

## MSXE371x soap api functions

Generated by Doxygen 1.7.1

Tue Oct 20 2015 12:12:48



# Contents

<b>1</b>	<b>MSXE371x SOAP functions documentation</b>	<b>1</b>
1.1	Introduction . . . . .	1
<b>2</b>	<b>Module Documentation</b>	<b>3</b>
2.1	MSXE371x functions . . . . .	3
2.2	Common functions . . . . .	3
2.3	Common general functions . . . . .	4
2.3.1	Function Documentation . . . . .	5
2.3.1.1	MXCommon__GetModuleType . . . . .	5
2.3.1.2	MXCommon__GetHostname . . . . .	5
2.3.1.3	MXCommon__SetHostname . . . . .	6
2.3.1.4	MXCommon__GetClientConnections . . . . .	6
2.3.1.5	MXCommon__Sterror . . . . .	6
2.3.1.6	MXCommon__Reboot . . . . .	8
2.3.1.7	MXCommon__ResetAllIOFunctionalities . . . . .	8
2.3.1.8	MXCommon__DataserverRestart . . . . .	8
2.3.1.9	MXCommon__GetEthernetLinksStates . . . . .	9
2.4	Common temperature functions . . . . .	10
2.4.1	Detailed Description . . . . .	10
2.4.2	Function Documentation . . . . .	10
2.4.2.1	MXCommon__GetModuleTemperatureValueAndStatus . . . . .	10
2.4.2.2	MXCommon__SetModuleTemperatureWarningLevels . . . . .	11
2.5	Common hardware trigger functions . . . . .	11
2.5.1	Function Documentation . . . . .	12
2.5.1.1	MXCommon__SetHardwareTriggerFilterTime . . . . .	12
2.5.1.2	MXCommon__GetHardwareTriggerFilterTime . . . . .	13
2.5.1.3	MXCommon__GetHardwareTriggerState . . . . .	13
2.6	Common security functions . . . . .	13

2.6.1	Detailed Description	14
2.6.2	Function Documentation	15
2.6.2.1	MXCommon__SetCustomerKey	15
2.6.2.2	MXCommon__TestCustomerID	15
2.7	Common time functions	15
2.7.1	Detailed Description	16
2.7.2	Function Documentation	16
2.7.2.1	MXCommon__SetTime	16
2.7.2.2	MXCommon__SysToHardwareClock	17
2.7.2.3	MXCommon__HardwareClockToSys	17
2.7.2.4	MXCommon__GetTime	17
2.7.2.5	MXCommon__GetUpTime	18
2.8	Common I/O auto configuration functions	18
2.8.1	Detailed Description	19
2.8.2	Function Documentation	19
2.8.2.1	MXCommon__GetAutoConfigurationFile	19
2.8.2.2	MXCommon__SetAutoConfigurationFile	19
2.8.2.3	MXCommon__StartAutoConfiguration	20
2.9	Common synchronisation timer functions	20
2.9.1	Function Documentation	21
2.9.1.1	MXCommon__InitAndStartSynchroTimer	21
2.9.1.2	MXCommon__StopAndReleaseSynchroTimer	22
2.10	Set/Backup/Restore general system configuration	22
2.10.1	Detailed Description	22
2.10.2	Function Documentation	23
2.10.2.1	MXCommon__GetConfigurationBackupFile	23
2.10.2.2	MXCommon__ApplyConfigurationBackupFile	23
2.10.2.3	MXCommon__ChangePassword	24
2.11	System state management	24
2.11.1	Detailed Description	25
2.11.2	Function Documentation	25
2.11.2.1	MXCommon__GetSubSystemState	25
2.11.2.2	MXCommon__GetSubsystemIDFromName	26
2.11.2.3	MXCommon__GetStateIDFromName	26
2.11.2.4	MXCommon__GetSubsystemNameFromID	27
2.11.2.5	MXCommon__GetStateNameFromID	27

2.12	Customer option management	27
2.12.1	Function Documentation	28
2.12.1.1	MXCommon__GetOptionInformation	28
2.13	Synchronisation management	28
2.13.1	Function Documentation	28
2.13.1.1	MXCommon__SetToMaster	28
2.13.1.2	MXCommon__GetSynchronizationStatus	29
2.14	input filter Filter management	29
2.14.1	Function Documentation	30
2.14.1.1	MXCommon__SetFilterChannels	30
2.15	MSXE371x Get informations functions	30
2.15.1	Function Documentation	30
2.15.1.1	MSXE371x__TransducerGetNbrOfType	30
2.15.1.2	MSXE371x__TransducerGetTypeInformation	31
2.16	MSXE371x Auto refresh functions	31
2.16.1	Detailed Description	32
2.16.2	Function Documentation	33
2.16.2.1	MSXE371x__TransducerInitAndStartAutoRefresh	33
2.16.2.2	MSXE371x__TransducerGetAutoRefreshValues	35
2.16.2.3	MSXE371x__TransducerGetAllAutoRefreshValues	35
2.16.2.4	MSXE371x__TransducerStopAndReleaseAutoRefresh	35
2.17	MSXE371x Sequence functions	36
2.17.1	Detailed Description	36
2.17.2	Function Documentation	38
2.17.2.1	MSXE371x__TransducerInitAndStartSequence	38
2.17.2.2	MSXE371x__TransducerStopAndReleaseSequence	40
2.17.2.3	MSXE371x__TransducerStopAndReleaseSequence2	40
2.18	MSXE371x diagnostic functions	41
2.18.1	Function Documentation	41
2.18.1.1	MSXE371x__TransducerInit	41
2.18.1.2	MSXE371x__TransducerInitPrimaryConnectionTest	42
2.18.1.3	MSXE371x__TransducerTestPrimaryConnection	42
2.18.1.4	MSXE371x__TransducerTestPrimaryShortCircuit	43
2.18.1.5	MSXE371x__TransducerRearmPrimary	43
2.18.1.6	MSXE371x__TransducerTestSecondaryConnection	44
2.18.1.7	MSXE371x__TransducerTestSecondaryShortCircuit	44

2.19	MSX-E371x incremental counter functions	44
2.19.1	Function Documentation	45
2.19.1.1	MSXE371x__IncCounterInit	45
2.19.1.2	MSXE371x__IncCounterRelease	46
2.19.1.3	MSXE371x__IncCounterRead32BitValue	47
2.19.1.4	MSXE371x__IncCounterWrite32BitValue	47
2.19.1.5	MSXE371x__IncCounterClear	47
2.19.1.6	MSXE371x__IncCounterInitAndEnableCompareLogic	48
2.19.1.7	MSXE371x__IncCounterDisableAndReleaseCompareLogic	48
2.19.1.8	MSXE371x__IncCounterInitAndEnableIndex	49
2.19.1.9	MSXE371x__IncCounterDisableAndReleaseIndex	50
2.20	MSX-E371x external temperature functions	50
2.20.1	Function Documentation	51
2.20.1.1	MSXE371x__ExternalTemperatureInit	51
2.20.1.2	MSXE371x__ExternalTemperatureRelease	52
2.20.1.3	MSXE371x__ExternalTemperatureRead	52
2.20.1.4	MSXE371x__ExternalTemperatureGainCalibration	53
2.21	MSXE371x calibration functions	53
2.21.1	Function Documentation	54
2.21.1.1	MSXE371x__CalibrationStart	54
2.21.1.2	MSXE371x__CalibrationGetCurrentStatus	54
2.21.1.3	MSXE371x__CalibrationNextStep	55
2.21.1.4	MSXE371x__CalibrationBreak	55
2.22	MSXE371x transducer database management functions	56
2.22.1	Function Documentation	57
2.22.1.1	MSXE371x__DataBaseGetNumberOfTransducers	57
2.22.1.2	MSXE371x__DataBaseGetTransducerType	57
2.22.1.3	MSXE371x__DataBaseGetTransducerInformation	57
2.22.1.4	MSXE371x__DataBaseAddTransducer	58
2.22.1.5	MSXE371x__DataBaseDelTransducer	59
2.22.1.6	MSXE371x__DataBaseSaveTransducers	59
<b>3</b>	<b>Data Structure Documentation</b>	<b>61</b>
3.1	ByteArray Struct Reference	61
3.1.1	Field Documentation	61
3.1.1.1	__ptr	61
3.1.1.2	__size	61

3.1.1.3	__offset	61
3.2	DefaultResponse Struct Reference	61
3.2.1	Field Documentation	62
3.2.1.1	iReturnValue	62
3.2.1.2	syserrno	62
3.3	MSXE371x__ByteArrayResponse Struct Reference	62
3.3.1	Field Documentation	62
3.3.1.1	sResponse	62
3.3.1.2	sArray	62
3.4	MSXE371x__CalibrationGetCurrentStatusResponse Struct Reference	62
3.4.1	Field Documentation	63
3.4.1.1	sResponse	63
3.4.1.2	ulStatus	63
3.4.1.3	ulChannel	63
3.4.1.4	ulDigitalValue	63
3.5	MSXE371x__DataBaseGetTransducerInformationResponse Struct Reference	63
3.5.1	Field Documentation	65
3.5.1.1	sResponse	65
3.5.1.2	cName	65
3.5.1.3	ulCalibrate	65
3.5.1.4	ulCalibratedChannels	65
3.5.1.5	ulType	65
3.5.1.6	ulFrequency	65
3.5.1.7	ulImpedance	65
3.5.1.8	dVeff	65
3.5.1.9	dSensitivity	65
3.5.1.10	dRange	65
3.6	MSXE371x__ExternalTemperatureReadResponse Struct Reference	65
3.6.1	Field Documentation	66
3.6.1.1	sResponse	66
3.6.1.2	ulValue	66
3.6.1.3	ulTimeStampHigh	66
3.6.1.4	ulTimeStampLow	66
3.7	MSXE371x__IncCounterRead32BitValueResponse Struct Reference	66
3.7.1	Field Documentation	66
3.7.1.1	sResponse	66

3.7.1.2	ulValue	66
3.7.1.3	ulTimeStampLow	66
3.7.1.4	ulTimeStampHigh	66
3.8	MSXE371x__Response Struct Reference	66
3.8.1	Field Documentation	67
3.8.1.1	iReturnValue	67
3.8.1.2	syserrno	67
3.9	MSXE371x__TransducerGetAllValuesResponse Struct Reference	67
3.9.1	Field Documentation	68
3.9.1.1	sResponse	68
3.9.1.2	ulAutoRefreshCounter	68
3.9.1.3	ulTransducersValue	68
3.9.1.4	ulTimeStampLow	68
3.9.1.5	ulTimeStampHigh	68
3.9.1.6	ulIncCounter	68
3.9.1.7	ulExternalTemperature	68
3.10	MSXE371x__TransducerGetTypeInformationResponse Struct Reference	68
3.10.1	Field Documentation	69
3.10.1.1	sResponse	69
3.10.1.2	ulTransducerSelectionIndex	69
3.10.1.3	pcName	69
3.10.1.4	ulCalibrationStatus	69
3.10.1.5	ulCalibratedChannels	69
3.10.1.6	ulType	69
3.10.1.7	ulFrequency	69
3.10.1.8	ulImpedance	69
3.10.1.9	dVeff	69
3.10.1.10	dSensibility	69
3.10.1.11	dRange	69
3.11	MSXE371x__unsignedLong8FixedArray Struct Reference	69
3.11.1	Field Documentation	70
3.11.1.1	ulValue	70
3.12	MSXE371x__unsignedlong9ArrayResponse Struct Reference	70
3.12.1	Field Documentation	70
3.12.1.1	sResponse	70
3.12.1.2	ulValue	70



3.13 MSXE371x__unsignedlongDefaultResponse Struct Reference . . . . .	70
3.13.1 Field Documentation . . . . .	70
3.13.1.1 sResponse . . . . .	70
3.13.1.2 ulValue . . . . .	70
3.14 MSXE371x__unsignedlongResponse Struct Reference . . . . .	70
3.14.1 Field Documentation . . . . .	71
3.14.1.1 sResponse . . . . .	71
3.14.1.2 ulValue . . . . .	71
3.15 MXCommon__ByteArrayResponse Struct Reference . . . . .	71
3.15.1 Field Documentation . . . . .	71
3.15.1.1 sResponse . . . . .	71
3.15.1.2 sArray . . . . .	71
3.16 MXCommon__FileResponse Struct Reference . . . . .	71
3.16.1 Field Documentation . . . . .	72
3.16.1.1 sResponse . . . . .	72
3.16.1.2 sArray . . . . .	72
3.16.1.3 ulEOF . . . . .	72
3.17 MXCommon__GetAutoConfigurationFileResponse Struct Reference . . . . .	72
3.17.1 Field Documentation . . . . .	72
3.17.1.1 sResponse . . . . .	72
3.17.1.2 bArray . . . . .	72
3.17.1.3 ulEOF . . . . .	72
3.18 MXCommon__GetEthernetLinksStatesResponse Struct Reference . . . . .	72
3.18.1 Field Documentation . . . . .	73
3.18.1.1 sResponse . . . . .	73
3.18.1.2 sPort0 . . . . .	73
3.18.1.3 sPort1 . . . . .	73
3.19 MXCommon__GetHardwareTriggerFilterTimeResponse Struct Reference . . . . .	73
3.19.1 Field Documentation . . . . .	73
3.19.1.1 sResponse . . . . .	73
3.19.1.2 ulFilterTime . . . . .	73
3.19.1.3 ulInfo01 . . . . .	73
3.19.1.4 ulInfo02 . . . . .	73
3.20 MXCommon__GetHardwareTriggerStateResponse Struct Reference . . . . .	73
3.20.1 Field Documentation . . . . .	74
3.20.1.1 sResponse . . . . .	74

3.20.1.2	ulState	74
3.20.1.3	ulInfo01	74
3.20.1.4	ulInfo02	74
3.21	MXCommon__GetModuleTemperatureValueAndStatusResponse Struct Reference	74
3.21.1	Field Documentation	75
3.21.1.1	sResponse	75
3.21.1.2	dTemperatureValue	75
3.21.1.3	ulTemperatureStatus	75
3.21.1.4	ulInfo	75
3.22	MXCommon__GetTimeResponse Struct Reference	75
3.22.1	Field Documentation	75
3.22.1.1	sResponse	75
3.22.1.2	ulLowTime	75
3.22.1.3	ulHighTime	75
3.23	MXCommon__GetUpTimeResponse Struct Reference	75
3.23.1	Field Documentation	76
3.23.1.1	sResponse	76
3.23.1.2	ulUpTime	76
3.24	MXCommon__Response Struct Reference	76
3.24.1	Field Documentation	76
3.24.1.1	iReturnValue	76
3.24.1.2	syserrno	76
3.25	MXCommon__TestCustomerIDResponse Struct Reference	76
3.25.1	Field Documentation	77
3.25.1.1	sResponse	77
3.25.1.2	bValueArray	77
3.25.1.3	bCryptedValueArray	77
3.26	MXCommon__unsignedLongResponse Struct Reference	77
3.26.1	Field Documentation	77
3.26.1.1	sResponse	77
3.26.1.2	ulValue	77
3.27	sGetEthernetLinksStatesPort Struct Reference	77
3.27.1	Field Documentation	78
3.27.1.1	ulState	78
3.27.1.2	ulSpeed	78
3.27.1.3	ulDuplex	78

3.27.1.4	ulInfo1	78
3.27.1.5	ulInfo2	78
3.28	UnsignedLongArray Struct Reference	78
3.28.1	Field Documentation	78
3.28.1.1	__ptr	78
3.28.1.2	__size	78
3.28.1.3	__offset	78
3.29	UnsignedShortArray Struct Reference	78
3.29.1	Field Documentation	79
3.29.1.1	__ptr	79
3.29.1.2	__size	79
3.29.1.3	__offset	79
3.30	xsd__base64Binary Struct Reference	79
3.30.1	Field Documentation	79
3.30.1.1	__ptr	79
3.30.1.2	__size	79
<b>4</b>	<b>File Documentation</b>	<b>81</b>
4.1	MSXE371x_public_doc.h File Reference	81
4.1.1	Typedef Documentation	90
4.1.1.1	xsd__string	90
4.1.1.2	xsd__char	90
4.1.1.3	xsd__float	90
4.1.1.4	xsd__double	90
4.1.1.5	xsd__int	90
4.1.1.6	xsd__long	90
4.1.1.7	xsd__unsignedByte	90
4.1.1.8	xsd__unsignedInt	90
4.1.1.9	xsd__unsignedShort	90
4.1.1.10	xsd__unsignedLong	90
4.1.2	Function Documentation	90
4.1.2.1	MXCommon__GetModuleType	90
4.1.2.2	MXCommon__GetHostname	90
4.1.2.3	MXCommon__SetHostname	91
4.1.2.4	MXCommon__GetClientConnections	91
4.1.2.5	MXCommon__Strerror	91
4.1.2.6	MXCommon__Reboot	93

4.1.2.7	MXCommon__ResetAllIOFunctionalities . . . . .	93
4.1.2.8	MXCommon__DataseverRestart . . . . .	93
4.1.2.9	MXCommon__GetEthernetLinksStates . . . . .	94
4.1.2.10	MXCommon__GetModuleTemperatureValueAndStatus . . . . .	95
4.1.2.11	MXCommon__SetModuleTemperatureWarningLevels . . . . .	95
4.1.2.12	MXCommon__SetHardwareTriggerFilterTime . . . . .	96
4.1.2.13	MXCommon__GetHardwareTriggerFilterTime . . . . .	96
4.1.2.14	MXCommon__GetHardwareTriggerState . . . . .	97
4.1.2.15	MXCommon__SetCustomerKey . . . . .	97
4.1.2.16	MXCommon__TestCustomerID . . . . .	98
4.1.2.17	MXCommon__SetTime . . . . .	98
4.1.2.18	MXCommon__SysToHardwareClock . . . . .	99
4.1.2.19	MXCommon__HardwareClockToSys . . . . .	99
4.1.2.20	MXCommon__GetTime . . . . .	99
4.1.2.21	MXCommon__GetUpTime . . . . .	100
4.1.2.22	MXCommon__GetAutoConfigurationFile . . . . .	100
4.1.2.23	MXCommon__SetAutoConfigurationFile . . . . .	101
4.1.2.24	MXCommon__StartAutoConfiguration . . . . .	101
4.1.2.25	MXCommon__InitAndStartSynchroTimer . . . . .	101
4.1.2.26	MXCommon__StopAndReleaseSynchroTimer . . . . .	102
4.1.2.27	MXCommon__GetConfigurationBackupFile . . . . .	103
4.1.2.28	MXCommon__ApplyConfigurationBackupFile . . . . .	103
4.1.2.29	MXCommon__ChangePassword . . . . .	104
4.1.2.30	MXCommon__GetSubSystemState . . . . .	104
4.1.2.31	MXCommon__GetSubsystemIDFromName . . . . .	105
4.1.2.32	MXCommon__GetStateIDFromName . . . . .	105
4.1.2.33	MXCommon__GetSubsystemNameFromID . . . . .	106
4.1.2.34	MXCommon__GetStateNameFromID . . . . .	106
4.1.2.35	MXCommon__GetOptionInformation . . . . .	106
4.1.2.36	MXCommon__SetToMaster . . . . .	107
4.1.2.37	MXCommon__GetSynchronizationStatus . . . . .	107
4.1.2.38	MXCommon__SetFilterChannels . . . . .	108
4.1.2.39	MSXE371x__TransducerGetNbrOfType . . . . .	108
4.1.2.40	MSXE371x__TransducerGetTypeInformation . . . . .	108
4.1.2.41	MSXE371x__TransducerInitAndStartAutoRefresh . . . . .	109
4.1.2.42	MSXE371x__TransducerGetAutoRefreshValues . . . . .	111

4.1.2.43	MSXE371x__TransducerGetAllAutoRefreshValues	111
4.1.2.44	MSXE371x__TransducerStopAndReleaseAutoRefresh	112
4.1.2.45	MSXE371x__TransducerInitAndStartSequence	112
4.1.2.46	MSXE371x__TransducerStopAndReleaseSequence	115
4.1.2.47	MSXE371x__TransducerStopAndReleaseSequence2	115
4.1.2.48	MSXE371x__TransducerInit	115
4.1.2.49	MSXE371x__TransducerInitPrimaryConnectionTest	116
4.1.2.50	MSXE371x__TransducerTestPrimaryConnection	116
4.1.2.51	MSXE371x__TransducerTestPrimaryShortCircuit	117
4.1.2.52	MSXE371x__TransducerRearmPrimary	117
4.1.2.53	MSXE371x__TransducerTestSecondaryConnection	117
4.1.2.54	MSXE371x__TransducerTestSecondaryShortCircuit	118
4.1.2.55	MSXE371x__IncCounterInit	118
4.1.2.56	MSXE371x__IncCounterRelease	119
4.1.2.57	MSXE371x__IncCounterRead32BitValue	119
4.1.2.58	MSXE371x__IncCounterWrite32BitValue	120
4.1.2.59	MSXE371x__IncCounterClear	120
4.1.2.60	MSXE371x__IncCounterInitAndEnableCompareLogic	121
4.1.2.61	MSXE371x__IncCounterDisableAndReleaseCompareLogic	121
4.1.2.62	MSXE371x__IncCounterInitAndEnableIndex	122
4.1.2.63	MSXE371x__IncCounterDisableAndReleaseIndex	122
4.1.2.64	MSXE371x__ExternalTemperatureInit	123
4.1.2.65	MSXE371x__ExternalTemperatureRelease	124
4.1.2.66	MSXE371x__ExternalTemperatureRead	124
4.1.2.67	MSXE371x__ExternalTemperatureGainCalibration	125
4.1.2.68	MSXE371x__CalibrationStart	125
4.1.2.69	MSXE371x__CalibrationGetCurrentStatus	126
4.1.2.70	MSXE371x__CalibrationNextStep	126
4.1.2.71	MSXE371x__CalibrationBreak	127
4.1.2.72	MSXE371x__DataBaseGetNumberOfTransducers	127
4.1.2.73	MSXE371x__DataBaseGetTransducerType	128
4.1.2.74	MSXE371x__DataBaseGetTransducerInformation	128
4.1.2.75	MSXE371x__DataBaseAddTransducer	129
4.1.2.76	MSXE371x__DataBaseDelTransducer	129
4.1.2.77	MSXE371x__DataBaseSaveTransducers	130



# Chapter 1

## MSXE371x SOAP functions documentation

**MainRevision:**

### 1.1 Introduction

The module is accessed via a TCP/IP socket:

The Ethernet I/O module has the following two servers: Command server (SOAP) > to send commands (acquisition,initialisation, etc.)

