unsignedLong ulOption, struct [MXCommon__GetModuleTemperatureValueAndStatusResponse](#) *Response)

Read the temperature on the module.

- int [MXCommon__SetModuleTemperatureWarningLevels](#) (xsd__double dMinimalWarningLevel, xsd__double dMaximalWarningLevel, xsd__unsignedLong ulOption, struct [MXCommon__Response](#) *Response)

Set the temperature warning level on the module.

2.6.1 Detailed Description

The role of this sub-system is to monitor the internal temperature of a module and issue a warning if it is below or above a threshold. If the internal temperature reaches a domain where the system is endangered, it switches automatically in a degraded working mode.

2.6.2 Function Documentation

2.6.2.1 `int MXCommon__GetModuleTemperatureValueAndStatus (xsd__unsignedLong ulOption, struct MXCommon__GetModuleTemperatureValueAndStatusResponse * Response)`

Parameters

- [in] *ulOption* : Reserved
- [out] *Response*
 - sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).
 - dValue : Temperature value in Degree Celsius
 - ulTemperatureStatus : Temperature Status :
 - TEMPERATURE_INITIAL = 0 : Temperature not ready
 - TEMPERATURE_TOLOW = 1 : Temperature too low !
 - TEMPERATURE_LOW = 2 : Temperature under the min warning value
 - TEMPERATURE_NOMINAL = 3 : Temperature in the nominal range
 - TEMPERATURE_HIGH = 4 : Temperature over the max warning value
 - TEMPERATURE_TOOHIGH = 5 : Temperature too high !
 - ulInfo : Reserved

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

2.6.2.2 `int MXCommon__SetModuleTemperatureWarningLevels (xsd__double dMinimalWarningLevel, xsd__double dMaximalWarningLevel, xsd__unsignedLong ulOption, struct MXCommon__Response * Response)`

Parameters

- [in] *dMinimalWarningLevel* : Minimal temperature warning level in Degree : 5 to 60 Degree Celsius
- [in] *dMaximalWarningLevel* : Maximal temperature warning level in Degree : 5 to 60 Degree Celsius
- [in] *ulOption* : Reserved
- [out] *Response*
 - sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

2.7 Common hardware trigger functions

These functions allow to set and request the current value of the hardware trigger.

Data Structures

- struct [MXCommon__GetHardwareTriggerFilterTimeResponse](#)
- struct [MXCommon__GetHardwareTriggerStateResponse](#)

Functions

- int [MXCommon__SetHardwareTriggerFilterTime](#) (xsd__unsignedLong ulFilterTime, xsd__unsignedLong ulOption, struct [MXCommon__Response](#) *Response)
Sets the filter time for the hardware trigger input in steps of 250 ns (max value: 65535).
- int [MXCommon__GetHardwareTriggerFilterTime](#) (xsd__unsignedLong ulOption, struct [MXCommon__GetHardwareTriggerFilterTimeResponse](#) *Response)
Get the filter time for the hardware trigger input.
- int [MXCommon__GetHardwareTriggerState](#) (xsd__unsignedLong ulOption, struct [MXCommon__GetHardwareTriggerStateResponse](#) *Response)
Get the hardware trigger state after the filter.

2.7.1 Function Documentation

2.7.1.1 int [MXCommon__SetHardwareTriggerFilterTime](#) (xsd__unsignedLong *ulFilterTime*, xsd__unsignedLong *ulOption*, struct [MXCommon__Response](#) * *Response*)

Sets the filter time for the hardware trigger input in steps of 250 ns (max value: 65535).

On the MSX-E3011 system, the step of the hardware trigger filter is **622ns**.

Parameters

[in] ***ulFilterTime*** Filter time for the hardware trigger input in steps of 250ns (max value : 65535).

- **0**: Disable the filter
- **1**: Sets the filter time to 250 ns
- **2**: Sets the filter time to 500 ns
- ...
- **65535**: Sets the filter time to 16 ms

[in] ***ulOption*** Reserved. Set to 0

[out] ***Response*** Response of the system

- ***sResponse.iReturnValue***
 - **0**: The remote function performed OK
 - **-1**: Internal system error occurred. See value of syserrno
- ***sResponse.syserrno*** system error code (the value of the libc "errno" code)

Return values*0* SOAP_OK*Others* See SOAP error**2.7.1.2 int MXCommon__GetHardwareTriggerFilterTime (xsd__unsignedLong ulOption, struct MXCommon__GetHardwareTriggerFilterTimeResponse * Response)**

Get the filter time for the hardware trigger input in **250ns** step (max value : 65535).

On the MSX-E3011 system, the step of the hardware trigger filter is **622ns**.

Parameters

[in] *ulOption* Reserved. Set to 0

[out] *Response* Response of the system

- *ulFilterTime* filter time for the hardware trigger input
 - **0**: filter disabled
 - **1**: filter of 250ns
 - **2**: filter of 500ns
 - ...
 - **65535**: filter of 16ms
- *sResponse.iReturnValue*
 - **0**: The remote function performed OK
 - **-1**: Internal system error occurred. See value of syserrno
- *sResponse.syserrno* system error code (the value of the libc "errno" code)

Return values*0* SOAP_OK*Others* See SOAP error**2.7.1.3 int MXCommon__GetHardwareTriggerState (xsd__unsignedLong ulOption, struct MXCommon__GetHardwareTriggerStateResponse * Response)****Parameters**

[in] *ulOption* : Reserved

[out] *Response* • *ulState* : Hardware trigger input state.

- **0**: Hardware trigger input is low
- **1**: Hardware trigger input is high.
- *sResponse.iReturnValue* : Return value
 - **0** : success
 - **-1**: system error (see syserrno)
- *sResponse.syserrno* : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).

Return values*SOAP_OK* SOAP call success*otherwise* SOAP protocol error

2.8 Common security functions

The "customer key" feature may for instance be used by a customer to be sure that his application communicates only with certified MSX-E modules.

Data Structures

- struct [MXCommon__TestCustomerIDResponse](#)

Functions

- int [MXCommon__SetCustomerKey](#) (struct [xsd__base64Binary](#) *bKey, struct [xsd__base64Binary](#) *bPublicKey, struct [MXCommon__Response](#) *Response)

Set the Customer key.

- int [MXCommon__TestCustomerID](#) (void *_ , struct [MXCommon__TestCustomerIDResponse](#) *Response)

Test the Customer ID (if the module has the right customer Key).

2.8.1 Detailed Description

A "customer key" consists of two strings of data stored on the certified MSX-E module, to be used by the function [MXCommon__TestCustomerID\(\)](#) to encrypt data.

These strings can not be read back. They are supposed to be kept secret by the user of this functionality.

To test if the MSX-E module you use is certified, you can request the MSX-E module to provide a set of randomly generated data and the result of the encryption (through the use of the stored "customer key") of the same data. Then your application must encrypt the delivered random data with its own "customer key" and compare it with the encrypted data delivered by the MSX-E module.

If the results are matching, the MSX-E module is certified for this application.

Detailed presentation of operations:

The user generates and stores on the module two keys (thanks to the software function : [MXCommon__SetCustomerKey\(\)](#)). This needs only to be done once:

- A public Key K1 (16 Bytes)
- A private Key K2 (32 Bytes)

When requested (with the software function : [MXCommon__TestCustomerID\(\)](#)), the module generates a 16 bytes random value and do an encryption of this value using the two saved keys and the AES algorithm (Rijndael).

The user receives then two arrays of 16 bytes :

- one with a random value [A]
- the second with encrypted random value [B]

$[B] = \text{AES}([A], K1, K2)$

The user performs then the same computation from $[A], K1, K2$ and compares his result with $[B]$. If it is the same, it means that the module he is using was already configured with the correct identification token.

The security of the method comes from that even knowing $[A]$ and $[B]$ no one can deduce $K1$ and $K2$ back in practical times. ADDI-DATA is not aware of a practical way to remotely retrieve the value of the key stored on a module.

It is the responsibility of the developer of the application to ensure that these tokens are suitably protected. The authorisation of the change of the "customer key" on the MSX-E module can be managed with the web interface.

The use of the "customer key" don't have an impact of the other functionalities of the MSX-E module.

2.8.2 Function Documentation

2.8.2.1 `int MXCommon__SetCustomerKey (struct xsd__base64Binary * bKey, struct xsd__base64Binary * bPublicKey, struct MXCommon__Response * Response)`

Parameters

- [in] *bKey* : Customer key (only writable on the module) [32 bytes containing a AES key]
- [in] *bPublicKey* : IV (Initialisation vector) for the AES cryptography [16 bytes containing a AES key]
- [out] *Response*
 - `sResponse.iReturnValue` : Return value
 - 0 : success
 - -1 : system error (see `syserrno`)
 - `sResponse.syserrno` : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

2.8.2.2 `int MXCommon__TestCustomerID (void * _, struct MXCommon__TestCustomerIDResponse * Response)`

Parameters

- [in] `_` : No Input
- [out] *Response*
 - `sResponse.iReturnValue` : Return value
 - 0 : success
 - -1 : system error (see `syserrno`)
 - `sResponse.syserrno` : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).
 - `bValueArray` : non encrypted value array [16 bytes of random data]
 - `bCryptedValueArray` : Encrypted value array [16 bytes of the encrypted random data]

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

2.9 Common time functions

A MSX-E module provides a "system clock" that may be in the simplest case set by the function [MXCommon__SetTime\(\)](#).

Data Structures

- struct [MXCommon__GetTimeResponse](#)
- struct [MXCommon__GetUpTimeResponse](#)

Functions

- int [MXCommon__SetTime](#) (xsd__unsignedLong ulLowTime, xsd__unsignedLong ulHighTime, struct [MXCommon__Response](#) *Response)
Set the time on the module.
- int [MXCommon__SysToHardwareClock](#) (void *_, struct [MXCommon__Response](#) *Response)
Set the hardware clock (if present) to the current system time.
- int [MXCommon__HardwareClockToSys](#) (void *_, struct [MXCommon__Response](#) *Response)
Set the system time from the hardware clock (if present).
- int [MXCommon__GetTime](#) (void *_, struct [MXCommon__GetTimeResponse](#) *Response)
Get the time on the module.
- int [MXCommon__GetUpTime](#) (void *_, struct [MXCommon__GetUpTimeResponse](#) *Response)
Ask the MSX-E module uptime (number of seconds since the last boot).

2.9.1 Detailed Description

If the module is configured to use NTP, the time received by the NTP server will have a greater priority. If the module is linked to another through a "synchronization bus" and is slave, then the time received from the master is the one taken into account.

Recent models also provide a "hardware clock", a component whose role is to track the time between reboots.

2.9.2 Function Documentation

2.9.2.1 int MXCommon__SetTime (xsd__unsignedLong ulLowTime, xsd__unsignedLong ulHighTime, struct MXCommon__Response * Response)

Parameters

- [in] **ulLowTime** : Number of microseconds since the begin of the second
- [in] **ulHighTime** : Number of seconds since the Epoch (1st January,1970)
- [out] **Response**
 - sResponse.iReturnValue : Return value
 - 0 : success

- -1: system error (see `syserrno`)
- `sResponse.syserrno` : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

2.9.2.2 `int MXCommon__SysToHardwareClock (void * _, struct MXCommon__Response * Response)`

Parameters

- [in] `_` No input parameter
- [out] *Response* • `sResponse.iReturnValue` : Return value
- 0 : success
 - -1: system error (see `syserrno`)
 - `sResponse.syserrno` : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

If this function fails, it means the module does not have a hardware RTC, or the hardware is not functional. Check the "hwclock" subsystem status.

2.9.2.3 `int MXCommon__HardwareClockToSys (void * _, struct MXCommon__Response * Response)`

When the hardware clock is present, the system time is automatically set to it when the module becomes master on the inter-module synchronisation bus.

Parameters

- [in] `_` No input parameter
- [out] *Response* • `sResponse.iReturnValue` : Return value
- 0 : success
 - -1: system error (see `syserrno`)
 - `sResponse.syserrno` : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

If this function fails, it means the module does not have a hardware RTC, or the hardware is not functional. Check the "hwclock" subsystem status.

2.9.2.4 int MXCommon__GetTime (void * _, struct MXCommon__GetTimeResponse * Response)

Parameters

- [in] _ : No input parameter
- [out] **Response**
- sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).
 - ulLowTime : Number of microseconds since the begin of the second
 - ulHighTime : Number of seconds since the Epoch (1st January,1970)

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

2.9.2.5 int MXCommon__GetUpTime (void * _, struct MXCommon__GetUpTimeResponse * Response)

Parameters

- [in] _ : no input parameter
- [out] **Response**
- sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).
 - ulUpTime : Number of seconds since the last boot of the system.

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

2.10 Common I/O auto configuration functions

On the web site of some MSX-E module, there is the possibility to define an auto-configuration and auto start of the I/O.

Data Structures

- struct [MXCommon__GetAutoConfigurationFileResponse](#)

Functions

- `int MXCommon__GetAutoConfigurationFile (void *__, struct MXCommon__GetAutoConfigurationFileResponse *Response)`
Get the auto configuration file of the module.
- `int MXCommon__SetAutoConfigurationFile (struct xsd__base64Binary *ByteArrayInput, xsd__unsignedLong ulEOF, struct MXCommon__Response *Response)`
Set the auto configuration file of the module.
- `int MXCommon__StartAutoConfiguration (void *__, struct MXCommon__ByteArrayResponse *Response)`
start/Restart the auto configuration

2.10.1 Detailed Description

- Auto-configuration means the system configures the I/O automatically at boot time.
- Auto-start means the system starts an acquisition automatically at boot time (this may no make sense for some systems). It implies auto-configuration.

This set of functions allows to:

- get the auto-configuration/start currently set on module, as a read-only binary file.
- set a auto-configuration/start on the module, using a previously saved file.
- start or restart the auto-configuration/start on the module, using the current configuration saved on the module.

2.10.2 Function Documentation

2.10.2.1 `int MXCommon__GetAutoConfigurationFile (void * ___, struct MXCommon__GetAutoConfigurationFileResponse * Response)`

Parameters

- [in] `__` : No input parameter
- [out] **Response** • `sResponse.iReturnValue` : Return value
- 0 : success
 - -1: system error (see `syserrno`)
 - -100 : Error of the read of the auto configuration file
 - `sResponse.syserrno` : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).
 - `bArray` : Array of Bytes of the file
 - `ulEOF` : End of file flag

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

2.10.2.2 `int MXCommon__SetAutoConfigurationFile (struct xsd__base64Binary * ByteArrayInput, xsd__unsignedLong ulEOF, struct MXCommon__Response * Response)`

Parameters

- [in] *ByteArrayInput* : Array of Bytes of the file
- [in] *ulEOF* : End of file flag
- [out] *Response*
 - sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

2.10.2.3 `int MXCommon__StartAutoConfiguration (void * _, struct MXCommon__ByteArrayResponse * Response)`

Parameters

- [in] *_* : No input parameter
- [out] *Response*
 - sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).
 - sArray : message returned by the auto configuration start

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

2.11 Common synchronisation timer functions

When modules are linked through a "synchronisation bus", the master can run a timer that generate a "synchro signal" on the slaves when overrun.

Functions

- `int MXCommon__InitAndStartSynchroTimer (xsd__unsignedLong ulTimeBase, xsd__unsignedLong ulReloadValue, xsd__unsignedLong ulNbrOfCycle, xsd__unsignedLong ulGenerateTriggerMode, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MXCommon__Response *Response)`

Initialises and starts the synchronisation timer of the module (not already available on all module).

- `int MXCommon__StopAndReleaseSynchroTimer (xsd__unsignedLong ulOption01, struct MXCommon__Response *Response)`
start/Restart the synchronisation timer (not already available on all module)

2.11.1 Function Documentation

2.11.1.1 `int MXCommon__InitAndStartSynchroTimer (xsd__unsignedLong ulTimeBase, xsd__unsignedLong ulReloadValue, xsd__unsignedLong ulNbrOfCycle, xsd__unsignedLong ulGenerateTriggerMode, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MXCommon__Response * Response)`

Parameters

- [in] **ulTimeBase** : Time base of the timer (0 for us, 1 for ms, 2 for s)
- [in] **ulReloadValue** : Timer reload value (0 to 0xFFFF), minimum reload time is 5 us
- [in] **ulNbrOfCycle** : Number of timer cycle
 - 0: continuous
 - > 0: defined number of cycle
- [in] **ulGenerateTriggerMode** :
 - 0: Wait the time overflow to set the synchronisation trigger
 - 1: Set the synchronisation trigger by the start of the timer and after each time overflow
- [in] **ulOption01** : Define the source of the trigger
 - 0 : Trigger disabled
 - 1 : Enable the hardware digital input trigger
- [in] **ulOption02** : Define the edge of the hardware trigger who generates a trigger action
 - 1 : rising edge (Only if hardware trigger selected)
 - 2 : falling edge (Only if hardware trigger selected)
 - 3 : Both front (Only if hardware trigger selected)
- [in] **ulOption03** : Define the number of trigger events before the action occur
 - 1 : all trigger event start the action
 - max value : 65535
- [in] **ulOption04** : Reserved
- [out] **Response**
 - sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
 - -2: not available time base
 - -3: timer reload value can not be greater than 65535
 - -4: minimum time reload is 5 us
 - -5: Number of cycle can not be greater than 65535
 - -6: Generate trigger mode error
 - -100: Init timer error
 - -101: Start timer error

- `sResponse.syserrno` : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#). May be `ENOSYS` : Function not implemented.

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

2.11.1.2 `int MXCommon__StopAndReleaseSynchroTimer (xsd__unsignedLong ulOption01, struct MXCommon__Response * Response)`

Parameters

- [in] *ulOption01* : Reserved
- [out] *Response* • `sResponse.iReturnValue` : Return value
- 0 : success
 - -1: system error (see `syserrno`)
 - -100: Start/Stop timer error
- `sResponse.syserrno` : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#). May be `ENOSYS` : Function not implemented.

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

2.12 Set/Backup/Restore general system configuration

Distinct of the I/O auto-configuration/auto-start functionality, these functions allows to manipulate the general system configuration.

Functions

- `int MXCommon__GetConfigurationBackupFile (void *, struct MXCommon__FileResponse *Response)`
Download a configuration backup file from the module.
- `int MXCommon__ApplyConfigurationBackupFile (struct xsd__base64Binary *ByteArrayInput, xsd__unsignedLong ulEOF, struct MXCommon__Response *Response)`
Upload a new configuration on the module.
- `int MXCommon__ChangePassword (struct xsd__base64Binary *PreviousUser, struct xsd__base64Binary *PreviousPassword, struct xsd__base64Binary *NewUser, struct xsd__base64Binary *NewPassword, struct MXCommon__Response *Response)`
Set a new id/password.

2.12.1 Detailed Description

It includes the network configuration, and generally everything that can be set up through the web interface.

These functions have been included to ease the automation of module customisation. They may be disabled using the web interface, in "Security/Remote general system configuration authorisation/remote sysconf changes"

2.12.2 Function Documentation

2.12.2.1 `int MXCommon__GetConfigurationBackupFile (void * _, struct MXCommon__FileResponse * Response)`

Parameters

- [in] `_` : No input parameter
- [out] ***Response***
 - `sResponse.iReturnValue` : Return value
 - 0 : success
 - -1: system error (see `syserrno`) (see `syserrno`)
 - `sResponse.syserrno` : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).
 - `bArray` : Array of Bytes of the file
 - `ulEOF` : End of file flag

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

This function is designed to be called repeatedly until no more data is available. At this point the flag `ulEOF` is set.

Below is an example in pseudo-C.

```
int dummy;
struct MXCommon__FileResponse Response;
while(1)
{
    if ( MXCommon__GetConfigurationBackupFile(&dummy, &Response) != SOAP_OK)
    {
        // handle soap error
    }
    if (Response.iReturnValue)
    {
        // handle remote error (Response.syserrno contains more information)
    }
    // do something with the data, for example save it in a file
    write(fd, Response.bArray.__ptr, Response.bArray.__size);
    // if this is the end of the file, quit the loop
    if(Response.ulEOF)
        break;
}
*
```

2.12.2.2 `int MXCommon__ApplyConfigurationBackupFile (struct xsd__base64Binary * ByteArrayInput, xsd__unsignedLong ulEOF, struct MXCommon__Response * Response)`

Parameters

- [in] *ByteArrayInput* : Array of Bytes of the file
- [in] *ulEOF* : End of file flag
- [out] *Response*
 - sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

This function is designed to be called repeatedly until all data is transferred. At this point the flag ulEOF must be set to 1. The new configuration is then applied.

2.12.2.3 `int MXCommon__ChangePassword (struct xsd__base64Binary * PreviousUser, struct xsd__base64Binary * PreviousPassword, struct xsd__base64Binary * NewUser, struct xsd__base64Binary * NewPassword, struct MXCommon__Response * Response)`

The changes are immediately active.

Parameters

- [in] *_* : No input parameter
- [out] *Response*
 - sResponse.iReturnValue : Return value
 - 0 : success
 - -1: string PreviousUser is invalid
 - -2: string PreviousPassword is invalid
 - -3: string NewUser is invalid
 - -4: string NewPassword is invalid
 - -5: authentication failed
 - -100: system error while saving tokens (use syserrno for more information)
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).
 - sArray : message returned by the auto configuration start

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

Warning

The parameters transit in clear text. Use this functionality only on trusted networks. Given that ADDI-DATA GmbH takes security seriously, there is no way to change the password without knowing it. No "hidden back-door". This function makes it all too easy to lock a module, if you don't remember the password you set on it.

2.13 System state management

Every MSX-E modules are composed of several sub-systems that work together to provide the system functionalities.

Functions

- `int MXCommon__GetSubSystemState (xsd__unsignedLong SubsystemID, struct MXCommon__unsignedLongResponse *Response)`
Returns the current state of the specified sub-system.
- `int MXCommon__GetSubsystemIDFromName (struct xsd__base64Binary *SubsystemName, struct MXCommon__unsignedLongResponse *Response)`
Returns the ID of the sub-system of symbolic name "SubsystemName".
- `int MXCommon__GetStateIDFromName (xsd__unsignedLong SubsystemID, struct xsd__base64Binary *StateName, struct MXCommon__unsignedLongResponse *Response)`
Returns the ID of the state of symbolic name "StateName" of the sub-system of ID "SubsystemID".
- `int MXCommon__GetSubsystemNameFromID (xsd__unsignedLong SubsystemID, struct MXCommon__ByteArrayResponse *Response)`
Returns the symbolic name of the sub-system of numerical ID "SubsystemName".
- `int MXCommon__GetStateNameFromID (xsd__unsignedLong SubsystemID, xsd__unsignedLong StateID, struct MXCommon__ByteArrayResponse *Response)`
Returns the symbolic name of the state of numerical ID "StateID" of the sub-system of ID "SubsystemID".

2.13.1 Detailed Description

These sub-systems have a state that, for example, indicate if it functions nominally.

A sub-system is identified by its ID (a positive integer) and its symbolic name. Each state in the set of possible states for a given sub-system has also an ID and a symbolic name.

Names are less likely to change between releases of the MSX-E operating system. That is why manipulating names should be preferred against indexes in an application. Still, manipulating ID is more efficient.

The functions in this section provide a way to retrieve the association between names and indexes. `MXCommon__GetSubSystemState()` requests the state of a given sub-system.

Notice that the event manager is the recommended way to be warned of a change of state.

The list of sub-systems and their ID and associated name can be consulted on the web site of the module.

2.13.2 Function Documentation

2.13.2.1 `int MXCommon__GetSubSystemState (xsd__unsignedLong SubsystemID, struct MXCommon__unsignedLongResponse * Response)`

Parameters

[in] *SubsystemID* sub-system numerical ID

- [out] **Response** • sResponse.iReturnValue : Return value
- 0 : success
 - -1: system error while executing the request (see syserrno)
 - -2: invalid parameter SubsystemID
- sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).
- Value The state of the sub-system "Id" at the moment of the execution of the request.

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

2.13.2.2 int MXCommon__GetSubsystemIDFromName (struct xsd__base64Binary * SubsystemName, struct MXCommon__unsignedLongResponse * Response)

Parameters

- [in] **SubsystemName** sub-system symbolic name.
- [out] **Response** • sResponse.iReturnValue :Return value
- 0 : success
 - -1: system error while executing the request (see syserrno)
 - -2: invalid parameter SubsystemName
- sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).
- Value The numerical ID of the sub-system "SubsystemName".

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

2.13.2.3 int MXCommon__GetStateIDFromName (xsd__unsignedLong SubsystemID, struct xsd__base64Binary * StateName, struct MXCommon__unsignedLongResponse * Response)

Parameters

- [in] **SubsystemID** sub-system numerical ID
- [in] **StateName** state symbolic name.
- [out] **Response** • sResponse.iReturnValue : Return value
- 0 : success
 - -1: system error while executing the request (see syserrno)
 - -2: invalid parameters SubsystemID or StateName
- sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).
- Value The numerical ID of the state "StateName".

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

2.13.2.4 `int MXCommon__GetSubsystemNameFromID (xsd__unsignedLong SubsystemID, struct MXCommon__ByteArrayResponse * Response)`

Parameters

- [in] **SubsystemID** sub-system numerical ID.
- [out] **Response**
 - sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error while executing the request (see syserrno)
 - -2: invalid parameter SubsystemName
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).
 - sArray : The symbolic name associated with the ID.

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

2.13.2.5 `int MXCommon__GetStateNameFromID (xsd__unsignedLong SubsystemID, xsd__unsignedLong StateID, struct MXCommon__ByteArrayResponse * Response)`

Parameters

- [in] **SubsystemID** sub-system numerical ID.
- [in] **StateID** sub-system numerical ID.
- [out] **Response**
 - sResponse.iReturnValue : Return value
 - 0 success
 - -1 system error while executing the request (see syserrno)
 - -2 invalid parameters SubsystemID or StateID
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).
 - sArray The symbolic name associated with the state numerical ID.

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

2.14 Customer option management

Enable to get informations about the options of the system.

Functions

- `int MXCommon__GetOptionInformation (void *, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MXCommon__ByteArrayResponse *Response)`
Enables to get information about the options available on the system.

2.14.1 Function Documentation

2.14.1.1 `int MXCommon__GetOptionInformation (void * __, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MXCommon__ByteArrayResponse * Response)`

Parameters

- [in] *ulOption01*,: not used, set it to 0
- [in] *ulOption02*,: not used, set it to 0
- [out] *Response*
 - sArray : Option information string
 - sResponse Composed of iReturnValue and syserrno

Return values

- SOAP_OK* SOAP call success
- otherwise* SOAP protocol error

2.15 Synchronisation management

Manage the synchronisation state of the system.

Functions

- `int MXCommon__SetToMaster (void * __, xsd__unsignedLong ulState, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MXCommon__Response *Response)`
Writes if the MSXE has to be always set to master The master mode (when enabled) make the system always detected as master.
- `int MXCommon__GetSynchronizationStatus (void * __, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MXCommon__unsignedLongResponse *Response)`
Reads the status of the synchronization for the corresponding MSXE The master mode (when enabled) make the system always detected as master.

2.15.1 Function Documentation

2.15.1.1 `int MXCommon__SetToMaster (void * __, xsd__unsignedLong ulState, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MXCommon__Response * Response)`

Parameters

- [in] *ulState* State of the supermaster mode
 - **0** automatic mode (default). The state of the system (master or slave) will be automatically detected by the system
 - **1** Set to master mode at all time. The system will always be detected as master
- [in] *ulOption01* Reserved. Set to 0
- [in] *ulOption02* Reserved. Set to 0
- [out] *Response* *iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The ulFilterTime parameter is wrong
- **-100** Internal system error occurred. See value of syserrno *syserrno* system error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

2.15.1.2 `int MXCommon__GetSynchronizationStatus (void * _, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MXCommon__unsignedLongResponse * Response)`

Parameters

[in] *ulOption01* Reserved. Set to 0

[in] *ulOption02* Reserved. Set to 0

[out] *Response sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-100** Internal system error occurred. See value of syserrno

sResponse.syserrno system error code (the value of the libc "errno" code)

ulValue State of the supermaster mode

- **0** Automatic mode (default). The state of the system (master or slave) will be automatically detected by the system
- **1** MSXE is always set as a master. The system will always be detected as master

Return values

0 SOAP_OK

Others See SOAP error

2.16 input filter Filter management

Manages the analog input filters in the system.

Functions

- `int MXCommon__SetFilterChannels (struct xsd__base64Binary *ChannelList, struct MXCommon__Response *Response)`

This function sets or resets a filter to a channel.

2.16.1 Function Documentation

2.16.1.1 `int MXCommon_SetFilterChannels (struct xsd__base64Binary * ChannelList, struct MXCommon_Response * Response)`

Parameters

[in] **ChannelList** Each index of the array represents a channel. A filter can be affected to each channel. If FilterID = 0, no filter is set (the filter is disabled on the corresponding channel). e.g.:
ChannelList[0] = FilterID // Set FilterID on channel 0.

[out] **Response**

- sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
- sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

2.17 MSX-E173x digital I/O functions

Data Structures

- struct [MSXE173x__DigitalIOGetNumberResponse](#)
Returns the number of digital IO available on the module.

Functions

- int [MSXE173x__DigitalIOGetNumber](#) (void ___, struct [MSXE173x__DigitalIOGetNumberResponse](#) *Response)
- int [MSXE173x__DigitalIOInitPortConfiguration](#) (xsd__unsignedLong ulPort, xsd__unsignedLong ulPortConfiguration, struct [MSXE173x__Response](#) *Response)
Initialise a digital i/o port (2 channels).
- int [MSXE173x__DigitalIOReadChannelValue](#) (xsd__unsignedLong ulChannel, struct [MSXE173x__unsignedLongResponse](#) *Response)
Read a digital i/o channel value.
- int [MSXE173x__DigitalIOReadAllChannelsValue](#) (void ___, struct [MSXE173x__unsignedLongResponse](#) *Response)
Read all digital i/o channels value. If channel is configured as output, then this function return the status of the output.
- int [MSXE173x__DigitalIOWriteChannelValue](#) (xsd__unsignedLong ulChannel, xsd__unsignedLong ulChannelValue, struct [MSXE173x__Response](#) *Response)
write a digital i/o channel value

- `int MSXE173x__DigitalIOWriteAllChannelsValue (xsd__unsignedLong ulChannelValue, struct MSXE173x__Response *Response)`
write all digital i/o channels value
- `int MSXE173x__DigitalIOReleasePortConfiguration (xsd__unsignedLong ulPort, struct MSXE173x__Response *Response)`
Release a digital i/o port (2 channels).
- `int MSXE173x__DigitalIOTestShortCircuit (xsd__unsignedLong ulOption, struct MSXE173x__Response *Response)`
Test short circuit status.
- `int MSXE173x__DigitalIORearmShortCircuit (xsd__unsignedLong ulOption, struct MSXE173x__Response *Response)`
Rearm digital outputs after a short circuit happened.

2.17.1 Function Documentation

2.17.1.1 `int MSXE173x__DigitalIOGetNumber (void * _, struct MSXE173x__DigitalIOGetNumberResponse * Response)`

2.17.1.2 `int MSXE173x__DigitalIOInitPortConfiguration (xsd__unsignedLong ulPort, xsd__unsignedLong ulPortConfiguration, struct MSXE173x__Response * Response)`

Parameters

[in] ***ulPort*** : Index of the digital i/o port (0 to 7)

[in] ***ulPortConfiguration*** : Define the port configuration

- 0 : input
- 1 : output

[out] ***Response*** :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -2: Digital i/o port selection error
- -3: Port configuration selection error
- -100: Init dig i/o port kernel function error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.17.1.3 int MSXE173x__DigitalIOReadChannelValue (xsd__unsignedLong ulChannel, struct MSXE173x__unsignedLongResponse * Response)

Parameters

[in] *ulChannel* : Index of the digital i/o channel (0 to 15)

[out] *Response* :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -2: Digital i/o channel selection error
- -100: Read dig i/o channel value kernel function error

syserrno : system-error code (the value of the libc "errno" code) *ulValue* : i/o channel value:

- 0
- 1

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.17.1.4 int MSXE173x__DigitalIOReadAllChannelsValue (void * _, struct MSXE173x__unsignedLongResponse * Response)

Parameters

[in] *_* : no input parameter

[out] *Response* :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -100: Read dig i/o channel value kernel function error

syserrno : system-error code (the value of the libc "errno" code) *ulValue* : i/o channels value(each bit correspond to one channel)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.17.1.5 int MSXE173x__DigitalIOWriteChannelValue (xsd__unsignedLong ulChannel, xsd__unsignedLong ulChannelValue, struct MSXE173x__Response * Response)

Parameters

[in] *ulChannel* : Index of the digital i/o channel (0 to 15)

[in] *ulChannelValue* : Channel value

- 0
- 1

[out] **Response** :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -2: Digital i/o channel selection error
- -3: Channel value error
- -100: Write dig i/o channel value kernel function error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.17.1.6 int MSXE173x_DigitalIOWriteAllChannelsValue (xsd__unsignedLong ulChannelValue, struct MSXE173x__Response * Response)

Parameters

[in] **ulChannelValue** : Channels value (each bit corresponds to a channel)

[out] **Response** :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -2: Channels value error
- -100: Write dig i/o channel value kernel function error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.17.1.7 int MSXE173x_DigitalIOReleasePortConfiguration (xsd__unsignedLong ulPort, struct MSXE173x__Response * Response)

Parameters

[in] **ulPort** : Index of the digital i/o port (0 to 7)

[out] **Response** :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -2: Digital i/o port selection error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.17.1.8 int MSXE173x__DigitalIOTestShortCircuit (xsd__unsignedLong *ulOption*, struct MSXE173x__unsignedLongResponse * *Response*)

Parameters

[in] *ulOption* : reserved

[out] *Response* :

iReturnValue :

- 0 : means the remote function performed OK
- -1: means an system error occurred

syserrno : system-error code (the value of the libc "errno" code)

ulValue : short circuit status: from 0 to 0xffff, one bit for each output

- 0 : no short circuit
- 1 : short circuit

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.17.1.9 int MSXE173x__DigitalIORearmShortCircuit (xsd__unsignedLong *ulOption*, struct MSXE173x__Response * *Response*)

Parameters

[in] *ulOption* : reserved

[out] *Response* :

iReturnValue :

- 0 : means the remote function performed OK
- -1: means an system error occurred

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.18 MSX-E173x IO watchdog functions

Data Structures

- struct [MSXE173x__IOWatchdogGetStatusAndValueResponse](#)

Functions

- int [MSXE173x__IOWatchdogInitAndStart](#) (xsd__unsignedLong ulTimeBase, xsd__unsignedLong ulTimeValue, xsd__unsignedLong ulOption1, xsd__unsignedLong ulOption2, struct [MSXE173x__Response](#) *Response)

Init and start the digital output IO watchdog.

- `int MSXE173x__IOWatchdogStopAndRelease (xsd__unsignedLong ulOption, struct MSXE173x__Response *Response)`

Stop and release the digital output watchdog.

- `int MSXE173x__IOWatchdogGetStatusAndValue (xsd__unsignedLong ulOption, struct MSXE173x__IOWatchdogGetStatusAndValueResponse *Response)`

Get watchdog current status and value information.

2.18.1 Function Documentation

- 2.18.1.1** `int MSXE173x__IOWatchdogInitAndStart (xsd__unsignedLong ulTimeBase, xsd__unsignedLong ulTimeValue, xsd__unsignedLong ulOption1, xsd__unsignedLong ulOption2, struct MSXE173x__Response * Response)`

Parameters

[in] *ulTimeBase* : Time base of the watchdog delay (0 for mus, 1 for ms, 2 for s)

[in] *ulTimeValue* : Time base of the watchdog delay (0 to 0xFFFF)

[in] *ulOption1* : Reserved

[in] *ulOption2* : Reserved

[out] *Response* :

iReturnValue :

- 0: remote function performed OK
- -1: an system error occurred
- -2: time base selection error
- -3: time value selection error
- -100: Init and start digital output watchdog kernel function error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

- 2.18.1.2** `int MSXE173x__IOWatchdogStopAndRelease (xsd__unsignedLong ulOption, struct MSXE173x__Response * Response)`

Parameters

[in] *ulOption* : reserved

[out] *Response* :

iReturnValue :

- 0: remote function performed OK
- -1: an system error occurred
- -100: Stop and release digital output watchdog kernel function error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.18.1.3 int MSXE173x__IOWatchdogGetStatusAndValue (xsd__unsignedLong ulOption, struct MSXE173x__IOWatchdogGetStatusAndValueResponse * Response)

Parameters

[in] *ulOption* : Reserved

[out] *Response* :

iReturnValue :

- 0: remote function performed OK
- -1: an system error occurred
- -2: channel selection error
- -100: Get diagnostic information kernel function error

ulStatus : current status information

- BIN XXXXXXXXXX XXXXXXXXXX XXXXXXXXXX XXXXXXXX0: is stopped,
- BIN XXXXXXXXXX XXXXXXXXXX XXXXXXXXXX XXXXXXXX1: is running,
- BIN XXXXXXXXXX XXXXXXXXXX XXXXXXXXXX XXXXXXXX0X: is not run down
- BIN XXXXXXXXXX XXXXXXXXXX XXXXXXXXXX XXXXXXXX1X: is run down

ulValue : current value information (0 to 0xFFFF)

ulInfo : reserved

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.19 MSX-E173x multifunction common functions

Functions

- int [MSXE173x__MFCommonSetInputsFilter](#) (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulInputAFilterValue, xsd__unsignedLong ulInputBFilterValue, xsd__unsignedLong ulInputCFilterValue, xsd__unsignedLong ulInputDFilterValue, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct [MSXE173x__Response](#) *Response)

Set a filter to the input of a multifunction sub module.

- int [MSXE173x__MFCommonReferenceVoltageActivation](#) (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulActivationFlag, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct [MSXE173x__Response](#) *Response)

Permit to activate the reference voltage to pin D-.

2.19.1 Function Documentation

2.19.1.1 `int MSXE173x_MFCommonSetInputsFilter (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulInputAFilterValue, xsd__unsignedLong ulInputBFilterValue, xsd__unsignedLong ulInputCFilterValue, xsd__unsignedLong ulInputDFilterValue, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response)`

Parameters

[in] *ulMFModuleIndex* : index of the multifunction sub module (0 to 3).

[in] *ulInputAFilterValue* : Filter value for input A (0 to 262143)

- 0: Filter nicht benutzt
- 1: 100 ns
- 2: 200 ns
- 3: 300 ns ...
- 262143 : 26,2143 ms

[in] *ulInputBFilterValue* : Filter value for input B (0 to 262143)

- 0: Filter nicht benutzt
- 1: 100 ns
- 2: 200 ns
- 3: 300 ns ...
- 262143 : 26,2143 ms

[in] *ulInputCFilterValue* : Filter value for input C (0 to 262143)

- 0: Filter nicht benutzt
- 1: 100 ns
- 2: 200 ns
- 3: 300 ns ...
- 262143 : 26,2143 ms

[in] *ulInputDFilterValue* : Filter value for input D (0 to 262143)

- 0: Filter nicht benutzt
- 1: 100 ns
- 2: 200 ns
- 3: 300 ns ...
- 262143 : 26,2143 ms

[in] *ulOption01* : Set it to 0

[in] *ulOption02* : Set it to 0

[in] *ulOption03* : Set it to 0

[in] *ulOption04* : Set it to 0

[out] *Response* :

iReturnValue :

- 0 : means the remote function performed OK
- -1: means an system error occurred
- -2: Multifunction sub module index selection error
- -3: Input A filter value selection error

- -4: Input B filter value selection error
- -5: Input C filter value selection error
- -6: Input D filter value selection error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.19.1.2 `int MSXE173x__MFCommonReferenceVoltageActivation (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulActivationFlag, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MSXE173x__Response * Response)`

Parameters

[in] **ulMFModuleIndex** : index of the multifunction sub module (0 to 3).

[in] **ulActivationFlag** :

- 0: normal mode from D- (Default mode)
- 1: activate the reference voltage to pin D-

[in] **ulOption01** : Set it to 0

[in] **ulOption02** : Set it to 0

[out] **Response** :

iReturnValue :

- 0 : means the remote function performed OK
- -1: means an system error occurred
- -2: Multifunction sub module index selection error
- -3: Activation flag selection error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.20 MSX-E173x ENDAT functions

Data Structures

- struct [MSXE173x__MFEndatGetPositionResponse](#)
- struct [MSXE173x__MFEndatGetSensorPropertiesResponse](#)
- struct [MSXE173x__MFEndatGetErrorSourcesResponse](#)
- struct [MSXE173x__MFEndatSensorSendParameterResponse](#)
- struct [MSXE173x__MFEndatSensorSendPosAndRecvSelMemAreaResponse](#)
- struct [MSXE173x__MFEndatGetPositionWithAddDataResponse](#)
- struct [MSXE173x__MFEndatGetSelectedAdditionalDataResponse](#)
- struct [MSXE173x__MFEndatGetCurrentLatchConfigurationResponse](#)

Functions

- `int MSXE173x__MFEndatInitSensor (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulFrequency, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response *Response)`

Initialises an EnDat sensor.

- `int MSXE173x__MFEndatGetPosition (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetPositionResponse *Response)`

Reads the position of the sensor.

- `int MSXE173x__MFEndatGetSensorProperties (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetSensorPropertiesResponse *Response)`

Reads the properties of the sensor.

- `int MSXE173x__MFEndatGetErrorSources (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetErrorSourcesResponse *Response)`

Reads the error sources.

- `int MSXE173x__MFEndatResetErrorBits (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response *Response)`

Resets the error bits.

- `int MSXE173x__MFEndatSensorReceiveReset (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response *Response)`

Resets the sensor.

- `int MSXE173x__MFEndatSelectMemoryArea (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response *Response)`

Selects a memory area (see page 31/121 of EnDat specifications).

- `int MSXE173x__MFEndatSensorSendParameter (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatSensorSendParameterResponse *Response)`

Reads a parameter from the memory area that was last selected.

- `int MSXE173x__MFEndatSensorReceiveParameter (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulParam, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response *Response)`

Writes a parameter to the memory area that was last selected.

- `int MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatSensorSendPosAndRecvSelMemAreaResponse *Response)`

Reads the position value of the sensor and selects a memory area in the same cycle.

- `int MSXE173x__MFEndatSelectAdditionalData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulMFChannelIndex, xsd__unsignedLong ulAddDataCount, xsd__unsignedLong ulMrsCodeAD1, xsd__unsignedLong ulMrsCodeAD2, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response *Response)`

Selects the additional data that will be sent by the sensor.

- `int MSXE173x__MFEndatGetPositionWithAddData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetPositionWithAddDataResponse *Response)`

Reads the position of the sensor with additional data.

- `int MSXE173x__MFEndatGetSelectedAdditionalData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetSelectedAdditionalDataResponse *Response)`

Reads the currently selected additional data of the sensor.

- `int MSXE173x__MFEndatInitAndEnableLatchPositionValues (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulLatchSource, xsd__unsignedLong ulTriggerEdgeCount, xsd__unsignedLong ulDataFormat, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response *Response)`

Initialises and enables the latch logic to get the position and additional informations from the EnDat sensor using a trigger source.

- `int MSXE173x__MFEndatGetCurrentLatchConfiguration (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetCurrentLatchConfigurationResponse *Response)`

Reads the current latch configuration, started using the function MSXE173x__MFEndatInitAndEnableLatchPositionValues.

- `int MSXE173x__MFEndatDisableAndReleaseLatchPositionValues (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response *Response)`

Disables and releases the latch logic started with the function MSXE173x__MFEndatInitAndEnableLatchPositionValues.

2.20.1 Function Documentation

2.20.1.1 `int MSXE173x__MFEndatInitSensor (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulFrequency, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response)`

This function should be called once, in order to call the other EnDat functions.

Parameters

- [in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] **ulChannelIndex** Index of the channel. Set to 0
- [in] **ulFrequency** Frequency to use in kHz (500, 900, 1500, 2500, 4500)
- [in] **ulOption01** Reserved. Set to 0
- [in] **ulOption02** Reserved. Set to 0
- [in] **ulOption03** Reserved. Set to 0
- [in] **ulOption04** Reserved. Set to 0
- [out] **Response iReturnValue**
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The ulConnectorIndex parameter is wrong
 - **-4** The ulChannelIndex parameter is wrong
 - **-5** The driver is in a wrong state (must be INITIALISED or UNINITIALISED)
 - **-6** The component is not programmed as EnDat
 - **-7** Error while resetting sensor. Note: If no sensor is plugged on the selected channel, you will get this error.
 - **-8** Error while selecting memory area 0xB9
 - **-9** Error while reading alarm space (address 0x0)
 - **-10** Error while reading warning space (address 0x1)
 - **-11** Error while clearing errors (write 0 at address 0x0)
 - **-12** Error while clearing warnings (write 0 at address 0x1)
 - **-13** Error while selecting memory area 0xA1
 - **-14** Error while reading number of clock pulses for transfer of position value (address 0x0D)
 - **-15** Error while selecting memory area 0xA5
 - **-16** Error while reading EnDat command set (address 0x5)
 - **-17** Error while reading support of error messages 1 (address 0x3)
 - **-18** Error while reading support of warnings (address 0x4)
 - **-19** Error while reading EnDat ordering designation (address 0x8)
 - **-20** ulFrequency parameter is too high for current sensor
 - **-21** The ulFrequency parameter is wrong
 - **-22** EnDat ordering designation is invalid

- **-41** Transmission error. Please call `MSXE173x__MFEndatGetErrorSources` to get more information
 - **-100** Internal system error occurred. See value of `syserrno`
- syserrno* system error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

2.20.1.2 `int MSXE173x__MFEndatGetPosition (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE173x__MFEndatGetPositionResponse * Response)`

Before calling this function, you must call the `MSXE173x__MFEndatInitSensor` function.

Parameters

[in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] *ulChannelIndex* Index of the channel. Set to 0

[in] *ulOption01* Reserved. Set to 0

[in] *ulOption02* Reserved. Set to 0

[in] *ulOption03* Reserved. Set to 0

[in] *ulOption04* Reserved. Set to 0

[out] *Response sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The *ulConnectorIndex* parameter is wrong
- **-4** The *ulChannelIndex* parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** Error while reading the position
- **-41** Transmission error. Please call `MSXE173x__MFEndatGetErrorSources` to get more information
- **-100** Internal system error occurred. See value of `syserrno`

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulPositionLow Position - low bits

ulPositionHigh Position - high bits

Return values

0 SOAP_OK

Others See SOAP error

2.20.1.3 `int MSXE173x__MFEndatGetSensorProperties (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetSensorPropertiesResponse * Response)`

Before calling this function, you must call the MSXE173x__MFEndatInitSensor function.

