MODBUS interface description

Table of Contents

Table of Contents

FC3 (read multiple register) Functions

Table of Contents

FC3 (read multiple register) Functions

Table of Contents

FC16 (write multiple register) Functions

Table of Contents

FC16 (write multiple register) Functions

Table of Contents

General description

Top

Introduction

This document describes the protocol used by the MODBUS server of the module.
The OPEN MODBUS protocol is based on the widely known MODBUS protocol.
OPEN MODBUS is an open protocol and is not manufacturer dependent.
It is mainly used to connect PLC and I/O devices.

Why a MODBUS Server on the MSX-E modules?

Thanks to the MODBUS server, it is possible to manage an MSX-E module with e.g.: a Siemens S7 PLC.
The S7 PLC can start acquisitions and read data from the MSX-E module!

Technical details

Please note that only MODBUS over TCP is standardized. Nonetheless in this present version the server implements OPEN MODBUS/TCP class 0 and one function of the class 2 even on UDP sockets.

The MODBUS/TCP class 0 defines two types of query: FC3 and FC16.

- **FC3 functions** read register content from the memory of the remote system
- **FC16 functions** write new register content on the memory of the remote system

The MODBUS/TCP server implement the following query of the class 2 : FC23.

- **FC23 functions** read/write registers content from/to the memory of the remote system

The MODBUS server offer a virtual memory organisation: registers (functions)
are mapped to be equivalent to SOAP functions.

Characteristics of this communication channel as the standardisation document describes it are:

- The default port used by the server is **512** in both UDP/IP and TCP/IP. You can change this via the web server.
- Data are sent in network order, i.e. **big endian (Motorola formata)**. Use the standard C functions atons/atonl and ntohs/ntohl to convert values bigger than 1 bytes.
- Datastructures used to describe parameters that are embedded in on-wire frames **must** be packed. How to do that is compiler-dependant.

The ADDI-DATA MSX-E Modbus server offers the following extension to the standard:

- It is possible to configure the server to accept data sent in **little endian (Intel format)** (native order)
- In this case, the default port used is **215**. You can change this via the web server.

As answer to query a client may receive an acknowledgement (named *standard response* onward) or an exception.
If an exception or an error occured, you can use the GetLastCommandStatus command to get the real error number (from the remote server).
Real error numbers are described for each command in the "Returns" field.

The chapter below describes the available functions and their parameters.
It also contains the precise description of all frames implied in a given action.

FC3 (read multiple register) Functions

Top

Functions in this group are used to read values on the module.

GetLastCommandStatus	Register: **0**
GetLastCommandStatusEx	Register: **10000**
MXCommon__GetModuleType	Register: **1**
MXCommon__GetModuleTypeEx	Register: **10200**
MXCommon__GetTime	Register: **2**
MXCommon__GetTimeEx	Register: **10500**
MXCommon__TestCustomerID	Register: **3**
MXCommon__TestCustomerIDEx	Register: **10550**
MX370x__getNumberOfChannels	Register: **100**
MX370x__getNumberOfChannelsEx	Register: **1000**
MX370x__TransducerGetAutoRefreshValues	Register: **101**
MX370x__TransducerGetAutoRefreshValuesEx	Register: **1050**
MX370x__TransducerGetNbrOfType	Register: **102**
MX370x__TransducerGetNbrOfTypeEx	Register: **1594**
MX370x__GetTransducerDatabaseCursor	Register: **103**
MX370x__GetTransducerDatabaseCursorEx	Register: **1598**
MX370x__TransducerGetTypeInformation	Register: **104**
MX370x__TransducerGetTypeInformationEx	Register: **1602**

Function GetLastCommandStatus

For new application(s) or automate communication it is recommended to use the function GetLastCommandStatusEx.

Description

Return the result of the last remote function call

Parameters:

[Response frame layout] ***ReturnValue:*** The return value of the remote function.

- 0 Always means success
- -100 means you should check Syserrno;
- for other values, check the documentation of the function

[Response frame layout] ***Syserrno:*** the value of the libc errno after the call to the remote function

[Response frame layout] ***Errstr:*** A nul-terminated string describing the error code Syserrno

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Reference number (=register)	2	16-bit integer	0	0x0000	0x0000
word count	2	16-bit integer	54	0x3600	0x0036

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	112	0x7000	0x0070
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	2	16-bit integer	108	0x6C00	0x006C
ReturnValue	4	32-bit integer	See the description above	0x????????	0x????????
Syserrno	4	32-bit integer	See the description above	0x????????	0x????????
Errstr	100	8-bit integer array	See the description above	0x??[100]	0x??[100]

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x83	0x83	0x83

Exception code	1	8-bit integer	See corresponding chapter	??	??

Function GetLastCommandStatusEx

Description

Return the result of the last remote function call

Parameters:

[Response frame layout] ***ReturnValue:*** The return value of the remote function.