Data server (TCP socket) > to obtain the values of the acquisition

Event server (TCP socket) > to obtain event from the module

MSXE371x server access information:

- SOAP server: Port number 5555
- Data server: Port number 8989
- Event server: Port number 6363

See the "Modules" chapter to view the functions





# Chapter 2

## Module Documentation

### 2.1 MSXE371x functions

#### Modules

- [MSXE371x Get informations functions](#)
- [MSXE371x Auto refresh functions](#)

*This function initialise and start the auto refresh acquisition.*

- [MSXE371x Sequence functions](#)

*A sequence is a list of channels (max 16) that are acquired.*

- [MSXE371x diagnostic functions](#)
- [MSX-E371x incremental counter functions](#)
- [MSX-E371x external temperature functions](#)
- [MSXE371x calibration functions](#)
- [MSXE371x transducer database management functions](#)

*These functions allows to manage the database of transducer types on the module.*

### 2.2 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.

## 2.3 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.*

### 2.3.1 Function Documentation

#### 2.3.1.1 `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

#### 2.3.1.2 `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

#### 2.3.1.3 `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

#### 2.3.1.4 `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

#### 2.3.1.5 `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] **errno** : 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) Linux Programmer's Manual  
 STRError(3)

**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

**2.3.1.6 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

**2.3.1.7 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

**2.3.1.8 int MXCommon\_\_DataseverRestart ( 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.

#### 2.3.1.9 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** Reserverd
- **ulInfo2** Reserverd

**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

## 2.4 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.*

### 2.4.1 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.

### 2.4.2 Function Documentation

- #### 2.4.2.1 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\_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

**2.4.2.2 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

## 2.5 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.*

### 2.5.1 Function Documentation

#### 2.5.1.1 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

### 2.5.1.2 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

### 2.5.1.3 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\\_\\_Strerror\(\)](#).

#### Return values

*SOAP\_OK* SOAP call success

*otherwise* SOAP protocol error

## 2.6 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 ).*

### 2.6.1 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]=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.

## 2.6.2 Function Documentation

**2.6.2.1** `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\\_\\_Sterror\(\)](#).

### Return values

*SOAP\_OK* SOAP call success  
*otherwise* SOAP protocol error

**2.6.2.2** `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\\_\\_Sterror\(\)](#).
  - 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

## 2.7 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).*

### 2.7.1 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.

### 2.7.2 Function Documentation

#### 2.7.2.1 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\\_\\_Strerror\(\)](#).

##### Return values

*SOAP\_OK* SOAP call success

*otherwise* SOAP protocol error

### 2.7.2.2 `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\\_\\_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.

### 2.7.2.3 `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\\_\\_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.

### 2.7.2.4 `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

#### 2.7.2.5 `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

## 2.8 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*

### 2.8.1 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.

### 2.8.2 Function Documentation

#### 2.8.2.1 `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

#### 2.8.2.2 `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\\_\\_Strerror\(\)](#).

#### Return values

**SOAP\_OK** SOAP call success

**otherwise** SOAP protocol error

### 2.8.2.3 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\\_\\_Strerror\(\)](#).

- sArray : message returned by the auto configuration start

#### Return values

**SOAP\_OK** SOAP call success

**otherwise** SOAP protocol error

## 2.9 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)*

## 2.9.1 Function Documentation

**2.9.1.1** `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\\_\\_Strerror\(\)](#). May be ENOSYS : Function not implemented.