Parameters

[in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] **ulChannelIndex** Index of the channel. Set to 0

[in] **ulOption01** Reserved. Set to 0

[in] **ulOption02** Reserved. Set to 0

[in] **ulOption03** Reserved. Set to 0

[in] **ulOption04** Reserved. Set to 0

[out] **Response** *sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The ulConnectorIndex parameter is wrong
- **-4** The ulChannelIndex parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** Error while selecting memory area 0xA3
- **-8** Error while reading ID number (address 0x8)
- **-9** Error while reading ID number (address 0x9)
- **-10** Error while reading ID number (address 0xA)
- **-11** Error while reading Serialnumber (address 0xB)
- **-12** Error while reading Serialnumber (address 0xC)
- **-13** Error while reading Serialnumber (address 0xD)
- **-14** Error while selecting memory area 0xA1
- **-15** Error while reading encoder model (address 0xE)
- **-16** Error while reading signal period length or signal periods per revolution for incremental output signals (address 0xF)
- **-17** Error while getting the position
- **-18** Error while selecting memory area 0xA3
- **-19** Error while reading signal period length or signal periods per revolution for incremental output signals (address 0x0)
- **-20** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x4)
- **-21** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x5)
- **-22** Error while selecting memory area 0xBD
- **-23** Error while reading scaling factor for resolution (address 0x1B)
- **-24** Error while reading measuring step, or measuring steps per revolution or subdivision values of a grating period (address 0x1C)
- **-25** Error while reading measuring step, or measuring steps per revolution or subdivision values of a grating period (address 0x1D)

- **-26** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x4)
- **-27** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x5)
- **-28** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x4)
- **-29** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x5)
- **-30** Error while reading distinguishable revolutions (address 0x1)
- **-31** Error while selecting memory area 0xBD
- **-32** Error while reading number of distinguishable revolutions with scaling factor (address 0x22)
- **-33** Error while selecting memory area 0xBD
- **-34** Error while reading status of additional datum 1 (address 0x0)
- **-35** Error while reading status of additional datum 2 (address 0x1)
- **-41** Transmission error. Please call `MSXE173x__MFEndatGetErrorSources` to get more information
- **-100** Internal system error occurred. See value of `syserrno`

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulIDNumberLsb ID Number - low bits (see page 67/121 of EnDat specifications)

ulIDNumberMsb ID Number - high bits (see page 67/121 of EnDat specifications)

ulSerialNumberLsb Serialnumber - low bits (see page 68/121 of EnDat specifications)

ulSerialNumberMsb Serialnumber - high bits (see page 68/121 of EnDat specifications)

ulModel Model/Type of the sensor (see page 79/84 of EnDat application notes 722024)

- **0, 1, 2, 3** Incremental linear encoder
- **4, 6** Absolute linear encoder
- **8, 9, 10, 11** Incremental rotary encoder or angle encoder
- **12** Singleturn encoder
- **13, 14** Multiturn encoder

ulMode Support of EnDat 2.2

- **0** Sensor does not support EnDat 2.2. commands
- **1** Sensor supports EnDat 2.2 commands

ulPositionSize Size of the position value send by the sensor (in bits)

ulSignalPeriod Signal period length or signal periods per revolution for incremental output signals (see page 79/84 of EnDat application notes 722024)

ulStepPerRevolution Measuring step length or measuring steps per revolution with serial data transfer (see page 79/84 of EnDat application notes 722024)

ulNumberOfRevolution Distinguishable revolutions - only for multiturn encoders (see page 79/84 of EnDat application notes 722024)

ulScalingFactor Scaling factor for resolution (see page 79/84 of EnDat application notes 722024)

ulAdditionalData Supported additional data (see page 85/121 of EnDat specifications)

- **Bit 0** "Position value 2" is available (MRS-Code: 0x42, 0x43, 0x44)
- **Bit 1** "Test values" is available (MRS-Code: 0x49, 0x4A, 0x4B)
- **Bit 2** "Temperature sensor 1 (external)" is available (MRS-Code: 0x4C)
- **Bit 3** "Temperature sensor 2 (external)" is available (MRS-Code: 0x4D)
- **Bit 4** "Additional sensors" is available (MRS-Code: 0x4E)
- **Bit 16** "Commutation" is available (MRS-Code: 0x51)

- **Bit 17** "Acceleration" is available (MRS-Code: 0x52)
- **Bit 18** "Limit position signals" is available (MRS-Code: 0x54)
- **Bit 19** "Asynchronous position value" is available (MRS-Code: 0x56, 0x57, 0x58)
- **Bit 20** "Operating status error sources" is available (MRS-Code: 0x59)
- **Bit 23** "Timestamp" is available (MRS-Code: 0x5B)

Return values

0 SOAP_OK

Others See SOAP error

2.20.1.4 `int MSXE173x_MFEndatGetErrorSources (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE173x_MFEndatGetErrorSourcesResponse * Response)`

The error are reseted after a call to MSXE173x_MFEndatResetErrorBits. If a function returns an error, please use this function to check if it is not a communication error.

Parameters

[in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] **ulChannelIndex** Index of the channel. Set to 0

[in] **ulOption01** Reserved. Set to 0

[in] **ulOption02** Reserved. Set to 0

[in] **ulOption03** Reserved. Set to 0

[in] **ulOption04** Reserved. Set to 0

[out] **Response** *sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The ulConnectorIndex parameter is wrong
- **-4** The ulChannelIndex parameter is wrong
- **-5** The component is not programmed as EnDat
- **-100** Internal system error occurred. See value of syserrno

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulErrorSrc Mask of bits that give the current error sources. Each bit represents an error. If the bit is set to 1, the error is present.

- **Bit 0** Invalid mode command
- **Bit 1** Invalid MRS-Code
- **Bit 2** Transmission is not completed
- **Bit 3** Communication command is not supported
- **Bit 4** MRS-Code is not allowed
- **Bit 6** Invalid address is selected or sensor's EEPROM is written while being busy
- **Bit 7** Try to write a protected memory place
- **Bit 8** Write-Protect configuration is tried to be reset (if a memory place is write-protected, it cannot be reset)

- **Bit 9** Block address is not available
- **Bit 10** Invalid address for the communication command
- **Bit 11** Invalid additional data (or additional data not supported by the sensor)

Return values

0 SOAP_OK

Others See SOAP error

2.20.1.5 `int MSXE173x__MFEndatResetErrorBits (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response)`

It can be used before each command in order to get (after the call of the command) the status of the system using MSXE173x__MFEndatGetErrorSources.

Once an error is detected using MSXE173x__MFEndatGetErrorSources, you have to clear it using this function.

Parameters

[in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] **ulChannelIndex** Index of the channel. Set to 0

[in] **ulOption01** Reserved. Set to 0

[in] **ulOption02** Reserved. Set to 0

[in] **ulOption03** Reserved. Set to 0

[in] **ulOption04** Reserved. Set to 0

[out] **Response iReturnValue**

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The ulConnectorIndex parameter is wrong
- **-4** The ulChannelIndex parameter is wrong
- **-5** The component is not programmed as EnDat
- **-100** Internal system error occurred. See value of syserrno

syserrno system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

2.20.1.6 `int MSXE173x__MFEndatSensorReceiveReset (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response)`

This function has the same effect as an hardware reboot of the sensor.

Parameters

- [in] ***ulConnectorIndex*** Index of the EnDat connector (0 to 3). See on the MSX-E system.
 - [in] ***ulChannelIndex*** Index of the channel. Set to 0
 - [in] ***ulOption01*** Reserved. Set to 0
 - [in] ***ulOption02*** Reserved. Set to 0
 - [in] ***ulOption03*** Reserved. Set to 0
 - [in] ***ulOption04*** Reserved. Set to 0
 - [out] ***Response iReturnValue***
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The *ulConnectorIndex* parameter is wrong
 - **-4** The *ulChannelIndex* parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be INITIALISED or UNINITIALISED or ERROR)
 - **-7** Error while resetting sensor
 - **-41** Transmission error. Please call `MSXE173x__MFEndatGetErrorSources` to get more information
 - **-100** Internal system error occurred. See value of `syserrno`
- syserrno*** system-error code (the value of the libc "errno" code)

Return values

- 0** SOAP_OK
- Others*** See SOAP error

2.20.1.7 `int MSXE173x__MFEndatSelectMemoryArea (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response)`

In order to send or read parameters, the memory area must first be selected.

Before calling this function, you must call the `MSXE173x__MFEndatInitSensor` function.

Parameters

- [in] ***ulConnectorIndex*** Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] ***ulChannelIndex*** Index of the channel. Set to 0
- [in] ***ulMrsCode*** The MRS-code corresponding to the memory area that you want to select (see page 31/121 and 84/121 of EnDat specifications)
 - **0xB9** Operating status (address area: 0x0 - 0x3)
 - **0xA1** Parameters of the encoder manufacturer - first part (address area: 0x4 - 0xF)
 - **0xA3** Parameters of the encoder manufacturer - second part (address area: 0x0 - 0xF)
 - **0xA5** Parameters of the encoder manufacturer - third part (address area: 0x0 - 0xF)
 - **0xA7** Operating parameters (address area: 0x0 - 0xF)

- **0xA9** Parameters of the OEM - first part (address area: depending on the sensor)
- **0xAB** Parameters of the OEM - second part (address area: depending on the sensor)
- **0xAD** Parameters of the OEM - third part (address area: depending on the sensor)
- **0xAF** Parameters of the OEM - fourth part (address area: depending on the sensor)
- **0xB1** Compensation values of the encoder manufacturer - first part (address area: depending on the sensor)
- **0xB3** Compensation values of the encoder manufacturer - second part (address area: depending on the sensor)
- **0xB5** Compensation values of the encoder manufacturer - third part (address area: depending on the sensor)
- **0xB7** Compensation values of the encoder manufacturer - fourth part (address area: depending on the sensor)

[in] *ulOption01* Reserved. Set to 0

[in] *ulOption02* Reserved. Set to 0

[in] *ulOption03* Reserved. Set to 0

[in] *ulOption04* Reserved. Set to 0

[out] *Response iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The *ulConnectorIndex* parameter is wrong
- **-4** The *ulChannelIndex* parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** The *ulMrsCode* parameter is wrong
- **-8** Error while selecting the memory area
- **-41** Transmission error. Please call *MSXE173x__MFEndatGetErrorSources* to get more information
- **-100** Internal system error occurred. See value of *syserrno*

syserrno system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

2.20.1.8 `int MSXE173x__MFEndatSensorSendParameter (xsd__unsignedLong
ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong
ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulOption01,
xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong
ulOption04, struct MSXE173x__MFEndatSensorSendParameterResponse * Response)`

Before calling this function, you must call the *MSXE173x__MFEndatInitSensor* function to initialise the sensor, and then *MSXE173x__MFEndatSelectMemoryArea*, or *MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea* to select the memory area that contains the parameter you want to read.

Parameters

- [in] ***ulConnectorIndex*** Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] ***ulChannelIndex*** Index of the channel. Set to 0
- [in] ***ulMrsCode*** The MRS-code corresponding to the last memory area that you have selected (see MSXE173x__MFEndatSelectMemoryArea or MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea)
- [in] ***ulAddress*** The address of the parameter that you want to read (0x0-0xFF)
- [in] ***ulOption01*** Reserved. Set to 0
- [in] ***ulOption02*** Reserved. Set to 0
- [in] ***ulOption03*** Reserved. Set to 0
- [in] ***ulOption04*** Reserved. Set to 0
- [out] ***Response*** ***sResponse.iReturnValue***
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The *ulConnectorIndex* parameter is wrong
 - **-4** The *ulChannelIndex* parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be INITIALISED)
 - **-7** The *ulMrsCode* parameter is wrong
 - **-8** The *ulAddress* parameter is wrong
 - **-9** Your sensor is not compatible with EnDat 2.2, but the parameter *ulMrsCode* is only available for EnDat 2.2
 - **-10** The last selected memory area do not corresponds to the parameter *ulMrsCode*. Please call MSXE173x__MFEndatSelectMemoryArea or MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea with the wished *ulMrsCode* before
 - **-11** Error while reading parameter
 - **-41** Transmission error. Please call MSXE173x__MFEndatGetErrorSources to get more information
 - **-100** Internal system error occurred. See value of *syserrno*
- sResponse.syserrno*** system-error code (the value of the libc "errno" code)
- ulParam*** Value of the parameter

Return values

- 0** SOAP_OK
- Others** See SOAP error

2.20.1.9 `int MSXE173x__MFEndatSensorReceiveParameter (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulParam, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response)`

Before calling this function, you must call the MSXE173x__MFEndatInitSensor function to initialise the sensor, and then MSXE173x__MFEndatSelectMemoryArea, or MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea to select the memory area that contains the parameter you want to write.

Parameters

- [in] ***ulConnectorIndex*** Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] ***ulChannelIndex*** Index of the channel. Set to 0
- [in] ***ulMrsCode*** The MRS-code corresponding to the last memory area that you have selected (see MSXE173x__MFEndatSelectMemoryArea or MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea)
- [in] ***ulAddress*** The address of the parameter that you want to write (0x0-0xFF)
- [in] ***ulParam*** The new value of the parameter
- [in] ***ulOption01*** Reserved. Set to 0
- [in] ***ulOption02*** Reserved. Set to 0
- [in] ***ulOption03*** Reserved. Set to 0
- [in] ***ulOption04*** Reserved. Set to 0
- [out] ***Response iReturnValue***
- **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The *ulConnectorIndex* parameter is wrong
 - **-4** The *ulChannelIndex* parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be INITIALISED)
 - **-7** The *ulMrsCode* parameter is wrong
 - **-8** The *ulAddress* parameter is wrong
 - **-9** The last selected memory area do not corresponds to the parameter *ulMrsCode*. Please call MSXE173x__MFEndatSelectMemoryArea or MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea with the wished *ulMrsCode* before
 - **-10** Error while writing parameter
 - **-41** Transmission error. Please call MSXE173x__MFEndatGetErrorSources to get more information
 - **-100** Internal system error occurred. See value of *syserrno*
- syserrno* system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

2.20.1.10 `int MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatSensorSendPosAndRecvSelMemAreaResponse * Response)`

In order to send or read parameters, the memory area must first be selected.

Before calling this function, you must call the MSXE173x__MFEndatInitSensor function.

This function differs from the MSXE173x_MFEndatSelectMemoryArea function, since it can select memory areas that are reserved for EnDat 2.2. sensors.

This function is reserved for EnDat 2.2 sensors. It will returns an error if the sensor does not support EnDat 2.2 commands.

Parameters

- [in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] **ulChannelIndex** Index of the channel. Set to 0
- [in] **ulMrsCode** The MRS-code corresponding to the memory area that you want to select (see page 31/121 and 84/121 of EnDat specifications)
 - **0xB9** Operating status (address area: 0x0 - 0x3)
 - **0xA1** Parameters of the encoder manufacturer - first part (address area: 0x4 - 0xF)
 - **0xA3** Parameters of the encoder manufacturer - second part (address area: 0x0 - 0xF)
 - **0xA5** Parameters of the encoder manufacturer - third part (address area: 0x0 - 0xF)
 - **0xA7** Operating parameters (address area: 0x0 - 0xF)
 - **0xA9** Parameters of the OEM - first part (address area: depending on the sensor)
 - **0xAB** Parameters of the OEM - second part (address area: depending on the sensor)
 - **0xAD** Parameters of the OEM - third part (address area: depending on the sensor)
 - **0xAF** Parameters of the OEM - fourth part (address area: depending on the sensor)
 - **0xB1** Compensation values of the encoder manufacturer - first part (address area: depending on the sensor)
 - **0xB3** Compensation values of the encoder manufacturer - second part (address area: depending on the sensor)
 - **0xB5** Compensation values of the encoder manufacturer - third part (address area: depending on the sensor)
 - **0xB7** Compensation values of the encoder manufacturer - fourth part (address area: depending on the sensor)
 - **0xBD** Parameters of the encoder manufacturer for EnDat 2.2 (address area: 0x0 - 0x3F)
 - **0xBB** Operating parameters 2 (address area: depending on the sensor)
 - **0xBF** Parameters of the section 2 memory area (address area: depending on the sensor)
- [in] **ulAddress** Selected block address (only used if ulMrsCode is set to 0xBF. If ulMrsCode is not 0xBF, set it to 0). It is used to address further section 2 memory areas (see page 46/121 of EnDat specifications)
- [in] **ulOption01** Reserved. Set to 0
- [in] **ulOption02** Reserved. Set to 0
- [in] **ulOption03** Reserved. Set to 0
- [in] **ulOption04** Reserved. Set to 0
- [out] **Response sResponse.iReturnValue**
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The ulConnectorIndex parameter is wrong
 - **-4** The ulChannelIndex parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be INITIALISED)
 - **-7** The ulMrsCode parameter is wrong

- **-8** The `ulAddress` parameter is wrong
- **-9** The sensor is not compatible with EnDat 2.2
- **-10** Error while getting position and selecting memory area
- **-41** Transmission error. Please call `MSXE173x__MFEndatGetErrorSources` to get more information
- **-100** Internal system error occurred. See value of `syserrno`

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulPositionLow Position - low bits

ulPositionHigh Position - high bits

Return values

0 SOAP_OK

Others See SOAP error

2.20.1.11 `int MSXE173x__MFEndatSelectAdditionalData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulMFChannelIndex, xsd__unsignedLong ulAddDataCount, xsd__unsignedLong ulMrsCodeAD1, xsd__unsignedLong ulMrsCodeAD2, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response)`

Some additional data are not available on all sensors. To get the available additional data of your sensor, please use the function `MSXE173x__MFEndatGetSensorProperties`.

If you select an additional data that is not available on your sensor, you will get the parameter `ucErrorSrc13` set to 1 when calling the function `MSXE173x__MFEndatGetErrorSources`.

The additional data are extra values that the sensor can send (in the same cycle as its position value).

This function is reserved for EnDat 2.2 sensors. It will returns an error if the sensor does not support EnDat 2.2 commands.

Parameters

[in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] *ulChannelIndex* Index of the channel. Set to 0

[in] *ulAddDataCount* The number of selected additional data (0 to 2)

[in] *ulMrsCodeAD1* The MRS-Code for the first additional data

- **0x40** Send additional info 1 without data contents
- **0x42** Position value 2 word 1 LSB
- **0x43** Position value 2 word 2
- **0x44** Position value 2 word 3 MSB
- **0x49** Test values word 1 LSB
- **0x4A** Test values word 2
- **0x4B** Test values word 3 MSB
- **0x4C** Temperature sensor 1 (external)
- **0x4D** Temperature sensor 2 (external)
- **0x4E** Additional sensors

[in] *ulMrsCodeAD2* The MRS-Code for the second additional data

- **0x50** Send additional datum 2 without data contents
- **0x51** Commutation
- **0x52** Acceleration
- **0x54** Limit position signals
- **0x56** Asynchronous position value word 1 LSB
- **0x57** Asynchronous position value word 2
- **0x58** Asynchronous position value word 3 MSB
- **0x59** Operating status error sources
- **0x5B** Timestamp

[in] *ulOption01* Reserved. Set to 0

[in] *ulOption02* Reserved. Set to 0

[in] *ulOption03* Reserved. Set to 0

[in] *ulOption04* Reserved. Set to 0

[out] *Response iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The *ulConnectorIndex* parameter is wrong
- **-4** The *ulChannelIndex* parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** The *ulAddDataCount* parameter is wrong
- **-8** The *ulMrsCodeAD1* parameter is wrong
- **-9** The *ulMrsCodeAD2* parameter is wrong
- **-10** The sensor is not compatible with EnDat 2.2
- **-11** Error while deactivating additional data 2
- **-12** Error while deactivating additional data 1
- **-13** Error while activating additional data 1
- **-14** Error while deactivating additional data 2
- **-15** Error while deactivating additional data 1
- **-16** Selected additional data 1 is wrong or not available on this sensor. Please call *MSXE173x__MFEndatGetErrorSources* to get more information
- **-17** Error while activating additional data 1. Please call *MSXE173x__MFEndatGetErrorSources* to get more information
- **-18** Error while getting the current position value
- **-19** Error while activating additional data 2
- **-20** Error while deactivating additional data 2
- **-21** Error while deactivating additional data 1
- **-22** Selected additional data 2 is wrong or not available on this sensor. Please call *MSXE173x__MFEndatGetErrorSources* to get more information
- **-23** Error while activating additional data 2. Please call *MSXE173x__MFEndatGetErrorSources* to get more information
- **-24** Error while getting the current position value
- **-41** Transmission error. Please call *MSXE173x__MFEndatGetErrorSources* to get more information
- **-100** Internal system error occurred. See value of *syserrno*

syserrno system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

2.20.1.12 `int MSXE173x__MFEndatGetPositionWithAddData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetPositionWithAddDataResponse * Response)`

Before calling this function, you must call the MSXE173x__MFEndatInitSensor function.

You must also call the function MSXE173x__MFEndatSelectAdditionalData to select the additional data that you want to receive.

This function is reserved for EnDat 2.2 sensors. It will returns an error if the sensor does not support EnDat 2.2 commands.

Parameters

[in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] *ulChannelIndex* Index of the channel. Set to 0

[in] *ulOption01* Reserved. Set to 0

[in] *ulOption02* Reserved. Set to 0

[in] *ulOption03* Reserved. Set to 0

[in] *ulOption04* Reserved. Set to 0

[out] *Response sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The *ulConnectorIndex* parameter is wrong
- **-4** The *ulChannelIndex* parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** Error while reading the position
- **-41** Transmission error. Please call MSXE173x__MFEndatGetErrorSources to get more information
- **-100** Internal system error occurred. See value of *syserrno*

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulPositionLow Position - low bits

ulPositionHigh Position - high bits

ulAddData1 Value of the additional data 1

ulAddData2 Value of the additional data 2

Return values

0 SOAP_OK

Others See SOAP error

2.20.1.13 `int MSXE173x__MFEndatGetSelectedAdditionalData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetSelectedAdditionalDataResponse * Response)`

The additional data are selected using the function MSXE173x__MFEndatSelectAdditionalData.

Parameters

- [in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] *ulChannelIndex* Index of the channel. Set to 0
- [in] *ulOption01* Reserved. Set to 0
- [in] *ulOption02* Reserved. Set to 0
- [in] *ulOption03* Reserved. Set to 0
- [in] *ulOption04* Reserved. Set to 0
- [out] *Response sResponse.iReturnValue*
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The *ulConnectorIndex* parameter is wrong
 - **-4** The *ulChannelIndex* parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-100** Internal system error occurred. See value of *syserrno*
- sResponse.syserrno* system-error code (the value of the libc "errno" code)
- ulAdCount* Number of selected additional data
- ulMrsCodeAD1* MRS code of the additional data 1
- ulMrsCodeAD2* MRS code of the additional data 2

Return values

- 0** SOAP_OK
- Others* See SOAP error

2.20.1.14 `int MSXE173x__MFEndatInitAndEnableLatchPositionValues (xsd__unsignedLong ulmFModuleIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulLatchSource, xsd__unsignedLong ulTriggerEdgeCount, xsd__unsignedLong ulDataFormat, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response)`

Before calling this function, you must call the MSXE173x__MFEndatInitSensor function.

When the selected trigger occurs, the position value of the selected EnDat channel will be measured and send through the dataserver.

If you chose additional data using the function MSXE173x__MFEndatSelectAdditionalData and if your selected *ulDataFormat* enables it, you will also receive the value of the selected additional data.

Note: using a synchro timer (and activating the synchro trigger as latch source) enables to get position of the EnDat sensor at a defined rate.

Parameters

[in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] **ulChannelIndex** Index of the channel. Set to 0

[in] **ulLatchSource** Mask of bits, that defines the trigger source.

Bit 0: Hardware trigger If you select the hardware trigger, you must also choose the edge detection (Bit 2 and 3). Of course, you can choose both.

- 1 activated
- 0 not used

Bit 1 : Synchro trigger

- 1 activated
- 0 not used

Bit 2 : Trigger rising edge (only if hardware trigger is selected)

- 1 activated
- 0 not used

Bit 3 : Trigger falling edge (only if hardware trigger is selected)

- 1 activated
- 0 not used

```
ulLatchSource = 2 (0b10)      -> the latch source is the synchro trigger
ulLatchSource = 5 (0b101)    -> the latch source is a rising edge of the hardware trigger
ulLatchSource = 13 (0b1101) -> the latch source is a rising or a falling edge of the hardware trigger
```

[in] **ulTriggerEdgeCount** Number of edges required to detect an hardware trigger (only if hardware trigger is selected)

[in] **ulDataFormat** Mask of bits, that defines the format of data that will be sent by the data server. It is a combination of bits. You can set all the bits to 1 if you want all the informations 0b11111 = 31.

You can also, for example, only wants the timestamp and the digital I/Os state (0b11 = 3).

You can also choose, for example, to get the additional data 2 (0b1000 = 8).

Bit 0: Timestamp

- 0 Not sent
- 1 timestamp sent by the dataserver

Bit 1: Digital I/Os state

- 0 Not sent
- 1 digital I/Os state sent by the data server

Bit 2: Additional data 1

- 0 Not sent
- 1 Additional data 1 sent by the data server

Bit 3: Additional data 2

- 0 Not sent
- 1 Additional data 2 sent by the data server

Bit 4: Format of the value

- 0 Raw value (as sent by the sensor)
- 1 Standardised value. The format of the frame will then depends on the model of the sensor. See system documentation.

If ulDataFormat is set to 0, the frame send by the dataserver will have the following definition

```

unsigned long eventsrc;           // High 16 bits (MSB): Channel concerned
    (0 to 3)

    // Low 16 bits (LSB): Bit mask representing the source of the event
    SB): Bit mask representing the source of the event
    rdware trigger                // Bit 0: Hardware trigger
    nchro trigger                 // Bit 1: Synchronization trigger
    mpare                         // Bit 2: Comparison trigger

unsigned long positionlow;        // Low bits of the position
unsigned long positionhigh;      // High bits of the position
unsigned long error;              // Status of the communication. Same value
    as the value returned by the function MSXE173x_MFEndatGetErrorSources

```

If you set `ulDataFormat` to 1 (send timestamp), then the frame send by the dataserver will be changed. At the end of the "basic" frame, we add the timestamp.

```
unsigned long ts;           // Second part of the timestamp
unsigned long tus;          // Microsecond part of the timestamp
```

If you set ulDataFormat to 2 = 0b10 (send digital I/Os state), then the frame send by the dataserver will be changed. At the end of the "basic" frame, we add the state of the digital I/Os.

```

unsigned long digiostate;          // State of the digital I/Os. Bit mask that encod
    es all digital I/Os.

                                   // 0b0
    All I/Os are set to low

                                   // 0b100
    I/O 2 is set to high, all others are set to 0

                                   // 0b11111111111111
    1111 : All I/Os are set to high

```

If you set `ulDataFormat` to `4 = 0b100` (send additional data 1), then the frame send by the `dataserver` will be changed. At the end of the "basic" frame, we add the value of the first additional data.

```

unsigned long ad1value;
\endcode

If you set ulDataFormat to 8 = 0b1000 (send additional data 2), then the frame s
end by the dataserver will be changed. At the end of the "basic" frame, we add th
e value of the second additional data. \n

\endcode
unsigned long ad2value;
\endcode

If you set the bits 0, 1, 2 and 3 to 1, the format of the frame will then be: \n

\endcode
unsigned long eventsrc;                // High 16 bits (MSB): Channel concerned
    (0 to 3)

                                // Low 16 bits (L
SB): Bit mask representing the source of the event
                                //      Bit 0: Ha
rdware trigger
                                //      Bit 1: Sy
nchro trigger
                                //      Bit 2: Co
mpare

unsigned long positionlow;             // Low bits of the position
unsigned long positionhigh;           // High bits of the position
unsigned long error;                  // Status of the communication. Same valu
e as the value returned by the function MSXE173x__MFEndatGetErrorSources
unsigned long ts;                     // Second part of the timestamp
unsigned long tus;                    // Microsecond part of the timestamp
unsigned long digiostate;              // State of the digital I/Os. Bit mask that encod

```

```

        es all digital I/Os.
        All I/Os are set to low
        I/O 2 is set to high, all others are set to 0
        1111 : All I/Os are set to high
unsigned long ad1value;
unsigned long ad2value;

```

The bit 4 of the `ulDataFormat` is more complex. The format of the frame that is sent by the `dataserver` will depend on the model of the sensor used. See the system documentation for more information, or the "Acquisition" menu of the web site.

[in] ***ulOption01*** Reserved. Set to 0

[in] ***ulOption02*** Reserved. Set to 0

[in] ***ulOption03*** Reserved. Set to 0

[in] ***ulOption04*** Reserved. Set to 0

[out] ***Response iReturnValue***

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The `ulConnectorIndex` parameter is wrong
- **-4** The `ulChannelIndex` parameter is wrong
- **-5** The component is not programmed as `EnDat`
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** The `ulLatchSource` parameter is wrong
- **-8** The `ulDataFormat` parameter is wrong
- **-9** Error while getting sensor properties
- **-10** The multiturn part size of the sensor is 0
- **-11** The singleturn part size of the sensor is 0
- **-12** The step per revolution properties of the sensor is 0
- **-41** Transmission error. Please call `MSXE173x__MFEndatGetErrorSources` to get more information
- **-100** Internal system error occurred. See value of `syserrno`

syserrno system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

2.20.1.15 `int MSXE173x__MFEndatGetCurrentLatchConfiguration (xsd__unsignedLong ulMModuleIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetCurrentLatchConfigurationResponse * Response)`

Parameters

[in] ***ulConnectorIndex*** Index of the `EnDat` connector (0 to 3). See on the MSX-E system.

[in] ***ulChannelIndex*** Index of the channel. Set to 0

[in] ***ulOption01*** Reserved. Set to 0

[in] ***ulOption02*** Reserved. Set to 0

[in] ***ulOption03*** Reserved. Set to 0

[in] ***ulOption04*** Reserved. Set to 0

[out] ***Response*** *sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The *ulConnectorIndex* parameter is wrong
- **-3** The *ulChannelIndex* parameter is wrong
- **-4** The component is not programmed as EnDat
- **-100** Internal system error occurred. See value of *syserrno*

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulRunning The running state.

- **0** Not running (do not use other return parameters, there will be set to 0)
- **1** Latch logic is running

ulLatchSource Mask of bits, that defines the trigger source. See documentation of MSXE173x__MFEndatInitAndEnableLatchPositionValues

ulTriggerEdgeCount Number of edges required to detect an hardware trigger (only if hardware trigger is selected)

ulDataFormat Mask of bits, that defines the format of data that will be sent by the data server. See documentation of MSXE173x__MFEndatInitAndEnableLatchPositionValues

ulModel (Used for computation) Model/Type of the sensor (see page 79/84 of EnDat application notes 722024)

- **0, 1, 2, 3** Incremental linear encoder
- **4, 6** Absolute linear encoder
- **8, 9, 10, 11** Incremental rotary encoder or angle encoder
- **12** Singleturn encoder
- **13, 14** Multiturn encoder

ulPositionSize (Used for computation) Size of the position value send by the sensor (in bits)

ulSignalPeriod (Used for computation) Signal period length or signal periods per revolution for incremental output signals

ulStepPerRevolution (Used for computation) Measuring step length or measuring steps per revolution with serial data transfer

ulNumberOfRevolution (Used for computation) Distinguishable revolutions - only for multiturn encoders

ulScalingFactor (Used for computation) Scaling factor for resolution

Return values

0 SOAP_OK

Others See SOAP error

2.20.1.16 `int MSXE173x__MFEndatDisableAndReleaseLatchPositionValues (`
`xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex,`
`xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong`
`ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response`
`)`

Parameters

- [in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.
 - [in] *ulChannelIndex* Index of the channel. Set to 0
 - [in] *ulOption01* Reserved. Set to 0
 - [in] *ulOption02* Reserved. Set to 0
 - [in] *ulOption03* Reserved. Set to 0
 - [in] *ulOption04* Reserved. Set to 0
 - [out] *Response iReturnValue*
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The *ulConnectorIndex* parameter is wrong
 - **-4** The *ulChannelIndex* parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be LATCH_RUNNING or LATCH_FIFO_OVERFLOW)
 - **-100** Internal system error occurred. See value of *syserrno*
- syserrno* system-error code (the value of the libc "errno" code)

Return values

- 0** SOAP_OK
- Others* See SOAP error

2.21 MSX-E17xx digital I/O functions

Data Structures

- struct [MSXE17xx__DigitalIOGetNumberResponse](#)

Functions

- `int MSXE17xx__DigitalIOGetNumber (void *, struct MSXE17xx__DigitalIOGetNumberResponse *Response)`
Returns the number of digital IO available on the module.
- `int MSXE17xx__DigitalIOInitPortConfiguration (xsd__unsignedLong ulPort, xsd__unsignedLong ulPortConfiguration, struct MSXE17xx__Response *Response)`
Initialise a digital i/o port (2 channels).

- `int MSXE17xx__DigitalIOReadChannelValue (xsd__unsignedLong ulChannel, struct MSXE17xx__unsignedLongResponse *Response)`
Read a digital i/o channel value.
- `int MSXE17xx__DigitalIOReadAllChannelsValue (void ___, struct MSXE17xx__unsignedLongResponse *Response)`
Read all digital i/o channels value. If channel is configured as output, then this function return the status of the output.
- `int MSXE17xx__DigitalIOWriteChannelValue (xsd__unsignedLong ulChannel, xsd__unsignedLong ulChannelValue, struct MSXE17xx__Response *Response)`
write a digital i/o channel value
- `int MSXE17xx__DigitalIOWriteAllChannelsValue (xsd__unsignedLong ulChannelValue, struct MSXE17xx__Response *Response)`
write all digital i/o channels value
- `int MSXE17xx__DigitalIOReleasePortConfiguration (xsd__unsignedLong ulPort, struct MSXE17xx__Response *Response)`
Release a digital i/o port (2 channels).
- `int MSXE17xx__DigitalIOTestShortCircuit (xsd__unsignedLong ulOption, struct MSXE17xx__unsignedLongResponse *Response)`
Test short circuit status.
- `int MSXE17xx__DigitalIORearmShortCircuit (xsd__unsignedLong ulOption, struct MSXE17xx__Response *Response)`
Rearm digital outputs after a short circuit happened.

2.21.1 Function Documentation

2.21.1.1 `int MSXE17xx__DigitalIOGetNumber (void * ___, struct MSXE17xx__DigitalIOGetNumberResponse * Response)`

Parameters

[in] *None*

[out] *Response* :

sResponse.iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred (check errno in this case)

sResponse.syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.21.1.2 `int MSXE17xx_DigitalIOInitPortConfiguration (xsd__unsignedLong ulPort, xsd__unsignedLong ulPortConfiguration, struct MSXE17xx_Response * Response)`

Parameters

[in] ***ulPort*** : Index of the digital i/o port (0 to 7)

[in] ***ulPortConfiguration*** : Define the port configuration

- 0 : input
- 1 : output

[out] ***Response*** :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -2: Digital i/o port selection error
- -3: Port configuration selection error
- -100: Init dig i/o port kernel function error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.21.1.3 `int MSXE17xx_DigitalIOReadChannelValue (xsd__unsignedLong ulChannel, struct MSXE17xx_unsignedLongResponse * Response)`

Parameters

[in] ***ulChannel*** : Index of the digital i/o channel (0 to 15)

[out] ***Response*** :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -2: Digital i/o channel selection error
- -100: Read dig i/o channel value kernel function error

syserrno : system-error code (the value of the libc "errno" code) ***ulValue*** : i/o channel value:

- 0
- 1

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.21.1.4 int MSXE17xx__DigitalIOReadAllChannelsValue (void * _, struct MSXE17xx__unsignedLongResponse * Response)

Parameters

[in] _ : no input parameter

[out] Response :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -100: Read dig i/o channel value kernel function error

syserrno : system-error code (the value of the libc "errno" code) *ulValue* : i/o channels value(each bit correspond to one channel)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.21.1.5 int MSXE17xx__DigitalIOWriteChannelValue (xsd__unsignedLong ulChannel, xsd__unsignedLong ulChannelValue, struct MSXE17xx__Response * Response)

Parameters

[in] *ulChannel* : Index of the digital i/o channel (0 to 15)

[in] *ulChannelValue* : Channel value

- 0
- 1

[out] Response :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -2: Digital i/o channel selection error
- -3: Channel value error
- -100: Write dig i/o channel value kernel function error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.21.1.6 int MSXE17xx__DigitalIOWriteAllChannelsValue (xsd__unsignedLong ulChannelValue, struct MSXE17xx__Response * Response)

Parameters

[in] *ulChannelValue* : Channels value (each bit corresponds to a channel)

[out] **Response** :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -2: Channels value error
- -100: Write dig i/o channel value kernel function error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.21.1.7 int MSXE17xx_DigitalIOReleasePortConfiguration (xsd__unsignedLong ulPort, struct MSXE17xx_Response * Response)

Parameters

[in] **ulPort** : Index of the digital i/o port (0 to 7)

[out] **Response** :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -2: Digital i/o port selection error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.21.1.8 int MSXE17xx_DigitalIOTestShortCircuit (xsd__unsignedLong ulOption, struct MSXE17xx_unsignedLongResponse * Response)

Parameters

[in] **ulOption** : reserved

[out] **Response** :

iReturnValue :

- 0 : means the remote function performed OK
- -1: means an system error occurred

syserrno : system-error code (the value of the libc "errno" code)

ulValue : short circuit status: from 0 to 0xffff, one bit for each output

- 0 : no short circuit
- 1 : short circuit

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.21.1.9 `int MSXE17xx_DigitalIORearmShortCircuit (xsd__unsignedLong ulOption, struct MSXE17xx__Response * Response)`

Parameters

[in] *ulOption* : reserved

[out] *Response* :

iReturnValue :

- 0 : means the remote function performed OK
- -1: means an system error occurred

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.22 MSX-E17xx IO watchdog functions

Data Structures

- struct [MSXE17xx_IOWatchdogGetStatusAndValueResponse](#)

Functions

- `int MSXE17xx_IOWatchdogInitAndStart (xsd__unsignedLong ulTimeBase, xsd__unsignedLong ulTimeValue, xsd__unsignedLong ulOption1, xsd__unsignedLong ulOption2, struct MSXE17xx__Response *Response)`

Init and start the digital output IO watchdog.

- `int MSXE17xx_IOWatchdogStopAndRelease (xsd__unsignedLong ulOption, struct MSXE17xx__Response *Response)`

Stop and release the digital output watchdog.

- `int MSXE17xx_IOWatchdogGetStatusAndValue (xsd__unsignedLong ulOption, struct MSXE17xx_IOWatchdogGetStatusAndValueResponse *Response)`

Get watchdog current status and value information.

2.22.1 Function Documentation

2.22.1.1 `int MSXE17xx_IOWatchdogInitAndStart (xsd__unsignedLong ulTimeBase, xsd__unsignedLong ulTimeValue, xsd__unsignedLong ulOption1, xsd__unsignedLong ulOption2, struct MSXE17xx__Response * Response)`

Parameters

[in] *ulTimeBase* : Time base of the watchdog delay (0 for mus, 1 for ms, 2 for s)

[in] *ulTimeValue* : Time base of the watchdog delay (0 to 0xFFFF)

[in] *ulOption1* : Reserved
 [in] *ulOption2* : Reserved
 [out] *Response* :
 iReturnValue :
 • 0: remote function performed OK
 • -1: an system error occurred
 • -2: time base selection error
 • -3: time value selection error
 • -100: Init and start digital output watchdog kernel function error
 syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.22.1.2 `int MSXE17xx_IOWatchdogStopAndRelease (xsd__unsignedLong ulOption, struct MSXE17xx__Response * Response)`

Parameters

[in] *ulOption* : reserved
 [out] *Response* :
 iReturnValue :
 • 0: remote function performed OK
 • -1: an system error occurred
 • -100: Stop and release digital output watchdog kernel function error
 syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.22.1.3 `int MSXE17xx_IOWatchdogGetStatusAndValue (xsd__unsignedLong ulOption, struct MSXE17xx__IOWatchdogGetStatusAndValueResponse * Response)`

Parameters

[in] *ulOption* : Reserved
 [out] *Response* :
 iReturnValue :
 • 0: remote function performed OK
 • -1: an system error occurred
 • -2: channel selection error
 • -100: Get diagnostic information kernel function error
 ulStatus : current status information

- BIN XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX0: is stopped,
 - BIN XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX1: is running,
 - BIN XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX0X: is not run down
 - BIN XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX1X: is run down
- ulValue* : current value information (0 to 0xFFFF)
ulInfo : reserved
syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.23 MSX-E17xx multifunction common functions

Functions

- int [MSXE17xx_MFCommonGetSubModuleFunctionality](#) (xsd__unsignedLong ulMFModuleIndex, struct [MSXE17xx__unsignedLongResponse](#) *Response)
Get the selected sub module functionality.
- int [MSXE17xx_MFCommonSetInputsFilter](#) (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulInputAFilterValue, xsd__unsignedLong ulInputBFilterValue, xsd__unsignedLong ulInputCFilterValue, xsd__unsignedLong ulInputDFilterValue, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct [MSXE17xx__Response](#) *Response)
Set a filter to the input of a multifunction sub module.
- int [MSXE17xx_MFCommonReferenceVoltageActivation](#) (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulActivationFlag, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct [MSXE17xx__Response](#) *Response)
Permit to activate the reference voltage to pin D-.
- int [MSXE17xx_MFCommonSetFIFO0Level](#) (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulFIFOLevel, xsd__unsignedLong ulTimeOutTimeBase, xsd__unsignedLong ulReloadValue, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct [MSXE17xx__Response](#) *Response)
Define the number of data bloc in the first FIFO before transmit the datas.

2.23.1 Function Documentation

2.23.1.1 int MSXE17xx_MFCommonGetSubModuleFunctionality (xsd__unsignedLong ulMFModuleIndex, struct MSXE17xx__unsignedLongResponse * Response)

Parameters

- [in] *ulMFModuleIndex* : index of the multifunction sub module (0 to 3).
 [out] *Response* :
ulValue :

- 0: Incremental counter
- -1: PWM

sResponse.iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred (check errno in this case)
- -2: Multifunction sub module index selection error

sResponse.syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.23.1.2 `int MSXE17xx_MFCommonSetInputsFilter (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulInputAFilterValue, xsd__unsignedLong ulInputBFilterValue, xsd__unsignedLong ulInputCFilterValue, xsd__unsignedLong ulInputDFilterValue, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_Response * Response)`

Parameters

[in] ***ulMFModuleIndex*** : index of the multifunction sub module (0 to 3).

[in] ***ulInputAFilterValue*** : Filter value for input A (0 to 262143)

- 0: Filter nicht benutzt
- 1: 100 ns
- 2: 200 ns
- 3: 300 ns ...
- 262143 : 26,2143 ms

[in] ***ulInputBFilterValue*** : Filter value for input B (0 to 262143)

- 0: Filter nicht benutzt
- 1: 100 ns
- 2: 200 ns
- 3: 300 ns ...
- 262143 : 26,2143 ms

[in] ***ulInputCFilterValue*** : Filter value for input C (0 to 262143)

- 0: Filter nicht benutzt
- 1: 100 ns
- 2: 200 ns
- 3: 300 ns ...
- 262143 : 26,2143 ms

[in] ***ulInputDFilterValue*** : Filter value for input D (0 to 262143)

- 0: Filter nicht benutzt
- 1: 100 ns
- 2: 200 ns
- 3: 300 ns ...

- 262143 : 26,2143 ms

[in] ***ulOption01*** : Set it to 0

[in] ***ulOption02*** : Set it to 0

[in] ***ulOption03*** : Set it to 0

[in] ***ulOption04*** : Set it to 0

[out] ***Response*** :

iReturnValue :

- 0 : means the remote function performed OK
- -1: means an system error occured
- -2: Multifunction sub module index selection error
- -3: Input A filter value selection error
- -4: Input B filter value selection error
- -5: Input C filter value selection error
- -6: Input D filter value selection error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.23.1.3 `int MSXE17xx_MFCommonReferenceVoltageActivation (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulActivationFlag, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MSXE17xx_Response * Response)`

Parameters

[in] ***ulMFModuleIndex*** : index of the multifunction sub module (0 to 3).

[in] ***ulActivationFlag*** :

- 0: normal mode from D- (Default mode)
- 1: activate the reference voltage to pin D-

[in] ***ulOption01*** : Set it to 0

[in] ***ulOption02*** : Set it to 0

[out] ***Response*** :

iReturnValue :

- 0 : means the remote function performed OK
- -1: means an system error occured
- -2: Multifunction sub module index selection error
- -3: Activation flag selection error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.23.1.4 `int MSXE17xx__MFCommonSetFIFO0Level (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulFIFOLevel, xsd__unsignedLong ulTimeOutTimeBase, xsd__unsignedLong ulReloadValue, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MSXE17xx__Response * Response)`

Parameters

- [in] ***ulMFModuleIndex*** : index of the multifunction sub module (0 to 3).
- [in] ***ulFIFOLevel*** : Define the FIFO level (1 to 200).
- [in] ***ulTimeOutTimeBase*** : Define a Time out : permit to receive the data from the FIFO before the FIFO level is reached.
Time base of the timer (0: disabled, 1 for us, 2 for ms, 3 for s)
- [in] ***ulReloadValue*** : Time out reload value (1 to 0xFFFF)
- [in] ***ulOption01*** : reserved (Set it to 0).
- [in] ***ulOption02*** : reserved (Set it to 0).
- [out] ***Response*** :
iReturnValue :
 - 0: means the remote function performed OK
 - -1: means an system error occurred
 - -2: Multifunction sub module index selection error
 - -3: FIFO level value is wrong
 - -4: Time out time base selection error
 - -5: Time out value can not be null, if a time base is selected
 - -100: Set FIFO level kernel function error
- syserrno*** : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

2.24 MSX-E17xx ENDAT functions

Data Structures

- struct [MSXE17xx__MFEndatGetPositionResponse](#)
- struct [MSXE17xx__MFEndatGetSensorPropertiesResponse](#)
- struct [MSXE17xx__MFEndatGetErrorSourcesResponse](#)
- struct [MSXE17xx__MFEndatSensorSendParameterResponse](#)
- struct [MSXE17xx__MFEndatSensorSendPosAndRecvSelMemAreaResponse](#)
- struct [MSXE17xx__MFEndatGetPositionWithAddDataResponse](#)
- struct [MSXE17xx__MFEndatGetSelectedAdditionalDataResponse](#)
- struct [MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse](#)

Functions

- `int MSXE17xx_MFEndatInitSensor (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulFrequency, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_Response *Response)`

Initialises an EnDat sensor.

- `int MSXE17xx_MFEndatGetPosition (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_MFEndatGetPositionResponse *Response)`

Reads the position of the sensor.

- `int MSXE17xx_MFEndatGetSensorProperties (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_MFEndatGetSensorPropertiesResponse *Response)`

Reads the properties of the sensor.

- `int MSXE17xx_MFEndatGetErrorSources (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_MFEndatGetErrorSourcesResponse *Response)`

Reads the error sources.

- `int MSXE17xx_MFEndatResetErrorBits (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_Response *Response)`

Resets the error bits.

- `int MSXE17xx_MFEndatSensorReceiveReset (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_Response *Response)`

Resets the sensor.

- `int MSXE17xx_MFEndatSelectMemoryArea (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_Response *Response)`

Selects a memory area (see page 31/121 of EnDat specifications).

- `int MSXE17xx_MFEndatSensorSendParameter (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_MFEndatSensorSendParameterResponse *Response)`

Reads a parameter from the memory area that was last selected.

- `int MSXE17xx_MFEndatSensorReceiveParameter (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulParam, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_Response *Response)`

Writes a parameter to the memory area that was last selected.

- `int MSXE17xx_MFEndatSensorSendPosAndRecvSelMemArea (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_MFEndatSensorSendPosAndRecvSelMemAreaResponse *Response)`

Reads the position value of the sensor and selects a memory area in the same cycle.

- `int MSXE17xx_MFEndatSelectAdditionalData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulMFChannelIndex, xsd__unsignedLong ulAddDataCount, xsd__unsignedLong ulMrsCodeAD1, xsd__unsignedLong ulMrsCodeAD2, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_Response *Response)`

Selects the additional data that will be sent by the sensor.

- `int MSXE17xx_MFEndatGetPositionWithAddData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_MFEndatGetPositionWithAddDataResponse *Response)`

Reads the position of the sensor with additional data.

- `int MSXE17xx_MFEndatGetSelectedAdditionalData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_MFEndatGetSelectedAdditionalDataResponse *Response)`

Reads the currently selected additional data of the sensor.

- `int MSXE17xx_MFEndatInitAndEnableLatchPositionValues (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulLatchSource, xsd__unsignedLong ulTriggerEdgeCount, xsd__unsignedLong ulDataFormat, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_Response *Response)`

Initialises and enables the latch logic to get the position and additional informations from the EnDat sensor using a trigger source.

- `int MSXE17xx_MFEndatGetCurrentLatchConfiguration (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_MFEndatGetCurrentLatchConfigurationResponse *Response)`

Reads the current latch configuration, started using the function MSXE17xx_MFEndatInitAndEnableLatchPositionValues.