- 0 Always means success
- -100 means you should check Syserrno;
- for other values, check the documentation of the function

[Response frame layout] ***Syserrno:*** the value of the libc errno after the call to the remote function

[Response frame layout] ***Errstr:*** A nul-terminated string describing the error code Syserrno

Query frame layout

Field	**Size (Bytes)**	**Type**	**Value**	**little endian (Intel)**	**big endian (Motorola)**
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Reference number (=register)	2	16-bit integer	10000	0x1027	0x2710
word count	2	16-bit integer	54	0x3600	0x0036

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	111	0x6F00	0x006F
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	1	8-bit integer	108	0x6C	0x6C
ReturnValue	4	32-bit integer	See the description above	0x????????	0x????????
Syserrno	4	32-bit integer	See the description above	0x????????	0x????????
Errstr	100	8-bit integer array	See the description above	0x??[100]	0x??[100]

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x83	0x83	0x83

Exception code	1	8-bit integer	See corresponding chapter	??	??

Function MXCommon__GetModuleType

For new application(s) or automate communication it is recommended to use the function MXCommon__GetModuleTypeEx.

Description

Returns the type of the MSX-E Module

Parameters:

[Response frame layout] ***str:*** A 200-characters string

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Reference number (=register)	2	16-bit integer	1	0x0100	0x0001
word count	2	16-bit integer	100	0x6400	0x0064

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)

transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	204	0xCC00	0x00CC
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	2	16-bit integer	200	0xC800	0x00C8
str	200	8-bit integer array	See the description above	0x??[200]	0x??[200]

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x83	0x83	0x83
Exception code	1	8-bit integer	See corresponding chapter	??	??

Function MXCommon__GetModuleTypeEx

Description

Returns the type of the MSX-E Module

Parameters:

[Response frame layout] ***str:*** A 200-characters string

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Reference number (=register)	2	16-bit integer	10200	0xD827	0x27D8
word count	2	16-bit integer	100	0x6400	0x0064

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	203	0xCB00	0x00CB
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	1	8-bit integer	200	0xC8	0xC8

str	200	8-bit integer array	See the description above	0x??[200]	0x??[200]

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x83	0x83	0x83
Exception code	1	8-bit integer	See corresponding chapter	??	??

Function MXCommon__GetTime

For new application(s) or automate communication it is recommended to use the function MXCommon__GetTimeEx.

Description

Get the time on the module

Parameters:

[Response frame layout] ***tv_sec:*** Number of seconds since the Epoch

[Response frame layout] ***tv_usec:*** Number of microseconds since the begin of the second

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied	0x0000	0x0000

			by server - usually 0		
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Reference number (=register)	2	16-bit integer	2	0x0200	0x0002
word count	2	16-bit integer	4	0x0400	0x0004

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	12	0x0C00	0x000C
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	2	16-bit integer	8	0x0800	0x0008
tv_sec	4	32-bit integer	See the description above	0x????????	0x????????
tv_usec	4	32-bit integer	See the description above	0x????????	0x????????

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x83	0x83	0x83
Exception code	1	8-bit integer	See corresponding chapter	??	??

Function MXCommon__GetTimeEx

Description

Get the time on the module

Parameters:

[Response frame layout] ***tv_sec:*** Number of seconds since the Epoch

[Response frame layout] ***tv_usec:*** Number of microseconds since the begin of the second

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000

length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Reference number (=register)	2	16-bit integer	10500	0x0429	0x2904
word count	2	16-bit integer	4	0x0400	0x0004

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	11	0x0B00	0x000B
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	1	8-bit integer	8	0x08	0x08
tv_sec	4	32-bit integer	See the description above	0x????????	0x????????
tv_usec	4	32-bit integer	See the description above	0x????????	0x????????

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000

protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x83	0x83	0x83
Exception code	1	8-bit integer	See corresponding chapter	??	??

Function MXCommon__TestCustomerID

For new application(s) or automate communication it is recommended to use the function MXCommon__TestCustomerIDEx.

Description

Permit to test the Customer ID (if the module has the right customer Key)

Parameters:

> [Response frame layout] ***bValueArray:*** non crypted value array [16 bytes of random data]
>
> [Response frame layout] ***bCryptedValueArray:*** Crypted value array [16 bytes of the crypted random data]

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
	1		0x03	0x03	0x03

MODBUS Function code		8-bit integer			
Reference number (=register)	2	16-bit integer	3	0x0300	0x0003
word count	2	16-bit integer	16	0x1000	0x0010

Response frame layout

Field	**Size (Bytes)**	**Type**	**Value**	**little endian (Intel)**	**big endian (Motorola)**
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	36	0x2400	0x0024
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	2	16-bit integer	32	0x2000	0x0020
bValueArray	16	8-bit integer array	See the description above	0x??[16]	0x??[16]
bCryptedValueArray	16	8-bit integer array	See the description above	0x??[16]	0x??[16]

Exception frame layout

Field	**Size (Bytes)**	**Type**	**Value**	**little endian (Intel)**	**big endian (Motorola)**
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
	1		0 or 1		

unit identifier		8-bit integer		0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x83	0x83	0x83
Exception code	1	8-bit integer	See corresponding chapter	??	??