### Return values

**SOAP\_OK** SOAP call success  
**otherwise** SOAP protocol error

### 2.9.1.2 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

## 2.10 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.*

### 2.10.1 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"

## 2.10.2 Function Documentation

### 2.10.2.1 `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\\_\\_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;
}
*
```

### 2.10.2.2 `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\\_\\_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 uEOF must be set to 1. The new configuration is then applied.

#### 2.10.2.3 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\\_\\_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.

## 2.11 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".

### 2.11.1 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.

### 2.11.2 Function Documentation

#### 2.11.2.1 `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__Strerror()`.

- 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

**2.11.2.2** `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

**2.11.2.3** `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



### 2.11.2.4 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

### 2.11.2.5 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

## 2.12 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.*

## 2.12.1 Function Documentation

**2.12.1.1** `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

## 2.13 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.*

## 2.13.1 Function Documentation

**2.13.1.1** `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

**2.13.1.2** `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

## 2.14 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.*

## 2.14.1 Function Documentation

### 2.14.1.1 `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\\_\\_Strerror\(\)](#).

#### Return values

- SOAP\_OK* SOAP call success
- otherwise* SOAP protocol error

## 2.15 MSXE371x Get informations functions

### Data Structures

- struct [MSXE371x\\_\\_TransducerGetTypeInformationResponse](#)

### Functions

- int [MSXE371x\\_\\_TransducerGetNbrOfType](#) (void \*\_, struct [MSXE371x\\_\\_unsignedlongResponse](#) \*Response)  
Returns the number of transducer types currently defined in the database.
- int [MSXE371x\\_\\_TransducerGetTypeInformation](#) (xsd\_\_unsignedLong ulIndex, struct [MSXE371x\\_\\_TransducerGetTypeInformationResponse](#) \*Response)  
Returns the information stored in the database about the type.

## 2.15.1 Function Documentation

### 2.15.1.1 `int MSXE371x__TransducerGetNbrOfType ( void * _, struct MSXE371x__unsignedlongResponse * Response )`

#### Parameters

- [in] \_ : no input parameter
- [out] ***Response*** :
- sResponse.iReturnValue* : Error value
- 0: success
  - -1: error

- -100: kernel function error  
*sResponse.syserrno* : system-error code (the value of the libc "errno" code) Its value is significant only when the iReturnValue returned an error (-1 or <= -100)  
Give this value to the MXCommon\_Sterror to get the string describing the error number.  
*ulValue* : number of transducers type.

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 2.15.1.2 int MSXE371x\_\_TransducerGetTypeInfoInformation ( xsd\_\_unsignedLong *ulIndex*, struct MSXE371x\_\_TransducerGetTypeInfoInformationResponse \* *Response* )

#### Parameters

[in] *ulIndex* : index of the transducer

[out] *Response* :

*sResponse.iReturnValue* : Error value

- 0: success
- -1: error
- -2: index is invalid
- -100: failure of kernel function "GetNumberOfTransducerTypes"
- -101: failure of kernel function "GetTransducerType"
- -101: failure of kernel function "GetTransducerInformation"

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

*ulTransducerSelectionIndex* : Selection value. Value to write for the transducer type selection

*pcName* : Name of the transducer type

*ulCalibrationStatus* : Calibration status

- 0 : Transducer type is not calibrated
- 1 : Transducer type is calibrated

*ulCalibratedChannels* : Bitmask of currently calibrated channels (D0 => channel 1, D1 => channel 1, ...) *ulType* : Type (0: HB 1: LVDT 2:Knaebel 3:HB-Mahr 4:LVDT-Mahr)

*ulFrequency* : Frequency (Hz)

*ulImpedance* : Impedance (Ohm)

*dVeff* : Nominal voltage (Vrms)

*dSensibility* : Sensibility (mv/V/mm)

*dRange* : Range (mm)

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

## 2.16 MSXE371x Auto refresh functions

This function initialise and start the auto refresh acquisition.

## Data Structures

- struct [MSXE371x\\_\\_TransducerGetAllValuesResponse](#)

## Functions

- int [MSXE371x\\_\\_TransducerInitAndStartAutoRefresh](#) (xsd\_\_unsignedLong ulTransducerSelection, xsd\_\_unsignedLong ulChannelMask, xsd\_\_unsignedLong ulAverageMode, xsd\_\_unsignedLong ulAverageValue, xsd\_\_unsignedLong ulTriggerMask, xsd\_\_unsignedLong ulTriggerMode, xsd\_\_unsignedLong ulHardwareTriggerEdge, xsd\_\_unsignedLong ulHardwareTriggerCount, xsd\_\_unsignedLong ulByTriggerNbrOfSeqToAcquire, xsd\_\_unsignedLong ulDataFormat, xsd\_\_unsignedLong ulOption1, xsd\_\_unsignedLong ulOption2, xsd\_\_unsignedLong ulOption3, xsd\_\_unsignedLong ulOption4, struct [MSXE371x\\_\\_Response](#) \*Response)

*Initialise and start the transducer auto refresh acquisition mode.*

- int [MSXE371x\\_\\_TransducerGetAutoRefreshValues](#) (void \_\_\_, struct [MSXE371x\\_\\_unsignedlong9ArrayResponse](#) \*Response)

*This function get the auto refresh counter value an the channels values.*

- int [MSXE371x\\_\\_TransducerGetAllAutoRefreshValues](#) (void \_\_\_, struct [MSXE371x\\_\\_TransducerGetAllValuesResponse](#) \*Response)

*This function return the actualy auto refresh values (Transducer auto refresh counter, transducer values, time stampe, ...).*

- int [MSXE371x\\_\\_TransducerStopAndReleaseAutoRefresh](#) (void \_\_\_, struct [MSXE371x\\_\\_Response](#) \*Response)

*Stop and release the transducer auto refresh acquisition mode.*

### 2.16.1 Detailed Description

In the auto refresh mode the measurement value is updated automatically after each acquisition. The acquisition is initialised and the values of each channels are stored in memory on the Ethernet E/A module MSX-E371x.

The PC reads the data asynchronously to the acquisition via the data socket or a SOAP function.

You can define a mask of all channels that should be acquired.

In the auto refresh mode you can activate the channel average value computation on the module:

Each channel is acquired x times to compute an average value for the channel.

You can start the acquisition by a hardware trigger or a synchro trigger.

The hardware trigger can react to a rising wave, falling wave or both edges.

You have the following possibility:

- Defining a number of edges before a trigger action is generated

There are two trigger modes:(for the hardware or synchro trigger)

a) One shot

b) Sequence

a) One shot:

After the software start, the module is waiting for a trigger signal to start the acquisition. After this the trigger signal is ignored.

b) Sequence:

After the software start the module is waiting for the trigger signal and acquires x sequences (also adjustable) and then wait again.

## 2.16.2 Function Documentation

**2.16.2.1** `int MSXE371x__TransducerInitAndStartAutoRefresh ( xsd__unsignedLong ulTransducerSelection, xsd__unsignedLong ulChannelMask, xsd__unsignedLong ulAverageMode, xsd__unsignedLong ulAverageValue, xsd__unsignedLong ulTriggerMask, xsd__unsignedLong ulTriggerMode, xsd__unsignedLong ulHardwareTriggerEdge, xsd__unsignedLong ulHardwareTriggerCount, xsd__unsignedLong ulByTriggerNbrOfSeqToAcquire, xsd__unsignedLong ulDataFormat, xsd__unsignedLong ulOption1, xsd__unsignedLong ulOption2, xsd__unsignedLong ulOption3, xsd__unsignedLong ulOption4, struct MSXE371x__Response * Response )`

### Parameters

- [in] ***ulTransducerSelection*** Transducer type selection
- [in] ***ulChannelMask*** Mask of the channel to acquire by the auto refresh (1 bit = 1 Channel, D0 channel 0 and D7 channel 7)
- [in] ***ulAverageMode*** Set the average mode :
  - 0 : not used
  - 1 : average per channel
- [in] ***ulAverageValue*** Set the average value (only used, when average is used) :
  - 0 : not used
  - max value : 255
- [in] ***ulTriggerMask*** Define the source of the trigger
  - 0 : trigger disabled
  - 1 : Enable Hardware Digital Input Trigger
  - 2 : Enable Synchro Trigger
  - 4 : Enable Incremental Counter Compare Interrupt
- [in] ***ulTriggerMode*** Define the trigger mode
  - 1 : One shot trigger
  - 2 : Sequence trigger
- [in] ***ulHardwareTriggerEdge*** 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] ***ulHardwareTriggerCount*** Define the number of trigger events before the action occur
  - 0 or 1 : all trigger event start the action
  - max value : 65535

[in] ***ulByTriggerNbrOfSeqToAcquire*** Define the number of sequence to acquire by each trigger event (1..65535)

[in] ***ulDataFormat*** D0 : Time stamp information

- 0: no time stamp information
- 1: time stamp information

D1 : Data format

- 0: Digital value
- 1: Analog value (in mm)

D2 : invert value

- 0 : don't invert the channel value
- 1 : invert the channel value (-2 mm -> + 2mm)

D3 : Temperature value

- 0 : don't acquire the temperature value
- 1 : acquire the temperature value

D4 : Incremental counter value

- 0 : don't acquire the incremental counter value
- 1 : acquire the incremental counter value

D5 : Diff. mode

- 0 : Diff mode disabled
- 1 : Diff mode enabled : Channel X value = Channel (X) value + Channel (X + 4) value

[in] ***ulOption1*** : Reserved

[in] ***ulOption2*** : Reserved

[in] ***ulOption3*** : Reserved

[in] ***ulOption4*** : Reserved

[out] ***Response*** :

***iReturnValue*** : Error value

- 0: success
- -1: means an system error occurred
- -2: Transducer selection error
- -3: The channel mask cannot be null
- -4: Channel Mask error
- -5: not available average mode
- -6: not available average value
- -7: Trigger source : 2 or more different source cannot be simultaneously be activated
- -8: Trigger mode selection error
- -9: Hardware trigger : front definition error
- -10: Hardware trigger count value not available
- -11: Nbr of sequence to acquire by trigger mode not available
- -12: Data format not available
- -13: Incremental counter not initialised
- -14: Incremental counter compare logic not initialised
- -15: Temperature channel not initialised
- -100: Auto refrseh initialisation and start kernel function error ***syserrno*** : system-error code (the value of the libc "errno" code)

## Returns

0: SOAP\_OK  
 <> 0: See SOAP error



### 2.16.2.2 int MSXE371x\_\_TransducerGetAutoRefreshValues ( void \* \_\_, struct MSXE371x\_\_unsignedlong9ArrayResponse \* *Response* )

#### Parameters

[in] \_\_ : no input parameter

[out] *Response* :

*sResponse.iReturnValue* :

- 0 : success
- -100 : Get auto refresh values kernel function error

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

*ulValue* : Array that contain the counter and channels values

- *ulValues* [0] : Auto refresh counter value
- *ulValues* [1] : Channel 0 value
- ...
- *ulValues* [16] : Channel 15 value

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

### 2.16.2.3 int MSXE371x\_\_TransducerGetAllAutoRefreshValues ( void \* \_\_, struct MSXE371x\_\_TransducerGetAllValuesResponse \* *Response* )

#### Parameters

[in] \_\_ : no input parameter

[out] *Response* :

*sResponse.iReturnValue* :

- 0 : success
- -100 : Get auto refresh all values kernel function error

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

*ulAutoRefreshCounter*: Auto refresh counter value *ulTransducersValue*: Transducers values (Array [8]) *ulTimeStampLow*: Time stampe low (micro s) *ulTimeStampHigh*: Time stampe high (s) *ulIncCounter* : Incremental counter value if initialised. *ulExternalTemperature*: External temperature if initialised.

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

### 2.16.2.4 int MSXE371x\_\_TransducerStopAndReleaseAutoRefresh ( void \* \_\_, struct MSXE371x\_\_Response \* *Response* )

#### Parameters

[in] \_\_ : no input parameter

[out] **Response** :

**iReturnValue** :

- 0 : success
- -1: means an system error occurred
- -100: "StopAutoRefresh" kernel function error

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

### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

## 2.17 MSXE371x Sequence functions

A sequence is a list of channels (max 16) that are acquired.

### Functions

- int [MSXE371x\\_\\_TransducerInitAndStartSequence](#) (xsd\_unsignedLong ulTransducerSelection, xsd\_unsignedLong ulNbrOfChannel, struct [MSXE371x\\_\\_unsignedLong8FixedArray](#) \*pulChannelList, xsd\_unsignedLong ulNbrOfSequence, xsd\_unsignedLong ulNbrMaxSequenceToTransfer, xsd\_unsignedLong ulDelayMode, xsd\_unsignedLong ulDelayTimeUnit, xsd\_unsignedLong ulDelayValue, xsd\_unsignedLong ulTriggerMask, xsd\_unsignedLong ulTriggerMode, xsd\_unsignedLong ulHardwareTriggerEdge, xsd\_unsignedLong ulHardwareTriggerCount, xsd\_unsignedLong ulByTriggerNbrOfSeqToAcquire, xsd\_unsignedLong ulDataFormat, xsd\_unsignedLong ulOption1, xsd\_unsignedLong ulOption2, xsd\_unsignedLong ulOption3, xsd\_unsignedLong ulOption4, struct [MSXE371x\\_\\_Response](#) \*Response)

*Initialise and start the transducer sequence acquisition mode.*

- int [MSXE371x\\_\\_TransducerStopAndReleaseSequence](#) (void \*\_ , struct [MSXE371x\\_\\_Response](#) \*Response)

*Stop and release the transducer sequence acquisition mode without sending an end packet.*

- int [MSXE371x\\_\\_TransducerStopAndReleaseSequence2](#) (unsigned long ulEmptyFIFO, unsigned long ulOption1, struct [MSXE371x\\_\\_Response](#) \*Response)

*Stop and release the transducer sequence acquisition mode and send an end packet with the same sequence number.*

### 2.17.1 Detailed Description

It can be any order of the channels in this list.

There are different sequence modes:

- Certain number of sequences / continuous
- With/Without delay

a) Certain number of sequences:

After the acquisition of the defined number of sequences, the acquisition is stopped automatically.

b) Continuous:

The sequences are acquired continuously until a software-stop-command occurs.

c) Without delay:

There is no waiting time between the acquisitions of 2 sequences.

d) With delay:

A delay between 2 sequences can be configured:

For this there are 2 delay types:

> Mode 1: The delay time defines the time between 2 sequence beginnings.

> Mode 2: The delay time defines the time between the end of a sequence until the beginning of the next sequence.

You can start the acquisition by a hardwaretrigger in the auto refresh and sequence mode.

The hardwaretrigger can react to a rising, falling or both edges.

You have the following possibilities:

- Initialising a filter on the trigger input to avoid errors
  
  
  
  
  
  
  
  
  
  
- Defining a number of edges before a trigger action is generated

There are two trigger modes:

a) One shot

b) Sequence

a) One shot:

After the software start, the module is waiting for a trigger signal to start the acquisition. After this the trigger signal is ignored.

b) Sequence:

After the software start the module is waiting for the trigger signal and acquires x sequences (also adjustable) and then wait again.

## 2.17.2 Function Documentation

**2.17.2.1** `int MSXE371x__TransducerInitAndStartSequence ( xsd__unsignedLong ulTransducerSelection, xsd__unsignedLong ulNbrOfChannel, struct MSXE371x__unsignedLong8FixedArray * pulChannelList, xsd__unsignedLong ulNbrOfSequence, xsd__unsignedLong ulNbrMaxSequenceToTransfer, xsd__unsignedLong ulDelayMode, xsd__unsignedLong ulDelayTimeUnit, xsd__unsignedLong ulDelayValue, xsd__unsignedLong ulTriggerMask, xsd__unsignedLong ulTriggerMode, xsd__unsignedLong ulHardwareTriggerEdge, xsd__unsignedLong ulHardwareTriggerCount, xsd__unsignedLong ulByTriggerNbrOfSeqToAcquire, xsd__unsignedLong ulDataFormat, xsd__unsignedLong ulOption1, xsd__unsignedLong ulOption2, xsd__unsignedLong ulOption3, xsd__unsignedLong ulOption4, struct MSXE371x__Response * Response )`

### Parameters

- [in] *ulTransducerSelection* : Transducer type selection
- [in] *ulNbrOfChannel* : nbr of channel in the sequence
- [in] *pulChannelList* : list of the channel who compose the sequence
- [in] *ulNbrOfSequence* : Number of sequence to acquire :
  - 0 : continuous mode
  - <> 0 : number of sequence
- [in] *ulNbrMaxSequenceToTransfer* : Max nbr of sequence to acquire before a data transfer
- [in] *ulDelayMode* : Delay Mode :
  - ADDIDATA\_DELAY\_NOT\_USED 0
  - ADDIDATA\_DELAY\_MODE1\_USED 1
- [in] *ulDelayTimeUnit* : Selection of the unit of *ulDelayValue*
  - 0: ms
  - 1: s
- [in] *ulDelayValue* : Delay Value
- [in] *ulTriggerMask* Define the source of the trigger
  - 0 : trigger disabled
  - 1 : Enable Hardware Digital Input Trigger
  - 2 : Enable Synchro Trigger
  - 4 : Enable Incremental Counter Compare Interrupt
- [in] *ulTriggerMode* Define the trigger mode
  - 1 : One shot trigger
  - 2 : Sequence trigger
- [in] *ulHardwareTriggerEdge* 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] *ulHardwareTriggerCount* Define the number of trigger events before the action occur
  - 0 or 1 : all trigger event start the action
  - max value : 65535

[in] ***ulByTriggerNbrOfSeqToAcquire*** Define the number of sequence to acquire by each trigger event (1..65535)

[in] ***ulDataFormat*** D0 : Time stamp information

- 0: no time stamp information
- 1: time stamp information

D1 : Sequence counter information

- 0 : No sequence counter information
- 1 : Sequence counter information

D2 : Data format

- 0: Digital value
- 1: Analog value (in mm)

D3 : invert value

- 0 : don't invert the channel value
- 1 : invert the channel value (-2 mm -> + 2mm)

D4 : Temperature value

- 0 : don't acquire the temperature value
- 1 : acquire the temperature value

D5 : Incremental counter value

- 0 : don't acquire the incremental counter value
- 1 : acquire the incremental counter value

D6 : Diff. mode

- 0 : Diff mode disabled
- 1 : Diff mode enabled : Channel X value = Channel (X) value + Channel (X + 4) value

D7 : receive the hardware trigger information

- 0 : no hardware trigger information
- 1 : hardware trigger information

D8 : receive the index input information

- 0 : no index input information
- 1 : index input information

[in] ***ulOption1*** : Reserved

[in] ***ulOption2*** : Reserved

[in] ***ulOption3*** : Reserved

[in] ***ulOption4*** : Reserved

[out] ***Response*** :

***iReturnValue*** :

- 0: success
- -1: means an system error occurred
- -2: Transducer selection error
- -3: Channel sequence size error
- -4: Channel array selection error
- -5: Delay mode selection error
- -6: Delay time base selection error

- -7: Delay wait time selection error
- -8: Max sequence to transfer > number of sequence
- -9: Trigger source : 2 or more different source cannot be simultaneously be activated
- -10: Trigger mode selection error
- -11: Hardware trigger : front definition error
- -12: Hardware trigger count value not available
- -13: Nbr of sequence to acquire by trigger mode not available
- -14: Data format not available
- -15: Incremental counter not initialised
- -16: Incremental counter compare logic not initialised
- -17: Temperature channel not initialised
- -100: Init and start sequence acquisition kernel function error *syserrno* : system-error code (the value of the libc "errno" code)

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 2.17.2.2 int MSXE371x\_\_TransducerStopAndReleaseSequence ( void \* \_\_, struct MSXE371x\_\_Response \* *Response* )

##### Parameters

[in] \_ : no input parameter

[out] *Response* :

*iReturnValue* :

- 0 : success
- -1 : means an system error occurred
- -100 : StartStopSequence kernel function error

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

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 2.17.2.3 int MSXE371x\_\_TransducerStopAndReleaseSequence2 ( unsigned long *ulEmptyFIFO*, unsigned long *ulOption1*, struct MSXE371x\_\_Response \* *Response* )

##### Parameters

[in] *ulEmptyFIFO* :

- 0 : Remove all available sequence datas
- 1 : Send any available sequence datas via the data server

[in] *ulOption1* : Reserved

[out] *Response* :

*iReturnValue* :

- 0 : success
- -1 : means an system error occurred
- -100 : StartStopSequence kernel function error

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

### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

## 2.18 MSXE371x diagnostic functions

### Functions

- int [MSXE371x\\_\\_TransducerInit](#) (xsd\_\_unsignedLong ulIndex, xsd\_\_unsignedLong ulFrequency, struct [MSXE371x\\_\\_Response](#) \*Response)  
*Init the transducer and set it to last selected.*
- int [MSXE371x\\_\\_TransducerInitPrimaryConnectionTest](#) (void \*\_, struct [MSXE371x\\_\\_Response](#) \*Response)  
*Initialise the primary connection test.*
- int [MSXE371x\\_\\_TransducerTestPrimaryConnection](#) (void \*\_, struct [MSXE371x\\_\\_unsignedlongDefaultResponse](#) \*Response)  
*Test primary connection.*
- int [MSXE371x\\_\\_TransducerTestPrimaryShortCircuit](#) (void \*\_, struct [MSXE371x\\_\\_unsignedlongDefaultResponse](#) \*Response)  
*Test primary short circuit status.*
- int [MSXE371x\\_\\_TransducerRearmPrimary](#) (void \*\_, struct [MSXE371x\\_\\_unsignedlongDefaultResponse](#) \*Response)  
*Rearm primary.*
- int [MSXE371x\\_\\_TransducerTestSecondaryConnection](#) (xsd\_\_unsignedLong ulChannel, struct [MSXE371x\\_\\_unsignedlongDefaultResponse](#) \*Response)  
*Test the secondary connection.*
- int [MSXE371x\\_\\_TransducerTestSecondaryShortCircuit](#) (xsd\_\_unsignedLong ulChannel, struct [MSXE371x\\_\\_unsignedlongDefaultResponse](#) \*Response)  
*Test the secondary short circuit.*

### 2.18.1 Function Documentation

#### 2.18.1.1 int [MSXE371x\\_\\_TransducerInit](#) ( xsd\_\_unsignedLong *ulIndex*, xsd\_\_unsignedLong *ulFrequency*, struct [MSXE371x\\_\\_Response](#) \* *Response* )

This function must be called before making a diagnostic test.

**Parameters**

- [in] *ulIndex* : Transducer index
- [in] *ulFrequency* : Transducer frequency
- [out] *Response* :
- sResponse.iReturnValue* : Error value
- 0: success
  - -100: failure of kernel function i\_IOKernel\_TransducerInit
  - -101: error driver not in correct state
  - -102: transducer index is invalid
  - -103: kernel copy error
- sResponse.syserrno* : system-error code (the value of the libc "errno" code)

**Returns**

- 0: SOAP\_OK
- <> 0: See SOAP error

### 2.18.1.2 int MSXE371x\_\_TransducerInitPrimaryConnectionTest ( void \* \_\_, struct MSXE371x\_\_Response \* *Response* )

**Parameters**

- [in] *\_\_* : no input parameter
- [out] *Response* :
- iReturnValue* : Error code
- 0: success
  - -1: means an system error occurred
  - -100: Primary connection test initialisation 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 MSXE371x\_\_TransducerTestPrimaryConnection ( void \* \_\_, struct MSXE371x\_\_unsignedlongDefaultResponse \* *Response* )

**Parameters**

- [in] *\_\_* : no input parameter
- [out] *Response* :
- sResponse.iReturnValue* : Error code
- 0: success
  - -1: means an system error occurred
  - -100: Primary connection test kernel function error
- sResponse.syserrno* : system-error code (the value of the libc "errno" code)
- ulValue* : Connection status:



- 0 : connection error
- 1 : connection ok

**Returns**

- 0: SOAP\_OK
- <> 0: See SOAP error

**2.18.1.4 int MSXE371x\_\_TransducerTestPrimaryShortCircuit ( void \* \_\_, struct MSXE371x\_\_unsignedlongDefaultResponse \* Response )****Parameters**

[in] \_\_ : no input parameter

[out] *Response* :

*sResponse.iReturnValue* : Error code

- 0: success
- -1: means an system error occurred
- -100: Primary short circuit test kernel function error *sResponse.syserrno* : system-error code (the value of the libc "errno" code)

*ulValue* : short circuit status:

- 0 : short circuit
- 1 : no short circuit

**Returns**

- 0: SOAP\_OK
- <> 0: See SOAP error

**2.18.1.5 int MSXE371x\_\_TransducerRearmPrimary ( void \* \_\_, struct MSXE371x\_\_unsignedlongDefaultResponse \* Response )****Parameters**

[in] \_\_ : no input parameter

[out] *Response* :

*sResponse.iReturnValue* : Error code

- 0: success
- -1: means an system error occurred
- -100: Primary short circuit rearm kernel function error *sResponse.syserrno* : system-error code (the value of the libc "errno" code)

*ulValue* : Rearm status:

- 0 : Rearm not ok
- 1 : Rearm ok

**Returns**

- 0: SOAP\_OK
- <> 0: See SOAP error

### 2.18.1.6 int MSXE371x\_\_TransducerTestSecondaryConnection ( xsd\_\_unsignedLong ulChannel, struct MSXE371x\_\_unsignedlongDefaultResponse \* Response )

#### Parameters

[in] *ulChannel*,: channel selection

[out] *Response* :

*sResponse.iReturnValue* : Error code

- 0: success
- -1: means an system error occurred
- -100: Secondary connection test kernel function error *sResponse.syserrno* : system-error code (the value of the libc "errno" code)
- ulValue* : Connection status:
  - 0 : connection error
  - 1 : connection ok

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

### 2.18.1.7 int MSXE371x\_\_TransducerTestSecondaryShortCircuit ( xsd\_\_unsignedLong ulChannel, struct MSXE371x\_\_unsignedlongDefaultResponse \* Response )

#### Parameters

[in] *ulChannel*,: channel selection

[out] *Response* :

*sResponse.iReturnValue* : Error code

- 0: success
- -1: means an system error occurred
- -100: Secondary short circuit test kernel function error *sResponse.syserrno* : system-error code (the value of the libc "errno" code)
- ulValue* : short circuit status:
  - 0 : short circuit
  - 1 : no short circuit

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

## 2.19 MSX-E371x incremental counter functions

### Data Structures

- struct [MSXE371x\\_\\_IncCounterRead32BitValueResponse](#)

## Functions

- `int MSXE371x__IncCounterInit (xsd__unsignedLong ulCounterMode, xsd__unsignedLong ulCounterOption, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE371x__Response *Response)`  
*Initialise the counter.*
- `int MSXE371x__IncCounterRelease (void *_ , struct MSXE371x__Response *Response)`  
*Release the counter.*
- `int MSXE371x__IncCounterRead32BitValue (void *_ , struct MSXE371x__IncCounterRead32BitValueResponse *Response)`  
*Read the 32 bits counter value.*
- `int MSXE371x__IncCounterWrite32BitValue (xsd__unsignedLong ulCounterValue, struct MSXE371x__Response *Response)`  
*write a 32 bits counter value*
- `int MSXE371x__IncCounterClear (void *_ , struct MSXE371x__Response *Response)`  
*Clear the 32 bits counter.*
- `int MSXE371x__IncCounterInitAndEnableCompareLogic (xsd__unsignedLong ulValue, xsd__unsignedLong ulMode, xsd__unsignedLong ulSynchroTrigger, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MSXE371x__Response *Response)`  
*Init and enable a counter compare value.*
- `int MSXE371x__IncCounterDisableAndReleaseCompareLogic (void *_ , struct MSXE371x__Response *Response)`  
*Disable and Release a counter compare value.*
- `int MSXE371x__IncCounterInitAndEnableIndex (xsd__unsignedLong ulReferenceAction, xsd__unsignedLong ulIndexOperation, xsd__unsignedLong ulAutoMode, xsd__unsignedLong ulOption01, struct MSXE371x__Response *Response)`  
*Init and enable the counter index.*
- `int MSXE371x__IncCounterDisableAndReleaseIndex (void *_ , struct MSXE371x__Response *Response)`  
*Disable and Release the counter index.*

### 2.19.1 Function Documentation

**2.19.1.1** `int MSXE371x__IncCounterInit ( xsd__unsignedLong ulCounterMode, xsd__unsignedLong ulCounterOption, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE371x__Response * Response )`

#### Parameters

[in] *ulCounterMode* : Set the counter mode : either

- MSXE371x\_COUNTER\_QUADRUPLE\_MODE (0x4)
- MSXE371x\_COUNTER\_DOUBLE\_MODE (0x2)
- MSXE371x\_COUNTER\_SIMPLE\_MODE (0x1)
- MSXE371x\_COUNTER\_DIRECT\_MODE (0x0)

[in] ***ulCounterOption*** : Set the counter option

if in QUADRUPLE/DOUBLE/SIMPLE mode : either

- MSXE371x\_COUNTER\_HYSTERESIS\_ON (0x1)
- MSXE371x\_COUNTER\_HYSTERESIS\_OFF (0x0)

if in DIRECT mode :

- MSXE371x\_COUNTER\_INCREMENT (0x0)
- MSXE371x\_COUNTER\_DECREMENT (0x1)

[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: success
- -1: means an system error occurred
- -2: Counter mode selection error
- -3: Counter option selection error
- -100: Init counter kernel function 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 MSXE371x\_\_IncCounterRelease ( void \* \_\_, struct MSXE371x\_\_Response \* Response )

## Parameters

[in] ***\_\_*** : no input parameter

[out] ***Response*** :

***iReturnValue*** :

- 0: success
- -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

### 2.19.1.3 int MSXE371x\_\_IncCounterRead32BitValue ( void \* \_\_, struct MSXE371x\_\_IncCounterRead32BitValueResponse \* Response )

#### Parameters

[in] \_\_ : no input parameter

[out] *Response* :

*sResponse.iReturnValue* :

- 0: success
- -1: means an system error occurred
- -100: Read counter value kernel function error

*sResponse.syserrno* : system-error code (the value of the libc "errno" code) *ulValue* : counter value *ulTimeStampLow* : 32 bit low part of time stamp (us) *ulTimeStampHigh* : 32 bit high part of time stamp (s)

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

### 2.19.1.4 int MSXE371x\_\_IncCounterWrite32BitValue ( xsd\_\_unsignedLong ulCounterValue, struct MSXE371x\_\_Response \* Response )

#### Parameters

[in] *ulCounterValue* : Counter value

[out] *Response* :

*iReturnValue* :

- 0: success
- -1: means an system error occurred
- -2: Counter value error
- -100: Write counter value kernel function error

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

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

### 2.19.1.5 int MSXE371x\_\_IncCounterClear ( void \* \_\_, struct MSXE371x\_\_Response \* Response )

#### Parameters

[in] \_\_ : no input parameter

[out] *Response* :

*iReturnValue* :

- 0: success
- -1: means an system error occurred

- -100: Write counter value kernel function error

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

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**2.19.1.6** `int MSXE371x__IncCounterInitAndEnableCompareLogic ( xsd__unsignedLong ulValue, xsd__unsignedLong ulMode, xsd__unsignedLong ulSynchroTrigger, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MSXE371x__Response * Response )`

#### Parameters

[in] *ulValue* : compare value ( 0 to 0xFFFFFFFF included )

[in] *ulMode* : compare mode

- 0: condition true when counter equals compare value
- 1: condition true when counter equals a multiple of the compare value

[in] *ulSynchroTrigger* • 0 : no synchro trigger

- 1 : generates a synchro trigger when condition is true

[in] *ulOption01* : set it to 0

[in] *ulOption02* : set it to 0

[out] *Response* :

*iReturnValue* :

- 0: success
- -1: means an system error occurred
- -2: Compare value error
- -3: Compare mode error
- -4: Synchro trigger error
- -100: Write counter compare value kernel function error
- -101: Enable counter compare kernel function error

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

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**2.19.1.7** `int MSXE371x__IncCounterDisableAndReleaseCompareLogic ( void * _, struct MSXE371x__Response * Response )`

#### Parameters

[in] *\_* : no input parameter

[out] *Response* :

*iReturnValue* :

- 0: success
- -1: means an system error occurred
- -100: Disable counter compare value kernel function error

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

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**2.19.1.8** `int MSXE371x__IncCounterInitAndEnableIndex ( xsd__unsignedLong  
ulReferenceAction, xsd__unsignedLong ulIndexOperation, xsd__unsignedLong  
ulAutoMode, xsd__unsignedLong ulOption01, struct MSXE371x__Response * Response  
)`

#### Parameters

[in] *ulReferenceAction* : Reference action

- 0: do not use the pin D as reference
- 1: use the pin D as reference

[in] *ulIndexOperation* : Index operation

- 0: latch and clear counter at low edge
- 1: latch and clear counter at high edge

[in] *ulAutoMode* : Auto mode

- 0 : do not use automode (action is done only once)
- 1 : use automode (action is done continuously)

[in] *ulOption01* : set it to 0

[out] *Response* :

*iReturnValue* :

- 0: success
- -1: means an system error occurred
- -2: Reference action selection error
- -3: Index operation selection error
- -4: Automode selection error
- -5: Index logic already initialized
- -100: Write counter compare value kernel function error

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

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

### 2.19.1.9 int MSXE371x\_\_IncCounterDisableAndReleaseIndex ( void \* \_\_\_, struct MSXE371x\_\_Response \* Response )

#### Parameters

[in] \_\_\_ : no input parameter

[out] **Response** :

**iReturnValue** :

- 0: success
- -1: means an system error occurred
- -2: Index logic already initialized
- -100: Disable counter compare value kernel function error

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

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

## 2.20 MSX-E371x external temperature functions

### Data Structures

- struct [MSXE371x\\_\\_ExternalTemperatureReadResponse](#)

### Functions

- int [MSXE371x\\_\\_ExternalTemperatureInit](#) (xsd\_\_unsignedLong ulConnectedTempSensor, xsd\_\_unsignedLong ulConvertMode, xsd\_\_unsignedLong ulGainSelection, xsd\_\_unsignedLong ulFrequencySelection, xsd\_\_unsignedLong ulPowerSaveMode, xsd\_\_unsignedLong ulOption01, xsd\_\_unsignedLong ulOption02, xsd\_\_unsignedLong ulOption03, xsd\_\_unsignedLong ulOption04, struct [MSXE371x\\_\\_Response](#) \*Response)

*Initialise the external temperature.*

- int [MSXE371x\\_\\_ExternalTemperatureRelease](#) (void \_\_\_, struct [MSXE371x\\_\\_Response](#) \*Response)

*Release the external temperature.*

- int [MSXE371x\\_\\_ExternalTemperatureRead](#) (void \_\_\_, struct [MSXE371x\\_\\_ExternalTemperatureReadResponse](#) \*Response)

*Read the external temperature value.*

- int [MSXE371x\\_\\_ExternalTemperatureGainCalibration](#) (unsigned long ulGainSelection, double dRefValue, struct [MSXE371x\\_\\_Response](#) \*Response)

*Temperature gain calibration.*



## 2.20.1 Function Documentation

**2.20.1.1** `int MSXE371x_ExternalTemperatureInit ( xsd__unsignedLong  
ulConnectedTempSensor, xsd__unsignedLong ulConvertMode, xsd__unsignedLong  
ulGainSelection, xsd__unsignedLong ulFrequencySelection, xsd__unsignedLong  
ulPowerSaveMode, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02,  
xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct  
MSXE371x_Response * Response )`

### Parameters

- [in] **ulConnectedTempSensor** Connected sensor type
- 1: TC type E
  - 2: TC type J
  - 3: TC type K
  - 4: TC type N
  - 5: TC type R
  - 6: TC type S
  - 7: TC type T
  - 100 for PT100
  - 200 for PT200
  - 500 for PT500
  - 1000 for PT1000
- [in] **ulConvertMode** Converting mode
- 0: Disabled. Return the value into the mOhm form
  - 1: °C
  - 2: °F
- If you use a TC sensor, you can only ask the value in °C or °F
- [in] **ulGainSelection** Gain selection (0 for auto mode)
- If you use a TC sensor, only use auto gain (0)
- [in] **ulFrequencySelection** Acquisition frequency selection (10,50, 240)
- If you use a TC sensor, you can only use frequencies 50 and 240
- [in] **ulPowerSaveMode** Power save mode selection
- 0 : Start the continuous acquisition
  - 1 : Disable the acquisition.
- [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: success
  - -1: means an system error occurred
  - -2: Connected temperature sensor selection error
  - -3: Convert mode selection error
  - -4: Gain selection error
  - -5: Frequency selection error
  - -6: Power save mode selection error

- -100: Init external temperature kernel function error

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

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**2.20.1.2 int MSXE371x\_\_ExternalTemperatureRelease ( void \* \_\_, struct MSXE371x\_\_Response \* Response )**

#### Parameters

[in] **\_\_** : no input parameter

[out] **Response** :

**iReturnValue** :

- 0: success
- -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

**2.20.1.3 int MSXE371x\_\_ExternalTemperatureRead ( void \* \_\_, struct MSXE371x\_\_ExternalTemperatureReadResponse \* Response )**

#### Parameters

[in] **\_\_** : no input parameter

[out] **Response** :

**sResponse.iReturnValue** :

- 0: success
- -1: means an system error occurred
- -2: Power save enabled and no transducer acquisition started
- -3: Power save enabled and transducer acquisition started but hardware trigger used
- -100: Read external temperature kernel function error

**sResponse.syserrno** : system-error code (the value of the libc "errno" code) **ulValue** : External temperature value. Format depend from the initialisation **ulTimeStampHigh** : 32 bit high part of time stamp (s) **ulTimeStampLow** : 32 bit low part of time stamp (us)

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 2.20.1.4 int MSXE371x\_\_ExternalTemperatureGainCalibration ( unsigned long ulGainSelection, double dRefValue, struct MSXE371x\_\_Response \* Response )

##### Parameters

[in] **ulGainSelection** : Selected gain (1, 2, 4, 8, 16, 32, 64 or 128)

[in] **dRefValue** : Ref voltage value the the calibration

[out] **Response** :

**iReturnValue** :

- 0 : success
- -1 : means an system error occurred
- -2 : Gain selection error
- -3 : Frequency selection error
- -4 : Ref value to high for the selected gain
- -5 : Calibration file access error
- -100 : IOCTL system call error

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

##### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

## 2.21 MSXE371x calibration functions

### Data Structures

- struct [MSXE371x\\_\\_CalibrationGetCurrentStatusResponse](#)

### Functions

- int [MSXE371x\\_\\_CalibrationStart](#) (xsd\_\_unsignedLong ulTransducerIndex, xsd\_\_double dPosition, xsd\_\_unsignedLong ulCalibrationMode, xsd\_\_unsignedLong ulInvertValue, xsd\_\_unsignedLong ulChannel, struct [MSXE371x\\_\\_Response](#) \*Response)

*This function start the calibration thread.*

- int [MSXE371x\\_\\_CalibrationGetCurrentStatus](#) (void \*\_, struct [MSXE371x\\_\\_CalibrationGetCurrentStatusResponse](#) \*Response)

*This function return the current calibration status.*

- int [MSXE371x\\_\\_CalibrationNextStep](#) (void \*\_, struct [MSXE371x\\_\\_Response](#) \*Response)

*This function start the next calibration step.*

- int [MSXE371x\\_\\_CalibrationBreak](#) (void \*\_, struct [MSXE371x\\_\\_Response](#) \*Response)

*This function break the current calibration.*

## 2.21.1 Function Documentation

**2.21.1.1** `int MSXE371x__CalibrationStart ( xsd__unsignedLong ulTransducerIndex, xsd__double dPosition, xsd__unsignedLong ulCalibrationMode, xsd__unsignedLong ulInvertValue, xsd__unsignedLong ulChannel, struct MSXE371x__Response * Response )`

### Parameters

- [in] ***ulTransducerIndex*** : Selected transducer to calibrate
- [in] ***dPosition*** : Selected user calibration position
- [in] ***ulCalibrationMode*** : Select the calibration mode
  - 0: Calibre the primary and all channels. ulChannel not used
  - 1: Calibre only the selected channel (ulChannel).
  - 2: Calibre the primary and all channels. ulChannel not used and reset the calibration bit