- `int MSXE17xx_MFEndatDisableAndReleaseLatchPositionValues (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_Response *Response)`

Disables and releases the latch logic started with the function MSXE17xx__MFEndatInitAndEnableLatchPositionValues.

2.24.1 Function Documentation

2.24.1.1 `int MSXE17xx__MFEndatInitSensor (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulFrequency, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__Response * Response)`

This function should be called once, in order to call the other EnDat functions.

Parameters

- [in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] **ulChannelIndex** Index of the channel. Set to 0
- [in] **ulFrequency** Frequency to use in kHz (500, 900, 1500, 2500, 4500)
- [in] **ulOption01** Reserved. Set to 0
- [in] **ulOption02** Reserved. Set to 0
- [in] **ulOption03** Reserved. Set to 0
- [in] **ulOption04** Reserved. Set to 0
- [out] **Response iReturnValue**
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The ulConnectorIndex parameter is wrong
 - **-4** The ulChannelIndex parameter is wrong
 - **-5** The driver is in a wrong state (must be INITIALISED or UNINITIALISED)
 - **-6** The component is not programmed as EnDat
 - **-7** Error while resetting sensor. Note: If no sensor is plugged on the selected channel, you will get this error.
 - **-8** Error while selecting memory area 0xB9
 - **-9** Error while reading alarm space (address 0x0)
 - **-10** Error while reading warning space (address 0x1)
 - **-11** Error while clearing errors (write 0 at address 0x0)
 - **-12** Error while clearing warnings (write 0 at address 0x1)
 - **-13** Error while selecting memory area 0xA1
 - **-14** Error while reading number of clock pulses for transfer of position value (address 0x0D)
 - **-15** Error while selecting memory area 0xA5
 - **-16** Error while reading EnDat command set (address 0x5)
 - **-17** Error while reading support of error messages 1 (address 0x3)
 - **-18** Error while reading support of warnings (address 0x4)
 - **-19** Error while reading EnDat ordering designation (address 0x8)
 - **-20** ulFrequency parameter is too high for current sensor
 - **-21** The ulFrequency parameter is wrong
 - **-22** EnDat ordering designation is invalid

- **-41** Transmission error. Please call `MSXE17xx__MFEndatGetErrorSources` to get more information
 - **-100** Internal system error occurred. See value of `syserrno`
- syserrno* system error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

2.24.1.2 `int MSXE17xx__MFEndatGetPosition (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE17xx__MFEndatGetPositionResponse * Response)`

Before calling this function, you must call the `MSXE17xx__MFEndatInitSensor` function.

Parameters

[in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] *ulChannelIndex* Index of the channel. Set to 0

[in] *ulOption01* Reserved. Set to 0

[in] *ulOption02* Reserved. Set to 0

[in] *ulOption03* Reserved. Set to 0

[in] *ulOption04* Reserved. Set to 0

[out] *Response sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The *ulConnectorIndex* parameter is wrong
- **-4** The *ulChannelIndex* parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** Error while reading the position
- **-41** Transmission error. Please call `MSXE17xx__MFEndatGetErrorSources` to get more information
- **-100** Internal system error occurred. See value of `syserrno`

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulPositionLow Position - low bits

ulPositionHigh Position - high bits

Return values

0 SOAP_OK

Others See SOAP error

2.24.1.3 `int MSXE17xx_MFEndatGetSensorProperties (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE17xx_MFEndatGetSensorPropertiesResponse * Response)`

Before calling this function, you must call the MSXE17xx_MFEndatInitSensor function.

Parameters

[in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] **ulChannelIndex** Index of the channel. Set to 0

[in] **ulOption01** Reserved. Set to 0

[in] **ulOption02** Reserved. Set to 0

[in] **ulOption03** Reserved. Set to 0

[in] **ulOption04** Reserved. Set to 0

[out] **Response** *sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The ulConnectorIndex parameter is wrong
- **-4** The ulChannelIndex parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** Error while selecting memory area 0xA3
- **-8** Error while reading ID number (address 0x8)
- **-9** Error while reading ID number (address 0x9)
- **-10** Error while reading ID number (address 0xA)
- **-11** Error while reading Serialnumber (address 0xB)
- **-12** Error while reading Serialnumber (address 0xC)
- **-13** Error while reading Serialnumber (address 0xD)
- **-14** Error while selecting memory area 0xA1
- **-15** Error while reading encoder model (address 0xE)
- **-16** Error while reading signal period length or signal periods per revolution for incremental output signals (address 0xF)
- **-17** Error while getting the position
- **-18** Error while selecting memory area 0xA3
- **-19** Error while reading signal period length or signal periods per revolution for incremental output signals (address 0x0)
- **-20** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x4)
- **-21** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x5)
- **-22** Error while selecting memory area 0xBD
- **-23** Error while reading scaling factor for resolution (address 0x1B)
- **-24** Error while reading measuring step, or measuring steps per revolution or subdivision values of a grating period (address 0x1C)
- **-25** Error while reading measuring step, or measuring steps per revolution or subdivision values of a grating period (address 0x1D)

- **-26** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x4)
- **-27** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x5)
- **-28** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x4)
- **-29** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x5)
- **-30** Error while reading distinguishable revolutions (address 0x1)
- **-31** Error while selecting memory area 0xBD
- **-32** Error while reading number of distinguishable revolutions with scaling factor (address 0x22)
- **-33** Error while selecting memory area 0xBD
- **-34** Error while reading status of additional datum 1 (address 0x0)
- **-35** Error while reading status of additional datum 2 (address 0x1)
- **-41** Transmission error. Please call `MSXE17xx__MFEndatGetErrorSources` to get more information
- **-100** Internal system error occurred. See value of `syserrno`

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulIDNumberLsb ID Number - low bits (see page 67/121 of EnDat specifications)

ulIDNumberMsb ID Number - high bits (see page 67/121 of EnDat specifications)

ulSerialNumberLsb Serialnumber - low bits (see page 68/121 of EnDat specifications)

ulSerialNumberMsb Serialnumber - high bits (see page 68/121 of EnDat specifications)

ulModel Model/Type of the sensor (see page 79/84 of EnDat application notes 722024)

- **0, 1, 2, 3** Incremental linear encoder
- **4, 6** Absolute linear encoder
- **8, 9, 10, 11** Incremental rotary encoder or angle encoder
- **12** Singleturn encoder
- **13, 14** Multiturn encoder

ulMode Support of EnDat 2.2

- **0** Sensor does not support EnDat 2.2. commands
- **1** Sensor supports EnDat 2.2 commands

ulPositionSize Size of the position value send by the sensor (in bits)

ulSignalPeriod Signal period length or signal periods per revolution for incremental output signals (see page 79/84 of EnDat application notes 722024)

ulStepPerRevolution Measuring step length or measuring steps per revolution with serial data transfer (see page 79/84 of EnDat application notes 722024)

ulNumberOfRevolution Distinguishable revolutions - only for multiturn encoders (see page 79/84 of EnDat application notes 722024)

ulScalingFactor Scaling factor for resolution (see page 79/84 of EnDat application notes 722024)

ulAdditionalData Supported additional data (see page 85/121 of EnDat specifications)

- **Bit 0** "Position value 2" is available (MRS-Code: 0x42, 0x43, 0x44)
- **Bit 1** "Test values" is available (MRS-Code: 0x49, 0x4A, 0x4B)
- **Bit 2** "Temperature sensor 1 (external)" is available (MRS-Code: 0x4C)
- **Bit 3** "Temperature sensor 2 (external)" is available (MRS-Code: 0x4D)
- **Bit 4** "Additional sensors" is available (MRS-Code: 0x4E)
- **Bit 16** "Commutation" is available (MRS-Code: 0x51)

- **Bit 17** "Acceleration" is available (MRS-Code: 0x52)
- **Bit 18** "Limit position signals" is available (MRS-Code: 0x54)
- **Bit 19** "Asynchronous position value" is available (MRS-Code: 0x56, 0x57, 0x58)
- **Bit 20** "Operating status error sources" is available (MRS-Code: 0x59)
- **Bit 23** "Timestamp" is available (MRS-Code: 0x5B)

Return values

0 SOAP_OK

Others See SOAP error

2.24.1.4 `int MSXE17xx_MFEndatGetErrorSources (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE17xx_MFEndatGetErrorSourcesResponse * Response)`

The error are reseted after a call to MSXE17xx_MFEndatResetErrorBits. If a function returns an error, please use this function to check if it is not a communication error.

Parameters

[in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] **ulChannelIndex** Index of the channel. Set to 0

[in] **ulOption01** Reserved. Set to 0

[in] **ulOption02** Reserved. Set to 0

[in] **ulOption03** Reserved. Set to 0

[in] **ulOption04** Reserved. Set to 0

[out] **Response** *sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The ulConnectorIndex parameter is wrong
- **-4** The ulChannelIndex parameter is wrong
- **-5** The component is not programmed as EnDat
- **-100** Internal system error occurred. See value of syserrno

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulErrorSrc Mask of bits that give the current error sources. Each bit represents an error. If the bit is set to 1, the error is present.

- **Bit 0** Invalid mode command
- **Bit 1** Invalid MRS-Code
- **Bit 2** Transmission is not completed
- **Bit 3** Communication command is not supported
- **Bit 4** MRS-Code is not allowed
- **Bit 6** Invalid address is selected or sensor's EEPROM is written while being busy
- **Bit 7** Try to write a protected memory place
- **Bit 8** Write-Protect configuration is tried to be reset (if a memory place is write-protected, it cannot be reset)

- **Bit 9** Block address is not available
- **Bit 10** Invalid address for the communication command
- **Bit 11** Invalid additional data (or additional data not supported by the sensor)

Return values

0 SOAP_OK

Others See SOAP error

2.24.1.5 `int MSXE17xx__MFEndatResetErrorBits (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__Response * Response)`

It can be used before each command in order to get (after the call of the command) the status of the system using MSXE17xx__MFEndatGetErrorSources.

Once an error is detected using MSXE17xx__MFEndatGetErrorSources, you have to clear it using this function.

Parameters

[in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] **ulChannelIndex** Index of the channel. Set to 0

[in] **ulOption01** Reserved. Set to 0

[in] **ulOption02** Reserved. Set to 0

[in] **ulOption03** Reserved. Set to 0

[in] **ulOption04** Reserved. Set to 0

[out] **Response iReturnValue**

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The ulConnectorIndex parameter is wrong
- **-4** The ulChannelIndex parameter is wrong
- **-5** The component is not programmed as EnDat
- **-100** Internal system error occurred. See value of syserrno

syserrno system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

2.24.1.6 `int MSXE17xx__MFEndatSensorReceiveReset (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__Response * Response)`

This function has the same effect as an hardware reboot of the sensor.

Parameters

- [in] ***ulConnectorIndex*** Index of the EnDat connector (0 to 3). See on the MSX-E system.
 - [in] ***ulChannelIndex*** Index of the channel. Set to 0
 - [in] ***ulOption01*** Reserved. Set to 0
 - [in] ***ulOption02*** Reserved. Set to 0
 - [in] ***ulOption03*** Reserved. Set to 0
 - [in] ***ulOption04*** Reserved. Set to 0
 - [out] ***Response iReturnValue***
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The *ulConnectorIndex* parameter is wrong
 - **-4** The *ulChannelIndex* parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be INITIALISED or UNINITIALISED or ERROR)
 - **-7** Error while resetting sensor
 - **-41** Transmission error. Please call `MSXE17xx__MFEndatGetErrorSources` to get more information
 - **-100** Internal system error occurred. See value of `syserrno`
- syserrno* system-error code (the value of the libc "errno" code)

Return values

- 0** SOAP_OK
- Others** See SOAP error

2.24.1.7 `int MSXE17xx__MFEndatSelectMemoryArea (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__Response * Response)`

In order to send or read parameters, the memory area must first be selected.

Before calling this function, you must call the `MSXE17xx__MFEndatInitSensor` function.

Parameters

- [in] ***ulConnectorIndex*** Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] ***ulChannelIndex*** Index of the channel. Set to 0
- [in] ***ulMrsCode*** The MRS-code corresponding to the memory area that you want to select (see page 31/121 and 84/121 of EnDat specifications)
 - **0xB9** Operating status (address area: 0x0 - 0x3)
 - **0xA1** Parameters of the encoder manufacturer - first part (address area: 0x4 - 0xF)
 - **0xA3** Parameters of the encoder manufacturer - second part (address area: 0x0 - 0xF)
 - **0xA5** Parameters of the encoder manufacturer - third part (address area: 0x0 - 0xF)
 - **0xA7** Operating parameters (address area: 0x0 - 0xF)

- **0xA9** Parameters of the OEM - first part (address area: depending on the sensor)
- **0xAB** Parameters of the OEM - second part (address area: depending on the sensor)
- **0xAD** Parameters of the OEM - third part (address area: depending on the sensor)
- **0xAF** Parameters of the OEM - fourth part (address area: depending on the sensor)
- **0xB1** Compensation values of the encoder manufacturer - first part (address area: depending on the sensor)
- **0xB3** Compensation values of the encoder manufacturer - second part (address area: depending on the sensor)
- **0xB5** Compensation values of the encoder manufacturer - third part (address area: depending on the sensor)
- **0xB7** Compensation values of the encoder manufacturer - fourth part (address area: depending on the sensor)

[in] *ulOption01* Reserved. Set to 0

[in] *ulOption02* Reserved. Set to 0

[in] *ulOption03* Reserved. Set to 0

[in] *ulOption04* Reserved. Set to 0

[out] *Response iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The *ulConnectorIndex* parameter is wrong
- **-4** The *ulChannelIndex* parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** The *ulMrsCode* parameter is wrong
- **-8** Error while selecting the memory area
- **-41** Transmission error. Please call *MSXE17xx__MFEndatGetErrorSources* to get more information
- **-100** Internal system error occurred. See value of *syserrno*

syserrno system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

2.24.1.8 `int MSXE17xx__MFEndatSensorSendParameter (xsd__unsignedLong
ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong
ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulOption01,
xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong
ulOption04, struct MSXE17xx__MFEndatSensorSendParameterResponse * Response)`

Before calling this function, you must call the *MSXE17xx__MFEndatInitSensor* function to initialise the sensor, and then *MSXE17xx__MFEndatSelectMemoryArea*, or *MSXE17xx__MFEndatSensorSendPosAndRecvSelMemArea* to select the memory area that contains the parameter you want to read.

Parameters

- [in] ***ulConnectorIndex*** Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] ***ulChannelIndex*** Index of the channel. Set to 0
- [in] ***ulMrsCode*** The MRS-code corresponding to the last memory area that you have selected (see MSXE17xx__MFEndatSelectMemoryArea or MSXE17xx__MFEndatSensorSendPosAndRecvSelMemArea)
- [in] ***ulAddress*** The address of the parameter that you want to read (0x0-0xFF)
- [in] ***ulOption01*** Reserved. Set to 0
- [in] ***ulOption02*** Reserved. Set to 0
- [in] ***ulOption03*** Reserved. Set to 0
- [in] ***ulOption04*** Reserved. Set to 0
- [out] ***Response*** ***sResponse.iReturnValue***
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The *ulConnectorIndex* parameter is wrong
 - **-4** The *ulChannelIndex* parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be INITIALISED)
 - **-7** The *ulMrsCode* parameter is wrong
 - **-8** The *ulAddress* parameter is wrong
 - **-9** Your sensor is not compatible with EnDat 2.2, but the parameter *ulMrsCode* is only available for EnDat 2.2
 - **-10** The last selected memory area do not corresponds to the parameter *ulMrsCode*. Please call MSXE17xx__MFEndatSelectMemoryArea or MSXE17xx__MFEndatSensorSendPosAndRecvSelMemArea with the wished *ulMrsCode* before
 - **-11** Error while reading parameter
 - **-41** Transmission error. Please call MSXE17xx__MFEndatGetErrorSources to get more information
 - **-100** Internal system error occurred. See value of *syserrno*
- sResponse.syserrno*** system-error code (the value of the libc "errno" code)
- ulParam*** Value of the parameter

Return values

- 0** SOAP_OK
- Others** See SOAP error

2.24.1.9 `int MSXE17xx__MFEndatSensorReceiveParameter (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulParam, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__Response * Response)`

Before calling this function, you must call the MSXE17xx__MFEndatInitSensor function to initialise the sensor, and then MSXE17xx__MFEndatSelectMemoryArea, or MSXE17xx__MFEndatSensorSendPosAndRecvSelMemArea to select the memory area that contains the parameter you want to write.

Parameters

- [in] ***ulConnectorIndex*** Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] ***ulChannelIndex*** Index of the channel. Set to 0
- [in] ***ulMrsCode*** The MRS-code corresponding to the last memory area that you have selected (see MSXE17xx__MFEndatSelectMemoryArea or MSXE17xx__MFEndatSensorSendPosAndRecvSelMemArea)
- [in] ***ulAddress*** The address of the parameter that you want to write (0x0-0xFF)
- [in] ***ulParam*** The new value of the parameter
- [in] ***ulOption01*** Reserved. Set to 0
- [in] ***ulOption02*** Reserved. Set to 0
- [in] ***ulOption03*** Reserved. Set to 0
- [in] ***ulOption04*** Reserved. Set to 0
- [out] ***Response iReturnValue***
- **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The *ulConnectorIndex* parameter is wrong
 - **-4** The *ulChannelIndex* parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be INITIALISED)
 - **-7** The *ulMrsCode* parameter is wrong
 - **-8** The *ulAddress* parameter is wrong
 - **-9** The last selected memory area do not corresponds to the parameter *ulMrsCode*. Please call MSXE17xx__MFEndatSelectMemoryArea or MSXE17xx__MFEndatSensorSendPosAndRecvSelMemArea with the wished *ulMrsCode* before
 - **-10** Error while writing parameter
 - **-41** Transmission error. Please call MSXE17xx__MFEndatGetErrorSources to get more information
 - **-100** Internal system error occurred. See value of *syserrno*
- syserrno* system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

2.24.1.10 `int MSXE17xx__MFEndatSensorSendPosAndRecvSelMemArea (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__MFEndatSensorSendPosAndRecvSelMemAreaResponse * Response)`

In order to send or read parameters, the memory area must first be selected.

Before calling this function, you must call the MSXE17xx__MFEndatInitSensor function.

This function differs from the MSXE17xx_MFEndatSelectMemoryArea function, since it can select memory areas that are reserved for EnDat 2.2. sensors.

This function is reserved for EnDat 2.2 sensors. It will returns an error if the sensor does not support EnDat 2.2 commands.

Parameters

- [in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] **ulChannelIndex** Index of the channel. Set to 0
- [in] **ulMrsCode** The MRS-code corresponding to the memory area that you want to select (see page 31/121 and 84/121 of EnDat specifications)
 - **0xB9** Operating status (address area: 0x0 - 0x3)
 - **0xA1** Parameters of the encoder manufacturer - first part (address area: 0x4 - 0xF)
 - **0xA3** Parameters of the encoder manufacturer - second part (address area: 0x0 - 0xF)
 - **0xA5** Parameters of the encoder manufacturer - third part (address area: 0x0 - 0xF)
 - **0xA7** Operating parameters (address area: 0x0 - 0xF)
 - **0xA9** Parameters of the OEM - first part (address area: depending on the sensor)
 - **0xAB** Parameters of the OEM - second part (address area: depending on the sensor)
 - **0xAD** Parameters of the OEM - third part (address area: depending on the sensor)
 - **0xAF** Parameters of the OEM - fourth part (address area: depending on the sensor)
 - **0xB1** Compensation values of the encoder manufacturer - first part (address area: depending on the sensor)
 - **0xB3** Compensation values of the encoder manufacturer - second part (address area: depending on the sensor)
 - **0xB5** Compensation values of the encoder manufacturer - third part (address area: depending on the sensor)
 - **0xB7** Compensation values of the encoder manufacturer - fourth part (address area: depending on the sensor)
 - **0xBD** Parameters of the encoder manufacturer for EnDat 2.2 (address area: 0x0 - 0x3F)
 - **0xBB** Operating parameters 2 (address area: depending on the sensor)
 - **0xBF** Parameters of the section 2 memory area (address area: depending on the sensor)
- [in] **ulAddress** Selected block address (only used if ulMrsCode is set to 0xBF. If ulMrsCode is not 0xBF, set it to 0). It is used to address further section 2 memory areas (see page 46/121 of EnDat specifications)
- [in] **ulOption01** Reserved. Set to 0
- [in] **ulOption02** Reserved. Set to 0
- [in] **ulOption03** Reserved. Set to 0
- [in] **ulOption04** Reserved. Set to 0
- [out] **Response sResponse.iReturnValue**
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The ulConnectorIndex parameter is wrong
 - **-4** The ulChannelIndex parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be INITIALISED)
 - **-7** The ulMrsCode parameter is wrong

- **-8** The `ulAddress` parameter is wrong
- **-9** The sensor is not compatible with EnDat 2.2
- **-10** Error while getting position and selecting memory area
- **-41** Transmission error. Please call `MSXE17xx__MFEndatGetErrorSources` to get more information
- **-100** Internal system error occurred. See value of `syserrno`

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulPositionLow Position - low bits

ulPositionHigh Position - high bits

Return values

0 SOAP_OK

Others See SOAP error

2.24.1.11 `int MSXE17xx__MFEndatSelectAdditionalData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulMFChannelIndex, xsd__unsignedLong ulAddDataCount, xsd__unsignedLong ulMrsCodeAD1, xsd__unsignedLong ulMrsCodeAD2, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__Response * Response)`

Some additional data are not available on all sensors. To get the available additional data of your sensor, please use the function `MSXE17xx__MFEndatGetSensorProperties`.

If you select an additional data that is not available on your sensor, you will get the parameter `ucErrorSrc13` set to 1 when calling the function `MSXE17xx__MFEndatGetErrorSources`.

The additional data are extra values that the sensor can send (in the same cycle as its position value).

This function is reserved for EnDat 2.2 sensors. It will returns an error if the sensor does not support EnDat 2.2 commands.

Parameters

[in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] *ulChannelIndex* Index of the channel. Set to 0

[in] *ulAddDataCount* The number of selected additional data (0 to 2)

[in] *ulMrsCodeAD1* The MRS-Code for the first additional data

- **0x40** Send additional info 1 without data contents
- **0x42** Position value 2 word 1 LSB
- **0x43** Position value 2 word 2
- **0x44** Position value 2 word 3 MSB
- **0x49** Test values word 1 LSB
- **0x4A** Test values word 2
- **0x4B** Test values word 3 MSB
- **0x4C** Temperature sensor 1 (external)
- **0x4D** Temperature sensor 2 (external)
- **0x4E** Additional sensors

[in] *ulMrsCodeAD2* The MRS-Code for the second additional data

- **0x50** Send additional datum 2 without data contents
- **0x51** Commutation
- **0x52** Acceleration
- **0x54** Limit position signals
- **0x56** Asynchronous position value word 1 LSB
- **0x57** Asynchronous position value word 2
- **0x58** Asynchronous position value word 3 MSB
- **0x59** Operating status error sources
- **0x5B** Timestamp

[in] *ulOption01* Reserved. Set to 0

[in] *ulOption02* Reserved. Set to 0

[in] *ulOption03* Reserved. Set to 0

[in] *ulOption04* Reserved. Set to 0

[out] *Response iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The *ulConnectorIndex* parameter is wrong
- **-4** The *ulChannelIndex* parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** The *ulAddDataCount* parameter is wrong
- **-8** The *ulMrsCodeAD1* parameter is wrong
- **-9** The *ulMrsCodeAD2* parameter is wrong
- **-10** The sensor is not compatible with EnDat 2.2
- **-11** Error while deactivating additional data 2
- **-12** Error while deactivating additional data 1
- **-13** Error while activating additional data 1
- **-14** Error while deactivating additional data 2
- **-15** Error while deactivating additional data 1
- **-16** Selected additional data 1 is wrong or not available on this sensor. Please call *MSXE17xx__MFEndatGetErrorSources* to get more information
- **-17** Error while activating additional data 1. Please call *MSXE17xx__MFEndatGetErrorSources* to get more information
- **-18** Error while getting the current position value
- **-19** Error while activating additional data 2
- **-20** Error while deactivating additional data 2
- **-21** Error while deactivating additional data 1
- **-22** Selected additional data 2 is wrong or not available on this sensor. Please call *MSXE17xx__MFEndatGetErrorSources* to get more information
- **-23** Error while activating additional data 2. Please call *MSXE17xx__MFEndatGetErrorSources* to get more information
- **-24** Error while getting the current position value
- **-41** Transmission error. Please call *MSXE17xx__MFEndatGetErrorSources* to get more information
- **-100** Internal system error occurred. See value of *syserrno*

syserrno system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

2.24.1.12 `int MSXE17xx_MFEndatGetPositionWithAddData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_MFEndatGetPositionWithAddDataResponse * Response)`

Before calling this function, you must call the MSXE17xx_MFEndatInitSensor function.

You must also call the function MSXE17xx_MFEndatSelectAdditionalData to select the additional data that you want to receive.

This function is reserved for EnDat 2.2 sensors. It will returns an error if the sensor does not support EnDat 2.2 commands.

Parameters

[in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] *ulChannelIndex* Index of the channel. Set to 0

[in] *ulOption01* Reserved. Set to 0

[in] *ulOption02* Reserved. Set to 0

[in] *ulOption03* Reserved. Set to 0

[in] *ulOption04* Reserved. Set to 0

[out] *Response sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The *ulConnectorIndex* parameter is wrong
- **-4** The *ulChannelIndex* parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** Error while reading the position
- **-41** Transmission error. Please call MSXE17xx_MFEndatGetErrorSources to get more information
- **-100** Internal system error occurred. See value of *syserrno*

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulPositionLow Position - low bits

ulPositionHigh Position - high bits

ulAddData1 Value of the additional data 1

ulAddData2 Value of the additional data 2

Return values

0 SOAP_OK

Others See SOAP error

2.24.1.13 `int MSXE17xx_MFEndatGetSelectedAdditionalData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_MFEndatGetSelectedAdditionalDataResponse * Response)`

The additional data are selected using the function MSXE17xx_MFEndatSelectAdditionalData.

Parameters

- [in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] *ulChannelIndex* Index of the channel. Set to 0
- [in] *ulOption01* Reserved. Set to 0
- [in] *ulOption02* Reserved. Set to 0
- [in] *ulOption03* Reserved. Set to 0
- [in] *ulOption04* Reserved. Set to 0
- [out] *Response sResponse.iReturnValue*
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The *ulConnectorIndex* parameter is wrong
 - **-4** The *ulChannelIndex* parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-100** Internal system error occurred. See value of *syserrno*
- sResponse.syserrno* system-error code (the value of the libc "errno" code)
- ulAdCount* Number of selected additional data
- ulMrsCodeAD1* MRS code of the additional data 1
- ulMrsCodeAD2* MRS code of the additional data 2

Return values

- 0** SOAP_OK
- Others* See SOAP error

2.24.1.14 `int MSXE17xx_MFEndatInitAndEnableLatchPositionValues (xsd__unsignedLong ulmFModuleIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulLatchSource, xsd__unsignedLong ulTriggerEdgeCount, xsd__unsignedLong ulDataFormat, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_Response * Response)`

Before calling this function, you must call the MSXE17xx_MFEndatInitSensor function.

When the selected trigger occurs, the position value of the selected EnDat channel will be measured and send through the dataserver.

If you chose additional data using the function MSXE17xx_MFEndatSelectAdditionalData and if your selected *ulDataFormat* enables it, you will also receive the value of the selected additional data.

Note: using a synchro timer (and activating the synchro trigger as latch source) enables to get position of the EnDat sensor at a defined rate.

Parameters

[in] ***ulConnectorIndex*** Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] ***ulChannelIndex*** Index of the channel. Set to 0

[in] ***ulLatchSource*** Mask of bits, that defines the trigger source.

Bit 0: Hardware trigger If you select the hardware trigger, you must also choose the edge detection (Bit 2 and 3). Of course, you can choose both.

- **1** activated
- **0** not used

Bit 1 : Synchro trigger

- **1** activated
- **0** not used

Bit 2 : Trigger rising edge (only if hardware trigger is selected)

- **1** activated
- **0** not used

Bit 3 : Trigger falling edge (only if hardware trigger is selected)

- **1** activated
- **0** not used

```
ulLatchSource = 2 (0b10)      -> the latch source is the synchro trigger
ulLatchSource = 5 (0b101)    -> the latch source is a rising edge of the hardware trigger
ulLatchSource = 13 (0b1101) -> the latch source is a rising or a falling edge of the hardware trigger
```

[in] ***ulTriggerEdgeCount*** Number of edges required to detect an hardware trigger (only if hardware trigger is selected)

[in] ***ulDataFormat*** Mask of bits, that defines the format of data that will be sent by the data server. It is a combination of bits. You can set all the bits to 1 if you want all the informations 0b11111 = 31.

You can also, for example, only wants the timestamp and the digital I/Os state (0b11 = 3).

You can also choose, for example, to get the additional data 2 (0b1000 = 8).

Bit 0: Timestamp

- **0** Not sent
- **1** timestamp sent by the dataserver

Bit 1: Digital I/Os state

- **0** Not sent
- **1** digital I/Os state sent by the data server

Bit 2: Additional data 1

- **0** Not sent
- **1** Additional data 1 sent by the data server

Bit 3: Additional data 2

- **0** Not sent
- **1** Additional data 2 sent by the data server

Bit 4: Format of the value

- **0** Raw value (as sent by the sensor)
- **1** Standardised value. The format of the frame will then depends on the model of the sensor. See system documentation.

If ulDataFormat is set to 0, the frame send by the dataserver will have the following definition

```

unsigned long eventsrc;           // High 16 bits (MSB): Channel concerned
    (0 to 3)

                                   // Low 16 bits (L
    SB): Bit mask representing the source of the event
                                   //      Bit 0: Ha
    rdware trigger
                                   //      Bit 1: Sy
    nchro trigger
                                   //      Bit 2: Co
    mpare

unsigned long positionlow;        // Low bits of the position
unsigned long positionhigh;      // High bits of the position
unsigned long error;             // Status of the communication. Same valu
    e as the value returned by the function MSXE17xx_MFEndatGetErrorSources

```

If you set `ulDataFormat` to 1 (send timestamp), then the frame send by the dataserver will be changed. At the end of the "basic" frame, we add the timestamp.

```
unsigned long ts;           // Second part of the timestamp
unsigned long tus;          // Microsecond part of the timestamp
```

If you set `ulDataFormat` to `2 = 0b10` (send digital I/Os state), then the frame send by the dataserver will be changed. At the end of the "basic" frame, we add the state of the digital I/Os.

```

unsigned long digiostate;      // State of the digital I/Os. Bit mask that encod
    es all digital I/Os.

                                // 0b0
    All I/Os are set to low

                                // 0b100
    I/O 2 is set to high, all others are set to 0

                                // 0b11111111111111
    1111 : All I/Os are set to high

```

If you set `ulDataFormat` to 4 = 0b100 (send additional data 1), then the frame send by the `dataserver` will be changed. At the end of the "basic" frame, we add the value of the first additional data.

```

unsigned long ad1value;
\endcode

If you set ulDataFormat to 8 = 0b1000 (send additional data 2), then the frame s
end by the dataserver will be changed. At the end of the "basic" frame, we add th
e value of the second additional data. \n

\code
unsigned long ad2value;
\endcode

If you set the bits 0, 1, 2 and 3 to 1, the format of the frame will then be: \n

\code
unsigned long eventsrc;                // High 16 bits (MSB): Channel concerned
    (0 to 3)

                                // Low 16 bits (L
SB): Bit mask representing the source of the event
                                //      Bit 0: Ha
rdware trigger
                                //      Bit 1: Sy
nchro trigger
                                //      Bit 2: Co
mpare

unsigned long positionlow;             // Low bits of the position
unsigned long positionhigh;           // High bits of the position
unsigned long error;                  // Status of the communication. Same valu
e as the value returned by the function MSXE17xx__MFEndatGetErrorSources
unsigned long ts;                     // Second part of the timestamp
unsigned long tus;                    // Microsecond part of the timestamp
unsigned long digiostate;              // State of the digital I/Os. Bit mask that encod

```

```

        es all digital I/Os.
        All I/Os are set to low
        I/O 2 is set to high, all others are set to 0
        1111 : All I/Os are set to high
unsigned long ad1value;
unsigned long ad2value;
// 0b0
// 0b100
// 0b11111111111111

```

The bit 4 of the `ulDataFormat` is more complex. The format of the frame that is sent by the `dataserver` will depend on the model of the sensor used. See the system documentation for more information, or the "Acquisition" menu of the web site.

[in] ***ulOption01*** Reserved. Set to 0

[in] ***ulOption02*** Reserved. Set to 0

[in] ***ulOption03*** Reserved. Set to 0

[in] ***ulOption04*** Reserved. Set to 0

[out] ***Response iReturnValue***

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The `ulConnectorIndex` parameter is wrong
- **-4** The `ulChannelIndex` parameter is wrong
- **-5** The component is not programmed as `EnDat`
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** The `ulLatchSource` parameter is wrong
- **-8** The `ulDataFormat` parameter is wrong
- **-9** Error while getting sensor properties
- **-10** The multiturn part size of the sensor is 0
- **-11** The singleturn part size of the sensor is 0
- **-12** The step per revolution properties of the sensor is 0
- **-41** Transmission error. Please call `MSXE17xx__MFEndatGetErrorSources` to get more information
- **-100** Internal system error occurred. See value of `syserrno`

syserrno system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

2.24.1.15 `int MSXE17xx__MFEndatGetCurrentLatchConfiguration (xsd__unsignedLong ulMModuleIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse * Response)`

Parameters

[in] ***ulConnectorIndex*** Index of the `EnDat` connector (0 to 3). See on the MSX-E system.

[in] ***ulChannelIndex*** Index of the channel. Set to 0

[in] ***ulOption01*** Reserved. Set to 0

[in] ***ulOption02*** Reserved. Set to 0

[in] ***ulOption03*** Reserved. Set to 0

[in] ***ulOption04*** Reserved. Set to 0

[out] ***Response sResponse.iReturnValue***

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The *ulConnectorIndex* parameter is wrong
- **-3** The *ulChannelIndex* parameter is wrong
- **-4** The component is not programmed as EnDat
- **-100** Internal system error occurred. See value of *syserrno*

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulRunning The running state.

- **0** Not running (do not use other return parameters, there will be set to 0)
- **1** Latch logic is running

ulLatchSource Mask of bits, that defines the trigger source. See documentation of MSXE17xx__MFEndatInitAndEnableLatchPositionValues

ulTriggerEdgeCount Number of edges required to detect an hardware trigger (only if hardware trigger is selected)

ulDataFormat Mask of bits, that defines the format of data that will be sent by the data server. See documentation of MSXE17xx__MFEndatInitAndEnableLatchPositionValues

ulModel (Used for computation) Model/Type of the sensor (see page 79/84 of EnDat application notes 722024)

- **0, 1, 2, 3** Incremental linear encoder
- **4, 6** Absolute linear encoder
- **8, 9, 10, 11** Incremental rotary encoder or angle encoder
- **12** Singleturn encoder
- **13, 14** Multiturn encoder

ulPositionSize (Used for computation) Size of the position value send by the sensor (in bits)

ulSignalPeriod (Used for computation) Signal period length or signal periods per revolution for incremental output signals

ulStepPerRevolution (Used for computation) Measuring step length or measuring steps per revolution with serial data transfer

ulNumberOfRevolution (Used for computation) Distinguishable revolutions - only for multiturn encoders

ulScalingFactor (Used for computation) Scaling factor for resolution

Return values

0 SOAP_OK

Others See SOAP error

```

2.24.1.16 int MSXE17xx_MFEndatDisableAndReleaseLatchPositionValues (
    xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex,
    xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong
    ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__Response * Response
)

```

Parameters

- [in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] **ulChannelIndex** Index of the channel. Set to 0
- [in] **ulOption01** Reserved. Set to 0
- [in] **ulOption02** Reserved. Set to 0
- [in] **ulOption03** Reserved. Set to 0
- [in] **ulOption04** Reserved. Set to 0
- [out] **Response iReturnValue**
- **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The ulConnectorIndex parameter is wrong
 - **-4** The ulChannelIndex parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be LATCH_RUNNING or LATCH_FIFO_OVERFLOW)
 - **-100** Internal system error occurred. See value of syserrno
- syserrno** system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

2.25 Software hints

Modules

- [SOAP function calls in C/C++ language](#)

Some hints on how to use the SOAP functions in a C/C++ program.

- [MSX-E systems servers](#)

The MSX-E embeds servers to provide configuration, management and monitoring over Ethernet.

2.26 SOAP function calls in C/C++ language

Some hints on how to use the SOAP functions in a C/C++ program.

2.26.1 C/C++ language SOAP function prototypes

In this documentation, functions are described as follows.

```
int MXCommon__GetModuleType(void * __, struct MXCommon__ByteArrayResponse
* Response);
```

Two main differences can be remarked in the function prototypes:

- The prefix:

```
soap_call__
```

- The first three parameters of the SOAP stub:

```
struct soap *soap
const char *soap_endpoint
const char *soap_action
```

The C function prototype has the following syntax:

```
int soap_call_MXCommon__GetModuleType(struct soap *soap, const char *soap_endpoin
t, const char *soap_action, void __, struct MXCommon__ByteArrayResponse *Response
);
```

Functions to call in C/C++ are called **stubs** and can be found in the **MSXEXXXXStub.h** file.

2.26.2 Rules and use of gSOAP calls in C/C++

When programming with gSOAP in C/C++ language, some rules have to be followed to avoid memory leaks, slow socket disconnections and multi-threading compliancy.

Note: Software code pieces in this chapter comes from [SOAP calls sample](#)

- **Opening and initializing an SOAP connection (using the gSOAP functions)**

```
struct soap *soapContext;

// IP address of the MSX-E and after the ':' is the MSX-E SOAP server port number

char soap_endpoint[] = IP_ADDRESS":5555";

// Allocates and initialize a new runtime context
if ((soapContext = soap_new ()) == NULL)
    return EXIT_FAILURE;

// Initializes the SOAP context with options
soap_init2 (soapContext, SOAP_IO_KEEPAIVE, SOAP_IO_KEEPAIVE);

// Sets timeouts in seconds
soapContext->send_timeout = 1;
soapContext->recv_timeout = 1;
soapContext->accept_timeout = 1;
```

Remark: A new runtime context is required in each new thread!

• Calling the MSX-E SOAP functions

Important

- In C/C++, each MSX-E SOAP function call is allocating memory (to manage deserialized data) and is not deallocating it automatically. The allocated memory has to be deallocated explicitly by calling, in this order, the `soap_destroy` and `soap_end` functions. These functions can be called after each MSX-E SOAP function call or after several calls. It is important to call these functions regularly to prevent memory leaks.
- If the SOAP function is returning pointer, call the `soap_destroy` and `soap_end` functions only after working with the pointers. If `soap_destroy` and `soap_end` are called before working with the pointers, the values pointed by the pointer will not be available. All dynamically allocated values are destroyed by `soap_destroy` and `soap_end`.

See [C/C++ language SOAP function prototypes](#) to call the MSX-E SOAP functions.

```
for ( ; ; )
{
    soap_error = soap_call_MXCommon__GetModuleTemperatureValueAndStatus (soapContext,
    soap_endpoint, NULL, option, &tempResponse);

    ...

    // As gSOAP doesn't released automatically the memory that it allocates t
o
    // deserialize SOAP data we have to released it explicitly.
    // There is no need to call the function in each loop cycle,
    // it depends on the application, and desired performance and
    // available memory.
    soap_destroy (soapContext);
    soap_end (soapContext);

    ...
}
```

• Error management

There are 3 types of errors that can be generated by the MSX-E SOAP functions:

- SOAP errors: It is the SOAP function return value. More details about the error can be obtained by using the `soap_faultstring` function.

```
soap_error = soap_call_MXCommon__GetModuleTemperatureValueAndStatus (soapContext,
    soap_endpoint, NULL, option, &tempResponse);

if (soap_error)
    printf ("message: %s\n", *soap_faultstring (soapContext));
```

Remark: `soap_faultstring` has to be called just after the SOAP function call that generates the SOAP error and before the call of `soap_destroy` and `soap_end`.

- The MSX-E error (`iReturnValue`): Returns errors that are not linux specific. The `iReturnValue` variable is located in the response structure. These errors are described in this documentation.

```
soap_error = soap_call_MXCommon__GetModuleTemperatureValueAndStatus (soapContext,
    soap_endpoint, NULL, option, &tempResponse);

// Check for errors, if there are, stop the loop
if (checkErrors (soapContext, soap_endpoint, NULL, soap_error, tempResponse.sResponse.iReturnValue, tempResponse.sResponse.syserrno))
    break;
```


- The MSX-E system error (syserrno): Returns linux specific errors. This variable has to be checked only if iReturnValue = -1 or iReturnValue <= -100. It returns the linux errno error. More details about the error can be obtained by using the soap_call_MXCommon__Strerror function.

```
struct MXCommon__ByteArrayResponse byteArrayResponse;

memset (&byteArrayResponse, 0, sizeof (struct MXCommon__ByteArrayResponse));

soap_error = soap_call_MXCommon__GetModuleTemperatureValueAndStatus (soapContext,
    soap_endpoint, NULL, option, &tempResponse);

if ((tempResponse.sResponse.iReturnValue == -1) || ((tempResponse.sResponse.iRetu
    rnValue <= -100) && tempResponse.sResponse.syserrno))
{
    soap_call_MXCommon__Strerror (soapContext, soap_endpoint, soap_action, te
    mpResponse.sResponse.syserrno, &byteArrayResponse);

    // Display the system error
    printf ("\nSystem error: %s\n", byteArrayResponse.sArray.__ptr);
}
```

• Close the SOAP connection

Before to quit the application or when no MSX-E SOAP function calls are necessary, the SOAP connection has to be closed and the context has to be released. Call following functions in this order: soap_destroy, soap_end, soap_free.

```
// Release SOAP
soap_destroy (soapContext);
soap_end (soapContext);

// Close the socket an release the SOAP context
soap_free (soapContext);
```

2.26.3 SOAP calls sample

Here a sample code, and it's makefile, to read the MSX-E internal temperature.

To compile it, use *make IP_ADDRESS=YOUR_MSX-E_IP_ADDRESS* don't forget to set the **INTERFACE_COMMON_LIBRARY_DIR** to point on the MSX-E Common Interface_Library directory.

Remark: Don't use blank spaces in the directories and file names.

temperature.c file

```
#include <unistd.h>
#include <stdio.h>
#include <stdbool.h>
#include <string.h>
#include <errno.h>
#include <MXCommon.nsmap> // this file must be included only once in the project
#include <assert.h>
#include <sys/stat.h>
#include <termios.h>

//-----
//-----

/** kbhit for linux.
```

```

@retval 0: No key pressed.
@retval <>0: Key pressed.
*/
int kbhit (void)
{
    struct termios oldt, newt;
    struct timeval tv;
    fd_set read_fd;
    int status;

    tcgetattr (STDIN_FILENO, &oldt);
    memcpy (&newt, &oldt, sizeof (newt));
    newt.c_lflag &= ~(ICANON | ECHO);
    tcsetattr (STDIN_FILENO, TCSANOW, &newt);

    tv.tv_sec = 0;
    tv.tv_usec = 0;
    FD_ZERO (&read_fd);
    FD_SET (0, &read_fd);

    status = select (1, &read_fd, NULL, NULL, &tv);
    tcsetattr (STDIN_FILENO, TCSANOW, &oldt);

    if (status < 0)
        return 0;
    else
        return (status);
}

//-----

/** getch for linux.

@retval key: Key number.
*/
int getch (void)
{
    struct termios oldt, newt;
    int ch;

    tcgetattr (STDIN_FILENO, &oldt);
    memcpy (&newt, &oldt, sizeof (newt));
    newt.c_lflag &= ~(ICANON | ECHO);
    tcsetattr (STDIN_FILENO, TCSANOW, &newt);
    ch = getchar();
    tcsetattr (STDIN_FILENO, TCSANOW, &oldt);

    return (ch);
}

//-----
//-----

/** Check if the SOAP call returns an error, display it.

@param[in] soapContext : SOAP context structure.
@param[in] soap_endpoint : IP and port number of the system to access.
@param[in] soap_action : SOAP action required by the Web service.
@param[in] soap_error: The SOAP function return value.
@param[in] msxe_error: The MSX-E system return value contained in the response s
    tructure (iReturnValue).
@param[in] msxe_syserro: The MSX-E system errno value contained in the response
    structure (syserro).

```

```

@retval 0: No error.
@retval 1: Error.
*/
int checkErrors (struct soap *soapContext, const char *soap_endpoint, const char
                 *soap_action, int soap_error, int msxe_error, int msxe_syserrno)
{
    if ((soap_error != 0) || (msxe_error != 0))
    {
        printf ("MSX-E function response error: %d\n", msxe_error);
        printf ("soap error: %d ", soap_error);

        // In case of an SOAP error, display it
        if (soap_error)
            printf ("message: %s\n", *soap_faultstring (soapContext))
;
        else
        {
            // It's not an SOAP error but a system error
            if ((msxe_error == -1) || ((msxe_error <= -100) && msxe_s
yserrno))
            {
                struct MXCommon__ByteArrayResponse byteArrayRespo
nse;
                memset (&byteArrayResponse, 0, sizeof (struct
MXCommon__ByteArrayResponse));

                // Get the system error
                if (soap_call_MXCommon__Strerror (soapContext, so
ap_endpoint, soap_action, msxe_syserrno, &byteArrayResponse))
                    printf ("\nsoap_call_MXCommon__Strerror e
rror: %s\n", *soap_faultstring (soapContext));
                else // Display the system error
                    printf ("\nSystem error: %s\n", byteArray
Response.sArray.__ptr);
            }
        }

        return 1;
    }

    return 0;
}

//-----
//-----

int main (void)
{
    struct soap *soapContext;
    unsigned long option = 0;
    struct MXCommon__GetModuleTemperatureValueAndStatusResponse tempResponse
;
    int soap_error = 0;
    char *tempStatus[] = {"NOT AVAILABLE", "TOO LOW", "LOW", "NOMINAL", "HIG
H", "TOO HIGH"};
    // IP address of the MSX-E and after the ':' is the MSX-E SOAP server po
rt number
    char soap_endpoint[] = IP_ADDRESS":5555";

    memset (&tempResponse, 0, sizeof (struct
MXCommon__GetModuleTemperatureValueAndStatusResponse));

    // Allocates and initialize a new runtime context
    if ((soapContext = soap_new ()) == NULL)

```

```

        return EXIT_FAILURE;

    // Initializes the SOAP context with options
    soap_init2 (soapContext, SOAP_IO_KEEPAIVE, SOAP_IO_KEEPAIVE);

    // Sets timeouts in seconds
    soapContext->send_timeout = 1;
    soapContext->recv_timeout = 1;
    soapContext->accept_timeout = 1;

    for ( ; ; )
    {
        soap_error = soap_call_MXCommon__GetModuleTemperatureValueAndSta
tus (soapContext, soap_endpoint, NULL, option, &tempResponse);

        // Check for errors, if there are, stop the loop
        if (checkErrors (soapContext, soap_endpoint, NULL, soap_error, t
empResponse.sResponse.iReturnValue, tempResponse.sResponse.syserrno))
            break;

        // There is no error
        printf ("MSX-E Temperature: %.2lf °C status: ", tempResponse.dTem
peratureValue);

        if (tempResponse.ulTemperatureStatus < 6)
            printf ("%s ", tempStatus[tempResponse.ulTemperatureStatu
s]);
        else
            printf ("unknown ");

        printf ("\n");

        // As gSOAP doesn't released automatically the memory that it al
locates to
        // deserialize SOAP data we have to released it explicitly.
        // There is no need to call the function in each loop cycle,
        // it depends on the application, and desired performance and
        // available memory.
        soap_destroy (soapContext);
        soap_end (soapContext);

        // Listen on keyboard actions
        if (kbhit ())
            if (getch () == 27) // Check if ESC key has been used
                break;
    }

    // Release SOAP
    soap_destroy (soapContext);
    soap_end (soapContext);
    // Close the socket an release the SOAP context
    soap_free (soapContext);

    return EXIT_SUCCESS;
}

```

Makefile

```

TOPDIR                                :=$(shell pwd)

CC                                    :=$(CROSS)$(CC)
LD                                    :=$(CROSS)ld
STRIP                                 :=$(CROSS)strip

ifneq ($(INTERFACE_COMMON_LIBRARY_DIR), "")
INTERFACE_COMMON_LIBRARY_DIR         :=$(TOPDIR)/../../Interface_Library
endif

```

```

# The MSX-E IP address
ifneq ($(IP_ADDRESS), "")
IP_ADDRESS                                     =192.168.99.99
endif

# File implementing SOAP client
SOAPSOURCES                                  := $(INTERFACE_COMMON_LIBRARY_DIR)/MSXEClient.o $
      (INTERFACE_COMMON_LIBRARY_DIR)/MSXEC.o $(INTERFACE_COMMON_LIBRARY_DIR)/stdsoap2.o

CFLAGS                                     += -pipe -Wall -O0 -Winline -I$(INTERFACE_COMMON_
      LIBRARY_DIR) -DIP_ADDRESS="\$(IP_ADDRESS)\\"

TEMPERATURE_SRC                             := temperature.o
BINS                                         := temperature

all: $(BINS)

temperature: $(SOAPSOURCES) $(TEMPERATURE_SRC)
      $(CC) $(CFLAGS) -o $@ $^
      $(STRIP) $@

clean:
      -rm -f $(BINS)
      -rm -f $(INTERFACE_COMMON_LIBRARY_DIR)/*.o
      -rm -f *.o

```

2.27 MSX-E systems servers

The MSX-E embeds servers to provide configuration, management and monitoring over Ethernet.