Function MXCommon__TestCustomerIDEx

Description

Permit to test the Customer ID (if the module has the right customer Key)

Parameters:

> [Response frame layout] ***bValueArray:*** non crypted value array [16 bytes of random data]
>
> [Response frame layout] ***bCryptedValueArray:*** Crypted value array [16 bytes of the crypted random data]

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Reference number (=register)	2	16-bit integer	10550	0x3629	0x2936
word count	2	16-bit integer	16	0x1000	0x0010

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	35	0x2300	0x0023
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	1	8-bit integer	32	0x20	0x20
bValueArray	16	8-bit integer array	See the description above	0x??[16]	0x??[16]
bCryptedValueArray	16	8-bit integer array	See the description above	0x??[16]	0x??[16]

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x83	0x83	0x83
Exception code	1	8-bit integer	See corresponding chapter	??	??

Function MX370x__getNumberOfChannels

For new application(s) or automate communication it is recommended to use the function MX370x__getNumberOfChannelsEx.

Description

Return the number of transducer channels on the module (4,8 or 16)

Parameters:

[Response frame layout]***ChannelNumber:*** Number of channels

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Reference number (=register)	2	16-bit integer	100	0x6400	0x0064
word count	2	16-bit integer	2	0x0200	0x0002

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server -	0x0000	0x0000

			usually 0		
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	8	0x0800	0x0008
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	2	16-bit integer	4	0x0400	0x0004
ChannelNumber	4	32-bit integer	See the description above	0x????????	0x????????

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x83	0x83	0x83
Exception code	1	8-bit integer	See corresponding chapter	??	??

Function MX370x__getNumberOfChannelsEx

Description

Return the number of transducer channels on the module (4,8 or 16)

Parameters:

[Response frame layout]***ChannelNumber:*** Number of channels

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Reference number (=register)	2	16-bit integer	1000	0xE803	0x03E8
word count	2	16-bit integer	2	0x0200	0x0002

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	7	0x0700	0x0007
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	1	8-bit integer	4	0x04	0x04
ChannelNumber	4	32-bit integer	See the description	0x????????	0x????????

			above		

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x83	0x83	0x83
Exception code	1	8-bit integer	See corresponding chapter	??	??

Function MX370x__TransducerGetAutoRefreshValues

For new application(s) or automate communication it is recommended to use the function MX370x__TransducerGetAutoRefreshValuesEx.

Description

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

Parameters:

[Response frame layout]***Value:*** Array that contain the counter and channels values (raw or converted, depending of the configuration)

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

Returns:

Possible return value on the remote system (read them with GetLastCommandStatus)

- 0 : success
- -100 : GetAutoRefreshAllValues kernel function error

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Reference number (=register)	2	16-bit integer	101	0x6500	0x0065
word count	2	16-bit integer	34	0x2200	0x0022

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	72	0x4800	0x0048
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	2	16-bit integer	68	0x4400	0x0044
Value	68	32-bit integer	See the description	0x????????[17]	0x????????[17]

		array	above		

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x83	0x83	0x83
Exception code	1	8-bit integer	See corresponding chapter	??	??

Function MX370x__TransducerGetAutoRefreshValuesEx

Description

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

Parameters:

[Response frame layout]***Value:*** Array that contain the counter and channels values (raw or converted, depending of the configuration)

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

Returns:

Possible return value on the remote system (read them with GetLastCommandStatusEx)

- 0 : success
- -100 : GetAutoRefreshAllValues kernel function error

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Reference number (=register)	2	16-bit integer	1050	0x1A04	0x041A
word count	2	16-bit integer	34	0x2200	0x0022

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	71	0x4700	0x0047
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	1	8-bit integer	68	0x44	0x44
Value	68	32-bit integer	See the description	0x????????[17]	0x????????[17]

		array	above		

Exception frame layout

Field	**Size (Bytes)**	**Type**	**Value**	**little endian (Intel)**	**big endian (Motorola)**
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x83	0x83	0x83
Exception code	1	8-bit integer	See corresponding chapter	??	??

Function MX370x__TransducerGetNbrOfType

For new application(s) or automate communication it is recommended to use the function MX370x__TransducerGetNbrOfTypeEx.

Description

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

Parameters:

[Query frame layout] ***NumberOfTransducerTypes:*** number of transducer types currently defined.