  - 3: Calibre only the selected channel (ulChannel) and reset the calibration bit If not other channel calibrate then calibrate the primary and 0mm position
- [in] ***ulInvertValue*** :
  - 0 : don't invert the channel value
  - 1 : invert the channel value (-2 mm -> + 2mm)
- [in] ***ulChannel*** : Selected calibration channel
- [out] ***Response*** :
  - iReturnValue*** : Error code :
    - 0: means the remote function performed OK
    - -1: means an system error occured
    - -2: Any calibration already started
    - -3: Selected transducer not available
    - -4: Calibration mode selection error
    - -5: Invert value selection mode not available
    - -6: Transducer channel selection error
    - -100: Start calibration kernel function error ***syserrno*** : system-error code (the value of the libc "errno" code)

### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**2.21.1.2** `int MSXE371x__CalibrationGetCurrentStatus ( void * _, struct MSXE371x__CalibrationGetCurrentStatusResponse * Response )`

### Parameters

- [in] ***\_*** : no input parameter
- [out] ***Response*** :
- [out] ***Response*** :
  - sResponse.iReturnValue*** : Error code :
    - 0: means the remote function performed OK

- -1: means an system error occurred
- -100: Get the calibration status kernel function error *sResponse.syserrno* : system-error code (the value of the libc "errno" code) *ulStatus* : Status
- 0: No calibration in progress
- 1: Primary calibration in progress
- 2: Wait user access null position setting
- 3: Null position calibration in progress
- 4: Wait user access user position setting
- 5: User position calibration in progress
- 6: Wait user connect only one transducer for the primary open line diagnostic
- 7: Primary open line diagnostic in progress
- 8: Calibration finiched

*ulChannel* : Current calibration channel

*ulDigitalValue* : Last measured digital value

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 2.21.1.3 int MSXE371x\_\_CalibrationNextStep ( void \* \_, struct MSXE371x\_\_Response \* Response )

##### Parameters

[in] \_ : no input parameter

[out] *Response* :

*iReturnValue* : Error code :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -100: Start netxt calibration step kernel function error *syserrno* : system-error code (the value of the libc "errno" code)

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 2.21.1.4 int MSXE371x\_\_CalibrationBreak ( void \* \_, struct MSXE371x\_\_Response \* Response )

##### Parameters

[in] \_ : no input parameter

[out] *Response* :

*iReturnValue* : Error code :

- 0: means the remote function performed OK

- -1: means an system error occurred
- -100: Break calibration kernel function error *syserrno* : system-error code (the value of the libc "errno" code)

### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

## 2.22 MSXE371x transducer database management functions

These functions allows to manage the database of transducer types on the module.

### Data Structures

- struct [MSXE371x\\_\\_DataBaseGetTransducerInformationResponse](#)

### Functions

- int [MSXE371x\\_\\_DataBaseGetNumberOfTransducers](#) (void \*\_\_, struct [MSXE371x\\_\\_unsignedlongResponse](#) \*Response)

*Returns the number of transducer types currently defined in the database.*

- int [MSXE371x\\_\\_DataBaseGetTransducerType](#) (xsd\_\_unsignedLong ulTransducerIndex, struct [MSXE371x\\_\\_unsignedlongResponse](#) \*Response)

*Returns the transducer identifier of the selected transducer.*

- int [MSXE371x\\_\\_DataBaseGetTransducerInformation](#) (xsd\_\_unsignedLong ulTransducerIndex, struct [MSXE371x\\_\\_DataBaseGetTransducerInformationResponse](#) \*Response)

*Returns the information stored in the database about the type.*

- int [MSXE371x\\_\\_DataBaseAddTransducer](#) (xsd\_\_unsignedLong ulTransducerIndex, xsd\_\_string cName, xsd\_\_unsignedLong ulType, xsd\_\_unsignedLong ulFrequency, xsd\_\_unsignedLong ulImpedance, xsd\_\_double dVeff, xsd\_\_double dSensitivity, xsd\_\_double dRange, struct [MSXE371x\\_\\_Response](#) \*Response)

*Adds a new transducer type definition into the database of the module.*

- int [MSXE371x\\_\\_DataBaseDelTransducer](#) (xsd\_\_unsignedLong ulTransducerIndex, struct [MSXE371x\\_\\_Response](#) \*Response)

*Deletes the selected transducer from the transducer database.*

- int [MSXE371x\\_\\_DataBaseSaveTransducers](#) (void \*\_\_, struct [MSXE371x\\_\\_ByteArrayResponse](#) \*Response)

*Commits the current changes in the transducer database, including the calibration values.*

## 2.22.1 Function Documentation

### 2.22.1.1 `int MSXE371x__DataBaseGetNumberOfTransducers ( void * _, struct MSXE371x__unsignedlongResponse * Response )`

#### Parameters

[in] `_` : no input parameter

[out] `Response` :

`sResponse.iReturnValue` : Error code :

- 0: success
- <> 0 : error
- -100: kernel function error

`sResponse.syserrno` : system-error code (the value of the libc "errno" code) `ulValue` : number of transducer types.

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

### 2.22.1.2 `int MSXE371x__DataBaseGetTransducerType ( xsd__unsignedLong ulTransducerIndex, struct MSXE371x__unsignedlongResponse * Response )`

#### Parameters

[in] `ulTransducerIndex` : Transducer index (0 to ulTransducerNumbers - 1)

[out] `Response` :

`sResponse.iReturnValue` : Error code :

- 0: success
- <> 0 : error
- -100: kernel function error

`sResponse.syserrno` : system-error code (the value of the libc "errno" code) Its value is significant only when the iReturnValue returned an error (-1 or <= -100)

Give this value to the MXCommon\_Sterror to get the string describing the error number. **ul-Value** : transducer identifier. Use this value as the "Index" parameter given to DataBaseGetTransducerInformationResponse().

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

### 2.22.1.3 `int MSXE371x__DataBaseGetTransducerInformation ( xsd__unsignedLong ulTransducerIndex, struct MSXE371x__DataBaseGetTransducerInformationResponse * Response )`

#### Parameters

[in] `ulTransducerIndex` : transducer identifier, as returned by DataBaseGetTransducerType().

[out] **Response** :

**sResponse.iReturnValue** : Error code :

- 0: success
- <> 0 : error
- -100: kernel function error

**sResponse.syserrno** : system-error code (the value of the libc "errno" code) Its value is significant only when the iReturnValue returned an error (-1 or <= -100)

Give this value to the MXCommon\_Strerror to get the string describing the error number.

**cName** : Name

**ulCalibrate** : Calibration state (0 : not calibrated 1 : calibrated)

**ulType** : Type (0: HB 1: LVDT 2:Knaebel 3:HB-Mahr 4:LVDT-Mahr)

**ulFrequency** : Nominal frequency (Hz)

**ulImpedance** : Impedance (Ohm)

**dVeff** : Nominal voltage (Vrms)

**dSensitivity** : Sensitivity (mV/V/mm)

**dRange** : Range (mm) **ulCalibratedChannels** : Bitmask of currently calibrated channels (D0 => channel 1, D1 => channel 1, ...)

## Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**2.22.1.4 int MSXE371x\_\_DataBaseAddTransducer ( xsd\_\_unsignedLong ulTransducerIndex, xsd\_\_string cName, xsd\_\_unsignedLong ulType, xsd\_\_unsignedLong ulFrequency, xsd\_\_unsignedLong ulImpedance, xsd\_\_double dVeff, xsd\_\_double dSensitivity, xsd\_\_double dRange, struct MSXE371x\_\_Response \* Response )**

## Parameters

[in] **ulTransducerIndex** : Identifier of the new type, user-defined value in the range 200 .. 255

[in] **cName** : Name

[in] **ulType** : Type (0: HB 1: LVDT 2:Knaebel 3:HB-Mahr 4:LVDT-Mahr)

[in] **ulFrequency** : Nominal frequency (Hz)

[in] **ulImpedance** : Impedance (Ohm)

[in] **dVeff** : Nominal voltage (Vrms)

[in] **dSensitivity** : Sensitivity (mV/V/mm)

[in] **dRange** : Range (mm)

[out] **Response** :

**iReturnValue** : Error code :

- 0: success
- <> 0 : error
- -100: kernel function error

**syserrno** : system-error code (the value of the libc "errno" code) Its value is significant only when the iReturnValue returned an error (-1 or <= -100)

Give this value to the MXCommon\_Strerror to get the string describing the error number.

## Returns

- 0: SOAP\_OK



- <> 0: See SOAP error

#### Note

This function returns an error if a transducer with the same identifier already exists in the database.

#### 2.22.1.5 int MSXE371x\_\_DataBaseDelTransducer ( xsd\_\_unsignedLong *ulTransducerIndex*, struct MSXE371x\_\_Response \* *Response* )

##### Parameters

[in] *ulTransducerIndex* : identifier, as returned by DataBaseGetTransducerType().

[out] *Response* :

*iReturnValue* : Error value :

- 0: success
- <> 0 : error
- -100: kernel function error

*syserrno* : system-error code (the value of the libc "errno" code) Its value is significant only when the *iReturnValue* returned an error (-1 or <= -100)

Give this value to the MXCommon\_Sterror to get the string describing the error number.

##### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### Note

This function returns an error if the identifier does not map to an existing transducer type.

#### 2.22.1.6 int MSXE371x\_\_DataBaseSaveTransducers ( void \* \_, struct MSXE371x\_\_ByteArrayResponse \* *Response* )

##### Parameters

[in] \_ : no input parameter

[out] *Response* : *sResponse.iReturnValue* : Error code

- 0 success
- <> 0 : error

*sResponse.syserrno* : system-error code (the value of the libc "errno" code) Its value is significant only when the *iReturnValue* returned an error (-1 or <= -100)

Give this value to the MXCommon\_Sterror to get the string describing the error number.

##### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error



## 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 MSXE371x\_\_ByteArrayResponse Struct Reference

### Data Fields

- struct [DefaultResponse sResponse](#)  
*Default return values.*
- struct [ByteArray sArray](#)  
*Dynamic Array of byte - encapsulates C-type strings.*

### 3.3.1 Field Documentation

#### 3.3.1.1 struct DefaultResponse MSXE371x\_\_ByteArrayResponse::sResponse

#### 3.3.1.2 struct ByteArray MSXE371x\_\_ByteArrayResponse::sArray

## 3.4 MSXE371x\_\_CalibrationGetCurrentStatusResponse Struct Reference

### Data Fields

- struct [DefaultResponse sResponse](#)  
*return value of the call :*
- [xsd\\_\\_unsignedLong ulStatus](#)  
*Calibration status :*
- [xsd\\_\\_unsignedLong ulChannel](#)  
*Current calibration channel.*
- [xsd\\_\\_unsignedLong ulDigitalValue](#)  
*Last measured digital value.*

### 3.4.1 Field Documentation

#### 3.4.1.1 struct DefaultResponse MSXE371x\_\_CalibrationGetCurrentStatusResponse::sResponse

- 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

#### 3.4.1.2 xsd\_\_unsignedLong MSXE371x\_\_CalibrationGetCurrentStatusResponse::ulStatus

- 0: No calibration in progress
- 1: Primary calibration in progress
- 2: Wait user access null position setting
- 3: Null position calibration in progress
- 4: Wait user access user position setting
- 5: User position calibration in progress
- 6: Wait user connect only one transducer for the primary open line diagnostic
- 7: Primary open line diagnostic in progress
- 8: Calibration finished
- 9: Wait user inverted position setting
- 10: Inverted user position calibration in progress

#### 3.4.1.3 xsd\_\_unsignedLong MSXE371x\_\_CalibrationGetCurrentStatusResponse::ulChannel

#### 3.4.1.4 xsd\_\_unsignedLong MSXE371x\_\_CalibrationGetCurrentStatusResponse::ulDigitalValue

## 3.5 MSXE371x\_\_DataBaseGetTransducerInformationResponse Struct Reference

### Data Fields

- struct [DefaultResponse](#) sResponse
- [xsd\\_\\_unsignedByte](#) cName [100]  
*Name.*
- [xsd\\_\\_unsignedLong](#) ulCalibrate  
*Calibration state (0 : not calibrated 1 : calibrated).*
- [xsd\\_\\_unsignedLong](#) ulCalibratedChannels  
*Bitmask of currently calibrated channels (D0 => channel 1, D1 => channel 1, ...).*

- [xsd\\_\\_unsignedLong ulType](#)

*Type (0: HB 1: LVDT 2:Knaebel 3:HB-Mahr 4:LVDT-Mahr).*

- [xsd\\_\\_unsignedLong ulFrequency](#)

*Nominal frequency (Hz).*

- [xsd\\_\\_unsignedLong ulImpedance](#)

*Load impedance (ohm).*

- [xsd\\_\\_double dVeff](#)

*Nominal voltage (Vrms).*

- [xsd\\_\\_double dSensitivity](#)

*Sensibility (mV/V/mm).*

- [xsd\\_\\_double dRange](#)

*Range (mm).*

### 3.5.1 Field Documentation

- 3.5.1.1 `struct DefaultResponse MSXE371x__ - DataBaseGetTransducerInformationResponse::sResponse`
- 3.5.1.2 `xsd__unsignedByte MSXE371x__ - DataBaseGetTransducerInformationResponse::cName[100]`
- 3.5.1.3 `xsd__unsignedLong MSXE371x__ - DataBaseGetTransducerInformationResponse::ulCalibrate`
- 3.5.1.4 `xsd__unsignedLong MSXE371x__ - DataBaseGetTransducerInformationResponse::ulCalibratedChannels`
- 3.5.1.5 `xsd__unsignedLong MSXE371x__ DataBaseGetTransducerInformationResponse::ulType`
- 3.5.1.6 `xsd__unsignedLong MSXE371x__ - DataBaseGetTransducerInformationResponse::ulFrequency`
- 3.5.1.7 `xsd__unsignedLong MSXE371x__ - DataBaseGetTransducerInformationResponse::ulImpedance`
- 3.5.1.8 `xsd__double MSXE371x__ DataBaseGetTransducerInformationResponse::dVeff`
- 3.5.1.9 `xsd__double MSXE371x__ DataBaseGetTransducerInformationResponse::dSensitivity`
- 3.5.1.10 `xsd__double MSXE371x__ DataBaseGetTransducerInformationResponse::dRange`

## 3.6 MSXE371x\_\_ExternalTemperatureReadResponse 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 ulTimeStampHigh`

*the meaning of this value is defined in the related header of the function who use this type (s)*

- `xsd__unsignedLong ulTimeStampLow`

*the meaning of this value is defined in the related header of the function who use this type (us)*

### 3.6.1 Field Documentation

3.6.1.1 struct `DefaultResponse MSXE371x__ExternalTemperatureReadResponse::sResponse`

3.6.1.2 `xsd__unsignedLong MSXE371x__ExternalTemperatureReadResponse::ulValue`

3.6.1.3 `xsd__unsignedLong MSXE371x__ExternalTemperatureReadResponse::ulTimeStampHigh`

3.6.1.4 `xsd__unsignedLong MSXE371x__ExternalTemperatureReadResponse::ulTimeStampLow`

## 3.7 MSXE371x\_\_IncCounterRead32BitValueResponse 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*

### 3.7.1 Field Documentation

3.7.1.1 struct `DefaultResponse MSXE371x__IncCounterRead32BitValueResponse::sResponse`

3.7.1.2 `xsd__unsignedLong MSXE371x__IncCounterRead32BitValueResponse::ulValue`

3.7.1.3 `xsd__unsignedLong MSXE371x__IncCounterRead32BitValueResponse::ulTimeStampLow`

3.7.1.4 `xsd__unsignedLong MSXE371x__IncCounterRead32BitValueResponse::ulTimeStampHigh`

## 3.8 MSXE371x\_\_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)*



### 3.8.1 Field Documentation

#### 3.8.1.1 xsd\_\_int MSXE371x\_\_Response::iReturnValue

- 0 success
- -1 means a system error occurred, the meaning of other values is function dependant and should be defined in the related header

#### 3.8.1.2 xsd\_\_int MSXE371x\_\_Response::syserrno

## 3.9 MSXE371x\_\_TransducerGetAllValuesResponse Struct Reference

### Data Fields

- struct [DefaultResponse](#) sResponse

*Default return values.*

- [xsd\\_\\_unsignedLong](#) ulAutoRefreshCounter

*Auto refresh counter value.*

- [xsd\\_\\_unsignedLong](#) ulTransducersValue [8]

*Transducers values.*

- [xsd\\_\\_unsignedLong](#) ulTimeStampLow

*Time stampe low (micro s).*

- [xsd\\_\\_unsignedLong](#) ulTimeStampHigh

*Time stampe high (s).*

- [xsd\\_\\_unsignedLong](#) ulIncCounter

*Incremental counter value if initialised.*

- [xsd\\_\\_unsignedLong](#) ulExternalTemperature

*External temperature if initialised.*

### 3.9.1 Field Documentation

3.9.1.1 struct `DefaultResponse MSXE371x__TransducerGetAllValuesResponse::sResponse`

3.9.1.2 `xsd__unsignedLong MSXE371x__TransducerGetAllValuesResponse::ulAutoRefreshCounter`

3.9.1.3 `xsd__unsignedLong MSXE371x__TransducerGetAllValuesResponse::ulTransducersValue[8]`

3.9.1.4 `xsd__unsignedLong MSXE371x__TransducerGetAllValuesResponse::ulTimeStampLow`

3.9.1.5 `xsd__unsignedLong MSXE371x__TransducerGetAllValuesResponse::ulTimeStampHigh`

3.9.1.6 `xsd__unsignedLong MSXE371x__TransducerGetAllValuesResponse::ulIncCounter`

3.9.1.7 `xsd__unsignedLong MSXE371x__TransducerGetAllValuesResponse::ulExternalTemperature`

## 3.10 MSXE371x\_\_TransducerGetTypeInformationResponse Struct Reference

### Data Fields

- struct `DefaultResponse sResponse`  
*Default return values.*
- `xsd__unsignedLong ulTransducerSelectionIndex`  
*Identifier.*
- `xsd__unsignedByte pcName [100]`  
*Name of the transducer type.*
- `xsd__unsignedLong ulCalibrationStatus`  
*Calibration status.*
- `xsd__unsignedLong ulCalibratedChannels`  
*Bitmask of currently calibrated channels (D0 => channel 1, D1 => channel 1, ...).*
- `xsd__unsignedLong ulType`  
*Type (0: HB 1: LVDT 2:Knaebel 3:HB-Mahr 4:LVDT-Mahr).*
- `xsd__unsignedLong ulFrequency`  
*Frequency (Hz).*
- `xsd__unsignedLong ulImpedance`  
*Impedance (Ohm).*
- `xsd__double dVeff`  
*Nominal voltage (Vrms).*

- [xsd\\_\\_double dSensibility](#)

*Sensibility (mv/V/mm).*

- [xsd\\_\\_double dRange](#)

*Range (mm).*

### 3.10.1 Field Documentation

**3.10.1.1 struct DefaultResponse MSXE371x\_\_ -  
TransducerGetTypeInformationResponse::sResponse**

**3.10.1.2 xsd\_\_unsignedLong MSXE371x\_\_ -  
TransducerGetTypeInformationResponse::ulTransducerSelectionIndex**

Value to use for the transducer type selection in the other SOAP functions.

**3.10.1.3 xsd\_\_unsignedByte MSXE371x\_\_ -  
TransducerGetTypeInformationResponse::pcName[100]**

**3.10.1.4 xsd\_\_unsignedLong MSXE371x\_\_ -  
TransducerGetTypeInformationResponse::ulCalibrationStatus**

- 0 : Transducer type is not calibrated
- 1 : Transducer type is calibrated

**3.10.1.5 xsd\_\_unsignedLong MSXE371x\_\_ -  
TransducerGetTypeInformationResponse::ulCalibratedChannels**

**3.10.1.6 xsd\_\_unsignedLong MSXE371x\_\_TransducerGetTypeInformationResponse::ulType**

**3.10.1.7 xsd\_\_unsignedLong MSXE371x\_\_ -  
TransducerGetTypeInformationResponse::ulFrequency**

**3.10.1.8 xsd\_\_unsignedLong MSXE371x\_\_ -  
TransducerGetTypeInformationResponse::ulImpedance**

**3.10.1.9 xsd\_\_double MSXE371x\_\_TransducerGetTypeInformationResponse::dVeff**

**3.10.1.10 xsd\_\_double MSXE371x\_\_TransducerGetTypeInformationResponse::dSensibility**

**3.10.1.11 xsd\_\_double MSXE371x\_\_TransducerGetTypeInformationResponse::dRange**

## 3.11 MSXE371x\_\_unsignedLong8FixedArray Struct Reference

### Data Fields

- [xsd\\_\\_unsignedLong ulValue \[8\]](#)

*the meaning of this value is defined in the related header of the function who use this type*

### 3.11.1 Field Documentation

3.11.1.1 `xsd__unsignedLong MSXE371x__unsignedLong8FixedArray::ulValue[8]`

## 3.12 MSXE371x\_\_unsignedlong9ArrayResponse Struct Reference

### Data Fields

- struct [DefaultResponse sResponse](#)

*Default return values.*

- `xsd__unsignedLong ulValue [9]`

*the meaning of this value is defined in the related header of the function who use this type*

### 3.12.1 Field Documentation

3.12.1.1 `struct DefaultResponse MSXE371x__unsignedlong9ArrayResponse::sResponse`

3.12.1.2 `xsd__unsignedLong MSXE371x__unsignedlong9ArrayResponse::ulValue[9]`

## 3.13 MSXE371x\_\_unsignedlongDefaultResponse 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*

### 3.13.1 Field Documentation

3.13.1.1 `struct DefaultResponse MSXE371x__unsignedlongDefaultResponse::sResponse`

3.13.1.2 `xsd__unsignedLong MSXE371x__unsignedlongDefaultResponse::ulValue`

## 3.14 MSXE371x\_\_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 MSXE371x\\_\\_unsignedlongResponse::sResponse](#)

3.14.1.2 [xsd\\_\\_unsignedLong MSXE371x\\_\\_unsignedlongResponse::ulValue](#)

## 3.15 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.*

### 3.15.1 Field Documentation

3.15.1.1 struct [DefaultResponse MXCommon\\_\\_ByteArrayResponse::sResponse](#)

3.15.1.2 struct [ByteArray MXCommon\\_\\_ByteArrayResponse::sArray](#)

## 3.16 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.*

### 3.16.1 Field Documentation

3.16.1.1 struct `DefaultResponse` `MXCommon__FileResponse::sResponse`

3.16.1.2 struct `ByteArray` `MXCommon__FileResponse::sArray`

3.16.1.3 `xsd__unsignedLong` `MXCommon__FileResponse::ulEOF`

## 3.17 `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.*

### 3.17.1 Field Documentation

3.17.1.1 struct `DefaultResponse` `MXCommon__GetAutoConfigurationFileResponse::sResponse`

3.17.1.2 struct `ByteArray` `MXCommon__GetAutoConfigurationFileResponse::bArray`

3.17.1.3 `xsd__unsignedLong` `MXCommon__GetAutoConfigurationFileResponse::ulEOF`

## 3.18 `MXCommon__GetEthernetLinksStatesResponse` Struct Reference

### Data Fields

- struct `DefaultResponse` `sResponse`

*Default return values.*

- struct `sGetEthernetLinksStatesPort` `sPort0`

- struct `sGetEthernetLinksStatesPort` `sPort1`

### 3.18.1 Field Documentation

3.18.1.1 struct DefaultResponse MXCommon\_\_GetEthernetLinksStatesResponse::sResponse

3.18.1.2 struct sGetEthernetLinksStatesPort MXCommon\_\_ -  
GetEthernetLinksStatesResponse::sPort0

3.18.1.3 struct sGetEthernetLinksStatesPort MXCommon\_\_ -  
GetEthernetLinksStatesResponse::sPort1

## 3.19 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.*

### 3.19.1 Field Documentation

3.19.1.1 struct DefaultResponse MXCommon\_\_ -  
GetHardwareTriggerFilterTimeResponse::sResponse

3.19.1.2 [xsd\\_\\_unsignedLong](#) MXCommon\_\_ -  
GetHardwareTriggerFilterTimeResponse::ulFilterTime

3.19.1.3 [xsd\\_\\_unsignedLong](#) MXCommon\_\_GetHardwareTriggerFilterTimeResponse::ulInfo01

3.19.1.4 [xsd\\_\\_unsignedLong](#) MXCommon\_\_GetHardwareTriggerFilterTimeResponse::ulInfo02

## 3.20 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.*

### 3.20.1 Field Documentation

3.20.1.1 struct [DefaultResponse](#) [MXCommon\\_\\_GetHardwareTriggerStateResponse::sResponse](#)

3.20.1.2 [xsd\\_\\_unsignedLong](#) [MXCommon\\_\\_GetHardwareTriggerStateResponse::ulState](#)

3.20.1.3 [xsd\\_\\_unsignedLong](#) [MXCommon\\_\\_GetHardwareTriggerStateResponse::ulInfo01](#)

3.20.1.4 [xsd\\_\\_unsignedLong](#) [MXCommon\\_\\_GetHardwareTriggerStateResponse::ulInfo02](#)

## 3.21 [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.*



### 3.21.1 Field Documentation

- 3.21.1.1 struct `DefaultResponse` `MXCommon__GetModuleTemperatureValueAndStatusResponse::sResponse`
- 3.21.1.2 `xsd__double` `MXCommon__GetModuleTemperatureValueAndStatusResponse::dTemperatureValue`
- 3.21.1.3 `xsd__unsignedLong` `MXCommon__GetModuleTemperatureValueAndStatusResponse::ulTemperatureStatus`
- 3.21.1.4 `xsd__unsignedLong` `MXCommon__GetModuleTemperatureValueAndStatusResponse::ulInfo`

## 3.22 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).*

### 3.22.1 Field Documentation

- 3.22.1.1 struct `DefaultResponse` `MXCommon__GetTimeResponse::sResponse`
- 3.22.1.2 `xsd__unsignedLong` `MXCommon__GetTimeResponse::ulLowTime`
- 3.22.1.3 `xsd__unsignedLong` `MXCommon__GetTimeResponse::ulHighTime`

## 3.23 MXCommon\_\_GetUpTimeResponse Struct Reference

### Data Fields

- struct `DefaultResponse` `sResponse`  
*Default return value.*
- `xsd__unsignedLong` `ulUpTime`  
*Reserved.*

### 3.23.1 Field Documentation

3.23.1.1 struct `DefaultResponse` `MXCommon__GetUpTimeResponse::sResponse`

3.23.1.2 `xsd__unsignedLong` `MXCommon__GetUpTimeResponse::ulUpTime`

## 3.24 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\(\)](#)).*

### 3.24.1 Field Documentation

3.24.1.1 `xsd__int` `MXCommon__Response::iReturnValue`

3.24.1.2 `xsd__int` `MXCommon__Response::syserrno`

## 3.25 MXCommon\_\_TestCustomerIDResponse Struct Reference

### Data Fields

- struct `DefaultResponse` `sResponse`

*Default return values.*

- struct `ByteArray` `bValueArray`

*non encrypted value*

- struct `ByteArray` `bCryptedValueArray`

*encrypted value*

### 3.25.1 Field Documentation

3.25.1.1 struct DefaultResponse MXCommon\_\_TestCustomerIDResponse::sResponse

3.25.1.2 struct ByteArray MXCommon\_\_TestCustomerIDResponse::bValueArray

3.25.1.3 struct ByteArray MXCommon\_\_TestCustomerIDResponse::bCryptedValueArray

## 3.26 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.*

### 3.26.1 Field Documentation

3.26.1.1 struct DefaultResponse MXCommon\_\_unsignedLongResponse::sResponse

3.26.1.2 [xsd\\_\\_unsignedLong](#) MXCommon\_\_unsignedLongResponse::ulValue

## 3.27 sGetEthernetLinksStatesPort Struct Reference

### Data Fields

- [xsd\\_\\_unsignedLong](#) ulState
- [xsd\\_\\_unsignedLong](#) ulSpeed
- [xsd\\_\\_unsignedLong](#) ulDuplex
- [xsd\\_\\_unsignedLong](#) ulInfo1
- [xsd\\_\\_unsignedLong](#) ulInfo2

### 3.27.1 Field Documentation

3.27.1.1 `xsd__unsignedLong sGetEthernetLinksStatesPort::ulState`

3.27.1.2 `xsd__unsignedLong sGetEthernetLinksStatesPort::ulSpeed`

3.27.1.3 `xsd__unsignedLong sGetEthernetLinksStatesPort::ulDuplex`

3.27.1.4 `xsd__unsignedLong sGetEthernetLinksStatesPort::ulInfo1`

3.27.1.5 `xsd__unsignedLong sGetEthernetLinksStatesPort::ulInfo2`

## 3.28 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*

### 3.28.1 Field Documentation

3.28.1.1 `xsd__unsignedLong* UnsignedLongArray::__ptr`

3.28.1.2 `int UnsignedLongArray::__size`

3.28.1.3 `int UnsignedLongArray::__offset`

## 3.29 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*

### 3.29.1 Field Documentation

3.29.1.1 `xsd__unsignedShort* UnsignedShortArray::__ptr`

3.29.1.2 `int UnsignedShortArray::__size`

3.29.1.3 `int UnsignedShortArray::__offset`

## 3.30 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*

### 3.30.1 Field Documentation

3.30.1.1 `unsigned char* xsd__base64Binary::__ptr`

3.30.1.2 `int xsd__base64Binary::__size`



# Chapter 4

## File Documentation

### 4.1 MSXE371x\_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 [MSXE371x\\_\\_Response](#)
- struct [MSXE371x\\_\\_unsignedlongResponse](#)
- struct [MSXE371x\\_\\_unsignedLong8FixedArray](#)
- struct [MSXE371x\\_\\_unsignedlong9ArrayResponse](#)
- struct [MSXE371x\\_\\_ByteArrayResponse](#)
- struct [MSXE371x\\_\\_unsignedlongDefaultResponse](#)
- struct [MSXE371x\\_\\_TransducerGetTypeInformationResponse](#)
- struct [MSXE371x\\_\\_TransducerGetAllValuesResponse](#)
- struct [MSXE371x\\_\\_IncCounterRead32BitValueResponse](#)
- struct [MSXE371x\\_\\_ExternalTemperatureReadResponse](#)
- struct [MSXE371x\\_\\_CalibrationGetCurrentStatusResponse](#)
- struct [MSXE371x\\_\\_DataBaseGetTransducerInformationResponse](#)

## 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 [MSXE371x\\_\\_TransducerGetNbrOfType](#) (void \*\_ , struct [MSXE371x\\_\\_unsignedlongResponse](#) \*Response)

*Returns the number of transducer types currently defined in the database.*

- int [MSXE371x\\_\\_TransducerGetTypeInfoInformation](#) (xsd\_\_unsignedLong ulIndex, struct [MSXE371x\\_\\_TransducerGetTypeInfoInformationResponse](#) \*Response)

*Returns the information stored in the database about the type.*

- int [MSXE371x\\_\\_TransducerInitAndStartAutoRefresh](#) (xsd\_\_unsignedLong ulTransducerSelection, xsd\_\_unsignedLong ulChannelMask, xsd\_\_unsignedLong ulAverageMode, xsd\_\_unsignedLong ulAverageValue, xsd\_\_unsignedLong ulTriggerMask, xsd\_\_unsignedLong ulTriggerMode, xsd\_\_unsignedLong ulHardwareTriggerEdge, xsd\_\_unsignedLong ulHardwareTriggerCount, xsd\_\_unsignedLong ulByTriggerNbrOfSeqToAcquire, xsd\_\_unsignedLong ulDataFormat, xsd\_\_unsignedLong ulOption1, xsd\_\_unsignedLong ulOption2, xsd\_\_unsignedLong ulOption3, xsd\_\_unsignedLong ulOption4, struct [MSXE371x\\_\\_Response](#) \*Response)

*Initialise and start the transducer auto refresh acquisition mode.*

- int [MSXE371x\\_\\_TransducerGetAutoRefreshValues](#) (void \*\_ , struct [MSXE371x\\_\\_unsignedlong9ArrayResponse](#) \*Response)

*This function get the auto refresh counter value an the channels values.*

- int [MSXE371x\\_\\_TransducerGetAllAutoRefreshValues](#) (void \*\_ , struct [MSXE371x\\_\\_TransducerGetAllValuesResponse](#) \*Response)

*This function return the actualy auto refresh values (Transducer auto refresh counter, transducer values, time stampe, ...).*

- int [MSXE371x\\_\\_TransducerStopAndReleaseAutoRefresh](#) (void \*\_ , struct [MSXE371x\\_\\_Response](#) \*Response)

*Stop and release the transducer auto refresh acquisition mode.*

- int [MSXE371x\\_\\_TransducerInitAndStartSequence](#) (xsd\_\_unsignedLong ulTransducerSelection, xsd\_\_unsignedLong ulNbrOfChannel, struct [MSXE371x\\_\\_unsignedLong8FixedArray](#) \*pulChannelList, xsd\_\_unsignedLong ulNbrOfSequence, xsd\_\_unsignedLong ulNbrMaxSequenceToTransfer, xsd\_\_unsignedLong ulDelayMode, xsd\_\_unsignedLong ulDelayTimeUnit, xsd\_\_unsignedLong ulDelayValue, xsd\_\_unsignedLong ulTriggerMask, xsd\_\_unsignedLong ulTriggerMode, xsd\_\_unsignedLong ulHardwareTriggerEdge, xsd\_\_unsignedLong ulHardwareTriggerCount, xsd\_\_unsignedLong ulByTriggerNbrOfSeqToAcquire, xsd\_\_unsignedLong ulDataFormat, xsd\_\_unsignedLong ulOption1, xsd\_\_unsignedLong ulOption2, xsd\_\_unsignedLong ulOption3, xsd\_\_unsignedLong ulOption4, struct [MSXE371x\\_\\_Response](#) \*Response)

*Initialise and start the transducer sequence acquisition mode.*

- int [MSXE371x\\_\\_TransducerStopAndReleaseSequence](#) (void \*\_ , struct [MSXE371x\\_\\_Response](#) \*Response)

*Stop and release the transducer sequence acquisition mode without sending an end packet.*

- int [MSXE371x\\_\\_TransducerStopAndReleaseSequence2](#) (unsigned long ulEmptyFIFO, unsigned long ulOption1, struct [MSXE371x\\_\\_Response](#) \*Response)

*Stop and release the transducer sequence acquisition mode and send an end packet with the same sequence number.*

- int [MSXE371x\\_\\_TransducerInit](#) (xsd\_\_unsignedLong ulIndex, xsd\_\_unsignedLong ulFrequency, struct [MSXE371x\\_\\_Response](#) \*Response)  
*Init the transducer and set it to last selected.*
- int [MSXE371x\\_\\_TransducerInitPrimaryConnectionTest](#) (void \*\_, struct [MSXE371x\\_\\_Response](#) \*Response)  
*Initialise the primary connection test.*
- int [MSXE371x\\_\\_TransducerTestPrimaryConnection](#) (void \*\_, struct [MSXE371x\\_\\_unsignedlongDefaultResponse](#) \*Response)  
*Test primary connection.*
- int [MSXE371x\\_\\_TransducerTestPrimaryShortCircuit](#) (void \*\_, struct [MSXE371x\\_\\_unsignedlongDefaultResponse](#) \*Response)  
*Test primary short circuit status.*
- int [MSXE371x\\_\\_TransducerRearmPrimary](#) (void \*\_, struct [MSXE371x\\_\\_unsignedlongDefaultResponse](#) \*Response)  
*Rearm primary.*
- int [MSXE371x\\_\\_TransducerTestSecondaryConnection](#) (xsd\_\_unsignedLong ulChannel, struct [MSXE371x\\_\\_unsignedlongDefaultResponse](#) \*Response)  
*Test the secondary connection.*
- int [MSXE371x\\_\\_TransducerTestSecondaryShortCircuit](#) (xsd\_\_unsignedLong ulChannel, struct [MSXE371x\\_\\_unsignedlongDefaultResponse](#) \*Response)  
*Test the secondary short circuit.*
- int [MSXE371x\\_\\_IncCounterInit](#) (xsd\_\_unsignedLong ulCounterMode, xsd\_\_unsignedLong ulCounterOption, xsd\_\_unsignedLong ulOption01, xsd\_\_unsignedLong ulOption02, xsd\_\_unsignedLong ulOption03, xsd\_\_unsignedLong ulOption04, struct [MSXE371x\\_\\_Response](#) \*Response)  
*Initialise the counter.*
- int [MSXE371x\\_\\_IncCounterRelease](#) (void \*\_, struct [MSXE371x\\_\\_Response](#) \*Response)  
*Release the counter.*
- int [MSXE371x\\_\\_IncCounterRead32BitValue](#) (void \*\_, struct [MSXE371x\\_\\_IncCounterRead32BitValueResponse](#) \*Response)  
*Read the 32 bits counter value.*
- int [MSXE371x\\_\\_IncCounterWrite32BitValue](#) (xsd\_\_unsignedLong ulCounterValue, struct [MSXE371x\\_\\_Response](#) \*Response)  
*write a 32 bits counter value*
- int [MSXE371x\\_\\_IncCounterClear](#) (void \*\_, struct [MSXE371x\\_\\_Response](#) \*Response)  
*Clear the 32 bits counter.*
- int [MSXE371x\\_\\_IncCounterInitAndEnableCompareLogic](#) (xsd\_\_unsignedLong ulValue, xsd\_\_unsignedLong ulMode, xsd\_\_unsignedLong ulSynchroTrigger, xsd\_\_unsignedLong ulOption01, xsd\_\_unsignedLong ulOption02, struct [MSXE371x\\_\\_Response](#) \*Response)

*Init and enable a counter compare value.*

- int [MSXE371x\\_\\_IncCounterDisableAndReleaseCompareLogic](#) (void \_\_\_, struct [MSXE371x\\_\\_Response](#) \*Response)

*Disable and Release a counter compare value.*

- int [MSXE371x\\_\\_IncCounterInitAndEnableIndex](#) (xsd\_\_unsignedLong ulReferenceAction, [xsd\\_\\_unsignedLong](#) ulIndexOperation, [xsd\\_\\_unsignedLong](#) ulAutoMode, [xsd\\_\\_unsignedLong](#) ulOption01, struct [MSXE371x\\_\\_Response](#) \*Response)

*Init and enable the counter index.*

- int [MSXE371x\\_\\_IncCounterDisableAndReleaseIndex](#) (void \_\_\_, struct [MSXE371x\\_\\_Response](#) \*Response)

*Disable and Release the counter index.*

- int [MSXE371x\\_\\_ExternalTemperatureInit](#) ([xsd\\_\\_unsignedLong](#) ulConnectedTempSensor, [xsd\\_\\_unsignedLong](#) ulConvertMode, [xsd\\_\\_unsignedLong](#) ulGainSelection, [xsd\\_\\_unsignedLong](#) ulFrequencySelection, [xsd\\_\\_unsignedLong](#) ulPowerSaveMode, [xsd\\_\\_unsignedLong](#) ulOption01, [xsd\\_\\_unsignedLong](#) ulOption02, [xsd\\_\\_unsignedLong](#) ulOption03, [xsd\\_\\_unsignedLong](#) ulOption04, struct [MSXE371x\\_\\_Response](#) \*Response)

*Initialise the external temperature.*

- int [MSXE371x\\_\\_ExternalTemperatureRelease](#) (void \_\_\_, struct [MSXE371x\\_\\_Response](#) \*Response)

*Release the external temperature.*

- int [MSXE371x\\_\\_ExternalTemperatureRead](#) (void \_\_\_, struct [MSXE371x\\_\\_ExternalTemperatureReadResponse](#) \*Response)

*Read the external temperature value.*

- int [MSXE371x\\_\\_ExternalTemperatureGainCalibration](#) (unsigned long ulGainSelection, double dRefValue, struct [MSXE371x\\_\\_Response](#) \*Response)

*Temperature gain calibration.*

- int [MSXE371x\\_\\_CalibrationStart](#) ([xsd\\_\\_unsignedLong](#) ulTransducerIndex, [xsd\\_\\_double](#) dPosition, [xsd\\_\\_unsignedLong](#) ulCalibrationMode, [xsd\\_\\_unsignedLong](#) ulInvertValue, [xsd\\_\\_unsignedLong](#) ulChannel, struct [MSXE371x\\_\\_Response](#) \*Response)

*This function start the calibration thread.*

- int [MSXE371x\\_\\_CalibrationGetCurrentStatus](#) (void \_\_\_, struct [MSXE371x\\_\\_CalibrationGetCurrentStatusResponse](#) \*Response)

*This function return the current calibration status.*

- int [MSXE371x\\_\\_CalibrationNextStep](#) (void \_\_\_, struct [MSXE371x\\_\\_Response](#) \*Response)

*This function start the next calibration step.*

- int [MSXE371x\\_\\_CalibrationBreak](#) (void \_\_\_, struct [MSXE371x\\_\\_Response](#) \*Response)

*This function break the current calibration.*

- int [MSXE371x\\_\\_DataBaseGetNumberOfTransducers](#) (void \_\_\_, struct [MSXE371x\\_\\_unsignedlongResponse](#) \*Response)

*Returns the number of transducer types currently defined in the database.*

- int MSXE371x\_\_DataBaseGetTransducerType (xsd\_\_unsignedLong ulTransducerIndex, struct MSXE371x\_\_unsignedlongResponse \*Response)

*Returns the transducer identifier of the selected transducer.*

- int MSXE371x\_\_DataBaseGetTransducerInformation (xsd\_\_unsignedLong ulTransducerIndex, struct MSXE371x\_\_DataBaseGetTransducerInformationResponse \*Response)

*Returns the information stored in the database about the type.*

- int MSXE371x\_\_DataBaseAddTransducer (xsd\_\_unsignedLong ulTransducerIndex, xsd\_\_string cName, xsd\_\_unsignedLong ulType, xsd\_\_unsignedLong ulFrequency, xsd\_\_unsignedLong ulImpedance, xsd\_\_double dVeff, xsd\_\_double dSensitivity, xsd\_\_double dRange, struct MSXE371x\_\_Response \*Response)

*Adds a new transducer type definition into the database of the module.*

- int MSXE371x\_\_DataBaseDelTransducer (xsd\_\_unsignedLong ulTransducerIndex, struct MSXE371x\_\_Response \*Response)

*Deletes the selected transducer from the transducer database.*

- int MSXE371x\_\_DataBaseSaveTransducers (void \*\_ , struct MSXE371x\_\_ByteArrayResponse \*Response)

*Commits the current changes in the transducer database, including the calibration values.*

### 4.1.1 Typedef Documentation

- 4.1.1.1 typedef char\* xsd\_\_string
- 4.1.1.2 typedef char xsd\_\_char
- 4.1.1.3 typedef float xsd\_\_float
- 4.1.1.4 typedef double xsd\_\_double
- 4.1.1.5 typedef int xsd\_\_int
- 4.1.1.6 typedef long xsd\_\_long
- 4.1.1.7 typedef unsigned char xsd\_\_unsignedByte
- 4.1.1.8 typedef unsigned int xsd\_\_unsignedInt
- 4.1.1.9 typedef unsigned short int xsd\_\_unsignedShort
- 4.1.1.10 typedef unsigned long xsd\_\_unsignedLong

### 4.1.2 Function Documentation

- 4.1.2.1 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

- 4.1.2.2 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

#### 4.1.2.3 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

#### 4.1.2.4 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

#### 4.1.2.5 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

#### 4.1.2.6 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

#### 4.1.2.7 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

#### 4.1.2.8 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.

#### 4.1.2.9 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

**4.1.2.10** `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\\_\\_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

**4.1.2.11** `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\\_\\_Sterror\(\)](#).

#### Return values

*SOAP\_OK* SOAP call success

*otherwise* SOAP protocol error

#### 4.1.2.12 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

#### 4.1.2.13 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

#### 4.1.2.14 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\\_\\_Strerror\(\)](#).

**Return values**

- SOAP\_OK* SOAP call success
- otherwise* SOAP protocol error

#### 4.1.2.15 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\\_\\_Sterror\(\)](#).

#### Return values

*SOAP\_OK* SOAP call success  
*otherwise* SOAP protocol error

#### 4.1.2.16 `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\\_\\_Sterror\(\)](#).
  - `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

#### 4.1.2.17 `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\\_\\_Sterror\(\)](#).

#### Return values

*SOAP\_OK* SOAP call success  
*otherwise* SOAP protocol error



#### 4.1.2.18 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.

#### 4.1.2.19 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.

#### 4.1.2.20 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\\_\\_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

#### 4.1.2.21 `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\\_\\_Sterror\(\)](#).
  - `ulUpTime` : Number of seconds since the last boot of the system.

#### Return values

*SOAP\_OK* SOAP call success  
*otherwise* SOAP protocol error

#### 4.1.2.22 `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\\_\\_Sterror\(\)](#).
  - `bArray` : Array of Bytes of the file
  - `ulEOF` : End of file flag

#### Return values

*SOAP\_OK* SOAP call success  
*otherwise* SOAP protocol error

**4.1.2.23** `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\\_\\_Strerror\(\)](#).

#### Return values

*SOAP\_OK* SOAP call success  
*otherwise* SOAP protocol error

**4.1.2.24** `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\\_\\_Strerror\(\)](#).
  - sArray : message returned by the auto configuration start

#### Return values

*SOAP\_OK* SOAP call success  
*otherwise* SOAP protocol error

**4.1.2.25** `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

#### 4.1.2.26 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\\_\\_Strerror\(\)](#). May be ENOSYS : Function not implemented.

#### Return values

***SOAP\_OK*** SOAP call success  
***otherwise*** SOAP protocol error

#### 4.1.2.27 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\\_\\_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;
}
*
```