2.27.1 SOAP server

SOAP means Simple Object Access Protocol. This protocol allows you to use the MSX-E software functions over Ethernet. It is providing **Web Services** that can easily be consumed in many programming languages like C, C++, C#, VB.Net... With the SOAP functions, all functionalities of the MSX-E system can be managed / configured / monitored. This server can be accessed on port 5555. All SOAP functions are described under Modules -> MSX-EXXXX functions.

2.27.2 Event server

MSX-E systems embed an event server that can be used to monitor the current MSX-E state. An MSX-E system is logically divided in subsystem states. A subsystem is uniquely identified by a number, as well as each possible state associated to it. These numerical identifiers can be monitored by using the event server, which signals when the state of a subsystem has changed. The MSX-E will send a new event frame as soon as a subsystem state changes on the MSX-E.

2.27.3 Modbus server

A Modbus server, accessible on port 512 permits, for example, to manage MSX-E systems from a PLC. The port number, endianness (Intel / Motorola) and protocol (TCP/UDP) can be configured through the MSX-E web interface.

Chapter 3

Data Structure Documentation

3.1 ByteArray Struct Reference

Dynamic Array of byte - encapsulates C-type strings.

Data Fields

- `xsd__unsignedByte * __ptr`
pointer of byte
- `int __size`
size of the byte array in bytes
- `int __offset`
not used

3.1.1 Field Documentation

3.1.1.1 `xsd__unsignedByte* ByteArray::__ptr`

3.1.1.2 `int ByteArray::__size`

3.1.1.3 `int ByteArray::__offset`

3.2 DefaultResponse Struct Reference

Data Fields

- `xsd__int iReturnValue`
return value of the call :
- `xsd__int syserrno`
system-error code (the value of the libc "errno" code)

3.2.1 Field Documentation

3.2.1.1 xsd__int DefaultResponse::iReturnValue

- 0 means the remote function performed OK
- -1 means a system error occurred, the meaning of other values is function dependant and should be defined in the related header

3.2.1.2 xsd__int DefaultResponse::syserrno

3.3 MSXE173x__DigitalIOGetNumberResponse Struct Reference

Returns the number of digital IO available on the module.

Data Fields

- struct [DefaultResponse](#) *sResponse*

Default return values.

- [xsd__unsignedLong](#) *ulNumberOfDigitalIO*

Number of digital IO available on the module (up to 16).

3.3.1 Detailed Description

Parameters

[in] *None*

[out] *Response* :

sResponse.iReturnValue :

- 0: means the remote function performed OK
 - -1: means an system error occurred (check errno in this case)
- sResponse.syserrno* : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

3.3.2 Field Documentation

3.3.2.1 struct `DefaultResponse` `MSXE173x__DigitalIOGetNumberResponse::sResponse`

3.3.2.2 `xsd__unsignedLong` `MSXE173x__DigitalIOGetNumberResponse::ulNumberOfDigitalIO`

3.4 MSXE173x__IOWatchdogGetStatusAndValueResponse Struct Reference

Data Fields

- struct `DefaultResponse` `sResponse`
Default return values.
- `xsd__unsignedLong` `ulStatus`
Watchdog current status information.
- `xsd__unsignedLong` `ulValue`
Watchdog current value information.
- `xsd__unsignedLong` `ulInfo`
reserved

3.4.1 Field Documentation

3.4.1.1 struct `DefaultResponse` `MSXE173x__IOWatchdogGetStatusAndValueResponse::sResponse`

3.4.1.2 `xsd__unsignedLong` `MSXE173x__IOWatchdogGetStatusAndValueResponse::ulStatus`

3.4.1.3 `xsd__unsignedLong` `MSXE173x__IOWatchdogGetStatusAndValueResponse::ulValue`

3.4.1.4 `xsd__unsignedLong` `MSXE173x__IOWatchdogGetStatusAndValueResponse::ulInfo`

3.5 MSXE173x__MFEndatGetCurrentLatchConfigurationResponse Struct Reference

Data Fields

- struct `DefaultResponse` `sResponse`
Default return values.
- `xsd__unsignedLong` `ulRunning`
The running state.
- `xsd__unsignedLong` `ulLatchSource`
Mask of bits, that defines the trigger source.

- [xsd__unsignedLong ulTriggerEdgeCount](#)

Number of edges required to detect an hardware trigger (only if hardware trigger is selected).

- [xsd__unsignedLong ulDataFormat](#)

Mask of bits, that defines the format of data that will be sent by the data server.

- [xsd__unsignedLong ulModel](#)

Model/Type of the sensor.

- [xsd__unsignedLong ulPositionSize](#)

Size of the position value send by the sensor (in bits).

- [xsd__unsignedLong ulSignalPeriod](#)

Signal period length or signal periods per revolution for incremental output signals.

- [xsd__unsignedLong ulStepPerRevolution](#)

Measuring step length or measuring steps per revolution with serial data transfer.

- [xsd__unsignedLong ulNumberOfRevolution](#)

Distinguishable revolutions - only for multiturn encoders.

- [xsd__unsignedLong ulScalingFactor](#)

Scaling factor for resolution.

3.5.1 Field Documentation

- 3.5.1.1 struct `DefaultResponse` `MSXE173x__MFEndatGetCurrentLatchConfigurationResponse::sResponse`
- 3.5.1.2 `xsd__unsignedLong` `MSXE173x__MFEndatGetCurrentLatchConfigurationResponse::ulRunning`
- 3.5.1.3 `xsd__unsignedLong` `MSXE173x__MFEndatGetCurrentLatchConfigurationResponse::ulLatchSource`
- 3.5.1.4 `xsd__unsignedLong` `MSXE173x__MFEndatGetCurrentLatchConfigurationResponse::ulTriggerEdgeCount`
- 3.5.1.5 `xsd__unsignedLong` `MSXE173x__MFEndatGetCurrentLatchConfigurationResponse::ulDataFormat`
- 3.5.1.6 `xsd__unsignedLong` `MSXE173x__MFEndatGetCurrentLatchConfigurationResponse::ulModel`
- 3.5.1.7 `xsd__unsignedLong` `MSXE173x__MFEndatGetCurrentLatchConfigurationResponse::ulPositionSize`
- 3.5.1.8 `xsd__unsignedLong` `MSXE173x__MFEndatGetCurrentLatchConfigurationResponse::ulSignalPeriod`
- 3.5.1.9 `xsd__unsignedLong` `MSXE173x__MFEndatGetCurrentLatchConfigurationResponse::ulStepPerRevolution`
- 3.5.1.10 `xsd__unsignedLong` `MSXE173x__MFEndatGetCurrentLatchConfigurationResponse::ulNumberOfRevolution`
- 3.5.1.11 `xsd__unsignedLong` `MSXE173x__MFEndatGetCurrentLatchConfigurationResponse::ulScalingFactor`

3.6 MSXE173x__MFEndatGetErrorSourcesResponse Struct Reference

Data Fields

- struct `DefaultResponse` `sResponse`
- `xsd__unsignedLong` `ulErrorSrc`

3.6.1 Field Documentation

3.6.1.1 struct `DefaultResponse MSXE173x__MFEndatGetErrorSourcesResponse::sResponse`

3.6.1.2 `xsd__unsignedLong MSXE173x__MFEndatGetErrorSourcesResponse::ulErrorSrc`

3.7 MSXE173x__MFEndatGetPositionResponse Struct Reference

Data Fields

- struct `DefaultResponse sResponse`
Default return values.
- `xsd__unsignedLong ulPositionLow`
Position - low bits.
- `xsd__unsignedLong ulPositionHigh`
Position - high bits.

3.7.1 Field Documentation

3.7.1.1 struct `DefaultResponse MSXE173x__MFEndatGetPositionResponse::sResponse`

3.7.1.2 `xsd__unsignedLong MSXE173x__MFEndatGetPositionResponse::ulPositionLow`

3.7.1.3 `xsd__unsignedLong MSXE173x__MFEndatGetPositionResponse::ulPositionHigh`

3.8 MSXE173x__MFEndatGetPositionWithAddDataResponse Struct Reference

Data Fields

- struct `DefaultResponse sResponse`
Default return values.
- `xsd__unsignedLong ulPositionLow`
Position - low bits.
- `xsd__unsignedLong ulPositionHigh`
Position - high bits.
- `xsd__unsignedLong ulAddData1`
Value of the additional data 1.
- `xsd__unsignedLong ulAddData2`
Value of the additional data 2.

3.8.1 Field Documentation

- 3.8.1.1 struct `DefaultResponse` `MSXE173x__MFEndatGetPositionWithAddDataResponse::sResponse`
- 3.8.1.2 `xsd__unsignedLong` `MSXE173x__MFEndatGetPositionWithAddDataResponse::ulPositionLow`
- 3.8.1.3 `xsd__unsignedLong` `MSXE173x__MFEndatGetPositionWithAddDataResponse::ulPositionHigh`
- 3.8.1.4 `xsd__unsignedLong` `MSXE173x__MFEndatGetPositionWithAddDataResponse::ulAddData1`
- 3.8.1.5 `xsd__unsignedLong` `MSXE173x__MFEndatGetPositionWithAddDataResponse::ulAddData2`

3.9 MSXE173x__MFEndatGetSelectedAdditionalDataResponse Struct Reference

Data Fields

- struct `DefaultResponse` `sResponse`

Default return values.

- `xsd__unsignedLong` `ulAdCount`

Number of selected additional data.

- `xsd__unsignedLong` `ulMrsCodeAD1`

MRS code of the additional data 1.

- `xsd__unsignedLong` `ulMrsCodeAD2`

MRS code of the additional data 2.

3.9.1 Field Documentation

- 3.9.1.1 struct `DefaultResponse` `MSXE173x__MFEndatGetSelectedAdditionalDataResponse::sResponse`
- 3.9.1.2 `xsd__unsignedLong` `MSXE173x__MFEndatGetSelectedAdditionalDataResponse::ulAdCount`
- 3.9.1.3 `xsd__unsignedLong` `MSXE173x__MFEndatGetSelectedAdditionalDataResponse::ulMrsCodeAD1`
- 3.9.1.4 `xsd__unsignedLong` `MSXE173x__MFEndatGetSelectedAdditionalDataResponse::ulMrsCodeAD2`

3.10 MSXE173x__MFEndatGetSensorPropertiesResponse Struct Reference

Data Fields

- struct `DefaultResponse` `sResponse`
Default return values.
- `xsd__unsignedLong` `ulIDNumberLsb`
ID Number - low bits.
- `xsd__unsignedLong` `ulIDNumberMsb`
ID Number - high bits.
- `xsd__unsignedLong` `ulSerialNumberLsb`
Serialnumber - low bits.
- `xsd__unsignedLong` `ulSerialNumberMsb`
Serialnumber - high bits.
- `xsd__unsignedLong` `ulModel`
Model/Type of the sensor.
- `xsd__unsignedLong` `ulMode`
Support of EnDat 2.2.
- `xsd__unsignedLong` `ulPositionSize`
Size of the position value send by the sensor (in bits).
- `xsd__unsignedLong` `ulSignalPeriod`
Signal period length or signal periods per revolution for incremental output signals.
- `xsd__unsignedLong` `ulStepPerRevolution`
Measuring step length or measuring steps per revolution with serial data transfer.
- `xsd__unsignedLong` `ulNumberOfRevolution`

Distinguishable revolutions - only for multiturn encoders.

- [xsd__unsignedLong ulScalingFactor](#)

Scaling factor for resolution.

- [xsd__unsignedLong ulAdditionalData](#)

Supported additional data.

3.10.1 Field Documentation

3.10.1.1 struct `DefaultResponse` `MSXE173x__MFEndatGetSensorPropertiesResponse::sResponse`

3.10.1.2 `xsd__unsignedLong` `MSXE173x__MFEndatGetSensorPropertiesResponse::ulIDNumberLsb`

3.10.1.3 `xsd__unsignedLong` `MSXE173x__MFEndatGetSensorPropertiesResponse::ulIDNumberMsb`

3.10.1.4 `xsd__unsignedLong` `MSXE173x__MFEndatGetSensorPropertiesResponse::ulSerialNumberLsb`

3.10.1.5 `xsd__unsignedLong` `MSXE173x__MFEndatGetSensorPropertiesResponse::ulSerialNumberMsb`

3.10.1.6 `xsd__unsignedLong` `MSXE173x__MFEndatGetSensorPropertiesResponse::ulModel`

3.10.1.7 `xsd__unsignedLong` `MSXE173x__MFEndatGetSensorPropertiesResponse::ulMode`

3.10.1.8 `xsd__unsignedLong` `MSXE173x__MFEndatGetSensorPropertiesResponse::ulPositionSize`

3.10.1.9 `xsd__unsignedLong` `MSXE173x__MFEndatGetSensorPropertiesResponse::ulSignalPeriod`

3.10.1.10 `xsd__unsignedLong` `MSXE173x__MFEndatGetSensorPropertiesResponse::ulStepPerRevolution`

3.10.1.11 `xsd__unsignedLong` `MSXE173x__MFEndatGetSensorPropertiesResponse::ulNumberOfRevolution`

3.10.1.12 `xsd__unsignedLong` `MSXE173x__MFEndatGetSensorPropertiesResponse::ulScalingFactor`

3.10.1.13 `xsd__unsignedLong` `MSXE173x__MFEndatGetSensorPropertiesResponse::ulAdditionalData`

3.11 MSXE173x__MFEndatSensorSendParameterResponse Struct Reference

Data Fields

- struct `DefaultResponse` `sResponse`
- `xsd__unsignedLong` `ulParam`

3.11.1 Field Documentation

3.11.1.1 struct `DefaultResponse` `MSXE173x__MFEndatSensorSendParameterResponse::sResponse`

3.11.1.2 `xsd__unsignedLong` `MSXE173x__MFEndatSensorSendParameterResponse::ulParam`

3.12 MSXE173x__MFEndatSensorSendPosAndRecvSelMemAreaResponse Struct Reference

Data Fields

- struct `DefaultResponse` `sResponse`
Default return values.
- `xsd__unsignedLong` `ulPositionLow`
Position - first 32 bits (31 down to 0).
- `xsd__unsignedLong` `ulPositionHigh`
Position - last 32 bits (63 down to 32).

3.12.1 Field Documentation

3.12.1.1 struct `DefaultResponse` `MSXE173x__MFEndatSensorSendPosAndRecvSelMemAreaResponse::sResponse`

3.12.1.2 `xsd__unsignedLong` `MSXE173x__MFEndatSensorSendPosAndRecvSelMemAreaResponse::ulPositionLow`

3.12.1.3 `xsd__unsignedLong` `MSXE173x__MFEndatSensorSendPosAndRecvSelMemAreaResponse::ulPositionHigh`

3.13 MSXE173x__Response Struct Reference

Data Fields

- `xsd__int` `iReturnValue`
return value of the call :
- `xsd__int` `syserrno`
system-error code (the value of the libc "errno" code)

3.13.1 Field Documentation

3.13.1.1 `xsd__int` `MSXE173x__Response::iReturnValue`

- 0 means the remote function performed OK

- -1 means a system error occurred, the meaning of other values is function dependant and should be defined in the related header

3.13.1.2 xsd__int MSXE173x__Response::syserrno

3.14 MSXE173x__unsignedLongResponse Struct Reference

Data Fields

- struct [DefaultResponse sResponse](#)

Default return values.

- [xsd__unsignedLong ulValue](#)

the meaning of this value is defined in the related header of the function who use this type

3.14.1 Field Documentation

3.14.1.1 struct DefaultResponse MSXE173x__unsignedLongResponse::sResponse

3.14.1.2 xsd__unsignedLong MSXE173x__unsignedLongResponse::ulValue

3.15 MSXE173x__unsignedLongTimeStampResponse Struct Reference

Data Fields

- struct [DefaultResponse sResponse](#)

Default return values.

- [xsd__unsignedLong ulValue](#)

the meaning of this value is defined in the related header of the function who use this type

- [xsd__unsignedLong ulTimeStampLow](#)

the meaning of this value is defined in the related header of the function who use this type

- [xsd__unsignedLong ulTimeStampHigh](#)

the meaning of this value is defined in the related header of the function who use this type

3.15.1 Field Documentation

- 3.15.1.1 struct DefaultResponse MSXE173x__unsignedLongTimeStampResponse::sResponse
- 3.15.1.2 xsd__unsignedLong MSXE173x__unsignedLongTimeStampResponse::ulValue
- 3.15.1.3 xsd__unsignedLong MSXE173x__unsignedLongTimeStampResponse::ulTimeStampLow
- 3.15.1.4 xsd__unsignedLong MSXE173x__-
unsignedLongTimeStampResponse::ulTimeStampHigh

3.16 MSXE17xx__DigitalIOGetNumberResponse Struct Reference

Data Fields

- struct [DefaultResponse sResponse](#)
Default return values.
- [xsd__unsignedLong ulNumberOfDigitalIO](#)
Number of digital IO available on the module (up to 16).

3.16.1 Field Documentation

- 3.16.1.1 struct DefaultResponse MSXE17xx__DigitalIOGetNumberResponse::sResponse
- 3.16.1.2 xsd__unsignedLong MSXE17xx__DigitalIOGetNumberResponse::ulNumberOfDigitalIO

3.17 MSXE17xx__IOWatchdogGetStatusAndValueResponse Struct Reference

Data Fields

- struct [DefaultResponse sResponse](#)
Default return values.
- [xsd__unsignedLong ulStatus](#)
Watchdog current status information.
- [xsd__unsignedLong ulValue](#)
Watchdog current value information.
- [xsd__unsignedLong ulInfo](#)
reserved

3.17.1 Field Documentation

- 3.17.1.1 struct `DefaultResponse MSXE17xx__IOWatchdogGetStatusAndValueResponse::sResponse`
- 3.17.1.2 `xsd__unsignedLong MSXE17xx__IOWatchdogGetStatusAndValueResponse::ulStatus`
- 3.17.1.3 `xsd__unsignedLong MSXE17xx__IOWatchdogGetStatusAndValueResponse::ulValue`
- 3.17.1.4 `xsd__unsignedLong MSXE17xx__IOWatchdogGetStatusAndValueResponse::ulInfo`

3.18 MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse Struct Reference

Data Fields

- struct `DefaultResponse sResponse`
Default return values.
- `xsd__unsignedLong ulRunning`
The running state.
- `xsd__unsignedLong ulLatchSource`
Mask of bits, that defines the trigger source.
- `xsd__unsignedLong ulTriggerEdgeCount`
Number of edges required to detect an hardware trigger (only if hardware trigger is selected).
- `xsd__unsignedLong ulDataFormat`
Mask of bits, that defines the format of data that will be sent by the data server.
- `xsd__unsignedLong ulModel`
Model/Type of the sensor.
- `xsd__unsignedLong ulPositionSize`
Size of the position value send by the sensor (in bits).
- `xsd__unsignedLong ulSignalPeriod`
Signal period length or signal periods per revolution for incremental output signals.
- `xsd__unsignedLong ulStepPerRevolution`
Measuring step length or measuring steps per revolution with serial data transfer.
- `xsd__unsignedLong ulNumberOfRevolution`
Distinguishable revolutions - only for multiturn encoders.
- `xsd__unsignedLong ulScalingFactor`
Scaling factor for resolution.

3.18.1 Field Documentation

- 3.18.1.1 struct `DefaultResponse` `MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse::sResponse`
- 3.18.1.2 `xsd__unsignedLong` `MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse::ulRunning`
- 3.18.1.3 `xsd__unsignedLong` `MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse::ulLatchSource`
- 3.18.1.4 `xsd__unsignedLong` `MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse::ulTriggerEdgeCount`
- 3.18.1.5 `xsd__unsignedLong` `MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse::ulDataFormat`
- 3.18.1.6 `xsd__unsignedLong` `MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse::ulModel`
- 3.18.1.7 `xsd__unsignedLong` `MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse::ulPositionSize`
- 3.18.1.8 `xsd__unsignedLong` `MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse::ulSignalPeriod`
- 3.18.1.9 `xsd__unsignedLong` `MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse::ulStepPerRevolution`
- 3.18.1.10 `xsd__unsignedLong` `MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse::ulNumberOfRevolution`
- 3.18.1.11 `xsd__unsignedLong` `MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse::ulScalingFactor`

3.19 MSXE17xx__MFEndatGetErrorSourcesResponse Struct Reference

Data Fields

- struct `DefaultResponse` `sResponse`
- `xsd__unsignedLong` `ulErrorSrc`

3.19.1 Field Documentation

3.19.1.1 struct `DefaultResponse MSXE17xx__MFEndatGetErrorSourcesResponse::sResponse`

3.19.1.2 `xsd__unsignedLong MSXE17xx__MFEndatGetErrorSourcesResponse::ulErrorSrc`

3.20 MSXE17xx__MFEndatGetPositionResponse Struct Reference

Data Fields

- struct `DefaultResponse sResponse`
Default return values.
- `xsd__unsignedLong ulPositionLow`
Position - low bits.
- `xsd__unsignedLong ulPositionHigh`
Position - high bits.

3.20.1 Field Documentation

3.20.1.1 struct `DefaultResponse MSXE17xx__MFEndatGetPositionResponse::sResponse`

3.20.1.2 `xsd__unsignedLong MSXE17xx__MFEndatGetPositionResponse::ulPositionLow`

3.20.1.3 `xsd__unsignedLong MSXE17xx__MFEndatGetPositionResponse::ulPositionHigh`

3.21 MSXE17xx__MFEndatGetPositionWithAddDataResponse Struct Reference

Data Fields

- struct `DefaultResponse sResponse`
Default return values.
- `xsd__unsignedLong ulPositionLow`
Position - low bits.
- `xsd__unsignedLong ulPositionHigh`
Position - high bits.
- `xsd__unsignedLong ulAddData1`
Value of the additional data 1.
- `xsd__unsignedLong ulAddData2`
Value of the additional data 2.

3.21.1 Field Documentation

- 3.21.1.1 struct `DefaultResponse` `MSXE17xx__MFEndatGetPositionWithAddDataResponse::sResponse`
- 3.21.1.2 `xsd__unsignedLong` `MSXE17xx__MFEndatGetPositionWithAddDataResponse::ulPositionLow`
- 3.21.1.3 `xsd__unsignedLong` `MSXE17xx__MFEndatGetPositionWithAddDataResponse::ulPositionHigh`
- 3.21.1.4 `xsd__unsignedLong` `MSXE17xx__MFEndatGetPositionWithAddDataResponse::ulAddData1`
- 3.21.1.5 `xsd__unsignedLong` `MSXE17xx__MFEndatGetPositionWithAddDataResponse::ulAddData2`

3.22 MSXE17xx__MFEndatGetSelectedAdditionalDataResponse Struct Reference

Data Fields

- struct `DefaultResponse` `sResponse`

Default return values.

- `xsd__unsignedLong` `ulAdCount`

Number of selected additional data.

- `xsd__unsignedLong` `ulMrsCodeAD1`

MRS code of the additional data 1.

- `xsd__unsignedLong` `ulMrsCodeAD2`

MRS code of the additional data 2.

3.22.1 Field Documentation

- 3.22.1.1 **struct DefaultResponse MSXE17xx__ - MFEndatGetSelectedAdditionalDataResponse::sResponse**
- 3.22.1.2 **xsd__unsignedLong MSXE17xx__ - MFEndatGetSelectedAdditionalDataResponse::ulAdCount**
- 3.22.1.3 **xsd__unsignedLong MSXE17xx__ - MFEndatGetSelectedAdditionalDataResponse::ulMrsCodeAD1**
- 3.22.1.4 **xsd__unsignedLong MSXE17xx__ - MFEndatGetSelectedAdditionalDataResponse::ulMrsCodeAD2**

3.23 MSXE17xx__MFEndatGetSensorPropertiesResponse Struct Reference

Data Fields

- struct [DefaultResponse sResponse](#)
Default return values.
- [xsd__unsignedLong ulIDNumberLsb](#)
ID Number - low bits.
- [xsd__unsignedLong ulIDNumberMsb](#)
ID Number - high bits.
- [xsd__unsignedLong ulSerialNumberLsb](#)
Serialnumber - low bits.
- [xsd__unsignedLong ulSerialNumberMsb](#)
Serialnumber - high bits.
- [xsd__unsignedLong ulModel](#)
Model/Type of the sensor.
- [xsd__unsignedLong ulMode](#)
Support of EnDat 2.2.
- [xsd__unsignedLong ulPositionSize](#)
Size of the position value send by the sensor (in bits).
- [xsd__unsignedLong ulSignalPeriod](#)
Signal period length or signal periods per revolution for incremental output signals.
- [xsd__unsignedLong ulStepPerRevolution](#)
Measuring step length or measuring steps per revolution with serial data transfer.
- [xsd__unsignedLong ulNumberOfRevolution](#)

Distinguishable revolutions - only for multiturn encoders.

- [xsd__unsignedLong ulScalingFactor](#)

Scaling factor for resolution.

- [xsd__unsignedLong ulAdditionalData](#)

Supported additional data.

3.23.1 Field Documentation

3.23.1.1 struct `DefaultResponse` `MSXE17xx__MFEndatGetSensorPropertiesResponse::sResponse`

3.23.1.2 `xsd__unsignedLong` `MSXE17xx__MFEndatGetSensorPropertiesResponse::ulIDNumberLsb`

3.23.1.3 `xsd__unsignedLong` `MSXE17xx__MFEndatGetSensorPropertiesResponse::ulIDNumberMsb`

3.23.1.4 `xsd__unsignedLong` `MSXE17xx__MFEndatGetSensorPropertiesResponse::ulSerialNumberLsb`

3.23.1.5 `xsd__unsignedLong` `MSXE17xx__MFEndatGetSensorPropertiesResponse::ulSerialNumberMsb`

3.23.1.6 `xsd__unsignedLong` `MSXE17xx__MFEndatGetSensorPropertiesResponse::ulModel`

3.23.1.7 `xsd__unsignedLong` `MSXE17xx__MFEndatGetSensorPropertiesResponse::ulMode`

3.23.1.8 `xsd__unsignedLong` `MSXE17xx__MFEndatGetSensorPropertiesResponse::ulPositionSize`

3.23.1.9 `xsd__unsignedLong` `MSXE17xx__MFEndatGetSensorPropertiesResponse::ulSignalPeriod`

3.23.1.10 `xsd__unsignedLong` `MSXE17xx__MFEndatGetSensorPropertiesResponse::ulStepPerRevolution`

3.23.1.11 `xsd__unsignedLong` `MSXE17xx__MFEndatGetSensorPropertiesResponse::ulNumberOfRevolution`

3.23.1.12 `xsd__unsignedLong` `MSXE17xx__MFEndatGetSensorPropertiesResponse::ulScalingFactor`

3.23.1.13 `xsd__unsignedLong` `MSXE17xx__MFEndatGetSensorPropertiesResponse::ulAdditionalData`

3.24 MSXE17xx__MFEndatSensorSendParameterResponse Struct Reference

Data Fields

- struct `DefaultResponse` `sResponse`
- `xsd__unsignedLong` `ulParam`

3.24.1 Field Documentation

3.24.1.1 struct `DefaultResponse` `MSXE17xx__MFEndatSensorSendParameterResponse::sResponse`

3.24.1.2 `xsd__unsignedLong` `MSXE17xx__MFEndatSensorSendParameterResponse::ulParam`

3.25 MSXE17xx__MFEndatSensorSendPosAndRecvSelMemAreaResponse Struct Reference

Data Fields

- struct `DefaultResponse` `sResponse`
Default return values.
- `xsd__unsignedLong` `ulPositionLow`
Position - first 32 bits (31 down to 0).
- `xsd__unsignedLong` `ulPositionHigh`
Position - last 32 bits (63 down to 32).

3.25.1 Field Documentation

3.25.1.1 struct `DefaultResponse` `MSXE17xx__MFEndatSensorSendPosAndRecvSelMemAreaResponse::sResponse`

3.25.1.2 `xsd__unsignedLong` `MSXE17xx__MFEndatSensorSendPosAndRecvSelMemAreaResponse::ulPositionLow`

3.25.1.3 `xsd__unsignedLong` `MSXE17xx__MFEndatSensorSendPosAndRecvSelMemAreaResponse::ulPositionHigh`

3.26 MSXE17xx__Response Struct Reference

Data Fields

- `xsd__int` `iReturnValue`
return value of the call :
- `xsd__int` `syserrno`
system-error code (the value of the libc "errno" code)

3.26.1 Field Documentation

3.26.1.1 `xsd__int` `MSXE17xx__Response::iReturnValue`

- 0 means the remote function performed OK

- -1 means a system error occurred, the meaning of other values is function dependant and should be defined in the related header

3.26.1.2 xsd__int MSXE17xx__Response::syserrno

3.27 MSXE17xx__unsignedLongResponse Struct Reference

Data Fields

- struct [DefaultResponse sResponse](#)

Default return values.

- [xsd__unsignedLong ulValue](#)

the meaning of this value is defined in the related header of the function who use this type

3.27.1 Field Documentation

3.27.1.1 struct DefaultResponse MSXE17xx__unsignedLongResponse::sResponse

3.27.1.2 xsd__unsignedLong MSXE17xx__unsignedLongResponse::ulValue

3.28 MSXE17xx__unsignedLongTimeStampResponse Struct Reference

Data Fields

- struct [DefaultResponse sResponse](#)

Default return values.

- [xsd__unsignedLong ulValue](#)

the meaning of this value is defined in the related header of the function who use this type

- [xsd__unsignedLong ulTimeStampLow](#)

the meaning of this value is defined in the related header of the function who use this type

- [xsd__unsignedLong ulTimeStampHigh](#)

the meaning of this value is defined in the related header of the function who use this type

3.28.1 Field Documentation

- 3.28.1.1 struct DefaultResponse MSXE17xx__unsignedLongTimeStampResponse::sResponse
- 3.28.1.2 xsd__unsignedLong MSXE17xx__unsignedLongTimeStampResponse::ulValue
- 3.28.1.3 xsd__unsignedLong MSXE17xx__unsignedLongTimeStampResponse::ulTimeStampLow
- 3.28.1.4 xsd__unsignedLong MSXE17xx__ -
unsignedLongTimeStampResponse::ulTimeStampHigh

3.29 MXCommon__ByteArrayResponse Struct Reference

Response containing a C-type string.

Data Fields

- struct [DefaultResponse sResponse](#)
Default return values.
- struct [ByteArray sArray](#)
Dynamic Array of byte - encapsulates C-type strings.

3.29.1 Field Documentation

- 3.29.1.1 struct DefaultResponse MXCommon__ByteArrayResponse::sResponse
- 3.29.1.2 struct ByteArray MXCommon__ByteArrayResponse::sArray

3.30 MXCommon__FileResponse Struct Reference

Response containing a chunk of a file.

Data Fields

- struct [DefaultResponse sResponse](#)
return values.
- struct [ByteArray sArray](#)
Dynamic Array of byte.
- [xsd__unsignedLong ulEOF](#)
flag indicating end of file.

3.30.1 Field Documentation

3.30.1.1 struct `DefaultResponse` `MXCommon__FileResponse::sResponse`

3.30.1.2 struct `ByteArray` `MXCommon__FileResponse::sArray`

3.30.1.3 `xsd__unsignedLong` `MXCommon__FileResponse::ulEOF`

3.31 `MXCommon__GetAutoConfigurationFileResponse` Struct Reference

Data Fields

- struct `DefaultResponse` `sResponse`

Default return values.

- struct `ByteArray` `bArray`

Array of byte of the file.

- `xsd__unsignedLong` `ulEOF`

End of file flag.

3.31.1 Field Documentation

3.31.1.1 struct `DefaultResponse` `MXCommon__GetAutoConfigurationFileResponse::sResponse`

3.31.1.2 struct `ByteArray` `MXCommon__GetAutoConfigurationFileResponse::bArray`

3.31.1.3 `xsd__unsignedLong` `MXCommon__GetAutoConfigurationFileResponse::ulEOF`

3.32 `MXCommon__GetEthernetLinksStatesResponse` Struct Reference

Data Fields

- struct `DefaultResponse` `sResponse`

Default return values.

- struct `sGetEthernetLinksStatesPort` `sPort0`

- struct `sGetEthernetLinksStatesPort` `sPort1`

3.32.1 Field Documentation

3.32.1.1 struct DefaultResponse MXCommon__GetEthernetLinksStatesResponse::sResponse

3.32.1.2 struct sGetEthernetLinksStatesPort MXCommon__ -
GetEthernetLinksStatesResponse::sPort0

3.32.1.3 struct sGetEthernetLinksStatesPort MXCommon__ -
GetEthernetLinksStatesResponse::sPort1

3.33 MXCommon__GetHardwareTriggerFilterTimeResponse Struct Reference

Data Fields

- struct [DefaultResponse](#) sResponse
Default return values.
- [xsd__unsignedLong](#) ulFilterTime
Hardware filter time (step of 250ns).
- [xsd__unsignedLong](#) ulInfo01
Reserved.
- [xsd__unsignedLong](#) ulInfo02
Reserved.

3.33.1 Field Documentation

3.33.1.1 struct DefaultResponse MXCommon__ -
GetHardwareTriggerFilterTimeResponse::sResponse

3.33.1.2 [xsd__unsignedLong](#) MXCommon__ -
GetHardwareTriggerFilterTimeResponse::ulFilterTime

3.33.1.3 [xsd__unsignedLong](#) MXCommon__GetHardwareTriggerFilterTimeResponse::ulInfo01

3.33.1.4 [xsd__unsignedLong](#) MXCommon__GetHardwareTriggerFilterTimeResponse::ulInfo02

3.34 MXCommon__GetHardwareTriggerStateResponse Struct Reference

Data Fields

- struct [DefaultResponse](#) sResponse
Default return values.
- [xsd__unsignedLong](#) ulState

0 : Trigger input is low / 1 : Trigger input is high

- [xsd__unsignedLong ulInfo01](#)

Reserved.

- [xsd__unsignedLong ulInfo02](#)

Reserved.

3.34.1 Field Documentation

3.34.1.1 struct [DefaultResponse](#) [MXCommon__GetHardwareTriggerStateResponse::sResponse](#)

3.34.1.2 [xsd__unsignedLong](#) [MXCommon__GetHardwareTriggerStateResponse::ulState](#)

3.34.1.3 [xsd__unsignedLong](#) [MXCommon__GetHardwareTriggerStateResponse::ulInfo01](#)

3.34.1.4 [xsd__unsignedLong](#) [MXCommon__GetHardwareTriggerStateResponse::ulInfo02](#)

3.35 [MXCommon__GetModuleTemperatureValueAndStatusResponse](#) Struct Reference

Data Fields

- struct [DefaultResponse](#) [sResponse](#)

Default return value.

- [xsd__double](#) [dTemperatureValue](#)

Temperature value.

- [xsd__unsignedLong](#) [ulTemperatureStatus](#)

Temperature status.

- [xsd__unsignedLong](#) [ulInfo](#)

Reserved.

3.35.1 Field Documentation

- 3.35.1.1 struct `DefaultResponse` `MXCommon__GetModuleTemperatureValueAndStatusResponse::sResponse`
- 3.35.1.2 `xsd__double` `MXCommon__GetModuleTemperatureValueAndStatusResponse::dTemperatureValue`
- 3.35.1.3 `xsd__unsignedLong` `MXCommon__GetModuleTemperatureValueAndStatusResponse::ulTemperatureStatus`
- 3.35.1.4 `xsd__unsignedLong` `MXCommon__GetModuleTemperatureValueAndStatusResponse::ulInfo`

3.36 MXCommon__GetTimeResponse Struct Reference

Data Fields

- struct `DefaultResponse` `sResponse`
Default return values.
- `xsd__unsignedLong` `ulLowTime`
Number of microseconds since the begin of the second.
- `xsd__unsignedLong` `ulHighTime`
Number of seconds since the Epoch (1st January,1970).

3.36.1 Field Documentation

- 3.36.1.1 struct `DefaultResponse` `MXCommon__GetTimeResponse::sResponse`
- 3.36.1.2 `xsd__unsignedLong` `MXCommon__GetTimeResponse::ulLowTime`
- 3.36.1.3 `xsd__unsignedLong` `MXCommon__GetTimeResponse::ulHighTime`

3.37 MXCommon__GetUpTimeResponse Struct Reference

Data Fields

- struct `DefaultResponse` `sResponse`
Default return value.
- `xsd__unsignedLong` `ulUpTime`
Reserved.

3.37.1 Field Documentation

3.37.1.1 struct `DefaultResponse` `MXCommon__GetUpTimeResponse::sResponse`

3.37.1.2 `xsd__unsignedLong` `MXCommon__GetUpTimeResponse::ulUpTime`

3.38 MXCommon__Response Struct Reference

contains return values

Data Fields

- `xsd__int` `iReturnValue`

return value of the call :

- *0 success*
- *-1 a system error occurred, the meaning of other values is function dependent and should be defined in the related header.*

- `xsd__int` `syserrno`

system-error code (the value of the libc "errno" code, see `MXCommon__Strerror()`).

3.38.1 Field Documentation

3.38.1.1 `xsd__int` `MXCommon__Response::iReturnValue`

3.38.1.2 `xsd__int` `MXCommon__Response::syserrno`

3.39 MXCommon__TestCustomerIDResponse Struct Reference

Data Fields

- struct `DefaultResponse` `sResponse`

Default return values.

- struct `ByteArray` `bValueArray`

non encrypted value

- struct `ByteArray` `bCryptedValueArray`

encrypted value

3.39.1 Field Documentation

3.39.1.1 struct DefaultResponse MXCommon__TestCustomerIDResponse::sResponse

3.39.1.2 struct ByteArray MXCommon__TestCustomerIDResponse::bValueArray

3.39.1.3 struct ByteArray MXCommon__TestCustomerIDResponse::bCryptedValueArray

3.40 MXCommon__unsignedLongResponse Struct Reference

Response containing a numerical value (ex: return code).

Data Fields

- struct [DefaultResponse](#) sResponse

Default return values.

- [xsd__unsignedLong](#) ulValue

The meaning of this value is defined in the related header of the function who use this type.

3.40.1 Field Documentation

3.40.1.1 struct DefaultResponse MXCommon__unsignedLongResponse::sResponse

3.40.1.2 [xsd__unsignedLong](#) MXCommon__unsignedLongResponse::ulValue

3.41 sGetEthernetLinksStatesPort Struct Reference

Data Fields

- [xsd__unsignedLong](#) ulState
- [xsd__unsignedLong](#) ulSpeed
- [xsd__unsignedLong](#) ulDuplex
- [xsd__unsignedLong](#) ulInfo1
- [xsd__unsignedLong](#) ulInfo2

3.41.1 Field Documentation

3.41.1.1 `xsd__unsignedLong sGetEthernetLinksStatesPort::ulState`

3.41.1.2 `xsd__unsignedLong sGetEthernetLinksStatesPort::ulSpeed`

3.41.1.3 `xsd__unsignedLong sGetEthernetLinksStatesPort::ulDuplex`

3.41.1.4 `xsd__unsignedLong sGetEthernetLinksStatesPort::ulInfo1`

3.41.1.5 `xsd__unsignedLong sGetEthernetLinksStatesPort::ulInfo2`

3.42 UnsignedLongArray Struct Reference

Dynamic Array of unsigned long.

Data Fields

- `xsd__unsignedLong * __ptr`
pointer of unsigned Long
- `int __size`
size of the unsigned Long array in Bytes
- `int __offset`
not used

3.42.1 Field Documentation

3.42.1.1 `xsd__unsignedLong* UnsignedLongArray::__ptr`

3.42.1.2 `int UnsignedLongArray::__size`

3.42.1.3 `int UnsignedLongArray::__offset`

3.43 UnsignedShortArray Struct Reference

Dynamic Array of unsigned short.

Data Fields

- `xsd__unsignedShort * __ptr`
pointer of unsigned short
- `int __size`
size of the unsigned short array in Bytes

- int [__offset](#)
not used

3.43.1 Field Documentation

3.43.1.1 `xsd__unsignedShort* UnsignedShortArray::__ptr`

3.43.1.2 `int UnsignedShortArray::__size`

3.43.1.3 `int UnsignedShortArray::__offset`

3.44 xsd__base64Binary Struct Reference

Dynamic Array of byte for input use.

Data Fields

- unsigned char * [__ptr](#)
pointer of byte
- int [__size](#)
size of the byte array

3.44.1 Field Documentation

3.44.1.1 `unsigned char* xsd__base64Binary::__ptr`

3.44.1.2 `int xsd__base64Binary::__size`

Chapter 4

File Documentation

4.1 MSXE173x_public_doc.h File Reference

Data Structures

- struct [xsd__base64Binary](#)
Dynamic Array of byte for input use.
- struct [UnsignedShortArray](#)
Dynamic Array of unsigned short.
- struct [UnsignedLongArray](#)
Dynamic Array of unsigned long.
- struct [ByteArray](#)
Dynamic Array of byte - encapsulates C-type strings.
- struct [DefaultResponse](#)
- struct [MXCommon__Response](#)
contains return values
- struct [MXCommon__ByteArrayResponse](#)
Response containing a C-type string.
- struct [MXCommon__FileResponse](#)
Response containing a chunk of a file.
- struct [MXCommon__unsignedLongResponse](#)
Response containing a numerical value (ex: return code).
- struct [sGetEthernetLinksStatesPort](#)
- struct [MXCommon__GetEthernetLinksStatesResponse](#)
- struct [MXCommon__GetModuleTemperatureValueAndStatusResponse](#)
- struct [MXCommon__GetHardwareTriggerFilterTimeResponse](#)
- struct [MXCommon__GetHardwareTriggerStateResponse](#)

- struct [MXCommon__TestCustomerIDResponse](#)
- struct [MXCommon__GetTimeResponse](#)
- struct [MXCommon__GetUpTimeResponse](#)
- struct [MXCommon__GetAutoConfigurationFileResponse](#)
- struct [MSXE173x__Response](#)
- struct [MSXE173x__unsignedLongResponse](#)
- struct [MSXE173x__unsignedLongTimeStampResponse](#)
- struct [MSXE173x__DigitalIOGetNumberResponse](#)

Returns the number of digital IO available on the module.

- struct [MSXE173x__IOWatchdogGetStatusAndValueResponse](#)
- struct [MSXE173x__MFEndatGetPositionResponse](#)
- struct [MSXE173x__MFEndatGetSensorPropertiesResponse](#)
- struct [MSXE173x__MFEndatGetErrorSourcesResponse](#)
- struct [MSXE173x__MFEndatSensorSendParameterResponse](#)
- struct [MSXE173x__MFEndatSensorSendPosAndRecvSelMemAreaResponse](#)
- struct [MSXE173x__MFEndatGetPositionWithAddDataResponse](#)
- struct [MSXE173x__MFEndatGetSelectedAdditionalDataResponse](#)
- struct [MSXE173x__MFEndatGetCurrentLatchConfigurationResponse](#)
- struct [MSXE17xx__Response](#)
- struct [MSXE17xx__unsignedLongResponse](#)
- struct [MSXE17xx__unsignedLongTimeStampResponse](#)
- struct [MSXE17xx__DigitalIOGetNumberResponse](#)
- struct [MSXE17xx__IOWatchdogGetStatusAndValueResponse](#)
- struct [MSXE17xx__MFEndatGetPositionResponse](#)
- struct [MSXE17xx__MFEndatGetSensorPropertiesResponse](#)
- struct [MSXE17xx__MFEndatGetErrorSourcesResponse](#)
- struct [MSXE17xx__MFEndatSensorSendParameterResponse](#)
- struct [MSXE17xx__MFEndatSensorSendPosAndRecvSelMemAreaResponse](#)
- struct [MSXE17xx__MFEndatGetPositionWithAddDataResponse](#)
- struct [MSXE17xx__MFEndatGetSelectedAdditionalDataResponse](#)
- struct [MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse](#)

Defines

- #define [MSXE170X_COUNTER_QUADRUPLE_MODE](#) 0x4
In the quadruple mode, the edge analysis circuit generates a counting pulse from each edge of two signals which are phase-shifted in relation to each other.
- #define [MSXE170X_COUNTER_DOUBLE_MODE](#) 0x2
Same function as quadruple mode, except only 2 of the 4 edges are analysed.
- #define [MSXE170X_COUNTER_SIMPLE_MODE](#) 0x1
Same function as quadruple mode, except one of the 4 edges is analysed in each period.
- #define [MSXE170X_COUNTER_DIRECT_MODE](#) 0x0
In the direct mode both edge analysis circuits become inactive.
- #define [MSXE170X_COUNTER_HYSTERESIS_ON](#) 0x1
On both edge analysis circuit a hysteresis switch is available.

- #define [MSXE170X_COUNTER_HYSTERESIS_OFF](#) 0x0
The first pulse will not be suppressed after a change of rotation.
- #define [MSXE170X_COUNTER_INCREMENT](#) 0x0
The counter increments after each counting pulse.
- #define [MSXE170X_COUNTER_DECREMENT](#) 0x1
The counter decrements after each counting pulse.
- #define [MSXE170X_COUNTER_LOW_EDGE_LATCH_AND_CLEAR_COUNTER](#) 0x0
After an index signal (Low level), the counter value (32-bit) is latched into the first latch register and then deleted (32-bit).
- #define [MSXE170X_COUNTER_HIGH_EDGE_LATCH_AND_CLEAR_COUNTER](#) 0x1
After an index signal (High level), the counter value (32-bit) is latched into the first latch register and then deleted (32-bit).
- #define [MSXE170X_COUNTER_LOW_EDGE_LATCH_COUNTER](#) 0x2
After an index signal (Low level), the counter value (32-bit) is latched into the first latch register.
- #define [MSXE170X_COUNTER_HIGH_EDGE_LATCH_COUNTER](#) 0x3
After an index signal (High level), the counter value (32-bit) is latched into the first latch register.