#### 4.1.2.28 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\\_\\_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 `uEOF` must be set to 1. The new configuration is then applied.

**4.1.2.29** `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\\_\\_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.

**4.1.2.30** `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

#### 4.1.2.31 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

#### 4.1.2.32 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

#### 4.1.2.33 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\\_\\_Strerror\(\)](#).
  - sArray : The symbolic name associated with the ID.

##### Return values

*SOAP\_OK* SOAP call success  
*otherwise* SOAP protocol error

#### 4.1.2.34 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\\_\\_Strerror\(\)](#).
  - sArray The symbolic name associated with the state numerical ID.

##### Return values

*SOAP\_OK* SOAP call success  
*otherwise* SOAP protocol error

#### 4.1.2.35 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



**4.1.2.36** `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

**4.1.2.37** `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

#### 4.1.2.38 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

#### 4.1.2.39 int MSXE371x\_\_TransducerGetNbrOfType ( void \* \_\_, struct MSXE371x\_\_unsignedlongResponse \* *Response* )

##### Parameters

- [in] ***\_\_*** : no input parameter
- [out] ***Response*** :
- sResponse.iReturnValue*** : Error value
- 0: success
  - -1: error
  - -100: kernel function error
- sResponse.syserrno*** : system-error code (the value of the libc "errno" code) Its value is significant only when the iReturnValue returned an error (-1 or <= -100)  
Give this value to the MXCommon\_Sterror to get the string describing the error number.
- ulValue*** : number of transducers type.

##### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 4.1.2.40 int MSXE371x\_\_TransducerGetTypeInfoInformation ( xsd\_\_unsignedLong *ulIndex*, struct MSXE371x\_\_TransducerGetTypeInfoInformationResponse \* *Response* )

##### Parameters

- [in] ***ulIndex*** : index of the transducer
- [out] ***Response*** :
- sResponse.iReturnValue*** : Error value
- 0: success
  - -1: error

- -2: index is invalid
  - -100: failure of kernel function "GetNumberOfTransducerTypes"
  - -101: failure of kernel function "GetTransducerType"
  - -101: failure of kernel function "GetTransducerInformation"
- sResponse.syserrno* : system-error code (the value of the libc "errno" code)

*ulTransducerSelectionIndex* : Selection value. Value to write for the transducer type selection

*pcName* : Name of the transducer type

*ulCalibrationStatus* : Calibration status

- 0 : Transducer type is not calibrated
- 1 : Transducer type is calibrated

*ulCalibratedChannels* : Bitmask of currently calibrated channels (D0 => channel 1, D1 => channel 1, ...)