Typedefs

- typedef char * [xsd__string](#)
encode xsd__string value as the xsd:string schema type
- typedef char [xsd__char](#)
encode xsd__string value as the xsd:char schema type
- typedef float [xsd__float](#)
encode xsd__float value as the xsd:float schema type
- typedef double [xsd__double](#)
encode xsd__double value as the xsd:double schema type
- typedef int [xsd__int](#)
encode xsd__int value as the xsd:int schema type
- typedef long [xsd__long](#)
encode xsd__long value as the xsd:long schema type
- typedef unsigned char [xsd__unsignedByte](#)
encode xsd__unsignedByte value as the xsd:unsignedByte schema type
- typedef unsigned int [xsd__unsignedInt](#)
encode xsd__unsignedInt value as the xsd:unsignedInt schema type

- typedef unsigned short int [xsd__unsignedShort](#)
encode xsd__unsignedShort value as the xsd:unsignedShort schema type
- typedef unsigned long [xsd__unsignedLong](#)
encode xsd__unsignedLong value as the xsd:unsignedLong schema type

Functions

- int [MXCommon__GetModuleType](#) (void ___, struct [MXCommon__ByteArrayResponse](#) *Response)
This function return the type of the MSX-E Module.
- int [MXCommon__GetHostname](#) (void ___, struct [MXCommon__ByteArrayResponse](#) *Response)
This function return the hostname of the MSX-E Module.
- int [MXCommon__SetHostname](#) (struct [xsd__base64Binary](#) *bHostname, struct [MXCommon__Response](#) *Response)
This function allows to set the hostname of the MSX-E Module.
- int [MXCommon__GetClientConnections](#) (void ___, struct [MXCommon__ByteArrayResponse](#) *Response)
This function return the client connection list.
- int [MXCommon__Sterror](#) (xsd__int errnum, struct [MXCommon__ByteArrayResponse](#) *Response)
Call the libc strerror() on the remote device (actually this is a call to strerror_r()).
- int [MXCommon__Reboot](#) (void ___, struct [MXCommon__Response](#) *Response)
Ask the MSX-E module to reboot.
- int [MXCommon__ResetAllIOFunctionalities](#) (xsd__unsignedLong ulOption, struct [MXCommon__Response](#) *Response)
Reset the I/O functionalities of the MSX-E system.
- int [MXCommon__DataseverRestart](#) (xsd__unsignedLong ulAction, xsd__unsignedLong ulOption, struct [MXCommon__Response](#) *Response)
Restart the data-server service.
- int [MXCommon__GetEthernetLinksStates](#) (void ___, struct [MXCommon__GetEthernetLinksStatesResponse](#) *Response)
Get MSX-E Ethernet links states.
- int [MXCommon__GetModuleTemperatureValueAndStatus](#) (xsd__unsignedLong ulOption, struct [MXCommon__GetModuleTemperatureValueAndStatusResponse](#) *Response)
Read the temperature on the module.
- int [MXCommon__SetModuleTemperatureWarningLevels](#) (xsd__double dMinimalWarningLevel, xsd__double dMaximalWarningLevel, xsd__unsignedLong ulOption, struct [MXCommon__Response](#) *Response)

Set the temperature warning level on the module.

- `int MXCommon__SetHardwareTriggerFilterTime (xsd__unsignedLong ulFilterTime, xsd__unsignedLong ulOption, struct MXCommon__Response *Response)`

Sets the filter time for the hardware trigger input in steps of 250 ns (max value: 65535).

- `int MXCommon__GetHardwareTriggerFilterTime (xsd__unsignedLong ulOption, struct MXCommon__GetHardwareTriggerFilterTimeResponse *Response)`

Get the filter time for the hardware trigger input.

- `int MXCommon__GetHardwareTriggerState (xsd__unsignedLong ulOption, struct MXCommon__GetHardwareTriggerStateResponse *Response)`

Get the hardware trigger state after the filter.

- `int MXCommon__SetCustomerKey (struct xsd__base64Binary *bKey, struct xsd__base64Binary *bPublicKey, struct MXCommon__Response *Response)`

Set the Customer key.

- `int MXCommon__TestCustomerID (void *_, struct MXCommon__TestCustomerIDResponse *Response)`

Test the Customer ID (if the module has the right customer Key).

- `int MXCommon__SetTime (xsd__unsignedLong ulLowTime, xsd__unsignedLong ulHighTime, struct MXCommon__Response *Response)`

Set the time on the module.

- `int MXCommon__SysToHardwareClock (void *_, struct MXCommon__Response *Response)`

Set the hardware clock (if present) to the current system time.

- `int MXCommon__HardwareClockToSys (void *_, struct MXCommon__Response *Response)`

Set the system time from the hardware clock (if present).

- `int MXCommon__GetTime (void *_, struct MXCommon__GetTimeResponse *Response)`

Get the time on the module.

- `int MXCommon__GetUpTime (void *_, struct MXCommon__GetUpTimeResponse *Response)`

Ask the MSX-E module uptime (number of seconds since the last boot).

- `int MXCommon__GetAutoConfigurationFile (void *_, struct MXCommon__GetAutoConfigurationFileResponse *Response)`

Get the auto configuration file of the module.

- `int MXCommon__SetAutoConfigurationFile (struct xsd__base64Binary *ByteArrayInput, xsd__unsignedLong ulEOF, struct MXCommon__Response *Response)`

Set the auto configuration file of the module.

- `int MXCommon__StartAutoConfiguration (void *_, struct MXCommon__ByteArrayResponse *Response)`

start/Restart the auto configuration

- `int MXCommon__InitAndStartSynchroTimer (xsd__unsignedLong ulTimeBase, xsd__unsignedLong ulReloadValue, xsd__unsignedLong ulNbrOfCycle, xsd__unsignedLong ulGenerateTriggerMode, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MXCommon__Response *Response)`
Initialises and starts the synchronisation timer of the module (not already available on all module).
- `int MXCommon__StopAndReleaseSynchroTimer (xsd__unsignedLong ulOption01, struct MXCommon__Response *Response)`
start/Restart the synchronisation timer (not already available on all module)
- `int MXCommon__GetConfigurationBackupFile (void *_ , struct MXCommon__FileResponse *Response)`
Download a configuration backup file from the module.
- `int MXCommon__ApplyConfigurationBackupFile (struct xsd__base64Binary *ByteArrayInput, xsd__unsignedLong ulEOF, struct MXCommon__Response *Response)`
Upload a new configuration on the module.
- `int MXCommon__ChangePassword (struct xsd__base64Binary *PreviousUser, struct xsd__base64Binary *PreviousPassword, struct xsd__base64Binary *NewUser, struct xsd__base64Binary *NewPassword, struct MXCommon__Response *Response)`
Set a new id/password.
- `int MXCommon__GetSubSystemState (xsd__unsignedLong SubsystemID, struct MXCommon__unsignedLongResponse *Response)`
Returns the current state of the specified sub-system.
- `int MXCommon__GetSubsystemIDFromName (struct xsd__base64Binary *SubsystemName, struct MXCommon__unsignedLongResponse *Response)`
Returns the ID of the sub-system of symbolic name "SubsystemName".
- `int MXCommon__GetStateIDFromName (xsd__unsignedLong SubsystemID, struct xsd__base64Binary *StateName, struct MXCommon__unsignedLongResponse *Response)`
Returns the ID of the state of symbolic name "StateName" of the sub-system of ID "SubsystemID".
- `int MXCommon__GetSubsystemNameFromID (xsd__unsignedLong SubsystemID, struct MXCommon__ByteArrayResponse *Response)`
Returns the symbolic name of the sub-system of numerical ID "SubsystemName".
- `int MXCommon__GetStateNameFromID (xsd__unsignedLong SubsystemID, xsd__unsignedLong StateID, struct MXCommon__ByteArrayResponse *Response)`
Returns the symbolic name of the state of numerical ID "StateID" of the sub-system of ID "SubsystemID".
- `int MXCommon__GetOptionInformation (void *_ , xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MXCommon__ByteArrayResponse *Response)`
Enables to get information about the options available on the system.
- `int MXCommon__SetToMaster (void *_ , xsd__unsignedLong ulState, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MXCommon__Response *Response)`

Writes if the MSXE has to be always set to master The master mode (when enabled) make the system always detected as master.

- int [MXCommon__GetSynchronizationStatus](#) (void ___, [xsd__unsignedLong](#) ulOption01, [xsd__unsignedLong](#) ulOption02, struct [MXCommon__unsignedLongResponse](#) *Response)

Reads the status of the synchronization for the corresponding MSXE The master mode (when enabled) make the system always detected as master.

- int [MXCommon__SetFilterChannels](#) (struct [xsd__base64Binary](#) *ChannelList, struct [MXCommon__Response](#) *Response)

This function sets or resets a filter to a channel.

- int [MSXE173x__DigitalIOGetNumber](#) (void ___, struct [MSXE173x__DigitalIOGetNumberResponse](#) *Response)
- int [MSXE173x__DigitalIOInitPortConfiguration](#) ([xsd__unsignedLong](#) ulPort, [xsd__unsignedLong](#) ulPortConfiguration, struct [MSXE173x__Response](#) *Response)

Initialise a digital i/o port (2 channels).

- int [MSXE173x__DigitalIOReadChannelValue](#) ([xsd__unsignedLong](#) ulChannel, struct [MSXE173x__unsignedLongResponse](#) *Response)

Read a digital i/o channel value.

- int [MSXE173x__DigitalIOReadAllChannelsValue](#) (void ___, struct [MSXE173x__unsignedLongResponse](#) *Response)

Read all digital i/o channels value.If channel is configured as output, then this function return the status of the output.

- int [MSXE173x__DigitalIOWriteChannelValue](#) ([xsd__unsignedLong](#) ulChannel, [xsd__unsignedLong](#) ulChannelValue, struct [MSXE173x__Response](#) *Response)

write a digital i/o channel value

- int [MSXE173x__DigitalIOWriteAllChannelsValue](#) ([xsd__unsignedLong](#) ulChannelValue, struct [MSXE173x__Response](#) *Response)

write all digital i/o channels value

- int [MSXE173x__DigitalIOReleasePortConfiguration](#) ([xsd__unsignedLong](#) ulPort, struct [MSXE173x__Response](#) *Response)

Release a digital i/o port (2 channels).

- int [MSXE173x__DigitalIOTestShortCircuit](#) ([xsd__unsignedLong](#) ulOption, struct [MSXE173x__unsignedLongResponse](#) *Response)

Test short circuit status.

- int [MSXE173x__DigitalIORearmShortCircuit](#) ([xsd__unsignedLong](#) ulOption, struct [MSXE173x__Response](#) *Response)

Rearm digital outputs after a short circuit happened.

- int [MSXE173x__IOWatchdogInitAndStart](#) ([xsd__unsignedLong](#) ulTimeBase, [xsd__unsignedLong](#) ulTimeValue, [xsd__unsignedLong](#) ulOption1, [xsd__unsignedLong](#) ulOption2, struct [MSXE173x__Response](#) *Response)

Init and start the digital output IO watchdog.

- `int MSXE173x__IOWatchdogStopAndRelease (xsd__unsignedLong ulOption, struct MSXE173x__Response *Response)`
Stop and release the digital output watchdog.
- `int MSXE173x__IOWatchdogGetStatusAndValue (xsd__unsignedLong ulOption, struct MSXE173x__IOWatchdogGetStatusAndValueResponse *Response)`
Get watchdog current status and value information.
- `int MSXE173x__MFCommonSetInputsFilter (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulInputAFilterValue, xsd__unsignedLong ulInputBFilterValue, xsd__unsignedLong ulInputCFilterValue, xsd__unsignedLong ulInputDFilterValue, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response *Response)`
Set a filter to the input of a multifunction sub module.
- `int MSXE173x__MFCommonReferenceVoltageActivation (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulActivationFlag, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MSXE173x__Response *Response)`
Permit to activate the reference voltage to pin D-.
- `int MSXE173x__MFEndatInitSensor (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulFrequency, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response *Response)`
Initialises an EnDat sensor.
- `int MSXE173x__MFEndatGetPosition (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetPositionResponse *Response)`
Reads the position of the sensor.
- `int MSXE173x__MFEndatGetSensorProperties (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetSensorPropertiesResponse *Response)`
Reads the properties of the sensor.
- `int MSXE173x__MFEndatGetErrorSources (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetErrorSourcesResponse *Response)`
Reads the error sources.
- `int MSXE173x__MFEndatResetErrorBits (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response *Response)`
Resets the error bits.

- int [MSXE173x__MFEndatSensorReceiveReset](#) (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct [MSXE173x__Response](#) *Response)

Resets the sensor.

- int [MSXE173x__MFEndatSelectMemoryArea](#) (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct [MSXE173x__Response](#) *Response)

Selects a memory area (see page 31/121 of EnDat specifications).

- int [MSXE173x__MFEndatSensorSendParameter](#) (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct [MSXE173x__MFEndatSensorSendParameterResponse](#) *Response)

Reads a parameter from the memory area that was last selected.

- int [MSXE173x__MFEndatSensorReceiveParameter](#) (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulParam, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct [MSXE173x__Response](#) *Response)

Writes a parameter to the memory area that was last selected.

- int [MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea](#) (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct [MSXE173x__MFEndatSensorSendPosAndRecvSelMemAreaResponse](#) *Response)

Reads the position value of the sensor and selects a memory area in the same cycle.

- int [MSXE173x__MFEndatSelectAdditionalData](#) (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulMFChannelIndex, xsd__unsignedLong ulAddDataCount, xsd__unsignedLong ulMrsCodeAD1, xsd__unsignedLong ulMrsCodeAD2, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct [MSXE173x__Response](#) *Response)

Selects the additional data that will be sent by the sensor.

- int [MSXE173x__MFEndatGetPositionWithAddData](#) (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct [MSXE173x__MFEndatGetPositionWithAddDataResponse](#) *Response)

Reads the position of the sensor with additional data.

- int [MSXE173x__MFEndatGetSelectedAdditionalData](#) (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct [MSXE173x__MFEndatGetSelectedAdditionalDataResponse](#) *Response)

Reads the currently selected additional data of the sensor.

- `int MSXE173x__MFEndatInitAndEnableLatchPositionValues (xsd__unsignedLong ulmFModuleIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulLatchSource, xsd__unsignedLong ulTriggerEdgeCount, xsd__unsignedLong ulDataFormat, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response *Response)`

Initialises and enables the latch logic to get the position and additional informations from the EnDat sensor using a trigger source.

- `int MSXE173x__MFEndatGetCurrentLatchConfiguration (xsd__unsignedLong ulmFModuleIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetCurrentLatchConfigurationResponse *Response)`

Reads the current latch configuration, started using the function MSXE173x__MFEndatInitAndEnableLatchPositionValues.

- `int MSXE173x__MFEndatDisableAndReleaseLatchPositionValues (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response *Response)`

Disables and releases the latch logic started with the function MSXE173x__MFEndatInitAndEnableLatchPositionValues.

- `int MSXE17xx__DigitalIOGetNumber (void *_ , struct MSXE17xx__DigitalIOGetNumberResponse *Response)`

Returns the number of digital IO available on the module.

- `int MSXE17xx__DigitalIOInitPortConfiguration (xsd__unsignedLong ulPort, xsd__unsignedLong ulPortConfiguration, struct MSXE17xx__Response *Response)`

Initialise a digital i/o port (2 channels).

- `int MSXE17xx__DigitalIOReadChannelValue (xsd__unsignedLong ulChannel, struct MSXE17xx__unsignedLongResponse *Response)`

Read a digital i/o channel value.

- `int MSXE17xx__DigitalIOReadAllChannelsValue (void *_ , struct MSXE17xx__unsignedLongResponse *Response)`

Read all digital i/o channels value. If channel is configured as output, then this function return the status of the output.

- `int MSXE17xx__DigitalIOWriteChannelValue (xsd__unsignedLong ulChannel, xsd__unsignedLong ulChannelValue, struct MSXE17xx__Response *Response)`

write a digital i/o channel value

- `int MSXE17xx__DigitalIOWriteAllChannelsValue (xsd__unsignedLong ulChannelValue, struct MSXE17xx__Response *Response)`

write all digital i/o channels value

- `int MSXE17xx__DigitalIOReleasePortConfiguration (xsd__unsignedLong ulPort, struct MSXE17xx__Response *Response)`

Release a digital i/o port (2 channels).

- `int MSXE17xx__DigitalIOTestShortCircuit (xsd__unsignedLong ulOption, struct MSXE17xx__unsignedLongResponse *Response)`
Test short circuit status.
- `int MSXE17xx__DigitalIORearmShortCircuit (xsd__unsignedLong ulOption, struct MSXE17xx__Response *Response)`
Rearm digital outputs after a short circuit happened.
- `int MSXE17xx__IOWatchdogInitAndStart (xsd__unsignedLong ulTimeBase, xsd__unsignedLong ulTimeValue, xsd__unsignedLong ulOption1, xsd__unsignedLong ulOption2, struct MSXE17xx__Response *Response)`
Init and start the digital output IO watchdog.
- `int MSXE17xx__IOWatchdogStopAndRelease (xsd__unsignedLong ulOption, struct MSXE17xx__Response *Response)`
Stop and release the digital output watchdog.
- `int MSXE17xx__IOWatchdogGetStatusAndValue (xsd__unsignedLong ulOption, struct MSXE17xx__IOWatchdogGetStatusAndValueResponse *Response)`
Get watchdog current status and value information.
- `int MSXE17xx__MFCommonGetSubModuleFunctionality (xsd__unsignedLong ulMFModuleIndex, struct MSXE17xx__unsignedLongResponse *Response)`
Get the selected sub module functionality.
- `int MSXE17xx__MFCommonSetInputsFilter (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulInputAFilterValue, xsd__unsignedLong ulInputBFilterValue, xsd__unsignedLong ulInputCFilterValue, xsd__unsignedLong ulInputDFilterValue, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__Response *Response)`
Set a filter to the input of a multifunction sub module.
- `int MSXE17xx__MFCommonReferenceVoltageActivation (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulActivationFlag, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MSXE17xx__Response *Response)`
Permit to activate the reference voltage to pin D-.
- `int MSXE17xx__MFCommonSetFIFO0Level (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulFIFOLevel, xsd__unsignedLong ulTimeOutTimeBase, xsd__unsignedLong ulReloadValue, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MSXE17xx__Response *Response)`
Define the number of data bloc in the first FIFO before transmit the datas.
- `int MSXE17xx__MFEndatInitSensor (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulFrequency, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__Response *Response)`
Initialises an EnDat sensor.

- `int MSXE17xx_MFEndatGetPosition (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE17xx_MFEndatGetPositionResponse *Response)`

Reads the position of the sensor.

- `int MSXE17xx_MFEndatGetSensorProperties (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE17xx_MFEndatGetSensorPropertiesResponse *Response)`

Reads the properties of the sensor.

- `int MSXE17xx_MFEndatGetErrorSources (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE17xx_MFEndatGetErrorSourcesResponse *Response)`

Reads the error sources.

- `int MSXE17xx_MFEndatResetErrorBits (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE17xx_Response *Response)`

Resets the error bits.

- `int MSXE17xx_MFEndatSensorReceiveReset (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE17xx_Response *Response)`

Resets the sensor.

- `int MSXE17xx_MFEndatSelectMemoryArea (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulMrsCode, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE17xx_Response *Response)`

Selects a memory area (see page 31/121 of EnDat specifications).

- `int MSXE17xx_MFEndatSensorSendParameter (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulMrsCode, xsd_unsignedLong ulAddress, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE17xx_MFEndatSensorSendParameterResponse *Response)`

Reads a parameter from the memory area that was last selected.

- `int MSXE17xx_MFEndatSensorReceiveParameter (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulMrsCode, xsd_unsignedLong ulAddress, xsd_unsignedLong ulParam, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE17xx_Response *Response)`

Writes a parameter to the memory area that was last selected.

- `int MSXE17xx__MFEndatSensorSendPosAndRecvSelMemArea (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__MFEndatSensorSendPosAndRecvSelMemAreaResponse *Response)`

Reads the position value of the sensor and selects a memory area in the same cycle.

- `int MSXE17xx__MFEndatSelectAdditionalData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulMFChannelIndex, xsd__unsignedLong ulAddDataCount, xsd__unsignedLong ulMrsCodeAD1, xsd__unsignedLong ulMrsCodeAD2, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__Response *Response)`

Selects the additional data that will be sent by the sensor.

- `int MSXE17xx__MFEndatGetPositionWithAddData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__MFEndatGetPositionWithAddDataResponse *Response)`

Reads the position of the sensor with additional data.

- `int MSXE17xx__MFEndatGetSelectedAdditionalData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__MFEndatGetSelectedAdditionalDataResponse *Response)`

Reads the currently selected additional data of the sensor.

- `int MSXE17xx__MFEndatInitAndEnableLatchPositionValues (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulLatchSource, xsd__unsignedLong ulTriggerEdgeCount, xsd__unsignedLong ulDataFormat, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__Response *Response)`

Initialises and enables the latch logic to get the position and additional informations from the EnDat sensor using a trigger source.

- `int MSXE17xx__MFEndatGetCurrentLatchConfiguration (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse *Response)`

Reads the current latch configuration, started using the function MSXE17xx__MFEndatInitAndEnableLatchPositionValues.

- `int MSXE17xx__MFEndatDisableAndReleaseLatchPositionValues (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__Response *Response)`

Disables and releases the latch logic started with the function MSXE17xx__MFEndatInitAndEnableLatchPositionValues.

4.1.1 Define Documentation

4.1.1.1 #define MSXE170X_COUNTER_QUADRUPLE_MODE 0x4

4.1.1.2 #define MSXE170X_COUNTER_DOUBLE_MODE 0x2

4.1.1.3 #define MSXE170X_COUNTER_SIMPLE_MODE 0x1

4.1.1.4 #define MSXE170X_COUNTER_DIRECT_MODE 0x0

The inputs A and B in 32-Bit mode or A, B and C,D in 16-Bit mode present, each, one clock pulse gate circuit. Thereby frequency and pulse duration measurements can be done.

4.1.1.5 #define MSXE170X_COUNTER_HYSTERESIS_ON 0x1

It suppresses the first counting pulse after a change of rotation.

4.1.1.6 **#define MSXE170X_COUNTER_HYSTERESIS_OFF 0x0**

4.1.1.7 **#define MSXE170X_COUNTER_INCREMENT 0x0**

4.1.1.8 **#define MSXE170X_COUNTER_DECREMENT 0x1**

4.1.1.9 **#define MSXE170X_COUNTER_LOW_EDGE_LATCH_AND_CLEAR_COUNTER 0x0**

4.1.1.10 **#define MSXE170X_COUNTER_HIGH_EDGE_LATCH_AND_CLEAR_COUNTER 0x1**

4.1.1.11 **#define MSXE170X_COUNTER_LOW_EDGE_LATCH_COUNTER 0x2**

4.1.1.12 **#define MSXE170X_COUNTER_HIGH_EDGE_LATCH_COUNTER 0x3**

4.1.2 Typedef Documentation

4.1.2.1 **typedef char* xsd__string**

4.1.2.2 **typedef char xsd__char**

4.1.2.3 **typedef float xsd__float**

4.1.2.4 **typedef double xsd__double**

4.1.2.5 **typedef int xsd__int**

4.1.2.6 **typedef long xsd__long**

4.1.2.7 **typedef unsigned char xsd__unsignedByte**

4.1.2.8 **typedef unsigned int xsd__unsignedInt**

4.1.2.9 **typedef unsigned short int xsd__unsignedShort**

4.1.2.10 **typedef unsigned long xsd__unsignedLong**

4.1.3 Function Documentation

4.1.3.1 **int MXCommon__GetModuleType (void * _, struct MXCommon__ByteArrayResponse * *Response*)**

Parameters

- [in] _ : no input parameter
- [out] *Response* • sArray : Module type string
• sResponse Composed of iReturnValue and syserrno

Return values

- SOAP_OK* SOAP call success
- otherwise* SOAP protocol error

4.1.3.2 int MXCommon__GetHostname (void * __, struct MXCommon__ByteArrayResponse * Response)

Parameters

- [in] *__* : no input parameter
- [out] **Response** • sArray : Hostname of the module
- iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
 - syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

4.1.3.3 int MXCommon__SetHostname (struct xsd__base64Binary * bHostname, struct MXCommon__Response * Response)

Parameters

- [in] *bHostname* : Hostname
- [out] **Response** • iReturnValue : Return value
- 0 : success
 - -1: system error (see syserrno)
 - syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

4.1.3.4 int MXCommon__GetClientConnections (void * __, struct MXCommon__ByteArrayResponse * Response)

Parameters

- [in] *__* : no input parameter
- [out] **Response** • sArray : string containing the list of connected clients.
- sResponse Composed of iReturnValue and syserrno

The sArray string is of the form IP-Address:first connection-second connection---- IP-Address:first connection-second connection----

Sample: 172.16.3.43:8989-5555 172.16.3.200:8989

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

4.1.3.5 int MXCommon__Strerror (xsd__int *errnum*, struct MXCommon__ByteArrayResponse * *Response*)

Usually SOAP functions return this value in a variable named syserror, which is meaningful only when the function return value, usually called iReturnValue, indicate an error (that is, have a value of -1 or -100, depending of the case).

Parameters

[in] *errnum* : Error number

[out] *Response* • sArray : See the description below.

- sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno).
- sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

STRERROR(3)
STRERROR(3)

Linux Programmer's Manual

NAME

strerror, strerror_r - return string describing error code

SYNOPSIS

```
#include <string.h>
```

```
char *strerror(int errnum);
```

```
#define _XOPEN_SOURCE 600
#include <string.h>
```

```
int strerror_r(int errnum, char *buf, size_t n);
```

DESCRIPTION

The `strerror()` function returns a string describing the error code passed in the argument `errnum`, possibly using the `LC_MESSAGES` part of the current locale to select the appropriate language.

This string must not be modified by the application, but may be modified by a subsequent call to `perror()` or `strerror()`. No library function will modify this string.

The `strerror_r()` function is similar to `strerror()`, but is thread safe. It returns the string in the user-supplied buffer `buf` of length `n`.

RETURN VALUE

The `strerror()` function returns the appropriate error description string, or an unknown error message if the error code is unknown.

The value of `errno` is not changed for a successful call, and is set to a non-zero value upon error.

The `strerror_r()` function returns 0 on success and -1 on failure, setting `errno`.

ERRORS

EINVAL The value of `errnum` is not a valid error number.

ERANGE Insufficient storage was supplied to contain the error description string.

CONFORMING TO

SVID 3, POSIX, 4.3BSD, ISO/IEC 9899:1990 (C89).

`strerror_r()` with prototype as given above is specified by SUSv3, and was in use under Digital Unix and HP Unix. An incompatible function, with prototype

```
char *strerror_r(int errnum, char *buf, size_t n);
```

is a GNU extension used by glibc (since 2.0), and must be regarded as obsolete in view of SUSv3.
 The GNU version may, but need not, use the user-supplied buffer.
 If it does, the result may be truncated in case the supplied buffer is too small.
 The result is always NUL-terminated.

SEE ALSO
 errno(3), perror(3), strsignal(3)

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

4.1.3.6 int MXCommon__Reboot (void * _, struct MXCommon__Response * *Response*)

Parameters

[in] **_** : no input parameter
 [out] **Response** • **iReturnValue** : Return value
 – 0 : success
 – -1 : system error (see syserrno)
 • **syserrno** : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

4.1.3.7 int MXCommon__ResetAllIOFunctionalities (xsd__unsignedLong *ulOption*, struct MXCommon__Response * *Response*)

The behavior of the function depends on the MSX-E system that is used.

On MSX-E3511: Stop the watchdogs and stop the generators
 On MSX-E3601: Stop the sequence acquisition and stop the calibration
 On MSX-E3701: Stop the acquisition

Parameters

[in] **ulOption** Reserved. Set to 0
 [out] **Response** **iReturnValue**
 • **0** The remote function performed OK
 • **-1** Internal system error occurred. See value of syserrno
 • **-100** Function not supported by the system
 syserrno system error code (the value of the libc "errno" code)

Return values

0 SOAP_OK
Others See SOAP error

4.1.3.8 int MXCommon__DataserverRestart (xsd__unsignedLong ulAction, xsd__unsignedLong ulOption, struct MXCommon__Response * Response)

Parameters

- [in] **ulAction** : action
- 0: normal restart
 - 1: with cache file reset
 - 2: with cache file deletion
- [in] **ulOption** : Reserved
- [out] **Response** • iReturnValue : Return value
- 0 : success
 - -1: system error (see syserrno)
 - syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

Note

(revision>6386) Depending on the system type, can be used to restart the data-recv service as well. In this case, parameter action is ignored.

4.1.3.9 int MXCommon__GetEthernetLinksStates (void * _, struct MXCommon__GetEthernetLinksStatesResponse * Response)

Parameters

- [in] **_** : no input parameter
- [out] **Response** Structure that contains the MSX-E Ethernet links states and errors:
- sResponse.iReturnValue**
- **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** Fail to get Ethernet links states
 - **-100** Internal system error occurred. See value of syserrno
- sResponse.syserrno** system error code (the value of the libc "errno" code)
- sPort0: Fisrt port informations**
- **ulState**
 - **0** Link down
 - **1** Link up
 - **ulSpeed**
 - **10** 10 Mb/s
 - **100** 100 Mb/s
 - **ulDuplex**
 - **0** Half duplex
 - **1** Full duplex

- **ulInfo1** Reserved
- **ulInfo2** Reserved

sPort1: Second port informations

- **ulState**
 - **0** Link down
 - **1** Link up
- **ulSpeed**
 - **10** 10 Mb/s
 - **100** 100 Mb/s
- **ulDuplex**
 - **0** Half duplex
 - **1** Full duplex
- **ulInfo1** Reserved
- **ulInfo2** Reserved

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.10 int MXCommon__GetModuleTemperatureValueAndStatus (xsd__unsignedLong ulOption, struct MXCommon__GetModuleTemperatureValueAndStatusResponse * Response)

Parameters

[in] *ulOption* : Reserved

[out] *Response* • sResponse.iReturnValue : Return value

– **0** : success

– **-1**: system error (see syserrno)

- sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).

– dValue : Temperature value in Degree Celsius

- ulTemperatureStatus : Temperature Status :

– TEMPERATURE_INITIAL = **0** : Temperature not ready

– TEMPERATURE_TOOLOW = **1** : Temperature too low !

– TEMPERATURE_LOW = **2** : Temperature under the min warning value

– TEMPERATURE_NOMINAL = **3** : Temperature in the nominal range

– TEMPERATURE_HIGH = **4** : Temperature over the max warning value

– TEMPERATURE_TOOHIGH = **5** : Temperature too high !

- ulInfo : Reserved

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

4.1.3.11 `int MXCommon__SetModuleTemperatureWarningLevels (xsd__double dMinimalWarningLevel, xsd__double dMaximalWarningLevel, xsd__unsignedLong ulOption, struct MXCommon__Response * Response)`

Parameters

- [in] *dMinimalWarningLevel* : Minimal temperature warning level in Degree : 5 to 60 Degree Celsius
- [in] *dMaximalWarningLevel* : Maximal temperature warning level in Degree : 5 to 60 Degree Celsius
- [in] *ulOption* : Reserved
- [out] *Response*
 - sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

Return values

- SOAP_OK* SOAP call success
- otherwise* SOAP protocol error

4.1.3.12 `int MXCommon__SetHardwareTriggerFilterTime (xsd__unsignedLong ulFilterTime, xsd__unsignedLong ulOption, struct MXCommon__Response * Response)`

Sets the filter time for the hardware trigger input in steps of 250 ns (max value: 65535).

On the MSX-E3011 system, the step of the hardware trigger filter is **622ns**.

Parameters

- [in] *ulFilterTime* Filter time for the hardware trigger input in steps of 250ns (max value : 65535).
 - **0**: Disable the filter
 - **1**: Sets the filter time to 250 ns
 - **2**: Sets the filter time to 500 ns
 - ...
 - **65535**: Sets the filter time to 16 ms
- [in] *ulOption* Reserved. Set to 0
- [out] *Response* Response of the system
 - *sResponse.iReturnValue*
 - **0**: The remote function performed OK
 - **-1**: Internal system error occurred. See value of syserrno
 - *sResponse.syserrno* system error code (the value of the libc "errno" code)

Return values

- 0** SOAP_OK
- Others* See SOAP error

4.1.3.13 `int MXCommon__GetHardwareTriggerFilterTime (xsd__unsignedLong ulOption, struct MXCommon__GetHardwareTriggerFilterTimeResponse * Response)`

Get the filter time for the hardware trigger input in **250ns** step (max value : 65535).

On the MSX-E3011 system, the step of the hardware trigger filter is **622ns**.

Parameters

[in] *ulOption* Reserved. Set to 0

[out] *Response* Response of the system

- *ulFilterTime* filter time for the hardware trigger input
 - 0: filter disabled
 - 1: filter of 250ns
 - 2: filter of 500ns
 - ...
 - 65535: filter of 16ms
- *sResponse.iReturnValue*
 - 0: The remote function performed OK
 - -1: Internal system error occurred. See value of syserrno
- *sResponse.syserrno* system error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.14 `int MXCommon__GetHardwareTriggerState (xsd__unsignedLong ulOption, struct MXCommon__GetHardwareTriggerStateResponse * Response)`

Parameters

[in] *ulOption* : Reserved

[out] *Response* • *ulState* : Hardware trigger input state.

- 0: Hardware trigger input is low
- 1: Hardware trigger input is high.
- *sResponse.iReturnValue* : Return value
 - 0 : success
 - -1: system error (see syserrno)
- *sResponse.syserrno* : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

4.1.3.15 int MXCommon__SetCustomerKey (struct xsd__base64Binary * *bKey*, struct xsd__base64Binary * *bPublicKey*, struct MXCommon__Response * *Response*)

Parameters

- [in] *bKey* : Customer key (only writable on the module) [32 bytes containing a AES key]
- [in] *bPublicKey* : IV (Initialisation vector) for the AES cryptography [16 bytes containing a AES key]
- [out] *Response* • sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
- sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

4.1.3.16 int MXCommon__TestCustomerID (void * _, struct MXCommon__TestCustomerIDResponse * *Response*)

Parameters

- [in] _ : No Input
- [out] *Response* • sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
- sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).
- bValueArray : non encrypted value array [16 bytes of random data]
- bCryptedValueArray : Encrypted value array [16 bytes of the encrypted random data]

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

4.1.3.17 int MXCommon__SetTime (xsd__unsignedLong *ulLowTime*, xsd__unsignedLong *ulHighTime*, struct MXCommon__Response * *Response*)

Parameters

- [in] *ulLowTime* : Number of microseconds since the begin of the second
- [in] *ulHighTime* : Number of seconds since the Epoch (1st January,1970)
- [out] *Response* • sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
- sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

4.1.3.18 int MXCommon__SysToHardwareClock (void * _, struct MXCommon__Response * Response)**Parameters**

[in] _ No input parameter
[out] **Response** • sResponse.iReturnValue : Return value
 – 0 : success
 – -1: system error (see syserrno)
• sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

If this function fails, it means the module does not have a hardware RTC, or the hardware is not functional. Check the "hwclock" subsystem status.

4.1.3.19 int MXCommon__HardwareClockToSys (void * _, struct MXCommon__Response * Response)

When the hardware clock is present, the system time is automatically set to it when the module becomes master on the inter-module synchronisation bus.

Parameters

[in] _ No input parameter
[out] **Response** • sResponse.iReturnValue : Return value
 – 0 : success
 – -1: system error (see syserrno)
• sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

If this function fails, it means the module does not have a hardware RTC, or the hardware is not functional. Check the "hwclock" subsystem status.

4.1.3.20 int MXCommon__GetTime (void * _, struct MXCommon__GetTimeResponse * Response)

Parameters

- [in] _ : No input parameter
- [out] **Response**
- sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).
 - ulLowTime : Number of microseconds since the begin of the second
 - ulHighTime : Number of seconds since the Epoch (1st January,1970)

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

4.1.3.21 int MXCommon__GetUpTime (void * _, struct MXCommon__GetUpTimeResponse * Response)

Parameters

- [in] _ : no input parameter
- [out] **Response**
- sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).
 - ulUpTime : Number of seconds since the last boot of the system.

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

4.1.3.22 int MXCommon__GetAutoConfigurationFile (void * _, struct MXCommon__GetAutoConfigurationFileResponse * Response)

Parameters

- [in] _ : No input parameter
- [out] **Response**
- sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
 - -100 : Error of the read of the auto configuration file
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#).
 - bArray : Array of Bytes of the file

- `ulEOF` : End of file flag

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

4.1.3.23 `int MXCommon__SetAutoConfigurationFile (struct xsd__base64Binary * ByteArrayInput, xsd__unsignedLong ulEOF, struct MXCommon__Response * Response)`

Parameters

[in] *ByteArrayInput* : Array of Bytes of the file

[in] *ulEOF* : End of file flag

[out] *Response* • `sResponse.iReturnValue` : Return value

– 0 : success

– -1: system error (see `syserrno`)

- `sResponse.syserrno` : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

4.1.3.24 `int MXCommon__StartAutoConfiguration (void * _, struct MXCommon__ByteArrayResponse * Response)`

Parameters

[in] `_` : No input parameter

[out] *Response* • `sResponse.iReturnValue` : Return value

– 0 : success

– -1: system error (see `syserrno`)

- `sResponse.syserrno` : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).
- `sArray` : message returned by the auto configuration start

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

4.1.3.25 `int MXCommon__InitAndStartSynchroTimer (xsd__unsignedLong ulTimeBase, xsd__unsignedLong ulReloadValue, xsd__unsignedLong ulNbrOfCycle, xsd__unsignedLong ulGenerateTriggerMode, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MXCommon__Response * Response)`

Parameters

[in] *ulTimeBase* : Time base of the timer (0 for us, 1 for ms, 2 for s)

- [in] ***ulReloadValue*** : Timer reload value (0 to 0xFFFF), minimum reload time is 5 us
- [in] ***ulNbrOfCycle*** : Number of timer cycle
 - 0: continuous
 - > 0: defined number of cycle
- [in] ***ulGenerateTriggerMode*** :
 - 0: Wait the time overflow to set the synchronisation trigger
 - 1: Set the synchronisation trigger by the start of the timer and after each time overflow
- [in] ***ulOption01*** : Define the source of the trigger
 - 0 : Trigger disabled
 - 1 : Enable the hardware digital input trigger
- [in] ***ulOption02*** : Define the edge of the hardware trigger who generates a trigger action
 - 1 : rising edge (Only if hardware trigger selected)
 - 2 : falling edge (Only if hardware trigger selected)
 - 3 : Both front (Only if hardware trigger selected)
- [in] ***ulOption03*** : Define the number of trigger events before the action occur
 - 1 : all trigger event start the action
 - max value : 65535
- [in] ***ulOption04*** : Reserved
- [out] ***Response***
 - sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
 - -2: not available time base
 - -3: timer reload value can not be greater than 65535
 - -4: minimum time reload is 5 us
 - -5: Number of cycle can not be greater than 65535
 - -6: Generate trigger mode error
 - -100: Init timer error
 - -101: Start timer error
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Strerror\(\)](#). May be ENOSYS : Function not implemented.

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

4.1.3.26 int MXCommon__StopAndReleaseSynchroTimer (xsd__unsignedLong ulOption01, struct MXCommon__Response * Response)

Parameters

- [in] ***ulOption01*** : Reserved
- [out] ***Response***
 - sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
 - -100: Start/Stop timer error

- `sResponse.syserrno` : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#). May be `ENOSYS` : Function not implemented.

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

4.1.3.27 `int MXCommon__GetConfigurationBackupFile (void * _, struct MXCommon__FileResponse * Response)`

Parameters

- [in] `_` : No input parameter
- [out] ***Response*** • `sResponse.iReturnValue` : Return value
- 0 : success
 - -1: system error (see `syserrno`) (see `syserrno`)
 - `sResponse.syserrno` : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).
 - `bArray` : Array of Bytes of the file
 - `ulEOF` : End of file flag

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

This function is designed to be called repeatedly until no more data is available. At this point the flag `ulEOF` is set.

Below is an example in pseudo-C.

```
int dummy;
struct MXCommon__FileResponse Response;
while(1)
{
    if ( MXCommon__GetConfigurationBackupFile(&dummy, &Response) != SOAP_OK)
    {
        // handle soap error
    }
    if (Response.iReturnValue)
    {
        // handle remote error (Response.syserrno contains more information)
    }
    // do something with the data, for example save it in a file
    write(fd, Response.bArray.__ptr, Response.bArray.__size);
    // if this is the end of the file, quit the loop
    if(Response.ulEOF)
        break;
}
*
```

4.1.3.28 `int MXCommon__ApplyConfigurationBackupFile (struct xsd__base64Binary * ByteArrayInput, xsd__unsignedLong ulEOF, struct MXCommon__Response * Response)`

Parameters

- [in] *ByteArrayInput* : Array of Bytes of the file
- [in] *ulEOF* : End of file flag
- [out] *Response*
 - sResponse.iReturnValue : Return value
 - 0 : success
 - -1: system error (see syserrno)
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

This function is designed to be called repeatedly until all data is transfered. At this point the flag ulEOF must be set to 1. The new configuration is then applied.

4.1.3.29 `int MXCommon__ChangePassword (struct xsd__base64Binary * PreviousUser, struct xsd__base64Binary * PreviousPassword, struct xsd__base64Binary * NewUser, struct xsd__base64Binary * NewPassword, struct MXCommon__Response * Response)`

The changes are immediately active.

Parameters

- [in] *_* : No input parameter
- [out] *Response*
 - sResponse.iReturnValue : Return value
 - 0 : success
 - -1: string PreviousUser is invalid
 - -2: string PreviousPassword is invalid
 - -3: string NewUser is invalid
 - -4: string NewPassword is invalid
 - -5: authentication failed
 - -100: system error while saving tokens (use syserrno for more information)
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).
 - sArray : message returned by the auto configuration start

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

Warning

The parameters transit in clear text. Use this functionality only on trusted networks.
Given that ADDI-DATA GmbH takes security seriously, there is no way to change the password without knowing it. No "hidden back-door". This function makes it all too easy to lock a module, if you don't remember the password you set on it.

4.1.3.30 `int MXCommon__GetSubSystemState (xsd__unsignedLong SubsystemID, struct MXCommon__unsignedLongResponse * Response)`

Parameters

- [in] *SubsystemID* sub-system numerical ID
- [out] *Response* • sResponse.iReturnValue : Return value
- 0 : success
 - -1: system error while executing the request (see syserrno)
 - -2: invalid parameter SubsystemID
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).
 - Value The state of the sub-system "Id" at the moment of the execution of the request.

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

4.1.3.31 `int MXCommon__GetSubsystemIDFromName (struct xsd__base64Binary * SubsystemName, struct MXCommon__unsignedLongResponse * Response)`

Parameters

- [in] *SubsystemName* sub-system symbolic name.
- [out] *Response* • sResponse.iReturnValue :Return value
- 0 : success
 - -1: system error while executing the request (see syserrno)
 - -2: invalid parameter SubsystemName
 - sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).
 - Value The numerical ID of the sub-system "SubsystemName".

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

4.1.3.32 `int MXCommon__GetStateIDFromName (xsd__unsignedLong SubsystemID, struct xsd__base64Binary * StateName, struct MXCommon__unsignedLongResponse * Response)`

Parameters

- [in] *SubsystemID* sub-system numerical ID
- [in] *StateName* state symbolic name.
- [out] *Response* • sResponse.iReturnValue : Return value
- 0 : success
 - -1: system error while executing the request (see syserrno)
 - -2: invalid parameters SubsystemID or StateName

- sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).
- Value The numerical ID of the state "StateName".

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

4.1.3.33 int MXCommon__GetSubsystemNameFromID (xsd__unsignedLong SubsystemID, struct MXCommon__ByteArrayResponse * Response)

Parameters

- [in] *SubsystemID* sub-system numerical ID.
- [out] *Response* • sResponse.iReturnValue : Return value
- 0 : success
 - -1: system error while executing the request (see syserrno)
 - -2: invalid parameter SubsystemName
- sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).
 - sArray : The symbolic name associated with the ID.

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

4.1.3.34 int MXCommon__GetStateNameFromID (xsd__unsignedLong SubsystemID, xsd__unsignedLong StateID, struct MXCommon__ByteArrayResponse * Response)

Parameters

- [in] *SubsystemID* sub-system numerical ID.
- [in] *StateID* sub-system numerical ID.
- [out] *Response* • sResponse.iReturnValue : Return value
- 0 success
 - -1 system error while executing the request (see syserrno)
 - -2 invalid parameters SubsystemID or StateID
- sResponse.syserrno : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).
 - sArray The symbolic name associated with the state numerical ID.

Return values

SOAP_OK SOAP call success
otherwise SOAP protocol error

4.1.3.35 `int MXCommon__GetOptionInformation (void * _, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MXCommon__ByteArrayResponse * Response)`

Parameters

- [in] *ulOption01*,: not used, set it to 0
- [in] *ulOption02*,: not used, set it to 0
- [out] *Response*
 - sArray : Option information string
 - sResponse Composed of iReturnValue and syserrno

Return values

- SOAP_OK* SOAP call success
- otherwise* SOAP protocol error

4.1.3.36 `int MXCommon__SetToMaster (void * _, xsd__unsignedLong ulState, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MXCommon__Response * Response)`

Parameters

- [in] *ulState* State of the supermaster mode
 - **0** automatic mode (default). The state of the system (master or slave) will be automatically detected by the system
 - **1** Set to master mode at all time. The system will always be detected as master
- [in] *ulOption01* Reserved. Set to 0
- [in] *ulOption02* Reserved. Set to 0
- [out] *Response iReturnValue*
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The ulFilterTime parameter is wrong
 - **-100** Internal system error occurred. See value of syserrno *syserrno* system error code (the value of the libc "errno" code)

Return values

- 0** SOAP_OK
- Others* See SOAP error

4.1.3.37 `int MXCommon__GetSynchronizationStatus (void * _, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MXCommon__unsignedLongResponse * Response)`

Parameters

- [in] *ulOption01* Reserved. Set to 0
- [in] *ulOption02* Reserved. Set to 0
- [out] *Response sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-100** Internal system error occurred. See value of `syserrno`

sResponse.syserrno system error code (the value of the libc "errno" code)

ulValue State of the supermaster mode

- **0** Automatic mode (default). The state of the system (master or slave) will be automatically detected by the system
- **1** MSXE is always set as a master. The system will always be detected as master

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.38 int MXCommon_SetFilterChannels (struct xsd__base64Binary * ChannelList, struct MXCommon_Response * Response)

Parameters

[in] **ChannelList** Each index of the array represents a channel. A filter can be affected to each channel. If FilterID = 0, no filter is set (the filter is disabled on the corresponding channel). e.g.:
ChannelList[0] = FilterID // Set FilterID on channel 0.

[out] **Response**

- *sResponse.iReturnValue* : Return value
 - 0 : success
 - -1: system error (see `syserrno`)
- *sResponse.syserrno* : System-error code. The value of the libc "errno" code, see [MXCommon__Sterror\(\)](#).