*ulType* : Type (0: HB 1: LVDT 2:Knaebel 3:HB-Mahr 4:LVDT-Mahr)

*ulFrequency* : Frequency (Hz)

*ulImpedance* : Impedance (Ohm)

*dVeff* : Nominal voltage (Vrms)

*dSensibility* : Sensibility (mv/V/mm)

*dRange* : Range (mm)

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**4.1.2.41** `int MSXE371x__TransducerInitAndStartAutoRefresh ( xsd__unsignedLong ulTransducerSelection, xsd__unsignedLong ulChannelMask, xsd__unsignedLong ulAverageMode, xsd__unsignedLong ulAverageValue, xsd__unsignedLong ulTriggerMask, xsd__unsignedLong ulTriggerMode, xsd__unsignedLong ulHardwareTriggerEdge, xsd__unsignedLong ulHardwareTriggerCount, xsd__unsignedLong ulByTriggerNbrOfSeqToAcquire, xsd__unsignedLong ulDataFormat, xsd__unsignedLong ulOption1, xsd__unsignedLong ulOption2, xsd__unsignedLong ulOption3, xsd__unsignedLong ulOption4, struct MSXE371x__Response * Response )`

#### Parameters

- [in] *ulTransducerSelection* Transducer type selection
- [in] *ulChannelMask* Mask of the channel to acquire by the auto refresh (1 bit = 1 Channel, D0 channel 0 and D7 channel 7)
- [in] *ulAverageMode* Set the average mode :
- 0 : not used
  - 1 : average per channel
- [in] *ulAverageValue* Set the average value (only used, when average is used) :
- 0 : not used
  - max value : 255
- [in] *ulTriggerMask* Define the source of the trigger
- 0 : trigger disabled
  - 1 : Enable Hardware Digital Input Trigger
  - 2 : Enable Synchro Trigger

- 4 : Enable Incremental Counter Compare Interrupt
- [in] ***ulTriggerMode*** Define the trigger mode
  - 1 : One shot trigger
  - 2 : Sequence trigger
- [in] ***ulHardwareTriggerEdge*** 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] ***ulHardwareTriggerCount*** Define the number of trigger events before the action occur
  - 0 or 1 : all trigger event start the action
  - max value : 65535
- [in] ***ulByTriggerNbrOfSeqToAcquire*** Define the number of sequence to acquire by each trigger event (1..65535)
- [in] ***ulDataFormat*** D0 : Time stamp information
  - 0: no time stamp information
  - 1: time stamp information
 D1 : Data format
  - 0: Digital value
  - 1: Analog value (in mm)
 D2 : invert value
  - 0 : don't invert the channel value
  - 1 : invert the channel value (-2 mm -> + 2mm)
 D3 : Temperature value
  - 0 : don't acquire the temperature value
  - 1 : acquire the temperature value
 D4 : Incremental counter value
  - 0 : don't acquire the incremental counter value
  - 1 : acquire the incremental counter value
 D5 : Diff. mode
  - 0 : Diff mode disabled
  - 1 : Diff mode enabled : Channel X value = Channel (X) value + Channel (X + 4) value
- [in] ***ulOption1*** : Reserved
- [in] ***ulOption2*** : Reserved
- [in] ***ulOption3*** : Reserved
- [in] ***ulOption4*** : Reserved
- [out] ***Response*** :
  - iReturnValue*** : Error value
    - 0: success
    - -1: means an system error occurred
    - -2: Transducer selection error
    - -3: The channel mask cannot be null
    - -4: Channel Mask error

- -5: not available average mode
- -6: not available average value
- -7: Trigger source : 2 or more different source cannot be simultaneously be activated
- -8: Trigger mode selection error
- -9: Hardware trigger : front definition error
- -10: Hardware trigger count value not available
- -11: Nbr of sequence to acquire by trigger mode not available
- -12: Data format not available
- -13: Incremental counter not initialised
- -14: Incremental counter compare logic not initialised
- -15: Temperature channel not initialised
- -100: Auto refresh initialisation and start kernel function error *syserrno* : system-error code (the value of the libc "errno" code)

#### Returns

0: SOAP\_OK  
<> 0: See SOAP error

#### 4.1.2.42 int MSXE371x\_\_TransducerGetAutoRefreshValues ( void \* \_\_, struct MSXE371x\_\_unsignedlong9ArrayResponse \* Response )

##### Parameters

[in] \_\_ : no input parameter  
[out] *Response* :  
    *sResponse.iReturnValue* :

- 0 : success
- -100 : Get auto refresh values kernel function error

*sResponse.syserrno* : system-error code (the value of the libc "errno" code)  
    *ulValue* : Array that contain the counter and channels values

- ulValues [0] : Auto refresh counter value
- ulValues [1] : Channel 0 value
- ...
- ulValues [16] : Channel 15 value

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 4.1.2.43 int MSXE371x\_\_TransducerGetAllAutoRefreshValues ( void \* \_\_, struct MSXE371x\_\_TransducerGetAllValuesResponse \* Response )

##### Parameters

[in] \_\_ : no input parameter  
[out] *Response* :  
    *sResponse.iReturnValue* :

- 0 : success
  - -100 : Get auto refresh all values kernel function error
- sResponse.syserrno* : system-error code (the value of the libc "errno" code)

*ulAutoRefreshCounter*: Auto refresh counter value *ulTransducersValue*: Transducers values (Array [8]) *ulTimeStampLow*: Time stampe low (micro s) *ulTimeStampHigh*: Time stampe high (s) *ulIncCounter* : Incremental counter value if initialised. *ulExternalTemperature*: External temperature if initialised.

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 4.1.2.44 int MSXE371x\_\_TransducerStopAndReleaseAutoRefresh ( void \* \_, struct MSXE371x\_\_Response \* Response )

##### Parameters

[in] \_ : no input parameter

[out] *Response* :

*iReturnValue* :

- 0 : success
- -1: means an system error occurred
- -100: "StopAutoRefresh" kernel function error

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

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 4.1.2.45 int MSXE371x\_\_TransducerInitAndStartSequence ( xsd\_\_unsignedLong ulTransducerSelection, xsd\_\_unsignedLong ulNbrOfChannel, struct MSXE371x\_\_unsignedLong8FixedArray \* pulChannelList, xsd\_\_unsignedLong ulNbrOfSequence, xsd\_\_unsignedLong ulNbrMaxSequenceToTransfer, xsd\_\_unsignedLong ulDelayMode, xsd\_\_unsignedLong ulDelayTimeUnit, xsd\_\_unsignedLong ulDelayValue, xsd\_\_unsignedLong ulTriggerMask, xsd\_\_unsignedLong ulTriggerMode, xsd\_\_unsignedLong ulHardwareTriggerEdge, xsd\_\_unsignedLong ulHardwareTriggerCount, xsd\_\_unsignedLong ulByTriggerNbrOfSeqToAcquire, xsd\_\_unsignedLong ulDataFormat, xsd\_\_unsignedLong ulOption1, xsd\_\_unsignedLong ulOption2, xsd\_\_unsignedLong ulOption3, xsd\_\_unsignedLong ulOption4, struct MSXE371x\_\_Response \* Response )

##### Parameters

[in] *ulTransducerSelection* : Transducer type selection

[in] *ulNbrOfChannel* : nbr of channel in the sequence

[in] *pulChannelList* : list of the channel who compose the sequence

[in] *ulNbrOfSequence* : Number of sequence to acquire :

- 0 : continuous mode
  - <> 0 : number of sequence
- [in] ***ulNbrMaxSequenceToTransfer*** : Max nbr of sequence to acquire before a data transfer
- [in] ***ulDelayMode*** : Delay Mode :
- ADDIDATA\_DELAY\_NOT\_USED 0
  - ADDIDATA\_DELAY\_MODE1\_USED 1
- [in] ***ulDelayTimeUnit*** : Selection of the unit of ulDelayValue
- 0: ms
  - 1: s
- [in] ***ulDelayValue*** : Delay Value
- [in] ***ulTriggerMask*** Define the source of the trigger
- 0 : trigger disabled
  - 1 : Enable Hardware Digital Input Trigger
  - 2 : Enable Synchro Trigger
  - 4 : Enable Incremental Counter Compare Interrupt
- [in] ***ulTriggerMode*** Define the trigger mode
- 1 : One shot trigger
  - 2 : Sequence trigger
- [in] ***ulHardwareTriggerEdge*** 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] ***ulHardwareTriggerCount*** Define the number of trigger events before the action occur
- 0 or 1 : all trigger event start the action
  - max value : 65535
- [in] ***ulByTriggerNbrOfSeqToAcquire*** Define the number of sequence to acquire by each trigger event (1..65535)
- [in] ***ulDataFormat*** D0 : Time stamp information
- 0: no time stamp information
  - 1: time stamp information
- D1 : Sequence counter information
- 0 : No sequence counter information
  - 1 : Sequence counter information
- D2 : Data format
- 0: Digital value
  - 1: Analog value (in mm)
- D3 : invert value
- 0 : don't invert the channel value
  - 1 : invert the channel value (-2 mm -> + 2mm)
- D4 : Temperature value
- 0 : don't acquire the temperature value
  - 1 : acquire the temperature value

D5 : Incremental counter value

- 0 : don't acquire the incremental counter value
- 1 : acquire the incremental counter value

D6 : Diff. mode

- 0 : Diff mode disabled
- 1 : Diff mode enabled : Channel X value = Channel (X) value + Channel (X + 4) value

D7 : receive the hardware trigger information

- 0 : no hardware trigger information
- 1 : hardware trigger information

D8 : receive the index input information

- 0 : no index input information
- 1 : index input information

[in] ***ulOption1*** : Reserved

[in] ***ulOption2*** : Reserved

[in] ***ulOption3*** : Reserved

[in] ***ulOption4*** : Reserved

[out] ***Response*** :

***iReturnValue*** :

- 0: success
- -1: means an system error occurred
- -2: Transducer selection error
- -3: Channel sequence size error
- -4: Channel array selection error
- -5: Delay mode selection error
- -6: Delay time base selection error
- -7: Delay wait time selection error
- -8: Max sequence to transfer > number of sequence
- -9: Trigger source : 2 or more different source cannot be simultaneously be activated
- -10: Trigger mode selection error
- -11: Hardware trigger : front definition error
- -12: Hardware trigger count value not available
- -13: Nbr of sequence to acquire by trigger mode not available
- -14: Data format not available
- -15: Incremental counter not initialised
- -16: Incremental counter compare logic not initialised
- -17: Temperature channel not initialised
- -100: Init and start sequence acquisition kernel function error ***syserrno*** : system-error code (the value of the libc "errno" code)

## Returns

- 0: SOAP\_OK
- <> 0: See SOAP error



#### 4.1.2.46 int MSXE371x\_\_TransducerStopAndReleaseSequence ( void \* \_\_, struct MSXE371x\_\_Response \* *Response* )

##### Parameters

- [in] *\_\_* : no input parameter
- [out] *Response* :
- iReturnValue* :
- 0 : success
  - -1 : means an system error occurred
  - -100 : StartStopSequence kernel function error
- syserrno* : system-error code (the value of the libc "errno" code)

##### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 4.1.2.47 int MSXE371x\_\_TransducerStopAndReleaseSequence2 ( unsigned long *ulEmptyFIFO*, unsigned long *ulOption1*, struct MSXE371x\_\_Response \* *Response* )

##### Parameters

- [in] *ulEmptyFIFO* :
- 0 : Remove all available sequence datas
  - 1 : Send any available sequence datas via the data server
- [in] *ulOption1* : Reserved
- [out] *Response* :
- iReturnValue* :
- 0 : success
  - -1 : means an system error occurred
  - -100 : StartStopSequence kernel function error
- syserrno* : system-error code (the value of the libc "errno" code)

##### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 4.1.2.48 int MSXE371x\_\_TransducerInit ( xsd\_\_unsignedLong *ulIndex*, xsd\_\_unsignedLong *ulFrequency*, struct MSXE371x\_\_Response \* *Response* )

This function must be called before making a diagnostic test.

##### Parameters

- [in] *ulIndex* : Transducer index
- [in] *ulFrequency* : Transducer frequency

[out] **Response** :

**sResponse.iReturnValue** : Error value

- 0: success
- -100: failure of kernel function i\_IOKernel\_TransducerInit
- -101: error driver not in correct state
- -102: transducer index is invalid
- -103: kernel copy error

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

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 4.1.2.49 int MSXE371x\_\_TransducerInitPrimaryConnectionTest ( void \* \_\_, struct MSXE371x\_\_Response \* *Response* )

##### Parameters

[in] **\_** : no input parameter

[out] **Response** :

**iReturnValue** : Error code

- 0: success
- -1: means an system error occurred
- -100: Primary connection test initialisation kernel function error

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

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 4.1.2.50 int MSXE371x\_\_TransducerTestPrimaryConnection ( void \* \_\_, struct MSXE371x\_\_unsignedlongDefaultResponse \* *Response* )

##### Parameters

[in] **\_** : no input parameter

[out] **Response** :

**sResponse.iReturnValue** : Error code

- 0: success
- -1: means an system error occurred
- -100: Primary connection test kernel function error

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

**ulValue** : Connection status:

- 0 : connection error
- 1 : connection ok

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 4.1.2.51 int MSXE371x\_\_TransducerTestPrimaryShortCircuit ( void \* \_\_, struct MSXE371x\_\_unsignedlongDefaultResponse \* Response )

##### Parameters

[in] \_\_ : no input parameter

[out] *Response* :

*sResponse.iReturnValue* : Error code

- 0: success
  - -1: means an system error occurred
  - -100: Primary short circuit test kernel function error *sResponse.syserrno* : system-error code (the value of the libc "errno" code)
- ulValue* : short circuit status:
- 0 : short circuit
  - 1 : no short circuit

##### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 4.1.2.52 int MSXE371x\_\_TransducerRearmPrimary ( void \* \_\_, struct MSXE371x\_\_unsignedlongDefaultResponse \* Response )

##### Parameters

[in] \_\_ : no input parameter

[out] *Response* :

*sResponse.iReturnValue* : Error code

- 0: success
  - -1: means an system error occurred
  - -100: Primary short circuit rearm kernel function error *sResponse.syserrno* : system-error code (the value of the libc "errno" code)
- ulValue* : Rearm status:
- 0 : Rearm not ok
  - 1 : Rearm ok

##### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 4.1.2.53 int MSXE371x\_\_TransducerTestSecondaryConnection ( xsd\_\_unsignedLong ulChannel, struct MSXE371x\_\_unsignedlongDefaultResponse \* Response )

##### Parameters

[in] *ulChannel*,: channel selection

[out] *Response* :

*sResponse.iReturnValue* : Error code

- 0: success
- -1: means an system error occurred
- -100: Secondary connection test kernel function error *sResponse.syserrno* : system-error code (the value of the libc "errno" code)  
*ulValue* : Connection status:
  - 0 : connection error
  - 1 : connection ok

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**4.1.2.54** `int MSXE371x__TransducerTestSecondaryShortCircuit ( xsd__unsignedLong ulChannel, struct MSXE371x__unsignedlongDefaultResponse * Response )`

#### Parameters

[in] *ulChannel*: channel selection

[out] *Response* :

*sResponse.iReturnValue* : Error code

- 0: success
- -1: means an system error occurred
- -100: Secondary short circuit test kernel function error *sResponse.syserrno* : system-error code (the value of the libc "errno" code)  
*ulValue* : short circuit status:
  - 0 : short circuit
  - 1 : no short circuit

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**4.1.2.55** `int MSXE371x__IncCounterInit ( xsd__unsignedLong ulCounterMode, xsd__unsignedLong ulCounterOption, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct MSXE371x__Response * Response )`

#### Parameters

[in] *ulCounterMode* : Set the counter mode : either

- MSXE371x\_COUNTER\_QUADRUPLE\_MODE (0x4)
- MSXE371x\_COUNTER\_DOUBLE\_MODE (0x2)
- MSXE371x\_COUNTER\_SIMPLE\_MODE (0x1)
- MSXE371x\_COUNTER\_DIRECT\_MODE (0x0)

[in] *ulCounterOption* : Set the counter option

if in QUADRUPLE/DOUBLE/SIMPLE mode : either

- MSXE371x\_COUNTER\_HYSTERESIS\_ON (0x1)

- MSXE371x\_COUNTER\_HYSTERESIS\_OFF (0x0)

if in DIRECT mode :

- MSXE371x\_COUNTER\_INCREMENT (0x0)
- MSXE371x\_COUNTER\_DECREMENT (0x1)

[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: success
- -1: means an system error occurred
- -2: Counter mode selection error
- -3: Counter option selection error
- -100: Init counter kernel function error

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

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 4.1.2.56 int MSXE371x\_\_IncCounterRelease ( void \* \_\_, struct MSXE371x\_\_Response \* *Response* )

##### Parameters

[in] *\_\_* : no input parameter

[out] *Response* :

*iReturnValue* :

- 0: success
- -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

#### 4.1.2.57 int MSXE371x\_\_IncCounterRead32BitValue ( void \* \_\_, struct MSXE371x\_\_IncCounterRead32BitValueResponse \* *Response* )

##### Parameters

[in] *\_\_* : no input parameter

[out] *Response* :

*sResponse.iReturnValue* :

- 0: success
- -1: means an system error occurred
- -100: Read counter value kernel function error

*sResponse.syserrno* : system-error code (the value of the libc "errno" code) *ulValue* : counter value *ulTimeStampLow* : 32 bit low part of time stamp (us) *ulTimeStampHigh* : 32 bit high part of time stamp (s)

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**4.1.2.58** `int MSXE371x__IncCounterWrite32BitValue ( xsd__unsignedLong ulCounterValue, struct MSXE371x__Response * Response )`

#### Parameters

[in] *ulCounterValue* : Counter value

[out] *Response* :

*iReturnValue* :

- 0: success
- -1: means an system error occurred
- -2: Counter value error
- -100: Write counter value kernel function error

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

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**4.1.2.59** `int MSXE371x__IncCounterClear ( void * _, struct MSXE371x__Response * Response )`

#### Parameters

[in] *\_* : no input parameter

[out] *Response* :

*iReturnValue* :

- 0: success
- -1: means an system error occurred
- -100: Write counter value kernel function error

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

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**4.1.2.60** `int MSXE371x__IncCounterInitAndEnableCompareLogic ( xsd__unsignedLong ulValue, xsd__unsignedLong ulMode, xsd__unsignedLong ulSynchroTrigger, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02, struct MSXE371x__Response * Response )`

#### Parameters

- [in] **ulValue** : compare value ( 0 to 0xFFFFFFFF included )
- [in] **ulMode** : compare mode
- 0: condition true when counter equals compare value
  - 1: condition true when counter equals a multiple of the compare value
- [in] **ulSynchroTrigger**    • 0 : no synchro trigger
- 1 : generates a synchro trigger when condition is true
- [in] **ulOption01** : set it to 0
- [in] **ulOption02** : set it to 0
- [out] **Response** :
- iReturnValue** :
- 0: success
  - -1: means an system error occurred
  - -2: Compare value error
  - -3: Compare mode error
  - -4: Synchro trigger error
  - -100: Write counter compare value kernel function error
  - -101: Enable counter compare kernel function error
- syserrno** : system-error code (the value of the libc "errno" code)

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**4.1.2.61** `int MSXE371x__IncCounterDisableAndReleaseCompareLogic ( void * _, struct MSXE371x__Response * Response )`

#### Parameters

- [in] **\_** : no input parameter
- [out] **Response** :
- iReturnValue** :
- 0: success
  - -1: means an system error occurred
  - -100: Disable counter compare value kernel function error
- syserrno** : system-error code (the value of the libc "errno" code)

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**4.1.2.62** `int MSXE371x__IncCounterInitAndEnableIndex ( xsd__unsignedLong  
ulReferenceAction, xsd__unsignedLong ulIndexOperation, xsd__unsignedLong  
ulAutoMode, xsd__unsignedLong ulOption01, struct MSXE371x__Response * Response  
)`

#### Parameters

- [in] **ulReferenceAction** : Reference action
- 0: do not use the pin D as reference
  - 1: use the pin D as reference
- [in] **ulIndexOperation** : Index operation
- 0: latch and clear counter at low edge
  - 1: latch and clear counter at high edge
- [in] **ulAutoMode** : Auto mode
- 0 : do not use automode (action is done only once)
  - 1 : use automode (action is done continuously)
- [in] **ulOption01** : set it to 0
- [out] **Response** :
- iReturnValue** :
- 0: success
  - -1: means an system error occurred
  - -2: Reference action selection error
  - -3: Index operation selection error
  - -4: Automode selection error
  - -5: Index logic already initialized
  - -100: Write counter compare value kernel function error
- syserrno** : system-error code (the value of the libc "errno" code)

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**4.1.2.63** `int MSXE371x__IncCounterDisableAndReleaseIndex ( void * _, struct  
MSXE371x__Response * Response )`

#### Parameters

- [in] **\_** : no input parameter
- [out] **Response** :
- iReturnValue** :
- 0: success
  - -1: means an system error occurred
  - -2: Index logic already initialized
  - -100: Disable counter compare value kernel function error
- syserrno** : system-error code (the value of the libc "errno" code)

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error



**4.1.2.64** `int MSXE371x_ExternalTemperatureInit ( xsd__unsignedLong  
ulConnectedTempSensor, xsd__unsignedLong ulConvertMode, xsd__unsignedLong  
ulGainSelection, xsd__unsignedLong ulFrequencySelection, xsd__unsignedLong  
ulPowerSaveMode, xsd__unsignedLong ulOption01, xsd__unsignedLong ulOption02,  
xsd__unsignedLong ulOption03, xsd__unsignedLong ulOption04, struct  
MSXE371x__Response * Response )`

#### Parameters

[in] ***ulConnectedTempSensor*** Connected sensor type

- 1: TC type E
- 2: TC type J
- 3: TC type K
- 4: TC type N
- 5: TC type R
- 6: TC type S
- 7: TC type T
- 100 for PT100
- 200 for PT200
- 500 for PT500
- 1000 for PT1000

[in] ***ulConvertMode*** Converting mode

- 0: Disabled. Return the value into the mOhm form
  - 1: oC
  - 2: oF
- If you use a TC sensor, you can only ask the value in °C or °F

[in] ***ulGainSelection*** Gain selection (0 for auto mode)

If you use a TC sensor, only use auto gain (0)

[in] ***ulFrequencySelection*** Acquisition frequency selection (10,50, 240)

If you use a TC sensor, you can only use frequencies 50 and 240

[in] ***ulPowerSaveMode*** Power save mode selection

- 0 : Start the continuous acquisition
- 1 : Disable the acquisition.