Return values

SOAP_OK SOAP call success

otherwise SOAP protocol error

4.1.3.39 int MSXE173x_DigitalIOGetNumber (void * _, struct MSXE173x_DigitalIOGetNumberResponse * Response)

4.1.3.40 int MSXE173x_DigitalIOInitPortConfiguration (xsd__unsignedLong ulPort, xsd__unsignedLong ulPortConfiguration, struct MSXE173x_Response * Response)

Parameters

[in] **ulPort** : Index of the digital i/o port (0 to 7)

[in] **ulPortConfiguration** : Define the port configuration

- 0 : input
- 1 : output

[out] **Response** :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -2: Digital i/o port selection error
- -3: Port configuration selection error
- -100: Init dig i/o port kernel function error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.41 int MSXE173x__DigitalIOReadChannelValue (xsd__unsignedLong ulChannel, struct MSXE173x__unsignedLongResponse * Response)

Parameters

[in] **ulChannel** : Index of the digital i/o channel (0 to 15)

[out] **Response** :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -2: Digital i/o channel selection error
- -100: Read dig i/o channel value kernel function error

syserrno : system-error code (the value of the libc "errno" code) **ulValue** : i/o channel value:

- 0
- 1

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.42 int MSXE173x__DigitalIOReadAllChannelsValue (void * __, struct MSXE173x__unsignedLongResponse * Response)

Parameters

[in] **__** : no input parameter

[out] **Response** :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -100: Read dig i/o channel value kernel function error

syserrno : system-error code (the value of the libc "errno" code) **ulValue** : i/o channels value(each bit correspond to one channel)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.43 `int MSXE173x__DigitalIOWriteChannelValue (xsd__unsignedLong ulChannel, xsd__unsignedLong ulChannelValue, struct MSXE173x__Response * Response)`

Parameters

- [in] *ulChannel* : Index of the digital i/o channel (0 to 15)
- [in] *ulChannelValue* : Channel value
- 0
 - 1
- [out] *Response* :
- iReturnValue* :
- 0: means the remote function performed OK
 - -1: means an system error occurred
 - -2: Digital i/o channel selection error
 - -3: Channel value error
 - -100: Write dig i/o channel value kernel function error
- syserrno* : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.44 `int MSXE173x__DigitalIOWriteAllChannelsValue (xsd__unsignedLong ulChannelValue, struct MSXE173x__Response * Response)`

Parameters

- [in] *ulChannelValue* : Channels value (each bit corresponds to a channel)
- [out] *Response* :
- iReturnValue* :
- 0: means the remote function performed OK
 - -1: means an system error occurred
 - -2: Channels value error
 - -100: Write dig i/o channel value kernel function error
- syserrno* : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.45 `int MSXE173x__DigitalIOReleasePortConfiguration (xsd__unsignedLong ulPort, struct MSXE173x__Response * Response)`

Parameters

- [in] *ulPort* : Index of the digital i/o port (0 to 7)

[out] **Response** :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occured
- -2: Digital i/o port selection error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.46 int MSXE173x__DigitalIOTestShortCircuit (xsd__unsignedLong ulOption, struct MSXE173x__unsignedLongResponse * Response)

Parameters

[in] **ulOption** : reserved

[out] **Response** :

iReturnValue :

- 0 : means the remote function performed OK
- -1: means an system error occured

syserrno : system-error code (the value of the libc "errno" code)

ulValue : short circuit status: from 0 to 0xffff, one bit for each output

- 0 : no short circuit
- 1 : short circuit

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.47 int MSXE173x__DigitalIORearmShortCircuit (xsd__unsignedLong ulOption, struct MSXE173x__Response * Response)

Parameters

[in] **ulOption** : reserved

[out] **Response** :

iReturnValue :

- 0 : means the remote function performed OK
- -1: means an system error occured

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.48 `int MSXE173x_IOWatchdogInitAndStart (xsd__unsignedLong ulTimeBase, xsd__unsignedLong ulTimeValue, xsd__unsignedLong ulOption1, xsd__unsignedLong ulOption2, struct MSXE173x__Response * Response)`

Parameters

[in] *ulTimeBase* : Time base of the watchdog delay (0 for mus, 1 for ms, 2 for s)

[in] *ulTimeValue* : Time base of the watchdog delay (0 to 0xFFFF)

[in] *ulOption1* : Reserved

[in] *ulOption2* : Reserved

[out] *Response* :

iReturnValue :

- 0: remote function performed OK
- -1: an system error occurred
- -2: time base selection error
- -3: time value selection error
- -100: Init and start digital output watchdog kernel function error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.49 `int MSXE173x_IOWatchdogStopAndRelease (xsd__unsignedLong ulOption, struct MSXE173x__Response * Response)`

Parameters

[in] *ulOption* : reserved

[out] *Response* :

iReturnValue :

- 0: remote function performed OK
- -1: an system error occurred
- -100: Stop and release digital output watchdog kernel function error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.50 `int MSXE173x_IOWatchdogGetStatusAndValue (xsd__unsignedLong ulOption, struct MSXE173x_IOWatchdogGetStatusAndValueResponse * Response)`

Parameters

[in] *ulOption* : Reserved

[out] **Response** :

iReturnValue :

- 0: remote function performed OK
- -1: an system error occurred
- -2: channel selection error
- -100: Get diagnostic information kernel function error

ulStatus : current status information

- BIN XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX0: is stopped,
- BIN XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX1: is running,
- BIN XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX0X: is not run down
- BIN XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX1X: is run down

ulValue : current value information (0 to 0xFFFF)

ulInfo : reserved

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.51 `int MSXE173x_MFCommonSetInputsFilter (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulInputAFilterValue, xsd__unsignedLong ulInputBFilterValue, xsd__unsignedLong ulInputCFilterValue, xsd__unsignedLong ulInputDFilterValue, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response)`

Parameters

[in] **ulMFModuleIndex** : index of the multifunction sub module (0 to 3).

[in] **ulInputAFilterValue** : Filter value for input A (0 to 262143)

- 0: Filter nicht benutzt
- 1: 100 ns
- 2: 200 ns
- 3: 300 ns ...
- 262143 : 26,2143 ms

[in] **ulInputBFilterValue** : Filter value for input B (0 to 262143)

- 0: Filter nicht benutzt
- 1: 100 ns
- 2: 200 ns
- 3: 300 ns ...
- 262143 : 26,2143 ms

[in] **ulInputCFilterValue** : Filter value for input C (0 to 262143)

- 0: Filter nicht benutzt
- 1: 100 ns
- 2: 200 ns
- 3: 300 ns ...

- 262143 : 26,2143 ms

[in] ***ulInputDFilterValue*** : Filter value for input D (0 to 262143)

- 0: Filter nicht benutzt
- 1: 100 ns
- 2: 200 ns
- 3: 300 ns ...
- 262143 : 26,2143 ms

[in] ***ulOption01*** : Set it to 0

[in] ***ulOption02*** : Set it to 0

[in] ***ulOption03*** : Set it to 0

[in] ***ulOption04*** : Set it to 0

[out] ***Response*** :

iReturnValue :

- 0 : means the remote function performed OK
- -1: means an system error occured
- -2: Multifunction sub module index selection error
- -3: Input A filter value selection error
- -4: Input B filter value selection error
- -5: Input C filter value selection error
- -6: Input D filter value selection error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.52 **int** MSXE173x__MFCommonReferenceVoltageActivation (xsd__unsignedLong *ulMFModuleIndex*, xsd__unsignedLong *ulActivationFlag*, xsd__unsignedLong *ulOption01*, xsd__unsignedLong *ulOption02*, struct MSXE173x__Response * *Response*)

Parameters

[in] ***ulMFModuleIndex*** : index of the multifunction sub module (0 to 3).

[in] ***ulActivationFlag*** :

- 0: normal mode from D- (Default mode)
- 1: activate the reference voltage to pin D-

[in] ***ulOption01*** : Set it to 0

[in] ***ulOption02*** : Set it to 0

[out] ***Response*** :

iReturnValue :

- 0 : means the remote function performed OK
- -1: means an system error occured
- -2: Multifunction sub module index selection error
- -3: Activation flag selection error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.53 `int MSXE173x__MFEndatInitSensor (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulFrequency, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response)`

This function should be called once, in order to call the other EnDat functions.

Parameters

- [in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] *ulChannelIndex* Index of the channel. Set to 0
- [in] *ulFrequency* Frequency to use in kHz (500, 900, 1500, 2500, 4500)
- [in] *ulOption01* Reserved. Set to 0
- [in] *ulOption02* Reserved. Set to 0
- [in] *ulOption03* Reserved. Set to 0
- [in] *ulOption04* Reserved. Set to 0
- [out] *Response iReturnValue*
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The ulConnectorIndex parameter is wrong
 - **-4** The ulChannelIndex parameter is wrong
 - **-5** The driver is in a wrong state (must be INITIALISED or UNINITIALISED)
 - **-6** The component is not programmed as EnDat
 - **-7** Error while resetting sensor. Note: If no sensor is plugged on the selected channel, you will get this error.
 - **-8** Error while selecting memory area 0xB9
 - **-9** Error while reading alarm space (address 0x0)
 - **-10** Error while reading warning space (address 0x1)
 - **-11** Error while clearing errors (write 0 at address 0x0)
 - **-12** Error while clearing warnings (write 0 at address 0x1)
 - **-13** Error while selecting memory area 0xA1
 - **-14** Error while reading number of clock pulses for transfer of position value (address 0x0D)
 - **-15** Error while selecting memory area 0xA5
 - **-16** Error while reading EnDat command set (address 0x5)
 - **-17** Error while reading support of error messages 1 (address 0x3)
 - **-18** Error while reading support of warnings (address 0x4)
 - **-19** Error while reading EnDat ordering designation (address 0x8)
 - **-20** ulFrequency parameter is too high for current sensor

- **-21** The `ulFrequency` parameter is wrong
 - **-22** EnDat ordering designation is invalid
 - **-41** Transmission error. Please call `MSXE173x__MFEndatGetErrorSources` to get more information
 - **-100** Internal system error occurred. See value of `syserrno`
- syserrno* system error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.54 `int MSXE173x__MFEndatGetPosition (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetPositionResponse * Response)`

Before calling this function, you must call the `MSXE173x__MFEndatInitSensor` function.

Parameters

[in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] *ulChannelIndex* Index of the channel. Set to 0

[in] *ulOption01* Reserved. Set to 0

[in] *ulOption02* Reserved. Set to 0

[in] *ulOption03* Reserved. Set to 0

[in] *ulOption04* Reserved. Set to 0

[out] *Response* *sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The `ulConnectorIndex` parameter is wrong
- **-4** The `ulChannelIndex` parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** Error while reading the position
- **-41** Transmission error. Please call `MSXE173x__MFEndatGetErrorSources` to get more information
- **-100** Internal system error occurred. See value of `syserrno`

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulPositionLow Position - low bits

ulPositionHigh Position - high bits

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.55 `int MSXE173x__MFEndatGetSensorProperties (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetSensorPropertiesResponse * Response)`

Before calling this function, you must call the MSXE173x__MFEndatInitSensor function.

Parameters

[in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] **ulChannelIndex** Index of the channel. Set to 0

[in] **ulOption01** Reserved. Set to 0

[in] **ulOption02** Reserved. Set to 0

[in] **ulOption03** Reserved. Set to 0

[in] **ulOption04** Reserved. Set to 0

[out] **Response** *sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The ulConnectorIndex parameter is wrong
- **-4** The ulChannelIndex parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** Error while selecting memory area 0xA3
- **-8** Error while reading ID number (address 0x8)
- **-9** Error while reading ID number (address 0x9)
- **-10** Error while reading ID number (address 0xA)
- **-11** Error while reading Serialnumber (address 0xB)
- **-12** Error while reading Serialnumber (address 0xC)
- **-13** Error while reading Serialnumber (address 0xD)
- **-14** Error while selecting memory area 0xA1
- **-15** Error while reading encoder model (address 0xE)
- **-16** Error while reading signal period length or signal periods per revolution for incremental output signals (address 0xF)
- **-17** Error while getting the position
- **-18** Error while selecting memory area 0xA3
- **-19** Error while reading signal period length or signal periods per revolution for incremental output signals (address 0x0)
- **-20** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x4)
- **-21** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x5)
- **-22** Error while selecting memory area 0xBD
- **-23** Error while reading scaling factor for resolution (address 0x1B)
- **-24** Error while reading measuring step, or measuring steps per revolution or subdivision values of a grating period (address 0x1C)
- **-25** Error while reading measuring step, or measuring steps per revolution or subdivision values of a grating period (address 0x1D)

- **-26** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x4)
- **-27** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x5)
- **-28** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x4)
- **-29** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x5)
- **-30** Error while reading distinguishable revolutions (address 0x1)
- **-31** Error while selecting memory area 0xBD
- **-32** Error while reading number of distinguishable revolutions with scaling factor (address 0x22)
- **-33** Error while selecting memory area 0xBD
- **-34** Error while reading status of additional datum 1 (address 0x0)
- **-35** Error while reading status of additional datum 2 (address 0x1)
- **-41** Transmission error. Please call `MSXE173x__MFEndatGetErrorSources` to get more information
- **-100** Internal system error occurred. See value of `syserrno`

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulIDNumberLsb ID Number - low bits (see page 67/121 of EnDat specifications)

ulIDNumberMsb ID Number - high bits (see page 67/121 of EnDat specifications)

ulSerialNumberLsb Serialnumber - low bits (see page 68/121 of EnDat specifications)

ulSerialNumberMsb Serialnumber - high bits (see page 68/121 of EnDat specifications)

ulModel Model/Type of the sensor (see page 79/84 of EnDat application notes 722024)

- **0, 1, 2, 3** Incremental linear encoder
- **4, 6** Absolute linear encoder
- **8, 9, 10, 11** Incremental rotary encoder or angle encoder
- **12** Singleturn encoder
- **13, 14** Multiturn encoder

ulMode Support of EnDat 2.2

- **0** Sensor does not support EnDat 2.2. commands
- **1** Sensor supports EnDat 2.2 commands

ulPositionSize Size of the position value send by the sensor (in bits)

ulSignalPeriod Signal period length or signal periods per revolution for incremental output signals (see page 79/84 of EnDat application notes 722024)

ulStepPerRevolution Measuring step length or measuring steps per revolution with serial data transfer (see page 79/84 of EnDat application notes 722024)

ulNumberOfRevolution Distinguishable revolutions - only for multiturn encoders (see page 79/84 of EnDat application notes 722024)

ulScalingFactor Scaling factor for resolution (see page 79/84 of EnDat application notes 722024)

ulAdditionalData Supported additional data (see page 85/121 of EnDat specifications)

- **Bit 0** "Position value 2" is available (MRS-Code: 0x42, 0x43, 0x44)
- **Bit 1** "Test values" is available (MRS-Code: 0x49, 0x4A, 0x4B)
- **Bit 2** "Temperature sensor 1 (external)" is available (MRS-Code: 0x4C)
- **Bit 3** "Temperature sensor 2 (external)" is available (MRS-Code: 0x4D)
- **Bit 4** "Additional sensors" is available (MRS-Code: 0x4E)
- **Bit 16** "Commutation" is available (MRS-Code: 0x51)

- **Bit 17** "Acceleration" is available (MRS-Code: 0x52)
- **Bit 18** "Limit position signals" is available (MRS-Code: 0x54)
- **Bit 19** "Asynchronous position value" is available (MRS-Code: 0x56, 0x57, 0x58)
- **Bit 20** "Operating status error sources" is available (MRS-Code: 0x59)
- **Bit 23** "Timestamp" is available (MRS-Code: 0x5B)

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.56 `int MSXE173x__MFEndatGetErrorSources (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE173x__MFEndatGetErrorSourcesResponse * Response)`

The error are reseted after a call to MSXE173x__MFEndatResetErrorBits. If a function returns an error, please use this function to check if it is not a communication error.

Parameters

[in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] **ulChannelIndex** Index of the channel. Set to 0

[in] **ulOption01** Reserved. Set to 0

[in] **ulOption02** Reserved. Set to 0

[in] **ulOption03** Reserved. Set to 0

[in] **ulOption04** Reserved. Set to 0

[out] **Response** *sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The ulConnectorIndex parameter is wrong
- **-4** The ulChannelIndex parameter is wrong
- **-5** The component is not programmed as EnDat
- **-100** Internal system error occurred. See value of syserrno

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulErrorSrc Mask of bits that give the current error sources. Each bit represents an error. If the bit is set to 1, the error is present.

- **Bit 0** Invalid mode command
- **Bit 1** Invalid MRS-Code
- **Bit 2** Transmission is not completed
- **Bit 3** Communication command is not supported
- **Bit 4** MRS-Code is not allowed
- **Bit 6** Invalid address is selected or sensor's EEPROM is written while being busy
- **Bit 7** Try to write a protected memory place
- **Bit 8** Write-Protect configuration is tried to be reset (if a memory place is write-protected, it cannot be reset)

- **Bit 9** Block address is not available
- **Bit 10** Invalid address for the communication command
- **Bit 11** Invalid additional data (or additional data not supported by the sensor)

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.57 `int MSXE173x__MFEndatResetErrorBits (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response)`

It can be used before each command in order to get (after the call of the command) the status of the system using MSXE173x__MFEndatGetErrorSources.

Once an error is detected using MSXE173x__MFEndatGetErrorSources, you have to clear it using this function.

Parameters

[in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] **ulChannelIndex** Index of the channel. Set to 0

[in] **ulOption01** Reserved. Set to 0

[in] **ulOption02** Reserved. Set to 0

[in] **ulOption03** Reserved. Set to 0

[in] **ulOption04** Reserved. Set to 0

[out] **Response iReturnValue**

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The ulConnectorIndex parameter is wrong
- **-4** The ulChannelIndex parameter is wrong
- **-5** The component is not programmed as EnDat
- **-100** Internal system error occurred. See value of syserrno

syserrno system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.58 `int MSXE173x__MFEndatSensorReceiveReset (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response)`

This function has the same effect as an hardware reboot of the sensor.

Parameters

- [in] ***ulConnectorIndex*** Index of the EnDat connector (0 to 3). See on the MSX-E system.
 - [in] ***ulChannelIndex*** Index of the channel. Set to 0
 - [in] ***ulOption01*** Reserved. Set to 0
 - [in] ***ulOption02*** Reserved. Set to 0
 - [in] ***ulOption03*** Reserved. Set to 0
 - [in] ***ulOption04*** Reserved. Set to 0
 - [out] ***Response iReturnValue***
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The *ulConnectorIndex* parameter is wrong
 - **-4** The *ulChannelIndex* parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be INITIALISED or UNINITIALISED or ERROR)
 - **-7** Error while resetting sensor
 - **-41** Transmission error. Please call `MSXE173x__MFEndatGetErrorSources` to get more information
 - **-100** Internal system error occurred. See value of `syserrno`
- syserrno* system-error code (the value of the libc "errno" code)

Return values

- 0** SOAP_OK
- Others** See SOAP error

4.1.3.59 `int MSXE173x__MFEndatSelectMemoryArea (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response)`

In order to send or read parameters, the memory area must first be selected.

Before calling this function, you must call the `MSXE173x__MFEndatInitSensor` function.

Parameters

- [in] ***ulConnectorIndex*** Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] ***ulChannelIndex*** Index of the channel. Set to 0
- [in] ***ulMrsCode*** The MRS-code corresponding to the memory area that you want to select (see page 31/121 and 84/121 of EnDat specifications)
 - **0xB9** Operating status (address area: 0x0 - 0x3)
 - **0xA1** Parameters of the encoder manufacturer - first part (address area: 0x4 - 0xF)
 - **0xA3** Parameters of the encoder manufacturer - second part (address area: 0x0 - 0xF)
 - **0xA5** Parameters of the encoder manufacturer - third part (address area: 0x0 - 0xF)
 - **0xA7** Operating parameters (address area: 0x0 - 0xF)

- **0xA9** Parameters of the OEM - first part (address area: depending on the sensor)
- **0xAB** Parameters of the OEM - second part (address area: depending on the sensor)
- **0xAD** Parameters of the OEM - third part (address area: depending on the sensor)
- **0xAF** Parameters of the OEM - fourth part (address area: depending on the sensor)
- **0xB1** Compensation values of the encoder manufacturer - first part (address area: depending on the sensor)
- **0xB3** Compensation values of the encoder manufacturer - second part (address area: depending on the sensor)
- **0xB5** Compensation values of the encoder manufacturer - third part (address area: depending on the sensor)
- **0xB7** Compensation values of the encoder manufacturer - fourth part (address area: depending on the sensor)

[in] *ulOption01* Reserved. Set to 0

[in] *ulOption02* Reserved. Set to 0

[in] *ulOption03* Reserved. Set to 0

[in] *ulOption04* Reserved. Set to 0

[out] *Response iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The *ulConnectorIndex* parameter is wrong
- **-4** The *ulChannelIndex* parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** The *ulMrsCode* parameter is wrong
- **-8** Error while selecting the memory area
- **-41** Transmission error. Please call `MSXE173x__MFEndatGetErrorSources` to get more information
- **-100** Internal system error occurred. See value of *syserrno*

syserrno system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.60 `int MSXE173x__MFEndatSensorSendParameter (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatSensorSendParameterResponse * Response)`

Before calling this function, you must call the `MSXE173x__MFEndatInitSensor` function to initialise the sensor, and then `MSXE173x__MFEndatSelectMemoryArea`, or `MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea` to select the memory area that contains the parameter you want to read.

Parameters

- [in] ***ulConnectorIndex*** Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] ***ulChannelIndex*** Index of the channel. Set to 0
- [in] ***ulMrsCode*** The MRS-code corresponding to the last memory area that you have selected (see MSXE173x__MFEndatSelectMemoryArea or MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea)
- [in] ***ulAddress*** The address of the parameter that you want to read (0x0-0xFF)
- [in] ***ulOption01*** Reserved. Set to 0
- [in] ***ulOption02*** Reserved. Set to 0
- [in] ***ulOption03*** Reserved. Set to 0
- [in] ***ulOption04*** Reserved. Set to 0
- [out] ***Response*** *sResponse.iReturnValue*
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The *ulConnectorIndex* parameter is wrong
 - **-4** The *ulChannelIndex* parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be INITIALISED)
 - **-7** The *ulMrsCode* parameter is wrong
 - **-8** The *ulAddress* parameter is wrong
 - **-9** Your sensor is not compatible with EnDat 2.2, but the parameter *ulMrsCode* is only available for EnDat 2.2
 - **-10** The last selected memory area do not corresponds to the parameter *ulMrsCode*. Please call MSXE173x__MFEndatSelectMemoryArea or MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea with the wished *ulMrsCode* before
 - **-11** Error while reading parameter
 - **-41** Transmission error. Please call MSXE173x__MFEndatGetErrorSources to get more information
 - **-100** Internal system error occurred. See value of *syserrno*
- sResponse.syserrno*** system-error code (the value of the libc "errno" code)
- ulParam*** Value of the parameter

Return values

- 0** SOAP_OK
- Others** See SOAP error

4.1.3.61 `int MSXE173x__MFEndatSensorReceiveParameter (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulParam, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response)`

Before calling this function, you must call the MSXE173x__MFEndatInitSensor function to initialise the sensor, and then MSXE173x__MFEndatSelectMemoryArea, or MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea to select the memory area that contains the parameter you want to write.

Parameters

- [in] ***ulConnectorIndex*** Index of the EnDat connector (0 to 3). See on the MSX-E system.
 - [in] ***ulChannelIndex*** Index of the channel. Set to 0
 - [in] ***ulMrsCode*** The MRS-code corresponding to the last memory area that you have selected (see MSXE173x__MFEndatSelectMemoryArea or MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea)
 - [in] ***ulAddress*** The address of the parameter that you want to write (0x0-0xFF)
 - [in] ***ulParam*** The new value of the parameter
 - [in] ***ulOption01*** Reserved. Set to 0
 - [in] ***ulOption02*** Reserved. Set to 0
 - [in] ***ulOption03*** Reserved. Set to 0
 - [in] ***ulOption04*** Reserved. Set to 0
 - [out] ***Response iReturnValue***
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The *ulConnectorIndex* parameter is wrong
 - **-4** The *ulChannelIndex* parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be INITIALISED)
 - **-7** The *ulMrsCode* parameter is wrong
 - **-8** The *ulAddress* parameter is wrong
 - **-9** The last selected memory area do not corresponds to the parameter *ulMrsCode*. Please call MSXE173x__MFEndatSelectMemoryArea or MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea with the wished *ulMrsCode* before
 - **-10** Error while writing parameter
 - **-41** Transmission error. Please call MSXE173x__MFEndatGetErrorSources to get more information
 - **-100** Internal system error occurred. See value of *syserrno*
- syserrno*** system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.62 `int MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulMrsCode, xsd__unsignedLong ulAddress, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatSensorSendPosAndRecvSelMemAreaResponse * Response)`

In order to send or read parameters, the memory area must first be selected.

Before calling this function, you must call the MSXE173x__MFEndatInitSensor function.

This function differs from the `MSXE173x_MFEndatSelectMemoryArea` function, since it can select memory areas that are reserved for EnDat 2.2. sensors.

This function is reserved for EnDat 2.2 sensors. It will returns an error if the sensor does not support EnDat 2.2 commands.

Parameters

- [in] ***ulConnectorIndex*** Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] ***ulChannelIndex*** Index of the channel. Set to 0
- [in] ***ulMrsCode*** The MRS-code corresponding to the memory area that you want to select (see page 31/121 and 84/121 of EnDat specifications)
 - **0xB9** Operating status (address area: 0x0 - 0xF)
 - **0xA1** Parameters of the encoder manufacturer - first part (address area: 0x4 - 0xF)
 - **0xA3** Parameters of the encoder manufacturer - second part (address area: 0x0 - 0xF)
 - **0xA5** Parameters of the encoder manufacturer - third part (address area: 0x0 - 0xF)
 - **0xA7** Operating parameters (address area: 0x0 - 0xF)
 - **0xA9** Parameters of the OEM - first part (address area: depending on the sensor)
 - **0xAB** Parameters of the OEM - second part (address area: depending on the sensor)
 - **0xAD** Parameters of the OEM - third part (address area: depending on the sensor)
 - **0xAF** Parameters of the OEM - fourth part (address area: depending on the sensor)
 - **0xB1** Compensation values of the encoder manufacturer - first part (address area: depending on the sensor)
 - **0xB3** Compensation values of the encoder manufacturer - second part (address area: depending on the sensor)
 - **0xB5** Compensation values of the encoder manufacturer - third part (address area: depending on the sensor)
 - **0xB7** Compensation values of the encoder manufacturer - fourth part (address area: depending on the sensor)
 - **0xBD** Parameters of the encoder manufacturer for EnDat 2.2 (address area: 0x0 - 0x3F)
 - **0xBB** Operating parameters 2 (address area: depending on the sensor)
 - **0xBF** Parameters of the section 2 memory area (address area: depending on the sensor)
- [in] ***ulAddress*** Selected block address (only used if *ulMrsCode* is set to 0xBF. If *ulMrsCode* is not 0xBF, set it to 0). It is used to address further section 2 memory areas (see page 46/121 of EnDat specifications)
- [in] ***ulOption01*** Reserved. Set to 0
- [in] ***ulOption02*** Reserved. Set to 0
- [in] ***ulOption03*** Reserved. Set to 0
- [in] ***ulOption04*** Reserved. Set to 0
- [out] ***Response sResponse.iReturnValue***
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The *ulConnectorIndex* parameter is wrong
 - **-4** The *ulChannelIndex* parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be INITIALISED)
 - **-7** The *ulMrsCode* parameter is wrong

- **-8** The ulAddress parameter is wrong
- **-9** The sensor is not compatible with EnDat 2.2
- **-10** Error while getting position and selecting memory area
- **-41** Transmission error. Please call MSXE173x__MFEndatGetErrorSources to get more information
- **-100** Internal system error occurred. See value of syserrno

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulPositionLow Position - low bits

ulPositionHigh Position - high bits

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.63 `int MSXE173x__MFEndatSelectAdditionalData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulMFChannelIndex, xsd__unsignedLong ulAddDataCount, xsd__unsignedLong ulMrsCodeAD1, xsd__unsignedLong ulMrsCodeAD2, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response)`

Some additional data are not available on all sensors. To get the available additional data of your sensor, please use the function MSXE173x__MFEndatGetSensorProperties.

If you select an additional data that is not available on your sensor, you will get the parameter ucErrorSrc13 set to 1 when calling the function MSXE173x__MFEndatGetErrorSources.

The additional data are extra values that the sensor can send (in the same cycle as its position value).

This function is reserved for EnDat 2.2 sensors. It will returns an error if the sensor does not support EnDat 2.2 commands.

Parameters

[in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] *ulChannelIndex* Index of the channel. Set to 0

[in] *ulAddDataCount* The number of selected additional data (0 to 2)

[in] *ulMrsCodeAD1* The MRS-Code for the first additional data

- **0x40** Send additional info 1 without data contents
- **0x42** Position value 2 word 1 LSB
- **0x43** Position value 2 word 2
- **0x44** Position value 2 word 3 MSB
- **0x49** Test values word 1 LSB
- **0x4A** Test values word 2
- **0x4B** Test values word 3 MSB
- **0x4C** Temperature sensor 1 (external)
- **0x4D** Temperature sensor 2 (external)
- **0x4E** Additional sensors

[in] *ulMrsCodeAD2* The MRS-Code for the second additional data

- **0x50** Send additional datum 2 without data contents
- **0x51** Commutation
- **0x52** Acceleration
- **0x54** Limit position signals
- **0x56** Asynchronous position value word 1 LSB
- **0x57** Asynchronous position value word 2
- **0x58** Asynchronous position value word 3 MSB
- **0x59** Operating status error sources
- **0x5B** Timestamp

[in] **ulOption01** Reserved. Set to 0

[in] **ulOption02** Reserved. Set to 0

[in] **ulOption03** Reserved. Set to 0

[in] **ulOption04** Reserved. Set to 0

[out] **Response iReturnValue**

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The ulConnectorIndex parameter is wrong
- **-4** The ulChannelIndex parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** The ulAddDataCount parameter is wrong
- **-8** The ulMrsCodeAD1 parameter is wrong
- **-9** The ulMrsCodeAD2 parameter is wrong
- **-10** The sensor is not compatible with EnDat 2.2
- **-11** Error while deactivating additional data 2
- **-12** Error while deactivating additional data 1
- **-13** Error while activating additional data 1
- **-14** Error while deactivating additional data 2
- **-15** Error while deactivating additional data 1
- **-16** Selected additional data 1 is wrong or not available on this sensor. Please call MSXE173x__MFEndatGetErrorSources to get more information
- **-17** Error while activating additional data 1. Please call MSXE173x__MFEndatGetErrorSources to get more information
- **-18** Error while getting the current position value
- **-19** Error while activating additional data 2
- **-20** Error while deactivating additional data 2
- **-21** Error while deactivating additional data 1
- **-22** Selected additional data 2 is wrong or not available on this sensor. Please call MSXE173x__MFEndatGetErrorSources to get more information
- **-23** Error while activating additional data 2. Please call MSXE173x__MFEndatGetErrorSources to get more information
- **-24** Error while getting the current position value
- **-41** Transmission error. Please call MSXE173x__MFEndatGetErrorSources to get more information
- **-100** Internal system error occurred. See value of syserrno

syserrno system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.64 `int MSXE173x__MFEndatGetPositionWithAddData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetPositionWithAddDataResponse * Response)`

Before calling this function, you must call the MSXE173x__MFEndatInitSensor function.

You must also call the function MSXE173x__MFEndatSelectAdditionalData to select the additional data that you want to receive.

This function is reserved for EnDat 2.2 sensors. It will returns an error if the sensor does not support EnDat 2.2 commands.

Parameters

[in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] *ulChannelIndex* Index of the channel. Set to 0

[in] *ulOption01* Reserved. Set to 0

[in] *ulOption02* Reserved. Set to 0

[in] *ulOption03* Reserved. Set to 0

[in] *ulOption04* Reserved. Set to 0

[out] *Response sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The *ulConnectorIndex* parameter is wrong
- **-4** The *ulChannelIndex* parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** Error while reading the position
- **-41** Transmission error. Please call MSXE173x__MFEndatGetErrorSources to get more information
- **-100** Internal system error occurred. See value of *syserrno*

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulPositionLow Position - low bits

ulPositionHigh Position - high bits

ulAddData1 Value of the additional data 1

ulAddData2 Value of the additional data 2

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.65 `int MSXE173x__MFEndatGetSelectedAdditionalData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__MFEndatGetSelectedAdditionalDataResponse * Response)`

The additional data are selected using the function MSXE173x__MFEndatSelectAdditionalData.

Parameters

- [in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] *ulChannelIndex* Index of the channel. Set to 0
- [in] *ulOption01* Reserved. Set to 0
- [in] *ulOption02* Reserved. Set to 0
- [in] *ulOption03* Reserved. Set to 0
- [in] *ulOption04* Reserved. Set to 0
- [out] *Response sResponse.iReturnValue*
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The *ulConnectorIndex* parameter is wrong
 - **-4** The *ulChannelIndex* parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-100** Internal system error occurred. See value of *syserrno*
- sResponse.syserrno* system-error code (the value of the libc "errno" code)
- ulAdCount* Number of selected additional data
- ulMrsCodeAD1* MRS code of the additional data 1
- ulMrsCodeAD2* MRS code of the additional data 2

Return values

- 0** SOAP_OK
- Others* See SOAP error

4.1.3.66 `int MSXE173x__MFEndatInitAndEnableLatchPositionValues (xsd__unsignedLong ulmFModuleIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulLatchSource, xsd__unsignedLong ulTriggerEdgeCount, xsd__unsignedLong ulDataFormat, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response)`

Before calling this function, you must call the MSXE173x__MFEndatInitSensor function.

When the selected trigger occurs, the position value of the selected EnDat channel will be measured and send through the dataserver.

If you chose additional data using the function MSXE173x__MFEndatSelectAdditionalData and if your selected *ulDataFormat* enables it, you will also receive the value of the selected additional data.

Note: using a synchro timer (and activating the synchro trigger as latch source) enables to get position of the EnDat sensor at a defined rate.

Parameters

[in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] **ulChannelIndex** Index of the channel. Set to 0

[in] **ulLatchSource** Mask of bits, that defines the trigger source.

Bit 0: Hardware trigger If you select the hardware trigger, you must also choose the edge detection (Bit 2 and 3). Of course, you can choose both.

- 1 activated
- 0 not used

Bit 1 : Synchro trigger

- 1 activated
- 0 not used

Bit 2 : Trigger rising edge (only if hardware trigger is selected)

- 1 activated
- 0 not used

Bit 3 : Trigger falling edge (only if hardware trigger is selected)

- 1 activated
- 0 not used

```
ulLatchSource = 2 (0b10)      -> the latch source is the synchro trigger
ulLatchSource = 5 (0b101)    -> the latch source is a rising edge of the hardware trigger
ulLatchSource = 13 (0b1101) -> the latch source is a rising or a falling edge of the hardware trigger
```

[in] **ulTriggerEdgeCount** Number of edges required to detect an hardware trigger (only if hardware trigger is selected)

[in] **ulDataFormat** Mask of bits, that defines the format of data that will be sent by the data server. It is a combination of bits. You can set all the bits to 1 if you want all the informations 0b11111 = 31.

You can also, for example, only wants the timestamp and the digital I/Os state (0b11 = 3).

You can also choose, for example, to get the additional data 2 (0b1000 = 8).

Bit 0: Timestamp

- 0 Not sent
- 1 timestamp sent by the dataserver

Bit 1: Digital I/Os state

- 0 Not sent
- 1 digital I/Os state sent by the data server

Bit 2: Additional data 1

- 0 Not sent
- 1 Additional data 1 sent by the data server

Bit 3: Additional data 2

- 0 Not sent
- 1 Additional data 2 sent by the data server

Bit 4: Format of the value

- 0 Raw value (as sent by the sensor)
- 1 Standardised value. The format of the frame will then depends on the model of the sensor. See system documentation.

If ulDataFormat is set to 0, the frame send by the dataserver will have the following definition

```

unsigned long eventsrc;           // High 16 bits (MSB): Channel concerned
    (0 to 3)

                                // Low 16 bits (LSB): Bit mask representing the source of the event
                                //      Bit 0: Hardware trigger
                                //      Bit 1: Synchro trigger
                                //      Bit 2: Compare

unsigned long positionlow;        // Low bits of the position
unsigned long positionhigh;      // High bits of the position
unsigned long error;             // Status of the communication. Same value as the value returned by the function MSXE173x__MFEndatGetErrorSources

```

If you set ulDataFormat to 1 (send timestamp), then the frame send by the dataserver will be changed. At the end of the "basic" frame, we add the timestamp.

```

unsigned long ts;                // Second part of the timestamp
unsigned long tus;              // Microsecond part of the timestamp

```

If you set ulDataFormat to 2 = 0b10 (send digital I/Os state), then the frame send by the dataserver will be changed. At the end of the "basic" frame, we add the state of the digital I/Os.

```

unsigned long digiostate;        // State of the digital I/Os. Bit mask that encodes all digital I/Os.
                                // 0b0: All I/Os are set to low
                                // 0b100: I/O 2 is set to high, all others are set to 0
                                // 0b11111111111111: All I/Os are set to high

```

If you set ulDataFormat to 4 = 0b100 (send additional data 1), then the frame send by the dataserver will be changed. At the end of the "basic" frame, we add the value of the first additional data.

```

unsigned long ad1value;
\encode
If you set ulDataFormat to 8 = 0b1000 (send additional data 2), then the frame send by the dataserver will be changed. At the end of the "basic" frame, we add the value of the second additional data. \n
\code
unsigned long ad2value;
\encode
If you set the bits 0, 1, 2 and 3 to 1, the format of the frame will then be: \n
\code
unsigned long eventsrc;           // High 16 bits (MSB): Channel concerned
    (0 to 3)

                                // Low 16 bits (LSB): Bit mask representing the source of the event
                                //      Bit 0: Hardware trigger
                                //      Bit 1: Synchro trigger
                                //      Bit 2: Compare
                                //      Bit 3: Compare

unsigned long positionlow;        // Low bits of the position
unsigned long positionhigh;      // High bits of the position
unsigned long error;             // Status of the communication. Same value as the value returned by the function MSXE173x__MFEndatGetErrorSources
unsigned long ts;                // Second part of the timestamp
unsigned long tus;              // Microsecond part of the timestamp
unsigned long digiostate;        // State of the digital I/Os. Bit mask that encodes all digital I/Os.

```

```

        es all digital I/Os.
        All I/Os are set to low
        I/O 2 is set to high, all others are set to 0
        1111 : All I/Os are set to high
unsigned long ad1value;
unsigned long ad2value;

```

The bit 4 of the ulDataFormat is more complex. The format of the frame that is sent by the dataserver will depend on the model of the sensor used. See the system documentation for more information, or the "Acquisition" menu of the web site.

[in] **ulOption01** Reserved. Set to 0

[in] **ulOption02** Reserved. Set to 0

[in] **ulOption03** Reserved. Set to 0

[in] **ulOption04** Reserved. Set to 0

[out] **Response iReturnValue**

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The ulConnectorIndex parameter is wrong
- **-4** The ulChannelIndex parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** The ulLatchSource parameter is wrong
- **-8** The ulDataFormat parameter is wrong
- **-9** Error while getting sensor properties
- **-10** The multiturn part size of the sensor is 0
- **-11** The singleturn part size of the sensor is 0
- **-12** The step per revolution properties of the sensor is 0
- **-41** Transmission error. Please call MSXE173x__MFEndatGetErrorSources to get more information
- **-100** Internal system error occurred. See value of syserrno

syserrno system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.67 **int** MSXE173x__MFEndatGetCurrentLatchConfiguration (**xsd__unsignedLong** *ulmFModuleIndex*, **xsd__unsignedLong** *ulChannelIndex*, **xsd__unsignedLong** *ulOption01*, **xsd__unsignedLong** *ulOption02*, **xsd__unsignedLong** *ulOption03*, **xsd__unsignedLong** *ulOption04*, **struct** MSXE173x__MFEndatGetCurrentLatchConfigurationResponse * *Response*)

Parameters

[in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] **ulChannelIndex** Index of the channel. Set to 0

[in] **ulOption01** Reserved. Set to 0

[in] **ulOption02** Reserved. Set to 0

[in] **ulOption03** Reserved. Set to 0

[in] **ulOption04** Reserved. Set to 0

[out] **Response** *sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The ulConnectorIndex parameter is wrong
- **-3** The ulChannelIndex parameter is wrong
- **-4** The component is not programmed as EnDat
- **-100** Internal system error occurred. See value of syserrno

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulRunning The running state.

- **0** Not running (do not use other return parameters, there will be set to 0)
- **1** Latch logic is running

ulLatchSource Mask of bits, that defines the trigger source. See documentation of MSXE173x__MFEndatInitAndEnableLatchPositionValues

ulTriggerEdgeCount Number of edges required to detect an hardware trigger (only if hardware trigger is selected)

ulDataFormat Mask of bits, that defines the format of data that will be sent by the data server. See documentation of MSXE173x__MFEndatInitAndEnableLatchPositionValues

ulModel (Used for computation) Model/Type of the sensor (see page 79/84 of EnDat application notes 722024)

- **0, 1, 2, 3** Incremental linear encoder
- **4, 6** Absolute linear encoder
- **8, 9, 10, 11** Incremental rotary encoder or angle encoder
- **12** Singleturn encoder
- **13, 14** Multiturn encoder

ulPositionSize (Used for computation) Size of the position value send by the sensor (in bits)

ulSignalPeriod (Used for computation) Signal period length or signal periods per revolution for incremental output signals

ulStepPerRevolution (Used for computation) Measuring step length or measuring steps per revolution with serial data transfer

ulNumberOfRevolution (Used for computation) Distinguishable revolutions - only for multiturn encoders

ulScalingFactor (Used for computation) Scaling factor for resolution

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.68 `int MSXE173x__MFEndatDisableAndReleaseLatchPositionValues (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE173x__Response * Response)`

Parameters

- [in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] *ulChannelIndex* Index of the channel. Set to 0
- [in] *ulOption01* Reserved. Set to 0
- [in] *ulOption02* Reserved. Set to 0
- [in] *ulOption03* Reserved. Set to 0
- [in] *ulOption04* Reserved. Set to 0
- [out] *Response iReturnValue*
- **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The ulConnectorIndex parameter is wrong
 - **-4** The ulChannelIndex parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be LATCH_RUNNING or LATCH_FIFO_OVERFLOW)
 - **-100** Internal system error occurred. See value of syserrno
- syserrno* system-error code (the value of the libc "errno" code)

Return values

- 0** SOAP_OK
- Others* See SOAP error

4.1.3.69 `int MSXE17xx__DigitalIOGetNumber (void * _, struct MSXE17xx__DigitalIOGetNumberResponse * Response)`

Parameters

- [in] *None*
- [out] *Response :*
- sResponse.iReturnValue :*
- **0**: means the remote function performed OK
 - **-1**: means an system error occured (check errno in this case)
- sResponse.syserrno :* system-error code (the value of the libc "errno" code)

Returns

- **0**: SOAP_OK
- **<> 0**: See SOAP error

4.1.3.70 `int MSXE17xx_DigitalIOInitPortConfiguration (xsd__unsignedLong ulPort, xsd__unsignedLong ulPortConfiguration, struct MSXE17xx_Response * Response)`

Parameters

[in] *ulPort* : Index of the digital i/o port (0 to 7)

[in] *ulPortConfiguration* : Define the port configuration

- 0 : input
- 1 : output

[out] *Response* :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -2: Digital i/o port selection error
- -3: Port configuration selection error
- -100: Init dig i/o port kernel function error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.71 `int MSXE17xx_DigitalIOReadChannelValue (xsd__unsignedLong ulChannel, struct MSXE17xx_unsignedLongResponse * Response)`

Parameters

[in] *ulChannel* : Index of the digital i/o channel (0 to 15)

[out] *Response* :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -2: Digital i/o channel selection error
- -100: Read dig i/o channel value kernel function error

syserrno : system-error code (the value of the libc "errno" code) *ulValue* : i/o channel value:

- 0
- 1

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.72 `int MSXE17xx__DigitalIOReadAllChannelsValue (void * _, struct MSXE17xx__unsignedLongResponse * Response)`

Parameters

[in] `_` : no input parameter

[out] `Response` :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -100: Read dig i/o channel value kernel function error

syserrno : system-error code (the value of the libc "errno" code) *ulValue* : i/o channels value(each bit correspond to one channel)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.73 `int MSXE17xx__DigitalIOWriteChannelValue (xsd__unsignedLong ulChannel, xsd__unsignedLong ulChannelValue, struct MSXE17xx__Response * Response)`

Parameters

[in] *ulChannel* : Index of the digital i/o channel (0 to 15)

[in] *ulChannelValue* : Channel value

- 0
- 1

[out] `Response` :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -2: Digital i/o channel selection error
- -3: Channel value error
- -100: Write dig i/o channel value kernel function error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.74 `int MSXE17xx__DigitalIOWriteAllChannelsValue (xsd__unsignedLong ulChannelValue, struct MSXE17xx__Response * Response)`

Parameters

[in] *ulChannelValue* : Channels value (each bit corresponds to a channel)

[out] **Response** :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -2: Channels value error
- -100: Write dig i/o channel value kernel function error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.75 `int MSXE17xx__DigitalIOReleasePortConfiguration (xsd__unsignedLong ulPort, struct MSXE17xx__Response * Response)`

Parameters

[in] **ulPort** : Index of the digital i/o port (0 to 7)

[out] **Response** :

iReturnValue :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -2: Digital i/o port selection error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.76 `int MSXE17xx__DigitalIOTestShortCircuit (xsd__unsignedLong ulOption, struct MSXE17xx__unsignedLongResponse * Response)`

Parameters

[in] **ulOption** : reserved

[out] **Response** :

iReturnValue :

- 0 : means the remote function performed OK
- -1: means an system error occurred

syserrno : system-error code (the value of the libc "errno" code)

ulValue : short circuit status: from 0 to 0xffff, one bit for each output

- 0 : no short circuit
- 1 : short circuit

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.77 `int MSXE17xx_DigitalIORearmShortCircuit (xsd__unsignedLong ulOption, struct MSXE17xx__Response * Response)`

Parameters

- [in] *ulOption* : reserved
- [out] *Response* :
- iReturnValue* :
- 0 : means the remote function performed OK
 - -1: means an system error occurred
- syserrno* : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.78 `int MSXE17xx_IOWatchdogInitAndStart (xsd__unsignedLong ulTimeBase, xsd__unsignedLong ulTimeValue, xsd__unsignedLong ulOption1, xsd__unsignedLong ulOption2, struct MSXE17xx__Response * Response)`

Parameters

- [in] *ulTimeBase* : Time base of the watchdog delay (0 for mus, 1 for ms, 2 for s)
- [in] *ulTimeValue* : Time base of the watchdog delay (0 to 0xFFFF)
- [in] *ulOption1* : Reserved
- [in] *ulOption2* : Reserved
- [out] *Response* :
- iReturnValue* :
- 0: remote function performed OK
 - -1: an system error occurred
 - -2: time base selection error
 - -3: time value selection error
 - -100: Init and start digital output watchdog kernel function error
- syserrno* : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.79 `int MSXE17xx_IOWatchdogStopAndRelease (xsd__unsignedLong ulOption, struct MSXE17xx__Response * Response)`

Parameters

- [in] *ulOption* : reserved
- [out] *Response* :
- iReturnValue* :

- 0: remote function performed OK
- -1: an system error occurred
- -100: Stop and release digital output watchdog kernel function error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.80 int MSXE17xx_IOWatchdogGetStatusAndValue (xsd__unsignedLong *ulOption*, struct MSXE17xx_IOWatchdogGetStatusAndValueResponse * *Response*)

Parameters

[in] *ulOption* : Reserved

[out] *Response* :

iReturnValue :

- 0: remote function performed OK
- -1: an system error occurred
- -2: channel selection error
- -100: Get diagnostic information kernel function error

ulStatus : current status information

- BIN XXXXXXXXXX XXXXXXXXXX XXXXXXXXXX XXXXXXXX0: is stopped,
- BIN XXXXXXXXXX XXXXXXXXXX XXXXXXXXXX XXXXXXXX1: is running,
- BIN XXXXXXXXXX XXXXXXXXXX XXXXXXXXXX XXXXXXXX0X: is not run down
- BIN XXXXXXXXXX XXXXXXXXXX XXXXXXXXXX XXXXXXXX1X: is run down

ulValue : current value information (0 to 0xFFFF)

ulInfo : reserved

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.81 int MSXE17xx_MFCommonGetSubModuleFunctionality (xsd__unsignedLong *ulMFModuleIndex*, struct MSXE17xx__unsignedLongResponse * *Response*)

Parameters

[in] *ulMFModuleIndex* : index of the multifunction sub module (0 to 3).

[out] *Response* :

ulValue :

- 0: Incremental counter
- -1: PWM

sResponse.iReturnValue :

- 0: means the remote function performed OK

- -1: means an system error occured (check errno in this case)
- -2: Multifunction sub module index selection error

sResponse.syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.82 `int MSXE17xx_MFCommonSetInputsFilter (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulInputAFilterValue, xsd__unsignedLong ulInputBFilterValue, xsd__unsignedLong ulInputCFilterValue, xsd__unsignedLong ulInputDFilterValue, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_Response * Response)`

Parameters

[in] *ulMFModuleIndex* : index of the multifunction sub module (0 to 3).

[in] *ulInputAFilterValue* : Filter value for input A (0 to 262143)

- 0: Filter nicht benutzt
- 1: 100 ns
- 2: 200 ns
- 3: 300 ns ...
- 262143 : 26,2143 ms

[in] *ulInputBFilterValue* : Filter value for input B (0 to 262143)

- 0: Filter nicht benutzt
- 1: 100 ns
- 2: 200 ns
- 3: 300 ns ...
- 262143 : 26,2143 ms

[in] *ulInputCFilterValue* : Filter value for input C (0 to 262143)

- 0: Filter nicht benutzt
- 1: 100 ns
- 2: 200 ns
- 3: 300 ns ...
- 262143 : 26,2143 ms

[in] *ulInputDFilterValue* : Filter value for input D (0 to 262143)

- 0: Filter nicht benutzt
- 1: 100 ns
- 2: 200 ns
- 3: 300 ns ...
- 262143 : 26,2143 ms

[in] *ulOption01* : Set it to 0

[in] *ulOption02* : Set it to 0

[in] *ulOption03* : Set it to 0

[in] ***ulOption04*** : Set it to 0

[out] ***Response*** :

iReturnValue :

- 0 : means the remote function performed OK
- -1: means an system error occurred
- -2: Multifunction sub module index selection error
- -3: Input A filter value selection error
- -4: Input B filter value selection error
- -5: Input C filter value selection error
- -6: Input D filter value selection error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.83 `int MSXE17xx__MFCommonReferenceVoltageActivation (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulActivationFlag, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MSXE17xx__Response * Response)`

Parameters

[in] ***ulMFModuleIndex*** : index of the multifunction sub module (0 to 3).

[in] ***ulActivationFlag*** :

- 0: normal mode from D- (Default mode)
- 1: activate the reference voltage to pin D-

[in] ***ulOption01*** : Set it to 0

[in] ***ulOption02*** : Set it to 0

[out] ***Response*** :

iReturnValue :

- 0 : means the remote function performed OK
- -1: means an system error occurred
- -2: Multifunction sub module index selection error
- -3: Activation flag selection error

syserrno : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.84 `int MSXE17xx_MFCommonSetFIFO0Level (xsd__unsignedLong ulMFModuleIndex, xsd__unsignedLong ulFIFOLevel, xsd__unsignedLong ulTimeOutTimeBase, xsd__unsignedLong ulReloadValue, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MSXE17xx_Response * Response)`

Parameters

- [in] *ulMFModuleIndex* : index of the multifunction sub module (0 to 3).
- [in] *ulFIFOLevel* : Define the FIFO level (1 to 200).
- [in] *ulTimeOutTimeBase* : Define a Time out : permit to receive the data from the FIFO before the FIFO level is reached.
Time base of the timer (0: disabled, 1 for us, 2 for ms, 3 for s)
- [in] *ulReloadValue* : Time out reload value (1 to 0xFFFF)
- [in] *ulOption01* : reserved (Set it to 0).
- [in] *ulOption02* : reserved (Set it to 0).
- [out] *Response* :
iReturnValue :
 - 0: means the remote function performed OK
 - -1: means an system error occurred
 - -2: Multifunction sub module index selection error
 - -3: FIFO level value is wrong
 - -4: Time out time base selection error
 - -5: Time out value can not be null, if a time base is selected
 - -100: Set FIFO level kernel function error*syserrno* : system-error code (the value of the libc "errno" code)

Returns

- 0: SOAP_OK
- <> 0: See SOAP error

4.1.3.85 `int MSXE17xx_MFEndatInitSensor (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulFrequency, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_Response * Response)`

This function should be called once, in order to call the other EnDat functions.

Parameters

- [in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] *ulChannelIndex* Index of the channel. Set to 0
- [in] *ulFrequency* Frequency to use in kHz (500, 900, 1500, 2500, 4500)
- [in] *ulOption01* Reserved. Set to 0
- [in] *ulOption02* Reserved. Set to 0
- [in] *ulOption03* Reserved. Set to 0
- [in] *ulOption04* Reserved. Set to 0

[out] **Response** *iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The *ulConnectorIndex* parameter is wrong
- **-4** The *ulChannelIndex* parameter is wrong
- **-5** The driver is in a wrong state (must be INITIALISED or UNINITIALISED)
- **-6** The component is not programmed as EnDat
- **-7** Error while resetting sensor. Note: If no sensor is plugged on the selected channel, you will get this error.
- **-8** Error while selecting memory area 0xB9
- **-9** Error while reading alarm space (address 0x0)
- **-10** Error while reading warning space (address 0x1)
- **-11** Error while clearing errors (write 0 at address 0x0)
- **-12** Error while clearing warnings (write 0 at address 0x1)
- **-13** Error while selecting memory area 0xA1
- **-14** Error while reading number of clock pulses for transfer of position value (address 0x0D)
- **-15** Error while selecting memory area 0xA5
- **-16** Error while reading EnDat command set (address 0x5)
- **-17** Error while reading support of error messages 1 (address 0x3)
- **-18** Error while reading support of warnings (address 0x4)
- **-19** Error while reading EnDat ordering designation (address 0x8)
- **-20** *ulFrequency* parameter is too high for current sensor
- **-21** The *ulFrequency* parameter is wrong
- **-22** EnDat ordering designation is invalid
- **-41** Transmission error. Please call `MSXE17xx__MFEndatGetErrorSources` to get more information
- **-100** Internal system error occurred. See value of `syserrno`

syserrno system error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.86 `int MSXE17xx__MFEndatGetPosition (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__MFEndatGetPositionResponse * Response)`

Before calling this function, you must call the `MSXE17xx__MFEndatInitSensor` function.

Parameters

- [in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] *ulChannelIndex* Index of the channel. Set to 0
- [in] *ulOption01* Reserved. Set to 0
- [in] *ulOption02* Reserved. Set to 0

[in] *ulOption03* Reserved. Set to 0

[in] *ulOption04* Reserved. Set to 0

[out] *Response sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The *ulConnectorIndex* parameter is wrong
- **-4** The *ulChannelIndex* parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** Error while reading the position
- **-41** Transmission error. Please call *MSXE17xx__MFEndatGetErrorSources* to get more information
- **-100** Internal system error occurred. See value of *syserrno*

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulPositionLow Position - low bits

ulPositionHigh Position - high bits

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.87 `int MSXE17xx__MFEndatGetSensorProperties (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__MFEndatGetSensorPropertiesResponse * Response)`

Before calling this function, you must call the *MSXE17xx__MFEndatInitSensor* function.

Parameters

[in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] *ulChannelIndex* Index of the channel. Set to 0

[in] *ulOption01* Reserved. Set to 0

[in] *ulOption02* Reserved. Set to 0

[in] *ulOption03* Reserved. Set to 0

[in] *ulOption04* Reserved. Set to 0

[out] *Response sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The *ulConnectorIndex* parameter is wrong
- **-4** The *ulChannelIndex* parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** Error while selecting memory area 0xA3

- **-8** Error while reading ID number (address 0x8)
- **-9** Error while reading ID number (address 0x9)
- **-10** Error while reading ID number (address 0xA)
- **-11** Error while reading Serialnumber (address 0xB)
- **-12** Error while reading Serialnumber (address 0xC)
- **-13** Error while reading Serialnumber (address 0xD)
- **-14** Error while selecting memory area 0xA1
- **-15** Error while reading encoder model (address 0xE)
- **-16** Error while reading signal period length or signal periods per revolution for incremental output signals (address 0xF)
- **-17** Error while getting the position
- **-18** Error while selecting memory area 0xA3
- **-19** Error while reading signal period length or signal periods per revolution for incremental output signals (address 0x0)
- **-20** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x4)
- **-21** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x5)
- **-22** Error while selecting memory area 0xBD
- **-23** Error while reading scaling factor for resolution (address 0x1B)
- **-24** Error while reading measuring step, or measuring steps per revolution or subdivision values of a grating period (address 0x1C)
- **-25** Error while reading measuring step, or measuring steps per revolution or subdivision values of a grating period (address 0x1D)
- **-26** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x4)
- **-27** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x5)
- **-28** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x4)
- **-29** Error while reading measuring step length or measuring steps per revolution with serial data transfer (address 0x5)
- **-30** Error while reading distinguishable revolutions (address 0x1)
- **-31** Error while selecting memory area 0xBD
- **-32** Error while reading number of distinguishable revolutions with scaling factor (address 0x22)
- **-33** Error while selecting memory area 0xBD
- **-34** Error while reading status of additional datum 1 (address 0x0)
- **-35** Error while reading status of additional datum 2 (address 0x1)
- **-41** Transmission error. Please call MSXE17xx__MFEndatGetErrorSources to get more information
- **-100** Internal system error occurred. See value of syserrno

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulIDNumberLsb ID Number - low bits (see page 67/121 of EnDat specifications)

ulIDNumberMsb ID Number - high bits (see page 67/121 of EnDat specifications)

ulSerialNumberLsb Serialnumber - low bits (see page 68/121 of EnDat specifications)

ulSerialNumberMsb Serialnumber - high bits (see page 68/121 of EnDat specifications)

ulModel Model/Type of the sensor (see page 79/84 of EnDat application notes 722024)

- **0, 1, 2, 3** Incremental linear encoder
- **4, 6** Absolute linear encoder
- **8, 9, 10, 11** Incremental rotary encoder or angle encoder
- **12** Singleturn encoder
- **13, 14** Multiturn encoder

ulMode Support of EnDat 2.2

- **0** Sensor does not support EnDat 2.2. commands
- **1** Sensor supports EnDat 2.2 commands

ulPositionSize Size of the position value send by the sensor (in bits)

ulSignalPeriod Signal period length or signal periods per revolution for incremental output signals (see page 79/84 of EnDat application notes 722024)

ulStepPerRevolution Measuring step length or measuring steps per revolution with serial data transfer (see page 79/84 of EnDat application notes 722024)

ulNumberOfRevolution Distinguishable revolutions - only for multiturn encoders (see page 79/84 of EnDat application notes 722024)

ulScalingFactor Scaling factor for resolution (see page 79/84 of EnDat application notes 722024)

ulAdditionalData Supported additional data (see page 85/121 of EnDat specifications)

- **Bit 0** "Position value 2" is available (MRS-Code: 0x42, 0x43, 0x44)
- **Bit 1** "Test values" is available (MRS-Code: 0x49, 0x4A, 0x4B)
- **Bit 2** "Temperature sensor 1 (external)" is available (MRS-Code: 0x4C)
- **Bit 3** "Temperature sensor 2 (external)" is available (MRS-Code: 0x4D)
- **Bit 4** "Additional sensors" is available (MRS-Code: 0x4E)
- **Bit 16** "Commutation" is available (MRS-Code: 0x51)
- **Bit 17** "Acceleration" is available (MRS-Code: 0x52)
- **Bit 18** "Limit position signals" is available (MRS-Code: 0x54)
- **Bit 19** "Asynchronous position value" is available (MRS-Code: 0x56, 0x57, 0x58)
- **Bit 20** "Operating status error sources" is available (MRS-Code: 0x59)
- **Bit 23** "Timestamp" is available (MRS-Code: 0x5B)

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.88 `int MSXE17xx_MFEndatGetErrorSources (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE17xx_MFEndatGetErrorSourcesResponse * Response)`

The error are reseted after a call to MSXE17xx_MFEndatResetErrorBits. If a function returns an error, please use this function to check if it is not a communication error.

Parameters

- [in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] **ulChannelIndex** Index of the channel. Set to 0
- [in] **ulOption01** Reserved. Set to 0
- [in] **ulOption02** Reserved. Set to 0

[in] *ulOption03* Reserved. Set to 0

[in] *ulOption04* Reserved. Set to 0

[out] *Response sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The *ulConnectorIndex* parameter is wrong
- **-4** The *ulChannelIndex* parameter is wrong
- **-5** The component is not programmed as EnDat
- **-100** Internal system error occurred. See value of *syserrno*

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulErrorSrc Mask of bits that give the current error sources. Each bit represents an error. If the bit is set to 1, the error is present.

- **Bit 0** Invalid mode command
- **Bit 1** Invalid MRS-Code
- **Bit 2** Transmission is not completed
- **Bit 3** Communication command is not supported
- **Bit 4** MRS-Code is not allowed
- **Bit 6** Invalid address is selected or sensor's EEPROM is written while being busy
- **Bit 7** Try to write a protected memory place
- **Bit 8** Write-Protect configuration is tried to be reset (if a memory place is write-protected, it cannot be reset)
- **Bit 9** Block address is not available
- **Bit 10** Invalid address for the communication command
- **Bit 11** Invalid additional data (or additional data not supported by the sensor)

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.89 `int MSXE17xx__MFEndatResetErrorBits (xsd__unsignedLong ulConnectorIndex,
xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01,
xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong
ulOption04, struct MSXE17xx__Response * Response)`

It can be used before each command in order to get (after the call of the command) the status of the system using `MSXE17xx__MFEndatGetErrorSources`.

Once an error is detected using `MSXE17xx__MFEndatGetErrorSources`, you have to clear it using this function.

Parameters

[in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] *ulChannelIndex* Index of the channel. Set to 0

[in] *ulOption01* Reserved. Set to 0

[in] *ulOption02* Reserved. Set to 0

- [in] *ulOption03* Reserved. Set to 0
- [in] *ulOption04* Reserved. Set to 0
- [out] *Response iReturnValue*
- **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The *ulConnectorIndex* parameter is wrong
 - **-4** The *ulChannelIndex* parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-100** Internal system error occurred. See value of *syserrno*
- syserrno* system-error code (the value of the libc "errno" code)

Return values

- 0** SOAP_OK
- Others* See SOAP error

4.1.3.90 `int MSXE17xx_MFEndatSensorReceiveReset (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE17xx_Response * Response)`

This function has the same effect as an hardware reboot of the sensor.

Parameters

- [in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] *ulChannelIndex* Index of the channel. Set to 0
- [in] *ulOption01* Reserved. Set to 0
- [in] *ulOption02* Reserved. Set to 0
- [in] *ulOption03* Reserved. Set to 0
- [in] *ulOption04* Reserved. Set to 0
- [out] *Response iReturnValue*
- **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The *ulConnectorIndex* parameter is wrong
 - **-4** The *ulChannelIndex* parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be INITIALISED or UNINITIALISED or ERROR)
 - **-7** Error while resetting sensor
 - **-41** Transmission error. Please call `MSXE17xx_MFEndatGetErrorSources` to get more information
 - **-100** Internal system error occurred. See value of *syserrno*
- syserrno* system-error code (the value of the libc "errno" code)

Return values

- 0** SOAP_OK
- Others* See SOAP error

```

4.1.3.91 int MSXE17xx_MFEndatSelectMemoryArea ( xsd_unsignedLong ulConnectorIndex,
xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulMrsCode,
xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong
ulOption03, xsd_unsignedLong ulOption04, struct MSXE17xx_Response * Response
)

```

In order to send or read parameters, the memory area must first be selected.

Before calling this function, you must call the MSXE17xx_MFEndatInitSensor function.

Parameters

- [in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] **ulChannelIndex** Index of the channel. Set to 0
- [in] **ulMrsCode** The MRS-code corresponding to the memory area that you want to select (see page 31/121 and 84/121 of EnDat specifications)
 - **0xB9** Operating status (address area: 0x0 - 0x3)
 - **0xA1** Parameters of the encoder manufacturer - first part (address area: 0x4 - 0xF)
 - **0xA3** Parameters of the encoder manufacturer - second part (address area: 0x0 - 0xF)
 - **0xA5** Parameters of the encoder manufacturer - third part (address area: 0x0 - 0xF)
 - **0xA7** Operating parameters (address area: 0x0 - 0xF)
 - **0xA9** Parameters of the OEM - first part (address area: depending on the sensor)
 - **0xAB** Parameters of the OEM - second part (address area: depending on the sensor)
 - **0xAD** Parameters of the OEM - third part (address area: depending on the sensor)
 - **0xAF** Parameters of the OEM - fourth part (address area: depending on the sensor)
 - **0xB1** Compensation values of the encoder manufacturer - first part (address area: depending on the sensor)
 - **0xB3** Compensation values of the encoder manufacturer - second part (address area: depending on the sensor)
 - **0xB5** Compensation values of the encoder manufacturer - third part (address area: depending on the sensor)
 - **0xB7** Compensation values of the encoder manufacturer - fourth part (address area: depending on the sensor)
- [in] **ulOption01** Reserved. Set to 0
- [in] **ulOption02** Reserved. Set to 0
- [in] **ulOption03** Reserved. Set to 0
- [in] **ulOption04** Reserved. Set to 0
- [out] **Response iReturnValue**
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The ulConnectorIndex parameter is wrong
 - **-4** The ulChannelIndex parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be INITIALISED)
 - **-7** The ulMrsCode parameter is wrong
 - **-8** Error while selecting the memory area
 - **-41** Transmission error. Please call MSXE17xx_MFEndatGetErrorSources to get more information

- **-100** Internal system error occurred. See value of `syserrno`
`syserrno` system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.92 `int MSXE17xx_MFEndatSensorSendParameter (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulMrsCode, xsd_unsignedLong ulAddress, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE17xx_MFEndatSensorSendParameterResponse * Response)`

Before calling this function, you must call the `MSXE17xx_MFEndatInitSensor` function to initialise the sensor, and then `MSXE17xx_MFEndatSelectMemoryArea`, or `MSXE17xx_MFEndatSensorSendPosAndRecvSelMemArea` to select the memory area that contains the parameter you want to read.

Parameters

[in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] **ulChannelIndex** Index of the channel. Set to 0

[in] **ulMrsCode** The MRS-code corresponding to the last memory area that you have selected (see `MSXE17xx_MFEndatSelectMemoryArea` or `MSXE17xx_MFEndatSensorSendPosAndRecvSelMemArea`)

[in] **ulAddress** The address of the parameter that you want to read (0x0-0xFF)

[in] **ulOption01** Reserved. Set to 0

[in] **ulOption02** Reserved. Set to 0

[in] **ulOption03** Reserved. Set to 0

[in] **ulOption04** Reserved. Set to 0

[out] **Response sResponse.iReturnValue**

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The `ulConnectorIndex` parameter is wrong
- **-4** The `ulChannelIndex` parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** The `ulMrsCode` parameter is wrong
- **-8** The `ulAddress` parameter is wrong
- **-9** Your sensor is not compatible with EnDat 2.2, but the parameter `ulMrsCode` is only available for EnDat 2.2
- **-10** The last selected memory area do not corresponds to the parameter `ulMrsCode`. Please call `MSXE17xx_MFEndatSelectMemoryArea` or `MSXE17xx_MFEndatSensorSendPosAndRecvSelMemArea` with the wished `ulMrsCode` before
- **-11** Error while reading parameter

- **-41** Transmission error. Please call MSXE17xx__MFEndatGetErrorSources to get more information
- **-100** Internal system error occurred. See value of syserrno

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulParam Value of the parameter

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.93 `int MSXE17xx__MFEndatSensorReceiveParameter (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulMrsCode, xsd_unsignedLong ulAddress, xsd_unsignedLong ulParam, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE17xx__Response * Response)`

Before calling this function, you must call the MSXE17xx__MFEndatInitSensor function to initialise the sensor, and then MSXE17xx__MFEndatSelectMemoryArea, or MSXE17xx__MFEndatSensorSendPosAndRecvSelMemArea to select the memory area that contains the parameter you want to write.

Parameters

[in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] *ulChannelIndex* Index of the channel. Set to 0

[in] *ulMrsCode* The MRS-code corresponding to the last memory area that you have selected (see MSXE17xx__MFEndatSelectMemoryArea or MSXE17xx__MFEndatSensorSendPosAndRecvSelMemArea)

[in] *ulAddress* The address of the parameter that you want to write (0x0-0xFF)

[in] *ulParam* The new value of the parameter

[in] *ulOption01* Reserved. Set to 0

[in] *ulOption02* Reserved. Set to 0

[in] *ulOption03* Reserved. Set to 0

[in] *ulOption04* Reserved. Set to 0

[out] *Response iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The ulConnectorIndex parameter is wrong
- **-4** The ulChannelIndex parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** The ulMrsCode parameter is wrong
- **-8** The ulAddress parameter is wrong
- **-9** The last selected memory area do not corresponds to the parameter ulMrsCode. Please call MSXE17xx__MFEndatSelectMemoryArea or MSXE17xx__MFEndatSensorSendPosAndRecvSelMemArea with the wished ulMrsCode before

- **-10** Error while writing parameter
- **-41** Transmission error. Please call MSXE17xx__MFEndatGetErrorSources to get more information
- **-100** Internal system error occurred. See value of syserrno

syserrno system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.94 `int MSXE17xx__MFEndatSensorSendPosAndRecvSelMemArea (xsd_unsignedLong ulConnectorIndex, xsd_unsignedLong ulChannelIndex, xsd_unsignedLong ulMrsCode, xsd_unsignedLong ulAddress, xsd_unsignedLong ulOption01, xsd_unsignedLong ulOption02, xsd_unsignedLong ulOption03, xsd_unsignedLong ulOption04, struct MSXE17xx__MFEndatSensorSendPosAndRecvSelMemAreaResponse * Response)`

In order to send or read parameters, the memory area must first be selected.

Before calling this function, you must call the MSXE17xx__MFEndatInitSensor function.

This function differs from the MSXE17xx__MFEndatSelectMemoryArea function, since it can select memory areas that are reserved for EnDat 2.2. sensors.

This function is reserved for EnDat 2.2 sensors. It will returns an error if the sensor does not support EnDat 2.2 commands.

Parameters

[in] **ulConnectorIndex** Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] **ulChannelIndex** Index of the channel. Set to 0

[in] **ulMrsCode** The MRS-code corresponding to the memory area that you want to select (see page 31/121 and 84/121 of EnDat specifications)

- **0xB9** Operating status (address area: 0x0 - 0x3)
- **0xA1** Parameters of the encoder manufacturer - first part (address area: 0x4 - 0xF)
- **0xA3** Parameters of the encoder manufacturer - second part (address area: 0x0 - 0xF)
- **0xA5** Parameters of the encoder manufacturer - third part (address area: 0x0 - 0xF)
- **0xA7** Operating parameters (address area: 0x0 - 0xF)
- **0xA9** Parameters of the OEM - first part (address area: depending on the sensor)
- **0xAB** Parameters of the OEM - second part (address area: depending on the sensor)
- **0xAD** Parameters of the OEM - third part (address area: depending on the sensor)
- **0xAF** Parameters of the OEM - fourth part (address area: depending on the sensor)
- **0xB1** Compensation values of the encoder manufacturer - first part (address area: depending on the sensor)
- **0xB3** Compensation values of the encoder manufacturer - second part (address area: depending on the sensor)
- **0xB5** Compensation values of the encoder manufacturer - third part (address area: depending on the sensor)
- **0xB7** Compensation values of the encoder manufacturer - fourth part (address area: depending on the sensor)

- **0xBD** Parameters of the encoder manufacturer for EnDat 2.2 (address area: 0x0 - 0x3F)
 - **0xBB** Operating parameters 2 (address area: depending on the sensor)
 - **0xBF** Parameters of the section 2 memory area (address area: depending on the sensor)
- [in] **ulAddress** Selected block address (only used if ulMrsCode is set to 0xBF. If ulMrsCode is not 0xBF, set it to 0). It is used to address further section 2 memory areas (see page 46/121 of EnDat specifications)
- [in] **ulOption01** Reserved. Set to 0
- [in] **ulOption02** Reserved. Set to 0
- [in] **ulOption03** Reserved. Set to 0
- [in] **ulOption04** Reserved. Set to 0
- [out] **Response sResponse.iReturnValue**
- **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The ulConnectorIndex parameter is wrong
 - **-4** The ulChannelIndex parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be INITIALISED)
 - **-7** The ulMrsCode parameter is wrong
 - **-8** The ulAddress parameter is wrong
 - **-9** The sensor is not compatible with EnDat 2.2
 - **-10** Error while getting position and selecting memory area
 - **-41** Transmission error. Please call MSXE17xx__MFEndatGetErrorSources to get more information
 - **-100** Internal system error occurred. See value of syserrno
- sResponse.syserrno** system-error code (the value of the libc "errno" code)
- ulPositionLow** Position - low bits
- ulPositionHigh** Position - high bits

Return values

- 0** SOAP_OK
- Others** See SOAP error

4.1.3.95 `int MSXE17xx__MFEndatSelectAdditionalData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulMFChannelIndex, xsd__unsignedLong ulAddDataCount, xsd__unsignedLong ulMrsCodeAD1, xsd__unsignedLong ulMrsCodeAD2, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__Response * Response)`

Some additional data are not available on all sensors. To get the available additional data of your sensor, please use the function MSXE17xx__MFEndatGetSensorProperties.

If you select an additional data that is not available on your sensor, you will get the parameter ucErrorSrc13 set to 1 when calling the function MSXE17xx__MFEndatGetErrorSources.

The additional data are extra values that the sensor can send (in the same cycle as its position value).

This function is reserved for EnDat 2.2 sensors. It will returns an error if the sensor does not support EnDat 2.2 commands.

Parameters

- [in] ***ulConnectorIndex*** Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] ***ulChannelIndex*** Index of the channel. Set to 0
- [in] ***ulAddDataCount*** The number of selected additional data (0 to 2)
- [in] ***ulMrsCodeAD1*** The MRS-Code for the first additional data
- **0x40** Send additional info 1 without data contents
 - **0x42** Position value 2 word 1 LSB
 - **0x43** Position value 2 word 2
 - **0x44** Position value 2 word 3 MSB
 - **0x49** Test values word 1 LSB
 - **0x4A** Test values word 2
 - **0x4B** Test values word 3 MSB
 - **0x4C** Temperature sensor 1 (external)
 - **0x4D** Temperature sensor 2 (external)
 - **0x4E** Additional sensors
- [in] ***ulMrsCodeAD2*** The MRS-Code for the second additional data
- **0x50** Send additional datum 2 without data contents
 - **0x51** Commutation
 - **0x52** Acceleration
 - **0x54** Limit position signals
 - **0x56** Asynchronous position value word 1 LSB
 - **0x57** Asynchronous position value word 2
 - **0x58** Asynchronous position value word 3 MSB
 - **0x59** Operating status error sources
 - **0x5B** Timestamp
- [in] ***ulOption01*** Reserved. Set to 0
- [in] ***ulOption02*** Reserved. Set to 0
- [in] ***ulOption03*** Reserved. Set to 0
- [in] ***ulOption04*** Reserved. Set to 0
- [out] ***Response iReturnValue***
- **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working
 - **-3** The *ulConnectorIndex* parameter is wrong
 - **-4** The *ulChannelIndex* parameter is wrong
 - **-5** The component is not programmed as EnDat
 - **-6** The driver is in a wrong state (must be INITIALISED)
 - **-7** The *ulAddDataCount* parameter is wrong
 - **-8** The *ulMrsCodeAD1* parameter is wrong
 - **-9** The *ulMrsCodeAD2* parameter is wrong
 - **-10** The sensor is not compatible with EnDat 2.2
 - **-11** Error while deactivating additional data 2
 - **-12** Error while deactivating additional data 1
 - **-13** Error while activating additional data 1
 - **-14** Error while deactivating additional data 2

- **-15** Error while deactivating additional data 1
- **-16** Selected additional data 1 is wrong or not available on this sensor. Please call MSXE17xx__MFEndatGetErrorSources to get more information
- **-17** Error while activating additional data 1. Please call MSXE17xx__MFEndatGetErrorSources to get more information
- **-18** Error while getting the current position value
- **-19** Error while activating additional data 2
- **-20** Error while deactivating additional data 2
- **-21** Error while deactivating additional data 1
- **-22** Selected additional data 2 is wrong or not available on this sensor. Please call MSXE17xx__MFEndatGetErrorSources to get more information
- **-23** Error while activating additional data 2. Please call MSXE17xx__MFEndatGetErrorSources to get more information
- **-24** Error while getting the current position value
- **-41** Transmission error. Please call MSXE17xx__MFEndatGetErrorSources to get more information
- **-100** Internal system error occurred. See value of syserrno

syserrno system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.96 `int MSXE17xx__MFEndatGetPositionWithAddData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__MFEndatGetPositionWithAddDataResponse * Response)`

Before calling this function, you must call the MSXE17xx__MFEndatInitSensor function.

You must also call the function MSXE17xx__MFEndatSelectAdditionalData to select the additional data that you want to receive.

This function is reserved for EnDat 2.2 sensors. It will returns an error if the sensor does not support EnDat 2.2 commands.

Parameters

- [in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.
- [in] *ulChannelIndex* Index of the channel. Set to 0
- [in] *ulOption01* Reserved. Set to 0
- [in] *ulOption02* Reserved. Set to 0
- [in] *ulOption03* Reserved. Set to 0
- [in] *ulOption04* Reserved. Set to 0
- [out] *Response sResponse.iReturnValue*
 - **0** The remote function performed OK
 - **-1** System error occurred
 - **-2** The PLD is not working

- **-3** The *ulConnectorIndex* parameter is wrong
- **-4** The *ulChannelIndex* parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** Error while reading the position
- **-41** Transmission error. Please call *MSXE17xx__MFEndatGetErrorSources* to get more information
- **-100** Internal system error occurred. See value of *syserrno*

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulPositionLow Position - low bits

ulPositionHigh Position - high bits

ulAddData1 Value of the additional data 1

ulAddData2 Value of the additional data 2

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.97 `int MSXE17xx__MFEndatGetSelectedAdditionalData (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__MFEndatGetSelectedAdditionalDataResponse * Response)`

The additional data are selected using the function *MSXE17xx__MFEndatSelectAdditionalData*.

Parameters

[in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] *ulChannelIndex* Index of the channel. Set to 0

[in] *ulOption01* Reserved. Set to 0

[in] *ulOption02* Reserved. Set to 0

[in] *ulOption03* Reserved. Set to 0

[in] *ulOption04* Reserved. Set to 0

[out] *Response sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The *ulConnectorIndex* parameter is wrong
- **-4** The *ulChannelIndex* parameter is wrong
- **-5** The component is not programmed as EnDat
- **-100** Internal system error occurred. See value of *syserrno*

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulAdCount Number of selected additional data

ulMrsCodeAD1 MRS code of the additional data 1

ulMrsCodeAD2 MRS code of the additional data 2

Return values*0* SOAP_OK*Others* See SOAP error

4.1.3.98 `int MSXE17xx__MfEndatInitAndEnableLatchPositionValues (xsd__unsignedLong ulmFModuleIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulLatchSource, xsd__unsignedLong ulTriggerEdgeCount, xsd__unsignedLong ulDataFormat, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__Response * Response)`

Before calling this function, you must call the MSXE17xx__MfEndatInitSensor function.

When the selected trigger occurs, the position value of the selected EnDat channel will be measured and send through the dataserver.

If you chose additional data using the function MSXE17xx__MfEndatSelectAdditionalData and if your selected ulDataFormat enables it, you will also receive the value of the selected additional data.

Note: using a synchro timer (and activating the synchro trigger as latch source) enables to get position of the EnDat sensor at a defined rate.

Parameters

[in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] *ulChannelIndex* Index of the channel. Set to 0

[in] *ulLatchSource* Mask of bits, that defines the trigger source.

Bit 0: Hardware trigger If you select the hardware trigger, you must also choose the edge detection (Bit 2 and 3). Of course, you can choose both.

- **1** activated
- **0** not used

Bit 1 : Synchro trigger

- **1** activated
- **0** not used

Bit 2 : Trigger rising edge (only if hardware trigger is selected)

- **1** activated
- **0** not used

Bit 3 : Trigger falling edge (only if hardware trigger is selected)

- **1** activated
- **0** not used

```
ulLatchSource = 2 (0b10)      -> the latch source is the synchro trigger
ulLatchSource = 5 (0b101)    -> the latch source is a rising edge of the hardware trigger
ulLatchSource = 13 (0b1101) -> the latch source is a rising or a falling edge of the hardware trigger
```

[in] *ulTriggerEdgeCount* Number of edges required to detect an hardware trigger (only if hardware trigger is selected)

[in] **ulDataFormat** Mask of bits, that defines the format of data that will be sent by the data server.

It is a combination of bits. You can set all the bits to 1 if you want all the informations (0b11111 = 31).

You can also, for example, only wants the timestamp and the digital I/Os state (0b11 = 3).

You can also choose, for example, to get the additional data 2 (0b1000 = 8).

Bit 0: Timestamp

- 0 Not sent
- 1 timestamp sent by the dataserver

Bit 1: Digital I/Os state

- 0 Not sent
- 1 digital I/Os state sent by the data server

Bit 2: Additional data 1

- 0 Not sent
- 1 Additional data 1 sent by the data server

Bit 3: Additional data 2

- 0 Not sent
- 1 Additional data 2 sent by the data server

Bit 4: Format of the value

- 0 Raw value (as sent by the sensor)
- 1 Standardised value. The format of the frame will then depends on the model of the sensor. See system documentation.

If ulDataFormat is set to 0, the frame send by the dataserver will have the following definition

```
unsigned long eventsrc;           // High 16 bits (MSB): Channel concerned
    (0 to 3)

                                // Low 16 bits (LSB): Bit mask representing the source of the event
                                //      Bit 0: Hardware trigger
                                //      Bit 1: Synchro trigger
                                //      Bit 2: Compare

unsigned long positionlow;        // Low bits of the position
unsigned long positionhigh;       // High bits of the position
unsigned long error;              // Status of the communication. Same value as the value returned by the function MSXE17xx_MFEndatGetErrorSources
```

If you set ulDataFormat to 1 (send timestamp), then the frame send by the dataserver will be changed. At the end of the "basic" frame, we add the timestamp.

```
unsigned long ts;                 // Second part of the timestamp
unsigned long tus;                // Microsecond part of the timestamp
```

If you set ulDataFormat to 2 = 0b10 (send digital I/Os state), then the frame send by the dataserver will be changed. At the end of the "basic" frame, we add the state of the digital I/Os.

```
unsigned long digiostate;         // State of the digital I/Os. Bit mask that encodes all digital I/Os.

                                // 0b0
                                // 0b100
                                // 0b1111111111111111
                                //      : All I/Os are set to high
```

If you set `ulDataFormat` to 4 = 0b100 (send additional data 1), then the frame send by the dataserver will be changed. At the end of the "basic" frame, we add the value of the first additional data.

```

unsigned long ad1value;
\encode
If you set ulDataFormat to 8 = 0b1000 (send additional data 2), then the frame s
end by the dataserver will be changed. At the end of the "basic" frame, we add th
e value of the second additional data. \n
\code
unsigned long ad2value;
\encode
If you set the bits 0, 1, 2 and 3 to 1, the format of the frame will then be: \n

\code
unsigned long eventsrc;           // High 16 bits (MSB): Channel concerned
    (0 to 3)

                                // Low 16 bits (LSB): Bit mask representing the source of the event
                                //      Bit 0: Hardware trigger
                                //      Bit 1: Synchro trigger
                                //      Bit 2: Compare

unsigned long positionlow;        // Low bits of the position
unsigned long positionhigh;      // High bits of the position
unsigned long error;              // Status of the communication. Same value as the value returned by the function MSXE17xx__MFEndatGetErrorSources
unsigned long ts;                 // Second part of the timestamp
unsigned long tus;                // Microsecond part of the timestamp
unsigned long digiostate;         // State of the digital I/Os. Bit mask that encodes all digital I/Os.

                                // 0b0
    All I/Os are set to low
                                // 0b100
    I/O 2 is set to high, all others are set to 0
                                // 0b11111111111111
    1111 : All I/Os are set to high
unsigned long ad1value;
unsigned long ad2value;
```

The bit 4 of the `ulDataFormat` is more complex. The format of the frame that is sent by the dataserver will depend on the model of the sensor used. See the system documentation for more information, or the "Acquisition" menu of the web site.

[in] ***ulOption01*** Reserved. Set to 0

[in] ***ulOption02*** Reserved. Set to 0

[in] ***ulOption03*** Reserved. Set to 0

[in] ***ulOption04*** Reserved. Set to 0

[out] ***Response iReturnValue***

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The `ulConnectorIndex` parameter is wrong
- **-4** The `ulChannelIndex` parameter is wrong
- **-5** The component is not programmed as `EnDat`
- **-6** The driver is in a wrong state (must be INITIALISED)
- **-7** The `ulLatchSource` parameter is wrong
- **-8** The `ulDataFormat` parameter is wrong

- **-9** Error while getting sensor properties
 - **-10** The multiturn part size of the sensor is 0
 - **-11** The singleturn part size of the sensor is 0
 - **-12** The step per revolution properties of the sensor is 0
 - **-41** Transmission error. Please call MSXE17xx__MFEndatGetErrorSources to get more information
 - **-100** Internal system error occurred. See value of syserrno
- syserrno* system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.99 `int MSXE17xx__MFEndatGetCurrentLatchConfiguration (xsd__unsignedLong ulmFModuleIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse * Response)`

Parameters

[in] *ulConnectorIndex* Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] *ulChannelIndex* Index of the channel. Set to 0

[in] *ulOption01* Reserved. Set to 0

[in] *ulOption02* Reserved. Set to 0

[in] *ulOption03* Reserved. Set to 0

[in] *ulOption04* Reserved. Set to 0

[out] *Response sResponse.iReturnValue*

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The *ulConnectorIndex* parameter is wrong
- **-3** The *ulChannelIndex* parameter is wrong
- **-4** The component is not programmed as EnDat
- **-100** Internal system error occurred. See value of syserrno

sResponse.syserrno system-error code (the value of the libc "errno" code)

ulRunning The running state.

- **0** Not running (do not use other return parameters, there will be set to 0)
- **1** Latch logic is running

ulLatchSource Mask of bits, that defines the trigger source. See documentation of MSXE17xx__MFEndatInitAndEnableLatchPositionValues

ulTriggerEdgeCount Number of edges required to detect an hardware trigger (only if hardware trigger is selected)

ulDataFormat Mask of bits, that defines the format of data that will be sent by the data server. See documentation of MSXE17xx__MFEndatInitAndEnableLatchPositionValues

ulModel (Used for computation) Model/Type of the sensor (see page 79/84 of EnDat application notes 722024)

- **0, 1, 2, 3** Incremental linear encoder
- **4, 6** Absolute linear encoder
- **8, 9, 10, 11** Incremental rotary encoder or angle encoder
- **12** Singleturn encoder
- **13, 14** Multiturn encoder

ulPositionSize (Used for computation) Size of the position value send by the sensor (in bits)

ulSignalPeriod (Used for computation) Signal period length or signal periods per revolution for incremental output signals

ulStepPerRevolution (Used for computation) Measuring step length or measuring steps per revolution with serial data transfer

ulNumberOfRevolution (Used for computation) Distinguishable revolutions - only for multiturn encoders

ulScalingFactor (Used for computation) Scaling factor for resolution

Return values

0 SOAP_OK

Others See SOAP error

4.1.3.100 `int MSXE17xx_MFEndatDisableAndReleaseLatchPositionValues (xsd__unsignedLong ulConnectorIndex, xsd__unsignedLong ulChannelIndex, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE17xx_Response * Response)`

Parameters

[in] ***ulConnectorIndex*** Index of the EnDat connector (0 to 3). See on the MSX-E system.

[in] ***ulChannelIndex*** Index of the channel. Set to 0

[in] ***ulOption01*** Reserved. Set to 0

[in] ***ulOption02*** Reserved. Set to 0

[in] ***ulOption03*** Reserved. Set to 0

[in] ***ulOption04*** Reserved. Set to 0

[out] ***Response iReturnValue***

- **0** The remote function performed OK
- **-1** System error occurred
- **-2** The PLD is not working
- **-3** The ulConnectorIndex parameter is wrong
- **-4** The ulChannelIndex parameter is wrong
- **-5** The component is not programmed as EnDat
- **-6** The driver is in a wrong state (must be LATCH_RUNNING or LATCH_FIFO_OVERFLOW)
- **-100** Internal system error occurred. See value of syserrno

syserrno system-error code (the value of the libc "errno" code)

Return values

0 SOAP_OK

Others See SOAP error

Index

- `__offset`
 - ByteArray, [101](#)
 - UnsignedLongArray, [130](#)
 - UnsignedShortArray, [131](#)
 - `__ptr`
 - ByteArray, [101](#)
 - UnsignedLongArray, [130](#)
 - UnsignedShortArray, [131](#)
 - xsd__base64Binary, [131](#)
 - `__size`
 - ByteArray, [101](#)
 - UnsignedLongArray, [130](#)
 - UnsignedShortArray, [131](#)
 - xsd__base64Binary, [131](#)
- Analog
 - MXCommon__SetFilterChannels, [30](#)
- bArray
 - MXCommon__-
 - GetAutoConfigurationFileResponse, [124](#)
- bCryptedValueArray
 - MXCommon__TestCustomerIDResponse, [129](#)
- bValueArray
 - MXCommon__TestCustomerIDResponse, [129](#)
- ByteArray, [101](#)
 - `__offset`, [101](#)
 - `__ptr`, [101](#)
 - `__size`, [101](#)
- Common functions, [3](#)
- Common general functions, [4](#)
- Common hardware trigger functions, [12](#)
- Common I/O auto configuration functions, [18](#)
- Common security functions, [14](#)
- Common synchronisation timer functions, [20](#)
- Common temperature functions, [10](#)
- Common time functions, [16](#)
- Common_autoconf
 - MXCommon__GetAutoConfigurationFile, [19](#)
 - MXCommon__SetAutoConfigurationFile, [19](#)
 - MXCommon__StartAutoConfiguration, [20](#)
- Common_configuration
 - MXCommon__-
 - ApplyConfigurationBackupFile, [23](#)
- MXCommon__ChangePassword, [24](#)
- MXCommon__GetConfigurationBackupFile, [23](#)
- Common_general
 - MXCommon__DataseverRestart, [8](#)
 - MXCommon__GetClientConnections, [6](#)
 - MXCommon__GetEthernetLinksStates, [9](#)
 - MXCommon__GetHostname, [5](#)
 - MXCommon__GetModuleType, [5](#)
 - MXCommon__Reboot, [8](#)
 - MXCommon__ResetAllIOFunctionalities, [8](#)
 - MXCommon__SetHostname, [6](#)
 - MXCommon__Strerror, [6](#)
- Common_hardware_trigger
 - MXCommon__-
 - GetHardwareTriggerFilterTime, [13](#)
 - MXCommon__GetHardwareTriggerState, [13](#)
 - MXCommon__-
 - SetHardwareTriggerFilterTime, [12](#)
- Common_security
 - MXCommon__SetCustomerKey, [15](#)
 - MXCommon__TestCustomerID, [15](#)
- Common_synchrotimer
 - MXCommon__InitAndStartSynchroTimer, [21](#)
 - MXCommon__-
 - StopAndReleaseSynchroTimer, [22](#)
- Common_temperature
 - MXCommon__-
 - GetModuleTemperatureValueAndStatus, [11](#)
 - MXCommon__-
 - SetModuleTemperatureWarningLevels, [11](#)
- Common_time
 - MXCommon__GetTime, [17](#)
 - MXCommon__GetUpTime, [18](#)
 - MXCommon__HardwareClockToSys, [17](#)
 - MXCommon__SetTime, [16](#)
 - MXCommon__SysToHardwareClock, [17](#)
- Customer option management, [27](#)
- CustomerOption
 - MXCommon__GetOptionInformation, [28](#)
- DefaultResponse, [101](#)
 - iReturnValue, [102](#)

- syserrno, [102](#)
- dTemperatureValue
 - MXCommon__-
 - GetModuleTemperatureValueAndStatusResponsesResponse, [103](#)
 - [127](#)
- input filter Filter management, [29](#)
- iReturnValue
 - DefaultResponse, [102](#)
 - MSXE173x__Response, [111](#)
 - MSXE17xx__Response, [121](#)
 - MXCommon__Response, [128](#)
- MSX-E systems servers, [99](#)
- MSX-E173x compatibility functions, [3](#)
- MSX-E173x digital I/O functions, [30](#)
- MSX-E173x ENDAT functions, [38](#)
- MSX-E173x IO watchdog functions, [34](#)
- MSX-E173x multifunction common functions, [36](#)
- MSX-E17xx digital I/O functions, [60](#)
- MSX-E17xx ENDAT functions, [70](#)
- MSX-E17xx functions, [3](#)
- MSX-E17xx IO watchdog functions, [65](#)
- MSX-E17xx multifunction common functions, [67](#)
- MSX-E17xx multifunction functions, [3](#)
- MSXE170X_COUNTER_DECREMENT
 - MSXE173x_public_doc.h, [147](#)
- MSXE170X_COUNTER_DIRECT_MODE
 - MSXE173x_public_doc.h, [146](#)
- MSXE170X_COUNTER_DOUBLE_MODE
 - MSXE173x_public_doc.h, [146](#)
- MSXE170X_COUNTER_HIGH_EDGE_-
 - LATCH_AND_CLEAR_COUNTER
 - MSXE173x_public_doc.h, [147](#)
- MSXE170X_COUNTER_HIGH_EDGE_-
 - LATCH_COUNTER
 - MSXE173x_public_doc.h, [147](#)
- MSXE170X_COUNTER_HYSTERESIS_OFF
 - MSXE173x_public_doc.h, [146](#)
- MSXE170X_COUNTER_HYSTERESIS_ON
 - MSXE173x_public_doc.h, [146](#)
- MSXE170X_COUNTER_INCREMENT
 - MSXE173x_public_doc.h, [147](#)
- MSXE170X_COUNTER_LOW_EDGE_LATCH_-
 - AND_CLEAR_COUNTER
 - MSXE173x_public_doc.h, [147](#)
- MSXE170X_COUNTER_LOW_EDGE_LATCH_-
 - COUNTER
 - MSXE173x_public_doc.h, [147](#)
- MSXE170X_COUNTER_QUADRUPLE_MODE
 - MSXE173x_public_doc.h, [146](#)
- MSXE170X_COUNTER_SIMPLE_MODE
 - MSXE173x_public_doc.h, [146](#)
- MSXE173x__DigitalIOGetNumber
 - MSXE173x_DigIO, [31](#)
 - MSXE173x_public_doc.h, [165](#)
- MSXE173x__DigitalIOGetNumberResponse, [102](#)
- ulNumberOfDigitalIO, [103](#)
- MSXE173x__DigitalIOInitPortConfiguration
 - MSXE173x_DigIO, [31](#)
 - MSXE173x_public_doc.h, [165](#)
- MSXE173x__DigitalIOReadAllChannelsValue
 - MSXE173x_DigIO, [32](#)
 - MSXE173x_public_doc.h, [166](#)
- MSXE173x__DigitalIOReadChannelValue
 - MSXE173x_DigIO, [31](#)
 - MSXE173x_public_doc.h, [166](#)
- MSXE173x__DigitalIORearmShortCircuit
 - MSXE173x_DigIO, [34](#)
 - MSXE173x_public_doc.h, [168](#)
- MSXE173x__DigitalIOReleasePortConfiguration
 - MSXE173x_DigIO, [33](#)
 - MSXE173x_public_doc.h, [167](#)
- MSXE173x__DigitalIOTestShortCircuit
 - MSXE173x_DigIO, [33](#)
 - MSXE173x_public_doc.h, [168](#)
- MSXE173x__DigitalIOWriteAllChannelsValue
 - MSXE173x_DigIO, [33](#)
 - MSXE173x_public_doc.h, [167](#)
- MSXE173x__DigitalIOWriteChannelValue
 - MSXE173x_DigIO, [32](#)
 - MSXE173x_public_doc.h, [166](#)
- MSXE173x__IOWatchdogGetStatusAndValue
 - MSXE173x_public_doc.h, [169](#)
 - MSXE173x_Watchdog, [36](#)
- MSXE173x__IOWatchdogGetStatusAndValueResponse, [103](#)
- sResponse, [103](#)
- ulInfo, [103](#)
- ulStatus, [103](#)
- ulValue, [103](#)
- MSXE173x__IOWatchdogInitAndStart
 - MSXE173x_public_doc.h, [168](#)
 - MSXE173x_Watchdog, [35](#)
- MSXE173x__IOWatchdogStopAndRelease
 - MSXE173x_public_doc.h, [169](#)
 - MSXE173x_Watchdog, [35](#)
- MSXE173x__MFCommonReferenceVoltageActivation
 - MSXE173x_MF_Common, [38](#)
 - MSXE173x_public_doc.h, [171](#)
- MSXE173x__MFCommonSetInputsFilter
 - MSXE173x_MF_Common, [37](#)
 - MSXE173x_public_doc.h, [170](#)
- MSXE173x__MFEndatDisableAndReleaseLatchPositionValues
 - MSXE173x_MF_Endat, [59](#)
 - MSXE173x_public_doc.h, [190](#)
- MSXE173x__MFEndatGetCurrentLatchConfiguration