[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: success
- -1: means an system error occurred
- -2: Connected temperature sensor selection error
- -3: Convert mode selection error
- -4: Gain selection error
- -5: Frequency selection error
- -6: Power save mode selection error
- -100: Init external temperature kernel function error

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

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**4.1.2.65** `int MSXE371x__ExternalTemperatureRelease ( void * _, struct MSXE371x__Response * Response )`

#### Parameters

[in] *\_* : no input parameter

[out] *Response* :

*iReturnValue* :

- 0: success
- -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

**4.1.2.66** `int MSXE371x__ExternalTemperatureRead ( void * _, struct MSXE371x__ExternalTemperatureReadResponse * Response )`

#### Parameters

[in] *\_* : no input parameter

[out] *Response* :

*sResponse.iReturnValue* :

- 0: success
- -1: means an system error occurred
- -2: Power save enabled and no transducer acquisition started
- -3: Power save enabled and transducer acquisition started but hardware trigger used
- -100: Read external temperature kernel function error

*sResponse.syserrno* : system-error code (the value of the libc "errno" code) *ulValue* : External temperature value. Format depend from the initialisation *ulTimeStampHigh* : 32 bit high part of time stamp (s) *ulTimeStampLow* : 32 bit low part of time stamp (us)

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**4.1.2.67** `int MSXE371x_ExternalTemperatureGainCalibration ( unsigned long ulGainSelection, double dRefValue, struct MSXE371x_Response * Response )`

#### Parameters

[in] *ulGainSelection* : Selected gain (1, 2, 4, 8, 16, 32, 64 or 128)

[in] *dRefValue* : Ref voltage value the the calibration

[out] *Response* :

*iReturnValue* :

- 0 : success
- -1 : means an system error occurred
- -2 : Gain selection error
- -3 : Frequency selection error
- -4 : Ref value to high for the selected gain
- -5 : Calibration file access error
- -100 : IOCTL system call error

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

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**4.1.2.68** `int MSXE371x_CalibrationStart ( xsd_unsignedLong ulTransducerIndex, xsd_double dPosition, xsd_unsignedLong ulCalibrationMode, xsd_unsignedLong ulInvertValue, xsd_unsignedLong ulChannel, struct MSXE371x_Response * Response )`

#### Parameters

[in] *ulTransducerIndex* : Selected transducer to calibrate

[in] *dPosition* : Selected user calibration position

[in] *ulCalibrationMode* : Select the calibration mode

- 0: Calibre the primary and all channels. *ulChannel* not used
- 1: Calibre only the selected channel (*ulChannel*).
- 2: Calibre the primary and all channels. *ulChannel* not used and reset the calibration bit
- 3: Calibre only the selected channel (*ulChannel*) and reset the calibration bit If not other channel calibrate then calibrate the primary and 0mm position

[in] *ulInvertValue* :

- 0 : don't invert the channel value
- 1 : invert the channel value (-2 mm -> + 2mm)

[in] *ulChannel* : Selected calibration channel

[out] *Response* :

*iReturnValue* : Error code :

- 0: means the remote function performed OK
- -1: means an system error ocured
- -2: Any calibration already started
- -3: Selected transducer not available

- -4: Calibration mode selection error
- -5: Invert value selection mode not available
- -6: Transducer channel selection error
- -100: Start calibration kernel function error *syserrno* : system-error code (the value of the libc "errno" code)

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**4.1.2.69** `int MSXE371x__CalibrationGetCurrentStatus ( void * __, struct MSXE371x__CalibrationGetCurrentStatusResponse * Response )`

#### Parameters

[in] *\_\_* : no input parameter

[out] *Response* :

[out] *Response* :

*sResponse.iReturnValue* : Error code :

- 0: means the remote function performed OK
- -1: means an system error occurred
- -100: Get the calibration status kernel function error *sResponse.syserrno* : system-error code (the value of the libc "errno" code) *ulStatus* : Status
- 0: No calibration in progress
- 1: Primary calibration in progress
- 2: Wait user access null position setting
- 3: Null position calibration in progress
- 4: Wait user access user position setting
- 5: User position calibration in progress
- 6: Wait user connect only one transducer for the primary open line diagnostic
- 7: Primary open line diagnostic in progress
- 8: Calibration finished

*ulChannel* : Current calibration channel

*ulDigitalValue* : Last measured digital value

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**4.1.2.70** `int MSXE371x__CalibrationNextStep ( void * __, struct MSXE371x__Response * Response )`

#### Parameters

[in] *\_\_* : no input parameter

[out] **Response** :

**iReturnValue** : Error code :

- 0: means the remote function performed OK
- -1: means an system error occured
- -100: Start next calibration step kernel function error **syserrno** : system-error code (the value of the libc "errno" code)

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**4.1.2.71** `int MSXE371x__CalibrationBreak ( void * _, struct MSXE371x__Response * Response )`

#### Parameters

[in] **\_** : no input parameter

[out] **Response** :

**iReturnValue** : Error code :

- 0: means the remote function performed OK
- -1: means an system error occured
- -100: Break calibration kernel function error **syserrno** : system-error code (the value of the libc "errno" code)

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**4.1.2.72** `int MSXE371x__DataBaseGetNumberOfTransducers ( void * _, struct MSXE371x__unsignedlongResponse * Response )`

#### Parameters

[in] **\_** : no input parameter

[out] **Response** :

**sResponse.iReturnValue** : Error code :

- 0: success
- <> 0 : error
- -100: kernel function error

**sResponse.syserrno** : system-error code (the value of the libc "errno" code) **ulValue** : number of transducer types.

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 4.1.2.73 `int MSXE371x__DataBaseGetTransducerType ( xsd__unsignedLong ulTransducerIndex, struct MSXE371x__unsignedlongResponse * Response )`

##### Parameters

[in] *ulTransducerIndex* : Transducer index (0 to *ulTransducerNumbers* - 1)

[out] *Response* :

*sResponse.iReturnValue* : Error code :

- 0: success
- <> 0 : error
- -100: kernel function error

*sResponse.syserrno* : system-error code (the value of the libc "errno" code) Its value is significant only when the *iReturnValue* returned an error (-1 or <= -100)

Give this value to the *MXCommon\_Sterror* to get the string describing the error number. *ul-Value* : transducer identifier. Use this value as the "Index" parameter given to *DataBaseGetTransducerInformationResponse()*.

##### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### 4.1.2.74 `int MSXE371x__DataBaseGetTransducerInformation ( xsd__unsignedLong ulTransducerIndex, struct MSXE371x__DataBaseGetTransducerInformationResponse * Response )`

##### Parameters

[in] *ulTransducerIndex* : transducer identifier, as returned by *DataBaseGetTransducerType()*.

[out] *Response* :

*sResponse.iReturnValue* : Error code :

- 0: success
- <> 0 : error
- -100: kernel function error

*sResponse.syserrno* : system-error code (the value of the libc "errno" code) Its value is significant only when the *iReturnValue* returned an error (-1 or <= -100)

Give this value to the *MXCommon\_Sterror* to get the string describing the error number.

*cName* : Name

*ulCalibrate* : Calibration state (0 : not calibrated 1 : calibrated)

*ulType* : Type (0: HB 1: LVDT 2:Knaebel 3:HB-Mahr 4:LVDT-Mahr)

*ulFrequency* : Nominal frequency (Hz)

*ulImpedance* : Impedance (Ohm)

*dVeff* : Nominal voltage (Vrms)

*dSensitivity* : Sensitivity (mV/V/mm)

*dRange* : Range (mm) *ulCalibratedChannels* : Bitmask of currently calibrated channels (D0 => channel 1, D1 => channel 1, ...)

##### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

**4.1.2.75** `int MSXE371x__DataBaseAddTransducer ( xsd__unsignedLong ulTransducerIndex, xsd__string cName, xsd__unsignedLong ulType, xsd__unsignedLong ulFrequency, xsd__unsignedLong ulImpedance, xsd__double dVeff, xsd__double dSensitivity, xsd__double dRange, struct MSXE371x__Response * Response )`

#### Parameters

- [in] *ulTransducerIndex* : Identifier of the new type, user-defined value in the range 200 .. 255
- [in] *cName* : Name
- [in] *ulType* : Type (0: HB 1: LVDT 2:Knaebel 3:HB-Mahr 4:LVDT-Mahr)
- [in] *ulFrequency* : Nominal frequency (Hz)
- [in] *ulImpedance* : Impedance (Ohm)
- [in] *dVeff* : Nominal voltage (Vrms)
- [in] *dSensitivity* : Sensitivity (mV/V/mm)
- [in] *dRange* : Range (mm)
- [out] *Response* :
  - iReturnValue* : Error code :
    - 0: success
    - <> 0 : error
    - -100: kernel function error
  - syserrno* : system-error code (the value of the libc "errno" code) Its value is significant only when the iReturnValue returned an error (-1 or <= -100)  
Give this value to the MXCommon\_Sterror to get the string describing the error number.

#### Returns

- 0: SOAP\_OK
- <> 0: See SOAP error

#### Note

This function returns an error if a transducer with the same identifier already exists in the database.

**4.1.2.76** `int MSXE371x__DataBaseDelTransducer ( xsd__unsignedLong ulTransducerIndex, struct MSXE371x__Response * Response )`

#### Parameters

- [in] *ulTransducerIndex* : identifier, as returned by DataBaseGetTransducerType().
- [out] *Response* :
  - iReturnValue* : Error value :
    - 0: success
    - <> 0 : error
    - -100: kernel function error
  - syserrno* : system-error code (the value of the libc "errno" code) Its value is significant only when the iReturnValue returned an error (-1 or <= -100)  
Give this value to the MXCommon\_Sterror to get the string describing the error number.

#### Returns

- 0: SOAP\_OK

- <> 0: See SOAP error

**Note**

This function returns an error if the identifier does not map to an existing transducer type.

**4.1.2.77** `int MSXE371x__DataBaseSaveTransducers ( void * __, struct MSXE371x__ByteArrayResponse * Response )`

**Parameters**

[in] `__` : no input parameter

[out] *Response* : *sResponse.iReturnValue* : Error code

- 0 success
- <> 0 : error

*sResponse.syserrno* : system-error code (the value of the libc "errno" code) Its value is significant only when the *iReturnValue* returned an error (-1 or <= -100)

Give this value to the `MXCommon_Sterror` to get the string describing the error number.

**Returns**

- 0: SOAP\_OK
- <> 0: See SOAP error



# Index

- `__offset`
    - ByteArray, [61](#)
    - UnsignedLongArray, [78](#)
    - UnsignedShortArray, [79](#)
  - `__ptr`
    - ByteArray, [61](#)
    - UnsignedLongArray, [78](#)
    - UnsignedShortArray, [79](#)
    - xsd\_\_base64Binary, [79](#)
  - `__size`
    - ByteArray, [61](#)
    - UnsignedLongArray, [78](#)
    - UnsignedShortArray, [79](#)
    - xsd\_\_base64Binary, [79](#)
- Analog
  - MXCommon\_\_SetFilterChannels, [30](#)
- bArray
  - MXCommon\_\_-
    - GetAutoConfigurationFileResponse, [72](#)
- bCryptedValueArray
  - MXCommon\_\_TestCustomerIDResponse, [77](#)
- bValueArray
  - MXCommon\_\_TestCustomerIDResponse, [77](#)
- ByteArray, [61](#)
  - `__offset`, [61](#)
  - `__ptr`, [61](#)
  - `__size`, [61](#)
- cName
  - MSXE371x\_\_-
    - DataBaseGetTransducerInformationResponse, [65](#)
- Common functions, [3](#)
- Common general functions, [4](#)
- Common hardware trigger functions, [11](#)
- Common I/O auto configuration functions, [18](#)
- Common security functions, [13](#)
- Common synchronisation timer functions, [20](#)
- Common temperature functions, [10](#)
- Common time functions, [15](#)
- 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\_\_DataserverRestart, [8](#)
  - MXCommon\_\_GetClientConnections, [6](#)
  - MXCommon\_\_GetEthernetLinksStates, [9](#)
  - MXCommon\_\_GetHostname, [5](#)
  - MXCommon\_\_GetModuleType, [5](#)
  - MXCommon\_\_Reboot, [8](#)
  - MXCommon\_\_ResetAllIOFunctionalities, [8](#)
  - MXCommon\_\_SetHostname, [6](#)
  - MXCommon\_\_Sterror, [6](#)
- Common\_hardware\_trigger
  - MXCommon\_\_-
    - GetHardwareTriggerFilterTime, [12](#)
    - MXCommon\_\_GetHardwareTriggerState, [13](#)
    - MXCommon\_\_-
      - SetHardwareTriggerFilterTime, [12](#)
- Common\_security
  - MXCommon\_\_SetCustomerKey, [15](#)
  - MXCommon\_\_TestCustomerID, [15](#)
- Common\_synchrotimer
  - MXCommon\_\_InitAndStartSynchroTimer, [21](#)
  - MXCommon\_\_-
    - StopAndReleaseSynchroTimer, [21](#)
- Common\_temperature
  - MXCommon\_\_-
    - GetModuleTemperatureValueAndStatus, [10](#)
    - MXCommon\_\_-
      - SetModuleTemperatureWarningLevels, [11](#)
- Common\_time
  - MXCommon\_\_GetTime, [17](#)
  - MXCommon\_\_GetUpTime, [18](#)
  - MXCommon\_\_HardwareClockToSys, [17](#)
  - MXCommon\_\_SetTime, [16](#)
  - MXCommon\_\_SysToHardwareClock, [16](#)
- Customer option management, [27](#)

- CustomerOption
  - MXCommon\_\_GetOptionInformation, 28
- DefaultResponse, 61
  - iReturnValue, 62
  - syserrno, 62
- dRange
  - MSXE371x\_\_-
    - DataBaseGetTransducerInformationResponse, 65
  - MSXE371x\_\_-
    - TransducerGetTypeInformationResponse, 69
- dSensitivity
  - MSXE371x\_\_-
    - TransducerGetTypeInformationResponse, 69
- dSensitivity
  - MSXE371x\_\_-
    - DataBaseGetTransducerInformationResponse, 65
- dTemperatureValue
  - MXCommon\_\_-
    - GetModuleTemperatureValueAndStatusResponse, 75
- dVeff
  - MSXE371x\_\_-
    - DataBaseGetTransducerInformationResponse, 65
  - MSXE371x\_\_-
    - TransducerGetTypeInformationResponse, 69
- input filter Filter management, 29
- iReturnValue
  - DefaultResponse, 62
  - MSXE371x\_\_Response, 67
  - MXCommon\_\_Response, 76
- MSX-E371x external temperature functions, 50
- MSX-E371x incremental counter functions, 44
- MSXE371x Auto refresh functions, 31
- MSXE371x calibration functions, 53
- MSXE371x diagnostic functions, 41
- MSXE371x functions, 3
- MSXE371x Get informations functions, 30
- MSXE371x Sequence functions, 36
- MSXE371x transducer database management functions, 56
- MSXE371x\_\_ByteArrayResponse, 62
  - sArray, 62
  - sResponse, 62
- MSXE371x\_\_CalibrationBreak
  - MSXE371x\_Calib, 55
  - MSXE371x\_public\_doc.h, 127
- MSXE371x\_\_CalibrationGetCurrentStatus
  - MSXE371x\_Calib, 54
  - MSXE371x\_public\_doc.h, 126
- MSXE371x\_\_CalibrationGetCurrentStatusResponse, 62
  - sResponse, 63
  - ulChannel, 63
  - ulDigitalValue, 63
  - ulStatus, 63
- MSXE371x\_\_CalibrationNextStep
  - MSXE371x\_Calib, 55
  - MSXE371x\_public\_doc.h, 126
- MSXE371x\_\_CalibrationStart
  - MSXE371x\_Calib, 54
  - MSXE371x\_public\_doc.h, 125
- MSXE371x\_\_DataBaseAddTransducer
  - MSXE371x\_public\_doc.h, 128
  - MSXE371x\_Transducer\_Database, 58
- MSXE371x\_\_DataBaseDelTransducer
  - MSXE371x\_public\_doc.h, 129
  - MSXE371x\_Transducer\_Database, 59
- MSXE371x\_\_DataBaseGetNumberOfTransducers
  - MSXE371x\_public\_doc.h, 127
  - MSXE371x\_Transducer\_Database, 57
- MSXE371x\_\_DataBaseGetTransducerInformation
  - MSXE371x\_public\_doc.h, 128
  - MSXE371x\_Transducer\_Database, 57
- MSXE371x\_\_DataBaseGetTransducerInformationResponse, 63
  - cName, 65
  - dRange, 65
  - dSensitivity, 65
  - dVeff, 65
  - sResponse, 65
  - ulCalibrate, 65
  - ulCalibratedChannels, 65
  - ulFrequency, 65
  - ulImpedance, 65
  - ulType, 65
- MSXE371x\_\_DataBaseGetTransducerType
  - MSXE371x\_public\_doc.h, 127
  - MSXE371x\_Transducer\_Database, 57
- MSXE371x\_\_DataBaseSaveTransducers
  - MSXE371x\_public\_doc.h, 130
  - MSXE371x\_Transducer\_Database, 59
- MSXE371x\_\_ExternalTemperatureGainCalibration
  - MSXE371x\_ExternalTemperature, 52
  - MSXE371x\_public\_doc.h, 124
- MSXE371x\_\_ExternalTemperatureInit
  - MSXE371x\_ExternalTemperature, 51
  - MSXE371x\_public\_doc.h, 122
- MSXE371x\_\_ExternalTemperatureRead
  - MSXE371x\_ExternalTemperature, 52

- MSXE371x\_public\_doc.h, [124](#)
- MSXE371x\_\_ExternalTemperatureReadResponse, [65](#)
  - sResponse, [66](#)
  - ulTimeStampHigh, [66](#)
  - ulTimeStampLow, [66](#)
  - ulValue, [66](#)
- MSXE371x\_\_ExternalTemperatureRelease
  - MSXE371x\_ExternalTemperature, [52](#)
  - MSXE371x\_public\_doc.h, [124](#)
- MSXE371x\_\_IncCounterClear
  - MSXE371x\_IncCounter, [47](#)
  - MSXE371x\_public\_doc.h, [120](#)
- MSXE371x\_\_IncCounterDisableAndReleaseCompareLogic
  - MSXE371x\_IncCounter, [48](#)
  - MSXE371x\_public\_doc.h, [121](#)
- MSXE371x\_\_IncCounterDisableAndReleaseIndex
  - MSXE371x\_IncCounter, [49](#)
  - MSXE371x\_public\_doc.h, [122](#)
- MSXE371x\_\_IncCounterInit
  - MSXE371x\_IncCounter, [45](#)
  - MSXE371x\_public\_doc.h, [118](#)
- MSXE371x\_\_IncCounterInitAndEnableCompareLogic
  - MSXE371x\_IncCounter, [48](#)
  - MSXE371x\_public\_doc.h, [120](#)
- MSXE371x\_\_IncCounterInitAndEnableIndex
  - MSXE371x\_IncCounter, [49](#)
  - MSXE371x\_public\_doc.h, [121](#)
- MSXE371x\_\_IncCounterRead32BitValue
  - MSXE371x\_IncCounter, [46](#)
  - MSXE371x\_public\_doc.h, [119](#)
- MSXE371x\_\_IncCounterRead32BitValueResponse, [66](#)
  - sResponse, [66](#)
  - ulTimeStampHigh, [66](#)
  - ulTimeStampLow, [66](#)
  - ulValue, [66](#)
- MSXE371x\_\_IncCounterRelease
  - MSXE371x\_IncCounter, [46](#)
  - MSXE371x\_public\_doc.h, [119](#)
- MSXE371x\_\_IncCounterWrite32BitValue
  - MSXE371x\_IncCounter, [47](#)
  - MSXE371x\_public\_doc.h, [120](#)
- MSXE371x\_\_Response, [66](#)
  - iReturnValue, [67](#)
  - syserrno, [67](#)
- MSXE371x\_\_TransducerGetAllAutoRefreshValues
  - MSXE371x\_AutoRefresh, [35](#)
  - MSXE371x\_public\_doc.h, [111](#)
- MSXE371x\_\_TransducerGetAllValuesResponse, [67](#)
  - sResponse, [68](#)
  - ulAutoRefreshCounter, [68](#)
  - ulExternalTemperature, [68](#)
  - ulIncCounter, [68](#)
  - ulTimeStampHigh, [68](#)
  - ulTimeStampLow, [68](#)
  - ulTransducersValue, [68](#)
- MSXE371x\_\_TransducerGetAutoRefreshValues
  - MSXE371x\_AutoRefresh, [34](#)
  - MSXE371x\_public\_doc.h, [111](#)
- MSXE371x\_\_TransducerGetNbrOfType
  - MSXE371x\_GetInfo, [30](#)
  - MSXE371x\_public\_doc.h, [108](#)
- MSXE371x\_\_TransducerGetTypeInformation
  - MSXE371x\_GetInfo, [31](#)
  - MSXE371x\_public\_doc.h, [108](#)
- MSXE371x\_\_TransducerGetTypeInformationResponse, [68](#)
  - dRange, [69](#)
  - dSensibility, [69](#)
  - dVeff, [69](#)
  - pcName, [69](#)
  - sResponse, [69](#)
  - ulCalibratedChannels, [69](#)
  - ulCalibrationStatus, [69](#)
  - ulFrequency, [69](#)
  - ulImpedance, [69](#)
  - ulTransducerSelectionIndex, [69](#)
  - ulType, [69](#)
- MSXE371x\_\_TransducerInit
  - MSXE371x\_Diagnose, [41](#)
  - MSXE371x\_public\_doc.h, [115](#)
- MSXE371x\_\_TransducerInitAndStartAutoRefresh
  - MSXE371x\_AutoRefresh, [33](#)
  - MSXE371x\_public\_doc.h, [109](#)
- MSXE371x\_\_TransducerInitAndStartSequence
  - MSXE371x\_public\_doc.h, [112](#)
  - MSXE371x\_Sequence, [38](#)
- MSXE371x\_\_TransducerInitPrimaryConnectionTest
  - MSXE371x\_Diagnose, [42](#)
  - MSXE371x\_public\_doc.h, [116](#)
- MSXE371x\_\_TransducerRearmPrimary
  - MSXE371x\_Diagnose, [43](#)
  - MSXE371x\_public\_doc.h, [117](#)
- MSXE371x\_\_TransducerStopAndReleaseAutoRefresh
  - MSXE371x\_AutoRefresh, [35](#)
  - MSXE371x\_public\_doc.h, [112](#)
- MSXE371x\_\_TransducerStopAndReleaseSequence
  - MSXE371x\_public\_doc.h, [114](#)
  - MSXE371x\_Sequence, [40](#)
- MSXE371x\_\_TransducerStopAndReleaseSequence2
  - MSXE371x\_public\_doc.h, [115](#)
  - MSXE371x\_Sequence, [40](#)
- MSXE371x\_\_TransducerTestPrimaryConnection
  - MSXE371x\_Diagnose, [42](#)
  - MSXE371x\_public\_doc.h, [116](#)
- MSXE371x\_\_TransducerTestPrimaryShortCircuit

- MSXE371x\_Diagnose, [43](#)
- MSXE371x\_public\_doc.h, [116](#)
- MSXE371x\_\_TransducerTestSecondaryConnection
  - MSXE371x\_Diagnose, [43](#)
  - MSXE371x\_public\_doc.h, [117](#)
- MSXE371x\_\_TransducerTestSecondaryShortCircuit
  - MSXE371x\_Diagnose, [44](#)
  - MSXE371x\_public\_doc.h, [118](#)
- MSXE371x\_\_unsignedLong8FixedArray, [69](#)
  - ulValue, [70](#)
- MSXE371x\_\_unsignedlong9ArrayResponse, [70](#)
  - sResponse, [70](#)
  - ulValue, [70](#)
- MSXE371x\_\_unsignedlongDefaultResponse, [70](#)
  - sResponse, [70](#)
  - ulValue, [70](#)
- MSXE371x\_\_unsignedlongResponse, [70](#)
  - sResponse, [71](#)
  - ulValue, [71](#)
- MSXE371x\_AutoRefresh
  - MSXE371x\_\_-
    - TransducerGetAllAutoRefreshValues, [35](#)
  - MSXE371x\_\_-
    - TransducerGetAutoRefreshValues, [34](#)
  - MSXE371x\_\_-
    - TransducerInitAndStartAutoRefresh, [33](#)
  - MSXE371x\_\_-
    - TransducerStopAndReleaseAutoRefresh, [35](#)
- MSXE371x\_Calib
  - MSXE371x\_\_CalibrationBreak, [55](#)
  - MSXE371x\_\_CalibrationGetCurrentStatus, [54](#)
  - MSXE371x\_\_CalibrationNextStep, [55](#)
  - MSXE371x\_\_CalibrationStart, [54](#)
- MSXE371x\_Diagnose
  - MSXE371x\_\_TransducerInit, [41](#)
  - MSXE371x\_\_-
    - TransducerInitPrimaryConnectionTest, [42](#)
  - MSXE371x\_\_TransducerRearmPrimary, [43](#)
  - MSXE371x\_\_-
    - TransducerTestPrimaryConnection, [42](#)
  - MSXE371x\_\_-
    - TransducerTestPrimaryShortCircuit, [43](#)
  - MSXE371x\_\_-
    - TransducerTestSecondaryConnection, [43](#)
  - MSXE371x\_\_-
    - TransducerTestSecondaryShortCircuit, [44](#)
- MSXE371x\_\_ExternalTemperature
  - MSXE371x\_\_-
    - ExternalTemperatureGainCalibration, [52](#)
  - MSXE371x\_\_ExternalTemperatureInit, [51](#)
  - MSXE371x\_\_ExternalTemperatureRead, [52](#)
  - MSXE371x\_\_ExternalTemperatureRelease, [52](#)
- MSXE371x\_GetInfo
  - MSXE371x\_\_TransducerGetNbrOfType, [30](#)
  - MSXE371x\_\_TransducerGetTypeInformation, [31](#)
- MSXE371x\_IncCounter
  - MSXE371x\_\_IncCounterClear, [47](#)
  - MSXE371x\_\_-
    - IncCounterDisableAndReleaseCompareLogic, [48](#)
  - MSXE371x\_\_-
    - IncCounterDisableAndReleaseIndex, [49](#)
  - MSXE371x\_\_IncCounterInit, [45](#)
  - MSXE371x\_\_-
    - IncCounterInitAndEnableCompareLogic, [48](#)
  - MSXE371x\_\_IncCounterInitAndEnableIndex, [49](#)
  - MSXE371x\_\_IncCounterRead32BitValue, [46](#)
  - MSXE371x\_\_IncCounterRelease, [46](#)
  - MSXE371x\_\_IncCounterWrite32BitValue, [47](#)
- MSXE371x\_public\_doc.h, [81](#)
- MSXE371x\_\_CalibrationBreak, [127](#)
- MSXE371x\_\_CalibrationGetCurrentStatus, [126](#)
- MSXE371x\_\_CalibrationNextStep, [126](#)
- MSXE371x\_\_CalibrationStart, [125](#)
- MSXE371x\_\_DataBaseAddTransducer, [128](#)
- MSXE371x\_\_DataBaseDelTransducer, [129](#)
- MSXE371x\_\_-
  - DataBaseGetNumberOfTransducers, [127](#)
- MSXE371x\_\_-
  - DataBaseGetTransducerInformation, [128](#)
- MSXE371x\_\_DataBaseGetTransducerType, [127](#)
- MSXE371x\_\_DataBaseSaveTransducers, [130](#)
- MSXE371x\_\_-
  - ExternalTemperatureGainCalibration, [124](#)
- MSXE371x\_\_ExternalTemperatureInit, [122](#)
- MSXE371x\_\_ExternalTemperatureRead, [124](#)
- MSXE371x\_\_ExternalTemperatureRelease, [124](#)