- MSXE173x_MF_Endat, [58](#)
- MSXE173x_public_doc.h, [189](#)
- MSXE173x__MFEndatGetCurrentLatchConfigurationResponse, [103](#)
 - sResponse, [105](#)
 - ulDataFormat, [105](#)
 - ulLatchSource, [105](#)
 - ulModel, [105](#)
 - ulNumberOfRevolution, [105](#)
 - ulPositionSize, [105](#)
 - ulRunning, [105](#)
 - ulScalingFactor, [105](#)
 - ulSignalPeriod, [105](#)
 - ulStepPerRevolution, [105](#)
 - ulTriggerEdgeCount, [105](#)
- MSXE173x__MFEndatGetErrorSources
 - MSXE173x_MF_Endat, [45](#)
 - MSXE173x_public_doc.h, [176](#)
- MSXE173x__MFEndatGetErrorSourcesResponse, [105](#)
 - sResponse, [106](#)
 - ulErrorSrc, [106](#)
- MSXE173x__MFEndatGetPosition
 - MSXE173x_MF_Endat, [42](#)
 - MSXE173x_public_doc.h, [173](#)
- MSXE173x__MFEndatGetPositionResponse, [106](#)
 - sResponse, [106](#)
 - ulPositionHigh, [106](#)
 - ulPositionLow, [106](#)
- MSXE173x__MFEndatGetPositionWithAddData
 - MSXE173x_MF_Endat, [54](#)
 - MSXE173x_public_doc.h, [185](#)
- MSXE173x__MFEndatGetPositionWithAddDataResponse, [106](#)
 - sResponse, [107](#)
 - ulAddData1, [107](#)
 - ulAddData2, [107](#)
 - ulPositionHigh, [107](#)
 - ulPositionLow, [107](#)
- MSXE173x__MFEndatGetSelectedAdditionalData
 - MSXE173x_MF_Endat, [54](#)
 - MSXE173x_public_doc.h, [185](#)
- MSXE173x__MFEndatGetSelectedAdditionalDataResponse, [107](#)
 - sResponse, [108](#)
 - ulAdCount, [108](#)
 - ulMrsCodeAD1, [108](#)
 - ulMrsCodeAD2, [108](#)
- MSXE173x__MFEndatGetSensorProperties
 - MSXE173x_MF_Endat, [42](#)
 - MSXE173x_public_doc.h, [173](#)
- MSXE173x__MFEndatGetSensorPropertiesResponse, [108](#)
 - sResponse, [110](#)
 - ulAdditionalData, [110](#)
 - ulIDNumberLsb, [110](#)
 - ulIDNumberMsb, [110](#)
 - ulMode, [110](#)
 - ulModel, [110](#)
 - ulNumberOfRevolution, [110](#)
 - ulPositionSize, [110](#)
 - ulScalingFactor, [110](#)
 - ulSerialNumberLsb, [110](#)
 - ulSerialNumberMsb, [110](#)
 - ulSignalPeriod, [110](#)
 - ulStepPerRevolution, [110](#)
- MSXE173x__MFEndatInitAndEnableLatchPositionValues
 - MSXE173x_MF_Endat, [55](#)
 - MSXE173x_public_doc.h, [186](#)
- MSXE173x__MFEndatInitSensor
 - MSXE173x_MF_Endat, [41](#)
 - MSXE173x_public_doc.h, [172](#)
- MSXE173x__MFEndatResetErrorBits
 - MSXE173x_MF_Endat, [46](#)
 - MSXE173x_public_doc.h, [177](#)
- MSXE173x__MFEndatSelectAdditionalData
 - MSXE173x_MF_Endat, [52](#)
 - MSXE173x_public_doc.h, [183](#)
- MSXE173x__MFEndatSelectMemoryArea
 - MSXE173x_MF_Endat, [47](#)
 - MSXE173x_public_doc.h, [178](#)
- MSXE173x__MFEndatSensorReceiveParameter
 - MSXE173x_MF_Endat, [49](#)
 - MSXE173x_public_doc.h, [180](#)
- MSXE173x__MFEndatSensorReceiveReset
 - MSXE173x_MF_Endat, [46](#)
- MSXE173x__MFEndatSensorSendParameter, [110](#)
 - sResponse, [111](#)
 - ulParam, [111](#)
- MSXE173x__MFEndatSensorSendPosAndRecvSelMemArea
 - MSXE173x_MF_Endat, [50](#)
 - MSXE173x_public_doc.h, [181](#)
- MSXE173x__MFEndatSensorSendPosAndRecvSelMemAreaResponse, [111](#)
 - sResponse, [111](#)
 - ulPositionHigh, [111](#)
 - ulPositionLow, [111](#)
- MSXE173x__Response, [111](#)
 - iReturnValue, [111](#)
 - syserrno, [112](#)
- MSXE173x__unsignedLongResponse, [112](#)
 - sResponse, [112](#)
 - ulValue, [112](#)

- MSXE173x__unsignedLongTimeStampResponse, 112
 - sResponse, 113
 - ulTimeStampHigh, 113
 - ulTimeStampLow, 113
 - ulValue, 113
- MSXE173x_DigIO
 - MSXE173x__DigitalIOGetNumber, 31
 - MSXE173x__DigitalIOInitPortConfiguration, 31
 - MSXE173x__-
 - DigitalIOReadAllChannelsValue, 32
 - MSXE173x__DigitalIOReadChannelValue, 31
 - MSXE173x__DigitalIORearmShortCircuit, 34
 - MSXE173x__-
 - DigitalIOReleasePortConfiguration, 33
 - MSXE173x__DigitalIOTestShortCircuit, 33
 - MSXE173x__-
 - DigitalIOWriteAllChannelsValue, 33
 - MSXE173x__DigitalIOWriteChannelValue, 32
- MSXE173x_MF_Common
 - MSXE173x__-
 - MFCommonReferenceVoltageActivation, 38
 - MSXE173x__MFCommonSetInputsFilter, 37
- MSXE173x_MF_Endat
 - MSXE173x__-
 - MFEndatDisableAndReleaseLatchPositionValues, 59
 - MSXE173x__-
 - MFEndatGetCurrentLatchConfiguration, 58
 - MSXE173x__MFEndatGetErrorSources, 45
 - MSXE173x__MFEndatGetPosition, 42
 - MSXE173x__-
 - MFEndatGetPositionWithAddData, 54
 - MSXE173x__-
 - MFEndatGetSelectedAdditionalData, 54
 - MSXE173x__MFEndatGetSensorProperties, 42
 - MSXE173x__-
 - MFEndatInitAndEnableLatchPositionValues, 55
 - MSXE173x__MFEndatInitSensor, 41
 - MSXE173x__MFEndatResetErrorBits, 46
 - MSXE173x__MFEndatSelectAdditionalData, 52
 - MSXE173x__MFEndatSelectMemoryArea, 47
- MSXE173x__-
 - MFEndatSensorReceiveParameter, 49
- MSXE173x__MFEndatSensorReceiveReset, 46
- MSXE173x__MFEndatSensorSendParameter, 48
- MSXE173x__-
 - MFEndatSensorSendPosAndRecvSelMemArea, 50
- MSXE173x_public_doc.h, 133
- MSXE170X_COUNTER_DECREMENT, 147
- MSXE170X_COUNTER_DIRECT_MODE, 146
- MSXE170X_COUNTER_DOUBLE_MODE, 146
- MSXE170X_COUNTER_HIGH_EDGE_-LATCH_AND_CLEAR_COUNTER, 147
- MSXE170X_COUNTER_HIGH_EDGE_-LATCH_COUNTER, 147
- MSXE170X_COUNTER_HYSTERESIS_-OFF, 146
- MSXE170X_COUNTER_HYSTERESIS_-ON, 146
- MSXE170X_COUNTER_INCREMENT, 147
- MSXE170X_COUNTER_LOW_EDGE_-LATCH_AND_CLEAR_COUNTER, 147
- MSXE170X_COUNTER_LOW_EDGE_-LATCH_COUNTER, 147
- MSXE170X_COUNTER_QUADRUPLE_-MODE, 146
- MSXE170X_COUNTER_SIMPLE_MODE, 146
- MSXE173x__DigitalIOGetNumber, 165
- MSXE173x__DigitalIOInitPortConfiguration, 165
- MSXE173x__-
 - DigitalIOReadAllChannelsValue, 166
- MSXE173x__DigitalIOReadChannelValue, 166
- MSXE173x__DigitalIORearmShortCircuit, 168
- MSXE173x__-
 - DigitalIOReleasePortConfiguration, 167
- MSXE173x__DigitalIOTestShortCircuit, 168
- MSXE173x__-
 - DigitalIOWriteAllChannelsValue, 167
- MSXE173x__DigitalIOWriteChannelValue, 166
- MSXE173x__-

- IOWatchdogGetStatusAndValue, [169](#)
- MSXE173x__IOWatchdogInitAndStart, [168](#)
- MSXE173x__IOWatchdogStopAndRelease, [169](#)
- MSXE173x__-
 - MFCCommonReferenceVoltageActivation, [171](#)
- MSXE173x__MFCCommonSetInputsFilter, [170](#)
- MSXE173x__-
 - MFEndatDisableAndReleaseLatchPositionValues, [190](#)
- MSXE173x__-
 - MFEndatGetCurrentLatchConfiguration, [189](#)
- MSXE173x__MFEndatGetErrorSources, [176](#)
- MSXE173x__MFEndatGetPosition, [173](#)
- MSXE173x__-
 - MFEndatGetPositionWithAddData, [185](#)
- MSXE173x__-
 - MFEndatGetSelectedAdditionalData, [185](#)
- MSXE173x__MFEndatGetSensorProperties, [173](#)
- MSXE173x__-
 - MFEndatInitAndEnableLatchPositionValues, [186](#)
- MSXE173x__MFEndatInitSensor, [172](#)
- MSXE173x__MFEndatResetErrorBits, [177](#)
- MSXE173x__MFEndatSelectAdditionalData, [183](#)
- MSXE173x__MFEndatSelectMemoryArea, [178](#)
- MSXE173x__-
 - MFEndatSensorReceiveParameter, [180](#)
- MSXE173x__MFEndatSensorReceiveReset, [177](#)
- MSXE173x__MFEndatSensorSendParameter, [179](#)
- MSXE173x__-
 - MFEndatSensorSendPosAndRecvSelMemArea, [181](#)
- MSXE17xx__DigitalIOGetNumber, [191](#)
- MSXE17xx__DigitalIOInitPortConfiguration, [191](#)
- MSXE17xx__-
 - DigitalIOReadAllChannelsValue, [192](#)
- MSXE17xx__DigitalIOReadChannelValue, [192](#)
- MSXE17xx__DigitalIORearmShortCircuit, [194](#)
- MSXE17xx__-
 - DigitalIOReleasePortConfiguration, [194](#)
- MSXE17xx__DigitalIOTestShortCircuit, [194](#)
- MSXE17xx__-
 - DigitalIOWriteAllChannelsValue, [193](#)
- MSXE17xx__DigitalIOWriteChannelValue, [193](#)
- MSXE17xx__-
 - IOWatchdogGetStatusAndValue, [196](#)
- MSXE17xx__IOWatchdogInitAndStart, [195](#)
- MSXE17xx__IOWatchdogStopAndRelease, [195](#)
- MSXE17xx__-
 - MFCCommonGetSubModuleFunctionality, [196](#)
- MSXE17xx__-
 - MFCCommonReferenceVoltageActivation, [198](#)
- MSXE17xx__MFCCommonSetFIFO0Level, [198](#)
- MSXE17xx__MFCCommonSetInputsFilter, [197](#)
- MSXE17xx__-
 - MFEndatDisableAndReleaseLatchPositionValues, [218](#)
- MSXE17xx__-
 - MFEndatGetCurrentLatchConfiguration, [217](#)
- MSXE17xx__MFEndatGetErrorSources, [203](#)
- MSXE17xx__MFEndatGetPosition, [200](#)
- MSXE17xx__-
 - MFEndatGetPositionWithAddData, [212](#)
- MSXE17xx__-
 - MFEndatGetSelectedAdditionalData, [213](#)
- MSXE17xx__MFEndatGetSensorProperties, [201](#)
- MSXE17xx__-
 - MFEndatInitAndEnableLatchPositionValues, [214](#)
- MSXE17xx__MFEndatInitSensor, [199](#)
- MSXE17xx__MFEndatResetErrorBits, [204](#)
- MSXE17xx__MFEndatSelectAdditionalData, [210](#)
- MSXE17xx__MFEndatSelectMemoryArea, [205](#)
- MSXE17xx__-
 - MFEndatSensorReceiveParameter, [208](#)
- MSXE17xx__MFEndatSensorReceiveReset, [205](#)
- MSXE17xx__MFEndatSensorSendParameter, [207](#)

- MSXE17xx_-
 - MFEndatSensorSendPosAndRecvSelMemArea, 209
- MXCommon_-
 - ApplyConfigurationBackupFile, 160
- MXCommon__ChangePassword, 161
- MXCommon__DataseverRestart, 150
- MXCommon__GetAutoConfigurationFile, 157
- MXCommon__GetClientConnections, 148
- MXCommon__GetConfigurationBackupFile, 160
- MXCommon__GetEthernetLinksStates, 151
- MXCommon_-
 - GetHardwareTriggerFilterTime, 153
- MXCommon__GetHardwareTriggerState, 154
- MXCommon__GetHostname, 147
- MXCommon_-
 - GetModuleTemperatureValueAndStatus, 152
- MXCommon__GetModuleType, 147
- MXCommon__GetOptionInformation, 163
- MXCommon__GetStateIDFromName, 162
- MXCommon__GetStateNameFromID, 163
- MXCommon__GetSubsystemIDFromName, 162
- MXCommon__GetSubsystemNameFromID, 163
- MXCommon__GetSubSystemState, 161
- MXCommon__GetSynchronizationStatus, 164
- MXCommon__GetTime, 156
- MXCommon__GetUpTime, 157
- MXCommon__HardwareClockToSys, 156
- MXCommon__InitAndStartSynchroTimer, 158
- MXCommon__Reboot, 150
- MXCommon__ResetAllIOFunctionalities, 150
- MXCommon__SetAutoConfigurationFile, 158
- MXCommon__SetCustomerKey, 154
- MXCommon__SetFilterChannels, 165
- MXCommon_-
 - SetHardwareTriggerFilterTime, 153
- MXCommon__SetHostname, 148
- MXCommon_-
 - SetModuleTemperatureWarningLevels, 152
- MXCommon__SetTime, 155
- MXCommon__SetToMaster, 164
- MXCommon__StartAutoConfiguration, 158
- MXCommon_-
 - StopAndReleaseSynchroTimer, 159
- MXCommon__Strerror, 148
- MXCommon__SysToHardwareClock, 156
- MXCommon__TestCustomerID, 155
- xsd__char, 147
- xsd__double, 147
- xsd__float, 147
- xsd__int, 147
- xsd__long, 147
- xsd__string, 147
- xsd__unsignedByte, 147
- xsd__unsignedInt, 147
- xsd__unsignedLong, 147
- xsd__unsignedShort, 147
- MSXE173x_Watchdog
 - MSXE173x_-
 - IOWatchdogGetStatusAndValue, 36
 - MSXE173x__IOWatchdogInitAndStart, 35
 - MSXE173x__IOWatchdogStopAndRelease, 35
- MSXE17xx__DigitalIOGetNumber
 - MSXE173x_public_doc.h, 191
 - MSXE17xx_DigIO, 61
- MSXE17xx__DigitalIOGetNumberResponse, 113
 - sResponse, 113
 - ulNumberOfDigitalIO, 113
- MSXE17xx__DigitalIOInitPortConfiguration
 - MSXE173x_public_doc.h, 191
 - MSXE17xx_DigIO, 61
- MSXE17xx__DigitalIOReadAllChannelsValue
 - MSXE173x_public_doc.h, 192
 - MSXE17xx_DigIO, 62
- MSXE17xx__DigitalIOReadChannelValue
 - MSXE173x_public_doc.h, 192
 - MSXE17xx_DigIO, 62
- MSXE17xx__DigitalIORearmShortCircuit
 - MSXE173x_public_doc.h, 194
 - MSXE17xx_DigIO, 64
- MSXE17xx__DigitalIOReleasePortConfiguration
 - MSXE173x_public_doc.h, 194
 - MSXE17xx_DigIO, 64
- MSXE17xx__DigitalIOTestShortCircuit
 - MSXE173x_public_doc.h, 194
 - MSXE17xx_DigIO, 64
- MSXE17xx__DigitalIOWriteAllChannelsValue
 - MSXE173x_public_doc.h, 193
 - MSXE17xx_DigIO, 63
- MSXE17xx__DigitalIOWriteChannelValue
 - MSXE173x_public_doc.h, 193
 - MSXE17xx_DigIO, 63
- MSXE17xx__IOWatchdogGetStatusAndValue
 - MSXE173x_public_doc.h, 196
 - MSXE17xx_Watchdog, 66
- MSXE17xx__IOWatchdogGetStatusAndValueResponse, 113
 - sResponse, 114

- ulInfo, [114](#)
- ulStatus, [114](#)
- ulValue, [114](#)
- MSXE17xx__IOWatchdogInitAndStart
 - MSXE173x_public_doc.h, [195](#)
 - MSXE17xx_Watchdog, [65](#)
- MSXE17xx__IOWatchdogStopAndRelease
 - MSXE173x_public_doc.h, [195](#)
 - MSXE17xx_Watchdog, [66](#)
- MSXE17xx__MFCommonGetSubModuleFunctionality
 - MSXE173x_public_doc.h, [196](#)
 - MSXE17xx_MF_Common, [67](#)
- MSXE17xx__MFCommonReferenceVoltageActivation
 - MSXE173x_public_doc.h, [198](#)
 - MSXE17xx_MF_Common, [69](#)
- MSXE17xx__MFCommonSetFIFO0Level
 - MSXE173x_public_doc.h, [198](#)
 - MSXE17xx_MF_Common, [69](#)
- MSXE17xx__MFCommonSetInputsFilter
 - MSXE173x_public_doc.h, [197](#)
 - MSXE17xx_MF_Common, [68](#)
- MSXE17xx__MFEndatDisableAndReleaseLatchPositionValues
 - MSXE173x_public_doc.h, [218](#)
 - MSXE17xx_MF_Endat, [91](#)
- MSXE17xx__MFEndatGetCurrentLatchConfiguration
 - MSXE173x_public_doc.h, [217](#)
 - MSXE17xx_MF_Endat, [90](#)
- MSXE17xx__MFEndatGetCurrentLatchConfigurationResponse, [114](#)
 - sResponse, [115](#)
 - ulDataFormat, [115](#)
 - ulLatchSource, [115](#)
 - ulModel, [115](#)
 - ulNumberOfRevolution, [115](#)
 - ulPositionSize, [115](#)
 - ulRunning, [115](#)
 - ulScalingFactor, [115](#)
 - ulSignalPeriod, [115](#)
 - ulStepPerRevolution, [115](#)
 - ulTriggerEdgeCount, [115](#)
- MSXE17xx__MFEndatGetErrorSources
 - MSXE173x_public_doc.h, [203](#)
 - MSXE17xx_MF_Endat, [77](#)
- MSXE17xx__MFEndatGetErrorSourcesResponse, [115](#)
 - sResponse, [116](#)
 - ulErrorSrc, [116](#)
- MSXE17xx__MFEndatGetPosition
 - MSXE173x_public_doc.h, [200](#)
 - MSXE17xx_MF_Endat, [74](#)
- MSXE17xx__MFEndatGetPositionResponse, [116](#)
 - sResponse, [116](#)
 - ulPositionHigh, [116](#)
 - ulPositionLow, [116](#)
- MSXE17xx__MFEndatGetPositionWithAddData
 - MSXE173x_public_doc.h, [212](#)
 - MSXE17xx_MF_Endat, [86](#)
- MSXE17xx__MFEndatGetPositionWithAddDataResponse, [116](#)
 - sResponse, [117](#)
 - ulAddData1, [117](#)
 - ulAddData2, [117](#)
 - ulPositionHigh, [117](#)
 - ulPositionLow, [117](#)
- MSXE17xx__MFEndatGetSelectedAdditionalData
 - MSXE173x_public_doc.h, [213](#)
 - MSXE17xx_MF_Endat, [86](#)
- MSXE17xx__MFEndatGetSelectedAdditionalDataResponse, [117](#)
 - sResponse, [118](#)
 - ulAdCount, [118](#)
 - ulMrsCodeAD1, [118](#)
 - ulMrsCodeAD2, [118](#)
- MSXE17xx__MFEndatGetSensorProperties
 - MSXE173x_public_doc.h, [201](#)
 - MSXE17xx_MF_Endat, [74](#)
- MSXE17xx__MFEndatGetSensorPropertiesResponse, [118](#)
 - sResponse, [120](#)
 - ulAdditionalData, [120](#)
 - ulIDNumberLsb, [120](#)
 - ulIDNumberMsb, [120](#)
 - ulMode, [120](#)
 - ulModel, [120](#)
 - ulNumberOfRevolution, [120](#)
 - ulPositionSize, [120](#)
 - ulScalingFactor, [120](#)
 - ulSerialNumberLsb, [120](#)
 - ulSerialNumberMsb, [120](#)
 - ulSignalPeriod, [120](#)
 - ulStepPerRevolution, [120](#)
- MSXE17xx__MFEndatInitAndEnableLatchPositionValues
 - MSXE173x_public_doc.h, [214](#)
 - MSXE17xx_MF_Endat, [87](#)
- MSXE17xx__MFEndatInitSensor
 - MSXE173x_public_doc.h, [199](#)
 - MSXE17xx_MF_Endat, [73](#)
- MSXE17xx__MFEndatResetErrorBits
 - MSXE173x_public_doc.h, [204](#)
 - MSXE17xx_MF_Endat, [78](#)
- MSXE17xx__MFEndatSelectAdditionalData
 - MSXE173x_public_doc.h, [210](#)
 - MSXE17xx_MF_Endat, [84](#)
- MSXE17xx__MFEndatSelectMemoryArea
 - MSXE173x_public_doc.h, [205](#)
 - MSXE17xx_MF_Endat, [79](#)
- MSXE17xx__MFEndatSensorReceiveParameter
 - MSXE173x_public_doc.h, [208](#)

- MSXE17xx_MF_Endat, 81
- MSXE17xx__MFEndatSensorReceiveReset
 - MSXE173x_public_doc.h, 205
 - MSXE17xx_MF_Endat, 78
- MSXE17xx__MFEndatSensorSendParameter
 - MSXE173x_public_doc.h, 207
 - MSXE17xx_MF_Endat, 80
- MSXE17xx__MFEndatSensorSendParameterResponse, 120
 - sResponse, 121
 - ulParam, 121
- MSXE17xx__MFEndatSensorSendPosAndRecvSelMemArea
 - MSXE173x_public_doc.h, 209
 - MSXE17xx_MF_Endat, 82
- MSXE17xx__MFEndatSensorSendPosAndRecvSelMemAreaResponse, 121
 - sResponse, 121
 - ulPositionHigh, 121
 - ulPositionLow, 121
- MSXE17xx__Response, 121
 - iReturnValue, 121
 - syserrno, 122
- MSXE17xx__unsignedLongResponse, 122
 - sResponse, 122
 - ulValue, 122
- MSXE17xx__unsignedLongTimeStampResponse, 122
 - sResponse, 123
 - ulTimeStampHigh, 123
 - ulTimeStampLow, 123
 - ulValue, 123
- MSXE17xx_DigIO
 - MSXE17xx__DigitalIOGetNumber, 61
 - MSXE17xx__DigitalIOInitPortConfiguration, 61
 - MSXE17xx__-
 - DigitalIOReadAllChannelsValue, 62
 - MSXE17xx__DigitalIOReadChannelValue, 62
 - MSXE17xx__DigitalIORearmShortCircuit, 64
 - MSXE17xx__-
 - DigitalIOReleasePortConfiguration, 64
 - MSXE17xx__DigitalIOTestShortCircuit, 64
 - MSXE17xx__-
 - DigitalIOWriteAllChannelsValue, 63
 - MSXE17xx__DigitalIOWriteChannelValue, 63
- MSXE17xx_MF_Common
 - MSXE17xx__-
 - MFCommonGetSubModuleFunctionality, 67
 - MSXE17xx__-
 - MFCommonReferenceVoltageActivation, 69
- MSXE17xx__MFCommonSetFIFO0Level, 69
- MSXE17xx__MFCommonSetInputsFilter, 68
- MSXE17xx_MF_Endat
 - MSXE17xx__-
 - MFEndatDisableAndReleaseLatchPositionValues, 91
 - MSXE17xx__-
 - MFEndatGetCurrentLatchConfiguration, 90
 - MSXE17xx__MFEndatGetErrorSources, 77
 - MSXE17xx__MFEndatGetPosition, 74
 - MSXE17xx__-
 - MFEndatGetPositionWithAddData, 86
 - MSXE17xx__-
 - MFEndatGetSelectedAdditionalData, 86
 - MSXE17xx__MFEndatGetSensorProperties, 74
 - MSXE17xx__-
 - MFEndatInitAndEnableLatchPositionValues, 87
 - MSXE17xx__MFEndatInitSensor, 73
 - MSXE17xx__MFEndatResetErrorBits, 78
 - MSXE17xx__MFEndatSelectAdditionalData, 84
 - MSXE17xx__MFEndatSelectMemoryArea, 79
 - MSXE17xx__-
 - MFEndatSensorReceiveParameter, 81
 - MSXE17xx__MFEndatSensorReceiveReset, 78
 - MSXE17xx__MFEndatSensorSendParameter, 80
 - MSXE17xx__-
 - MFEndatSensorSendPosAndRecvSelMemArea, 82
- MSXE17xx_Watchdog
 - MSXE17xx__-
 - IOWatchdogGetStatusAndValue, 66
 - MSXE17xx__IOWatchdogInitAndStart, 65
 - MSXE17xx__IOWatchdogStopAndRelease, 66
- MXCommon__ApplyConfigurationBackupFile
 - Common_configuration, 23
 - MSXE173x_public_doc.h, 160
- MXCommon__ByteArrayResponse, 123
 - sArray, 123
 - sResponse, 123
- MXCommon__ChangePassword
 - Common_configuration, 24
 - MSXE173x_public_doc.h, 161
- MXCommon__DataseverRestart

- Common_general, 8
- MSXE173x_public_doc.h, 150
- MXCommon__FileResponse, 123
 - sArray, 124
 - sResponse, 124
 - ulEOF, 124
- MXCommon__GetAutoConfigurationFile
 - Common_autoconf, 19
 - MSXE173x_public_doc.h, 157
- MXCommon__GetAutoConfigurationFileResponse, 124
 - bArray, 124
 - sResponse, 124
 - ulEOF, 124
- MXCommon__GetClientConnections
 - Common_general, 6
 - MSXE173x_public_doc.h, 148
- MXCommon__GetConfigurationBackupFile
 - Common_configuration, 23
 - MSXE173x_public_doc.h, 160
- MXCommon__GetEthernetLinksStates
 - Common_general, 9
 - MSXE173x_public_doc.h, 151
- MXCommon__GetEthernetLinksStatesResponse, 124
 - sPort0, 125
 - sPort1, 125
 - sResponse, 125
- MXCommon__GetHardwareTriggerFilterTime
 - Common_hardware_trigger, 13
 - MSXE173x_public_doc.h, 153
- MXCommon__GetHardwareTriggerFilterTimeResponse, 125
 - sResponse, 125
 - ulFilterTime, 125
 - ulInfo01, 125
 - ulInfo02, 125
- MXCommon__GetHardwareTriggerState
 - Common_hardware_trigger, 13
 - MSXE173x_public_doc.h, 154
- MXCommon__GetHardwareTriggerStateResponse, 125
 - sResponse, 126
 - ulInfo01, 126
 - ulInfo02, 126
 - ulState, 126
- MXCommon__GetHostname
 - Common_general, 5
 - MSXE173x_public_doc.h, 147
- MXCommon__GetModuleTemperatureValueAndStatus
 - Common_temperature, 11
 - MSXE173x_public_doc.h, 152
- MXCommon__GetModuleTemperatureValueAndStatusResponse, 126
 - dTemperatureValue, 127
 - sResponse, 127
 - ulInfo, 127
 - ulTemperatureStatus, 127
- MXCommon__GetModuleType
 - Common_general, 5
 - MSXE173x_public_doc.h, 147
- MXCommon__GetOptionInformation
 - CustomerOption, 28
 - MSXE173x_public_doc.h, 163
- MXCommon__GetStateIDFromName
 - MSXE173x_public_doc.h, 162
 - SystemStatemanagement, 26
- MXCommon__GetStateNameFromID
 - MSXE173x_public_doc.h, 163
 - SystemStatemanagement, 27
- MXCommon__GetSubsystemIDFromName
 - MSXE173x_public_doc.h, 162
 - SystemStatemanagement, 26
- MXCommon__GetSubsystemNameFromID
 - MSXE173x_public_doc.h, 163
 - SystemStatemanagement, 26
- MXCommon__GetSubSystemState
 - MSXE173x_public_doc.h, 161
 - SystemStatemanagement, 25
- MXCommon__GetSynchronizationStatus
 - MSXE173x_public_doc.h, 164
 - Synchronisation, 29
- MXCommon__GetTime
 - Common_time, 17
 - MSXE173x_public_doc.h, 156
- MXCommon__GetTimeResponse, 127
 - sResponse, 127
 - ulHighTime, 127
 - ulLowTime, 127
- MXCommon__GetUpTime
 - Common_time, 18
 - MSXE173x_public_doc.h, 157
- MXCommon__GetUpTimeResponse, 127
 - sResponse, 128
 - ulUpTime, 128
- MXCommon__HardwareClockToSys
 - Common_time, 17
 - MSXE173x_public_doc.h, 156
- MXCommon__InitAndStartSynchroTimer
 - Common_synchrotimer, 21
 - MSXE173x_public_doc.h, 158
- MXCommon__Reboot
 - Common_general, 8
 - MSXE173x_public_doc.h, 150
- MXCommon__ResetAllIOFunctionalities
 - Common_general, 8
 - MSXE173x_public_doc.h, 150
- MXCommon__Response, 128

- iReturnValue, 128
- syserrno, 128
- MXCommon__SetAutoConfigurationFile
 - Common_autoconf, 19
 - MSXE173x_public_doc.h, 158
- MXCommon__SetCustomerKey
 - Common_security, 15
 - MSXE173x_public_doc.h, 154
- MXCommon__SetFilterChannels
 - Analog, 30
 - MSXE173x_public_doc.h, 165
- MXCommon__SetHardwareTriggerFilterTime
 - Common_hardware_trigger, 12
 - MSXE173x_public_doc.h, 153
- MXCommon__SetHostname
 - Common_general, 6
 - MSXE173x_public_doc.h, 148
- MXCommon__SetModuleTemperatureWarningLevels
 - Common_temperature, 11
 - MSXE173x_public_doc.h, 152
- MXCommon__SetTime
 - Common_time, 16
 - MSXE173x_public_doc.h, 155
- MXCommon__SetToMaster
 - MSXE173x_public_doc.h, 164
 - Synchronisation, 28
- MXCommon__StartAutoConfiguration
 - Common_autoconf, 20
 - MSXE173x_public_doc.h, 158
- MXCommon__StopAndReleaseSynchroTimer
 - Common_synchrotimer, 22
 - MSXE173x_public_doc.h, 159
- MXCommon__Strerror
 - Common_general, 6
 - MSXE173x_public_doc.h, 148
- MXCommon__SysToHardwareClock
 - Common_time, 17
 - MSXE173x_public_doc.h, 156
- MXCommon__TestCustomerID
 - Common_security, 15
 - MSXE173x_public_doc.h, 155
- MXCommon__TestCustomerIDResponse, 128
 - bCryptedValueArray, 129
 - bValueArray, 129
 - sResponse, 129
- MXCommon__unsignedLongResponse, 129
 - sResponse, 129
 - ulValue, 129
- sArray
 - MXCommon__ByteArrayResponse, 123
 - MXCommon__FileResponse, 124
- Set/Backup/Restore general system configuration, 22
- sGetEthernetLinksStatesPort, 129
 - ulDuplex, 130
 - ulInfo1, 130
 - ulInfo2, 130
 - ulSpeed, 130
 - ulState, 130
- SOAP function calls in C/C++ language, 92
- Software hints, 92
- sPort0
 - MXCommon__-GetEthernetLinksStatesResponse, 125
- sPort1
 - MXCommon__-GetEthernetLinksStatesResponse, 125
- sResponse
 - MSXE173x__DigitalIOGetNumberResponse, 103
 - MSXE173x__-IOWatchdogGetStatusAndValueResponse, 103
 - MSXE173x__-MFEndatGetCurrentLatchConfigurationResponse, 105
 - MSXE173x__-MFEndatGetErrorSourcesResponse, 106
 - MSXE173x__MFEndatGetPositionResponse, 106
 - MSXE173x__-MFEndatGetPositionWithAddDataResponse, 107
 - MSXE173x__-MFEndatGetSelectedAdditionalDataResponse, 108
 - MSXE173x__-MFEndatGetSensorPropertiesResponse, 110
 - MSXE173x__-MFEndatSensorSendParameterResponse, 111
 - MSXE173x__-MFEndatSensorSendPosAndRecvSelMemAreaResponse, 111
 - MSXE173x__unsignedLongResponse, 112
 - MSXE173x__-unsignedLongTimeStampResponse, 113
 - MSXE17xx__DigitalIOGetNumberResponse, 113
 - MSXE17xx__-IOWatchdogGetStatusAndValueResponse, 114
 - MSXE17xx__-MFEndatGetCurrentLatchConfigurationResponse,

- 115
- MSXE17xx__-
 - MFEndatGetErrorSourcesResponse, 116
- MSXE17xx__MFEndatGetPositionResponse, 116
- MSXE17xx__-
 - MFEndatGetPositionWithAddDataResponse, 117
- MSXE17xx__-
 - MFEndatGetSelectedAdditionalDataResponse, 118
- MSXE17xx__-
 - MFEndatGetSensorPropertiesResponse, 120
- MSXE17xx__-
 - MFEndatSensorSendParameterResponse, 121
- MSXE17xx__-
 - MFEndatSensorSendPosAndRecvSelMemAreaResponse, 121
- MSXE17xx__unsignedLongResponse, 122
- MSXE17xx__-
 - unsignedLongTimeStampResponse, 123
- MXCommon__ByteArrayResponse, 123
- MXCommon__FileResponse, 124
- MXCommon__-
 - GetAutoConfigurationFileResponse, 124
- MXCommon__-
 - GetEthernetLinksStatesResponse, 125
- MXCommon__-
 - GetHardwareTriggerFilterTimeResponse, 125
- MXCommon__-
 - GetHardwareTriggerStateResponse, 126
- MXCommon__-
 - GetModuleTemperatureValueAndStatusResponse, 127
- MXCommon__GetTimeResponse, 127
- MXCommon__GetUpTimeResponse, 128
- MXCommon__TestCustomerIDResponse, 129
- MXCommon__unsignedLongResponse, 129
- Synchronisation
 - MXCommon__GetSynchronizationStatus, 29
 - MXCommon__SetToMaster, 28
- Synchronisation management, 28
- syserrno
 - DefaultResponse, 102
 - MSXE173x__Response, 112
 - MSXE17xx__Response, 122
 - MXCommon__Response, 128
- System state management, 25
- SystemStateManagement
 - MXCommon__GetStateIDFromName, 26
 - MXCommon__GetStateNameFromID, 27
 - MXCommon__GetSubsystemIDFromName, 26
 - MXCommon__GetSubsystemNameFromID, 26
 - MXCommon__GetSubSystemState, 25
- ulAdCount
 - MSXE173x__-
 - MFEndatGetSelectedAdditionalDataResponse, 108
 - MSXE17xx__-
 - MFEndatGetSelectedAdditionalDataResponse, 118
- ulAddData1
 - MSXE173x__-
 - MFEndatGetPositionWithAddDataResponse, 107
 - MSXE17xx__-
 - MFEndatGetPositionWithAddDataResponse, 117
- ulAddData2
 - MSXE173x__-
 - MFEndatGetPositionWithAddDataResponse, 107
 - MSXE17xx__-
 - MFEndatGetPositionWithAddDataResponse, 117
- ulAdditionalData
 - MSXE173x__-
 - MFEndatGetSensorPropertiesResponse, 110
 - MSXE17xx__-
 - MFEndatGetSensorPropertiesResponse, 120
- ulDataFormat
 - MSXE173x__-
 - MFEndatGetCurrentLatchConfigurationResponse, 105
 - MSXE17xx__-
 - MFEndatGetCurrentLatchConfigurationResponse, 115
- ulDuplex
 - sGetEthernetLinksStatesPort, 130
- ulEOF
 - MXCommon__FileResponse, 124
 - MXCommon__-
 - GetAutoConfigurationFileResponse, 124
- ulErrorSrc

- MSXE173x__-
 - MFEndatGetErrorSourcesResponse, [106](#)
- MSXE17xx__-
 - MFEndatGetErrorSourcesResponse, [116](#)
- ulFilterTime
 - MXCommon__-
 - GetHardwareTriggerFilterTimeResponse, [125](#)
- ulHighTime
 - MXCommon__GetTimeResponse, [127](#)
- ulIDNumberLsb
 - MSXE173x__-
 - MFEndatGetSensorPropertiesResponse, [110](#)
 - MSXE17xx__-
 - MFEndatGetSensorPropertiesResponse, [120](#)
- ulIDNumberMsb
 - MSXE173x__-
 - MFEndatGetSensorPropertiesResponse, [110](#)
 - MSXE17xx__-
 - MFEndatGetSensorPropertiesResponse, [120](#)
- ulInfo
 - MSXE173x__-
 - IOWatchdogGetStatusAndValueResponse, [103](#)
 - MSXE17xx__-
 - IOWatchdogGetStatusAndValueResponse, [114](#)
 - MXCommon__-
 - GetModuleTemperatureValueAndStatusResponse, [127](#)
- ulInfo01
 - MXCommon__-
 - GetHardwareTriggerFilterTimeResponse, [125](#)
 - MXCommon__-
 - GetHardwareTriggerStateResponse, [126](#)
- ulInfo02
 - MXCommon__-
 - GetHardwareTriggerFilterTimeResponse, [125](#)
 - MXCommon__-
 - GetHardwareTriggerStateResponse, [126](#)
- ulInfo1
 - sGetEthernetLinksStatesPort, [130](#)
- ulInfo2
 - sGetEthernetLinksStatesPort, [130](#)
- ulLatchSource
 - MSXE173x__-
 - MFEndatGetCurrentLatchConfigurationResponse, [105](#)
 - MSXE17xx__-
 - MFEndatGetCurrentLatchConfigurationResponse, [115](#)
- ulLowTime
 - MXCommon__GetTimeResponse, [127](#)
- ulMode
 - MSXE173x__-
 - MFEndatGetSensorPropertiesResponse, [110](#)
 - MSXE17xx__-
 - MFEndatGetSensorPropertiesResponse, [120](#)
- ulModel
 - MSXE173x__-
 - MFEndatGetCurrentLatchConfigurationResponse, [105](#)
 - MSXE173x__-
 - MFEndatGetSensorPropertiesResponse, [110](#)
 - MSXE17xx__-
 - MFEndatGetCurrentLatchConfigurationResponse, [115](#)
 - MSXE17xx__-
 - MFEndatGetSensorPropertiesResponse, [120](#)
- ulMrsCodeAD1
 - MSXE173x__-
 - MFEndatGetSelectedAdditionalDataResponse, [108](#)
 - MSXE17xx__-
 - MFEndatGetSelectedAdditionalDataResponse, [118](#)
- ulMrsCodeAD2
 - MSXE173x__-
 - MFEndatGetSelectedAdditionalDataResponse, [108](#)
 - MSXE17xx__-
 - MFEndatGetSelectedAdditionalDataResponse, [118](#)
- ulNumberOfDigitalIO
 - MSXE173x__DigitalIOGetNumberResponse, [103](#)
 - MSXE17xx__DigitalIOGetNumberResponse, [113](#)
- ulNumberOfRevolution
 - MSXE173x__-
 - MFEndatGetCurrentLatchConfigurationResponse, [105](#)
 - MSXE173x__-
 - MFEndatGetSensorPropertiesResponse, [120](#)

- [110](#)
 MSXE17xx__-
 MFEndatGetCurrentLatchConfigurationResponse, [115](#)
 MSXE17xx__-
 MFEndatGetSensorPropertiesResponse, [120](#)
- ulParam
 MSXE173x__-
 MFEndatSensorSendParameterResponse, [111](#)
 MSXE17xx__-
 MFEndatSensorSendParameterResponse, [121](#)
- ulPositionHigh
 MSXE173x__MFEndatGetPositionResponse, [106](#)
 MSXE173x__-
 MFEndatGetPositionWithAddDataResponse, [107](#)
 MSXE173x__-
 MFEndatSensorSendPosAndRecvSelMemAreaResponse, [111](#)
 MSXE17xx__MFEndatGetPositionResponse, [116](#)
 MSXE17xx__-
 MFEndatGetPositionWithAddDataResponse, [117](#)
 MSXE17xx__-
 MFEndatSensorSendPosAndRecvSelMemAreaResponse, [121](#)
- ulPositionLow
 MSXE173x__MFEndatGetPositionResponse, [106](#)
 MSXE173x__-
 MFEndatGetPositionWithAddDataResponse, [107](#)
 MSXE173x__-
 MFEndatSensorSendPosAndRecvSelMemAreaResponse, [111](#)
 MSXE17xx__MFEndatGetPositionResponse, [116](#)
 MSXE17xx__-
 MFEndatGetPositionWithAddDataResponse, [117](#)
 MSXE17xx__-
 MFEndatSensorSendPosAndRecvSelMemAreaResponse, [121](#)
- ulPositionSize
 MSXE173x__-
 MFEndatGetCurrentLatchConfigurationResponse, [105](#)
 MSXE173x__-
 MFEndatGetSensorPropertiesResponse, [120](#)
- [110](#)
 MSXE17xx__-
 MFEndatGetCurrentLatchConfigurationResponse, [115](#)
 MSXE17xx__-
 MFEndatGetSensorPropertiesResponse, [120](#)
- ulRunning
 MSXE173x__-
 MFEndatGetCurrentLatchConfigurationResponse, [105](#)
 MSXE17xx__-
 MFEndatGetCurrentLatchConfigurationResponse, [115](#)
- ulScalingFactor
 MSXE173x__-
 MFEndatGetCurrentLatchConfigurationResponse, [105](#)
 MSXE173x__-
 MFEndatGetSensorPropertiesResponse, [110](#)
 MSXE17xx__-
 MFEndatGetCurrentLatchConfigurationResponse, [115](#)
 MSXE17xx__-
 MFEndatGetSensorPropertiesResponse, [120](#)
- ulSerialNumberLsb
 MSXE173x__-
 MFEndatGetSensorPropertiesResponse, [110](#)
- ulSerialNumberMsb
 MSXE173x__-
 MFEndatGetSensorPropertiesResponse, [110](#)
 MSXE17xx__-
 MFEndatGetSensorPropertiesResponse, [120](#)
- ulSignalPeriod
 MSXE173x__-
 MFEndatGetCurrentLatchConfigurationResponse, [105](#)
 MSXE173x__-
 MFEndatGetSensorPropertiesResponse, [110](#)
 MSXE17xx__-
 MFEndatGetCurrentLatchConfigurationResponse, [115](#)
 MSXE17xx__-
 MFEndatGetSensorPropertiesResponse, [120](#)

- ulSpeed
 - sGetEthernetLinksStatesPort, [130](#)
- ulState
 - MXCommon__-
 - GetHardwareTriggerStateResponse, [126](#)
 - sGetEthernetLinksStatesPort, [130](#)
- ulStatus
 - MSXE173x__-
 - IOWatchdogGetStatusAndValueResponse, [103](#)
 - MSXE17xx__-
 - IOWatchdogGetStatusAndValueResponse, [114](#)
- ulStepPerRevolution
 - MSXE173x__-
 - MFEndatGetCurrentLatchConfigurationResponse, [105](#)
 - MSXE173x__-
 - MFEndatGetSensorPropertiesResponse, [110](#)
 - MSXE17xx__-
 - MFEndatGetCurrentLatchConfigurationResponse, [115](#)
 - MSXE17xx__-
 - MFEndatGetSensorPropertiesResponse, [120](#)
- ulTemperatureStatus
 - MXCommon__-
 - GetModuleTemperatureValueAndStatusResponse, [127](#)
- ulTimeStampHigh
 - MSXE173x__-
 - unsignedLongTimeStampResponse, [113](#)
 - MSXE17xx__-
 - unsignedLongTimeStampResponse, [123](#)
- ulTimeStampLow
 - MSXE173x__-
 - unsignedLongTimeStampResponse, [113](#)
 - MSXE17xx__-
 - unsignedLongTimeStampResponse, [123](#)
- ulTriggerEdgeCount
 - MSXE173x__-
 - MFEndatGetCurrentLatchConfigurationResponse, [105](#)
 - MSXE17xx__-
 - MFEndatGetCurrentLatchConfigurationResponse, [115](#)
- ulUpTime
 - MXCommon__GetUpTimeResponse, [128](#)
- ulValue
 - MSXE173x__-
 - IOWatchdogGetStatusAndValueResponse, [103](#)
 - MSXE173x__unsignedLongResponse, [112](#)
 - MSXE173x__-
 - unsignedLongTimeStampResponse, [113](#)
 - MSXE17xx__-
 - IOWatchdogGetStatusAndValueResponse, [114](#)
 - MSXE17xx__unsignedLongResponse, [122](#)
 - MSXE17xx__-
 - unsignedLongTimeStampResponse, [123](#)
 - MXCommon__unsignedLongResponse, [129](#)
 - unsignedLongArray, [130](#)
 - __offset, [130](#)
 - __ptr, [130](#)
 - __size, [130](#)
 - UnsignedShortArray, [130](#)
 - __offset, [131](#)
 - __ptr, [131](#)
 - __size, [131](#)
 - xsd__base64Binary, [131](#)
 - __ptr, [131](#)
 - __size, [131](#)
 - xsd__char
 - MSXE173x_public_doc.h, [147](#)
 - xsd__double
 - MSXE173x_public_doc.h, [147](#)
 - xsd__float
 - MSXE173x_public_doc.h, [147](#)
 - xsd__int
 - MSXE173x_public_doc.h, [147](#)
 - xsd__long
 - MSXE173x_public_doc.h, [147](#)
 - xsd__string
 - MSXE173x_public_doc.h, [147](#)
 - xsd__unsignedByte
 - MSXE173x_public_doc.h, [147](#)
 - xsd__unsignedInt
 - MSXE173x_public_doc.h, [147](#)
 - xsd__unsignedLong
 - MSXE173x_public_doc.h, [147](#)
 - xsd__unsignedShort
 - MSXE173x_public_doc.h, [147](#)