- MSXE371x\_\_IncCounterClear, [120](#)
- MSXE371x\_\_-
  - IncCounterDisableAndReleaseCompareLogic, [121](#)
- MSXE371x\_\_-
  - IncCounterDisableAndReleaseIndex, [122](#)
- MSXE371x\_\_IncCounterInit, [118](#)
- MSXE371x\_\_-
  - IncCounterInitAndEnableCompareLogic, [120](#)
- MSXE371x\_\_IncCounterInitAndEnableIndex, [121](#)
- MSXE371x\_\_IncCounterRead32BitValue, [119](#)
- MSXE371x\_\_IncCounterRelease, [119](#)
- MSXE371x\_\_IncCounterWrite32BitValue, [120](#)
- MSXE371x\_\_-
  - TransducerGetAllAutoRefreshValues, [111](#)
- MSXE371x\_\_-
  - TransducerGetAutoRefreshValues, [111](#)
- MSXE371x\_\_TransducerGetNbrOfType, [108](#)
- MSXE371x\_\_TransducerGetTypeInformation, [108](#)
- MSXE371x\_\_TransducerInit, [115](#)
- MSXE371x\_\_-
  - TransducerInitAndStartAutoRefresh, [109](#)
- MSXE371x\_\_-
  - TransducerInitAndStartSequence, [112](#)
- MSXE371x\_\_-
  - TransducerInitPrimaryConnectionTest, [116](#)
- MSXE371x\_\_TransducerRearmPrimary, [117](#)
- MSXE371x\_\_-
  - TransducerStopAndReleaseAutoRefresh, [112](#)
- MSXE371x\_\_-
  - TransducerStopAndReleaseSequence, [114](#)
- MSXE371x\_\_-
  - TransducerStopAndReleaseSequence2, [115](#)
- MSXE371x\_\_-
  - TransducerTestPrimaryConnection, [116](#)
- MSXE371x\_\_-
  - TransducerTestPrimaryShortCircuit, [116](#)
- MSXE371x\_\_-
  - TransducerTestSecondaryConnection, [117](#)
- MSXE371x\_\_-
  - TransducerTestSecondaryShortCircuit, [118](#)
- MXCommon\_\_-
  - ApplyConfigurationBackupFile, [103](#)
  - MXCommon\_\_ChangePassword, [104](#)
  - MXCommon\_\_DataseverRestart, [93](#)
  - MXCommon\_\_GetAutoConfigurationFile, [100](#)
  - MXCommon\_\_GetClientConnections, [91](#)
  - MXCommon\_\_GetConfigurationBackupFile, [102](#)
  - MXCommon\_\_GetEthernetLinksStates, [94](#)
  - MXCommon\_\_-
    - GetHardwareTriggerFilterTime, [96](#)
  - MXCommon\_\_GetHardwareTriggerState, [97](#)
  - MXCommon\_\_GetHostname, [90](#)
  - MXCommon\_\_-
    - GetModuleTemperatureValueAndStatus, [95](#)
  - MXCommon\_\_GetModuleType, [90](#)
  - MXCommon\_\_GetOptionInformation, [106](#)
  - MXCommon\_\_GetStateIDFromName, [105](#)
  - MXCommon\_\_GetStateNameFromID, [106](#)
  - MXCommon\_\_GetSubsystemIDFromName, [105](#)
  - MXCommon\_\_GetSubsystemNameFromID, [105](#)
  - MXCommon\_\_GetSubSystemState, [104](#)
  - MXCommon\_\_GetSynchronizationStatus, [107](#)
  - MXCommon\_\_GetTime, [99](#)
  - MXCommon\_\_GetUpTime, [100](#)
  - MXCommon\_\_HardwareClockToSys, [99](#)
  - MXCommon\_\_InitAndStartSynchroTimer, [101](#)
  - MXCommon\_\_Reboot, [93](#)
  - MXCommon\_\_ResetAllIOFunctionalities, [93](#)
  - MXCommon\_\_SetAutoConfigurationFile, [100](#)
  - MXCommon\_\_SetCustomerKey, [97](#)
  - MXCommon\_\_SetFilterChannels, [107](#)
  - MXCommon\_\_-
    - SetHardwareTriggerFilterTime, [96](#)
  - MXCommon\_\_SetHostname, [91](#)
  - MXCommon\_\_-
    - SetModuleTemperatureWarningLevels, [95](#)
  - MXCommon\_\_SetTime, [98](#)
  - MXCommon\_\_SetToMaster, [106](#)
  - MXCommon\_\_StartAutoConfiguration, [101](#)
  - MXCommon\_\_-
    - StopAndReleaseSynchroTimer, [102](#)
  - MXCommon\_\_Sterror, [91](#)

- MXCommon\_\_SysToHardwareClock, 98
- MXCommon\_\_TestCustomerID, 98
- xsd\_\_char, 90
- xsd\_\_double, 90
- xsd\_\_float, 90
- xsd\_\_int, 90
- xsd\_\_long, 90
- xsd\_\_string, 90
- xsd\_\_unsignedByte, 90
- xsd\_\_unsignedInt, 90
- xsd\_\_unsignedLong, 90
- xsd\_\_unsignedShort, 90
- MSXE371x\_Sequence
  - MSXE371x\_\_-
    - TransducerInitAndStartSequence, 38
  - MSXE371x\_\_-
    - TransducerStopAndReleaseSequence, 40
  - MSXE371x\_\_-
    - TransducerStopAndReleaseSequence2, 40
- MSXE371x\_Transducer\_Database
  - MSXE371x\_\_DataBaseAddTransducer, 58
  - MSXE371x\_\_DataBaseDelTransducer, 59
  - MSXE371x\_\_-
    - DataBaseGetNumberOfTransducers, 57
  - MSXE371x\_\_-
    - DataBaseGetTransducerInformation, 57
  - MSXE371x\_\_DataBaseGetTransducerType, 57
  - MSXE371x\_\_DataBaseSaveTransducers, 59
- MXCommon\_\_ApplyConfigurationBackupFile
  - Common\_configuration, 23
  - MSXE371x\_public\_doc.h, 103
- MXCommon\_\_ByteArrayResponse, 71
  - sArray, 71
  - sResponse, 71
- MXCommon\_\_ChangePassword
  - Common\_configuration, 24
  - MSXE371x\_public\_doc.h, 104
- MXCommon\_\_DataserverRestart
  - Common\_general, 8
  - MSXE371x\_public\_doc.h, 93
- MXCommon\_\_FileResponse, 71
  - sArray, 72
  - sResponse, 72
  - ulEOF, 72
- MXCommon\_\_GetAutoConfigurationFile
  - Common\_autoconf, 19
  - MSXE371x\_public\_doc.h, 100
- MXCommon\_\_GetAutoConfigurationFileResponse, 72
- bArray, 72
- sResponse, 72
- ulEOF, 72
- MXCommon\_\_GetClientConnections
  - Common\_general, 6
  - MSXE371x\_public\_doc.h, 91
- MXCommon\_\_GetConfigurationBackupFile
  - Common\_configuration, 23
  - MSXE371x\_public\_doc.h, 102
- MXCommon\_\_GetEthernetLinksStates
  - Common\_general, 9
  - MSXE371x\_public\_doc.h, 94
- MXCommon\_\_GetEthernetLinksStatesResponse, 72
  - sPort0, 73
  - sPort1, 73
  - sResponse, 73
- MXCommon\_\_GetHardwareTriggerFilterTime
  - Common\_hardware\_trigger, 12
  - MSXE371x\_public\_doc.h, 96
- MXCommon\_\_GetHardwareTriggerFilterTimeResponse, 73
  - sResponse, 73
  - ulFilterTime, 73
  - ulInfo01, 73
  - ulInfo02, 73
- MXCommon\_\_GetHardwareTriggerState
  - Common\_hardware\_trigger, 13
  - MSXE371x\_public\_doc.h, 97
- MXCommon\_\_GetHardwareTriggerStateResponse, 73
  - sResponse, 74
  - ulInfo01, 74
  - ulInfo02, 74
  - ulState, 74
- MXCommon\_\_GetHostname
  - Common\_general, 5
  - MSXE371x\_public\_doc.h, 90
- MXCommon\_\_GetModuleTemperatureValueAndStatus
  - Common\_temperature, 10
  - MSXE371x\_public\_doc.h, 95
- MXCommon\_\_GetModuleTemperatureValueAndStatusResponse, 74
  - dTemperatureValue, 75
  - sResponse, 75
  - ulInfo, 75
  - ulTemperatureStatus, 75
- MXCommon\_\_GetModuleType
  - Common\_general, 5
  - MSXE371x\_public\_doc.h, 90
- MXCommon\_\_GetOptionInformation
  - CustomerOption, 28
  - MSXE371x\_public\_doc.h, 106
- MXCommon\_\_GetStateIDFromName



- MSXE371x\_public\_doc.h, 105
- SystemStatemanagement, 26
- MXCommon\_\_GetStateNameFromID
  - MSXE371x\_public\_doc.h, 106
  - SystemStatemanagement, 27
- MXCommon\_\_GetSubsystemIDFromName
  - MSXE371x\_public\_doc.h, 105
  - SystemStatemanagement, 26
- MXCommon\_\_GetSubsystemNameFromID
  - MSXE371x\_public\_doc.h, 105
  - SystemStatemanagement, 26
- MXCommon\_\_GetSubSystemState
  - MSXE371x\_public\_doc.h, 104
  - SystemStatemanagement, 25
- MXCommon\_\_GetSynchronizationStatus
  - MSXE371x\_public\_doc.h, 107
  - Synchronisation, 29
- MXCommon\_\_GetTime
  - Common\_time, 17
  - MSXE371x\_public\_doc.h, 99
- MXCommon\_\_GetTimeResponse, 75
  - sResponse, 75
  - ulHighTime, 75
  - ulLowTime, 75
- MXCommon\_\_GetUpTime
  - Common\_time, 18
  - MSXE371x\_public\_doc.h, 100
- MXCommon\_\_GetUpTimeResponse, 75
  - sResponse, 76
  - ulUpTime, 76
- MXCommon\_\_HardwareClockToSys
  - Common\_time, 17
  - MSXE371x\_public\_doc.h, 99
- MXCommon\_\_InitAndStartSynchroTimer
  - Common\_synchrotimer, 21
  - MSXE371x\_public\_doc.h, 101
- MXCommon\_\_Reboot
  - Common\_general, 8
  - MSXE371x\_public\_doc.h, 93
- MXCommon\_\_ResetAllIOFunctionalities
  - Common\_general, 8
  - MSXE371x\_public\_doc.h, 93
- MXCommon\_\_Response, 76
  - iReturnValue, 76
  - syserrno, 76
- MXCommon\_\_SetAutoConfigurationFile
  - Common\_autoconf, 19
  - MSXE371x\_public\_doc.h, 100
- MXCommon\_\_SetCustomerKey
  - Common\_security, 15
  - MSXE371x\_public\_doc.h, 97
- MXCommon\_\_SetFilterChannels
  - Analog, 30
  - MSXE371x\_public\_doc.h, 107
- MXCommon\_\_SetHardwareTriggerFilterTime
  - Common\_hardware\_trigger, 12
  - MSXE371x\_public\_doc.h, 96
- MXCommon\_\_SetHostname
  - Common\_general, 6
  - MSXE371x\_public\_doc.h, 91
- MXCommon\_\_SetModuleTemperatureWarningLevels
  - Common\_temperature, 11
  - MSXE371x\_public\_doc.h, 95
- MXCommon\_\_SetTime
  - Common\_time, 16
  - MSXE371x\_public\_doc.h, 98
- MXCommon\_\_SetToMaster
  - MSXE371x\_public\_doc.h, 106
  - Synchronisation, 28
- MXCommon\_\_StartAutoConfiguration
  - Common\_autoconf, 20
  - MSXE371x\_public\_doc.h, 101
- MXCommon\_\_StopAndReleaseSynchroTimer
  - Common\_synchrotimer, 21
  - MSXE371x\_public\_doc.h, 102
- MXCommon\_\_Sterror
  - Common\_general, 6
  - MSXE371x\_public\_doc.h, 91
- MXCommon\_\_SysToHardwareClock
  - Common\_time, 16
  - MSXE371x\_public\_doc.h, 98
- MXCommon\_\_TestCustomerID
  - Common\_security, 15
  - MSXE371x\_public\_doc.h, 98
- MXCommon\_\_TestCustomerIDResponse, 76
  - bCryptedValueArray, 77
  - bValueArray, 77
  - sResponse, 77
- MXCommon\_\_unsignedLongResponse, 77
  - sResponse, 77
  - ulValue, 77
- pcName
  - MSXE371x\_\_-
    - TransducerGetTypeInformationResponse, 69
- sArray
  - MSXE371x\_\_ByteArrayResponse, 62
  - MXCommon\_\_ByteArrayResponse, 71
  - MXCommon\_\_FileResponse, 72
- Set/Backup/Restore general system configuration, 22
- sGetEthernetLinksStatesPort, 77
  - ulDuplex, 78
  - ulInfo1, 78
  - ulInfo2, 78
  - ulSpeed, 78

- ulState, 78
- sPort0
  - MXCommon\_\_-
    - GetEthernetLinksStatesResponse, 73
- sPort1
  - MXCommon\_\_-
    - GetEthernetLinksStatesResponse, 73
- sResponse
  - MSXE371x\_\_ByteArrayResponse, 62
  - MSXE371x\_\_-
    - CalibrationGetCurrentStatusResponse, 63
  - MSXE371x\_\_-
    - DataBaseGetTransducerInformationResponse, 65
  - MSXE371x\_\_-
    - ExternalTemperatureReadResponse, 66
  - MSXE371x\_\_-
    - IncCounterRead32BitValueResponse, 66
  - MSXE371x\_\_-
    - TransducerGetAllValuesResponse, 68
  - MSXE371x\_\_-
    - TransducerGetTypeInformationResponse, 69
  - MSXE371x\_\_unsignedlong9ArrayResponse, 70
  - MSXE371x\_\_unsignedlongDefaultResponse, 70
  - MSXE371x\_\_unsignedlongResponse, 71
  - MXCommon\_\_ByteArrayResponse, 71
  - MXCommon\_\_FileResponse, 72
  - MXCommon\_\_-
    - GetAutoConfigurationFileResponse, 72
  - MXCommon\_\_-
    - GetEthernetLinksStatesResponse, 73
  - MXCommon\_\_-
    - GetHardwareTriggerFilterTimeResponse, 73
  - MXCommon\_\_-
    - GetHardwareTriggerStateResponse, 74
  - MXCommon\_\_-
    - GetModuleTemperatureValueAndStatusResponse, 75
  - MXCommon\_\_GetTimeResponse, 75
  - MXCommon\_\_GetUpTimeResponse, 76
  - MXCommon\_\_TestCustomerIDResponse, 77
  - MXCommon\_\_unsignedLongResponse, 77
- Synchronisation
  - MXCommon\_\_GetSynchronizationStatus, 29
  - MXCommon\_\_SetToMaster, 28
- Synchronisation management, 28
- syserrno
  - DefaultResponse, 62
  - MSXE371x\_\_Response, 67
  - MXCommon\_\_Response, 76
- System state management, 24
- SystemStateManagement
  - MXCommon\_\_GetStateIDFromName, 26
  - MXCommon\_\_GetStateNameFromID, 27
  - MXCommon\_\_GetSubsystemIDFromName, 26
  - MXCommon\_\_GetSubsystemNameFromID, 26
  - MXCommon\_\_GetSubSystemState, 25
- ulAutoRefreshCounter
  - MSXE371x\_\_-
    - TransducerGetAllValuesResponse, 68
- ulCalibrate
  - MSXE371x\_\_-
    - DataBaseGetTransducerInformationResponse, 65
- ulCalibratedChannels
  - MSXE371x\_\_-
    - DataBaseGetTransducerInformationResponse, 65
  - MSXE371x\_\_-
    - TransducerGetTypeInformationResponse, 69
- ulCalibrationStatus
  - MSXE371x\_\_-
    - TransducerGetTypeInformationResponse, 69
- ulChannel
  - MSXE371x\_\_-
    - CalibrationGetCurrentStatusResponse, 63
- ulDigitalValue
  - MSXE371x\_\_-
    - CalibrationGetCurrentStatusResponse, 63
- ulDuplex
  - sGetEthernetLinksStatesPort, 78
- ulEOF
  - MXCommon\_\_FileResponse, 72
  - MXCommon\_\_-
    - GetAutoConfigurationFileResponse, 72
- ulExternalTemperature
  - MSXE371x\_\_-
    - TransducerGetAllValuesResponse, 68



- ulFilterTime
  - MXCommon\_\_-
    - GetHardwareTriggerFilterTimeResponse, 73
- ulFrequency
  - MSXE371x\_\_-
    - DataBaseGetTransducerInformationResponse, 65
  - MSXE371x\_\_-
    - TransducerGetTypeInformationResponse, 69
- ulHighTime
  - MXCommon\_\_GetTimeResponse, 75
- ulImpedance
  - MSXE371x\_\_-
    - DataBaseGetTransducerInformationResponse, 65
  - MSXE371x\_\_-
    - TransducerGetTypeInformationResponse, 69
- ulIncCounter
  - MSXE371x\_\_-
    - TransducerGetAllValuesResponse, 68
- ulInfo
  - MXCommon\_\_-
    - GetModuleTemperatureValueAndStatusResponse, 75
- ulInfo01
  - MXCommon\_\_-
    - GetHardwareTriggerFilterTimeResponse, 73
  - MXCommon\_\_-
    - GetHardwareTriggerStateResponse, 74
- ulInfo02
  - MXCommon\_\_-
    - GetHardwareTriggerFilterTimeResponse, 73
  - MXCommon\_\_-
    - GetHardwareTriggerStateResponse, 74
- ulInfo1
  - sGetEthernetLinksStatesPort, 78
- ulInfo2
  - sGetEthernetLinksStatesPort, 78
- ulLowTime
  - MXCommon\_\_GetTimeResponse, 75
- ulSpeed
  - sGetEthernetLinksStatesPort, 78
- ulState
  - MXCommon\_\_-
    - GetHardwareTriggerStateResponse, 74
- sGetEthernetLinksStatesPort, 78
- ulStatus
  - MSXE371x\_\_-
    - CalibrationGetCurrentStatusResponse, 63
- ulTemperatureStatus
  - MXCommon\_\_-
    - GetModuleTemperatureValueAndStatusResponse, 75
- ulTimeStampHigh
  - MSXE371x\_\_-
    - ExternalTemperatureReadResponse, 66
- MSXE371x\_\_-
  - IncCounterRead32BitValueResponse, 66
- MSXE371x\_\_-
  - TransducerGetAllValuesResponse, 68
- ulTimeStampLow
  - MSXE371x\_\_-
    - ExternalTemperatureReadResponse, 66
- MSXE371x\_\_-
  - IncCounterRead32BitValueResponse, 66
- MSXE371x\_\_-
  - TransducerGetAllValuesResponse, 68
- ulTransducerSelectionIndex
  - MSXE371x\_\_-
    - TransducerGetTypeInformationResponse, 69
- ulTransducersValue
  - MSXE371x\_\_-
    - TransducerGetAllValuesResponse, 68
- ulType
  - MSXE371x\_\_-
    - DataBaseGetTransducerInformationResponse, 65
  - MSXE371x\_\_-
    - TransducerGetTypeInformationResponse, 69
- ulUpTime
  - MXCommon\_\_GetUpTimeResponse, 76
- ulValue
  - MSXE371x\_\_-
    - ExternalTemperatureReadResponse, 66
  - MSXE371x\_\_-
    - IncCounterRead32BitValueResponse, 66
  - MSXE371x\_\_unsignedLong8FixedArray, 70

- MSXE371x\_\_unsignedlong9ArrayResponse,  
70
- MSXE371x\_\_unsignedlongDefaultResponse,  
70
- MSXE371x\_\_unsignedlongResponse, 71
- MXCommon\_\_unsignedLongResponse, 77
- UnsignedLongArray, 78
  - \_\_offset, 78
  - \_\_ptr, 78
  - \_\_size, 78
- UnsignedShortArray, 78
  - \_\_offset, 79
  - \_\_ptr, 79
  - \_\_size, 79
- xsd\_\_base64Binary, 79
  - \_\_ptr, 79
  - \_\_size, 79
- xsd\_\_char
  - MSXE371x\_public\_doc.h, 90
- xsd\_\_double
  - MSXE371x\_public\_doc.h, 90
- xsd\_\_float
  - MSXE371x\_public\_doc.h, 90
- xsd\_\_int
  - MSXE371x\_public\_doc.h, 90
- xsd\_\_long
  - MSXE371x\_public\_doc.h, 90
- xsd\_\_string
  - MSXE371x\_public\_doc.h, 90
- xsd\_\_unsignedByte
  - MSXE371x\_public\_doc.h, 90
- xsd\_\_unsignedInt
  - MSXE371x\_public\_doc.h, 90
- xsd\_\_unsignedLong
  - MSXE371x\_public\_doc.h, 90
- xsd\_\_unsignedShort
  - MSXE371x\_public\_doc.h, 90