Returns:

Possible return value on the remote system (read them with GetLastCommandStatus)

- 0 : success
- otherwise : internal error

Query frame layout

Field	**Size (Bytes)**	**Type**	**Value**	**little endian (Intel)**	**big endian (Motorola)**

transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Reference number (=register)	2	16-bit integer	102	0x6600	0x0066
word count	2	16-bit integer	2	0x0200	0x0002

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	8	0x0800	0x0008
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	2	16-bit integer	4	0x0400	0x0004
NumberOfTransducerTypes	4	32-bit integer	See the description above	0x????????	0x????????

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian	big endian (Motorola)

				(Intel)	
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x83	0x83	0x83
Exception code	1	8-bit integer	See corresponding chapter	??	??

Function MX370x__TransducerGetNbrOfTypeEx

Description

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

Parameters:

[Query frame layout] ***NumberOfTransducerTypes:*** number of transducer types currently defined.

Returns:

Possible return value on the remote system (read them with GetLastCommandStatusEx)

- 0 : success
- otherwise : internal error

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000

length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Reference number (=register)	2	16-bit integer	1594	0x3A06	0x063A
word count	2	16-bit integer	2	0x0200	0x0002

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	7	0x0700	0x0007
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	1	8-bit integer	4	0x04	0x04
NumberOfTransducerTypes	4	32-bit integer	See the description above	0x????????	0x????????

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003

unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x83	0x83	0x83
Exception code	1	8-bit integer	See corresponding chapter	??	??

Function MX370x__GetTransducerDatabaseCursor

For new application(s) or automate communication it is recommended to use the function MX370x__GetTransducerDatabaseCursorEx.

Description

Returns the current cursor of the transducer database.

Parameters:

[Query frame layout] ***TransducerDatabaseCursor:*** Current cursor. This is an integer from 0 .. (NumberOfTransducerTypes-1)

Returns:

Possible return value on the remote system (read them with GetLastCommandStatus)

- 0 : success
- otherwise : internal error

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01

MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Reference number (=register)	2	16-bit integer	103	0x6700	0x0067
word count	2	16-bit integer	2	0x0200	0x0002

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	8	0x0800	0x0008
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	2	16-bit integer	4	0x0400	0x0004
TransducerDatabaseCursor	4	32-bit integer	See the description above	0x????????	0x????????

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function	1	8-bit integer	0x83	0x83	0x83

code					
Exception code	1	8-bit integer	See corresponding chapter	??	??

Function MX370x__GetTransducerDatabaseCursorEx

Description

Returns the current cursor of the transducer database.

Parameters:

[Query frame layout] ***TransducerDatabaseCursor:*** Current cursor. This is an integer from 0 .. (NumberOfTransducerTypes-1)

Returns:

Possible return value on the remote system (read them with GetLastCommandStatusEx)

- 0 : success
- otherwise : internal error

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Reference number (=register)	2	16-bit integer	1598	0x3E06	0x063E
word count	2	16-bit integer	2	0x0200	0x0002

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	7	0x0700	0x0007
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	1	8-bit integer	4	0x04	0x04
TransducerDatabaseCursor	4	32-bit integer	See the description above	0x????????	0x????????

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x83	0x83	0x83
Exception code	1	8-bit integer	See corresponding chapter	??	??

Function MX370x__TransducerGetTypeInformation

For new application(s) or automate communication it is recommended to use the function MX370x__TransducerGetTypeInformationEx.

Description

Returns the information stored in the database about the type selected by the current TransducerDatabaseCursor.

Parameters:

SelectionIndex : Identifier. Value to use for the transducer type selection in the other SOAP functions.
Name : Name of the transducer type
CalibrationStatus : Calibration status \li 0 : Transducer type is not calibrated
\li 1 : Transducer type is calibrated
Type : Type (0: HB 1: LVDT 2:Knaebel 3:HB-Mahr 4:LVDT-Mahr) ***Frequency :*** Frequency (Hz)
Impedance : Impedance (Ohm)
Veff : Nominal voltage (Vrms)
Sensibility : Sensibility (mv/V/mm)
Range : Range (mm)

Returns:

Possible return value on the remote system (read them with GetLastCommandStatus)

- 0 : success
- otherwise : internal error

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
	2		104	0x6800	0x0068

Reference number (=register)		16-bit integer			
word count	2	16-bit integer	65	0x4100	0x0041

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	133	0x8500	0x0085
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	2	16-bit integer	129	0x8100	0x0081
SelectionIndex	4	32-bit integer	See the description above	0x????????	0x????????
Name	100	8-bit integer array	See the description above	0x??[100]	0x??[100]
CalibrationStatus	1	8-bit integer	See the description above	0x??	0x??
Type	4	32-bit integer	See the description above	0x????????	0x????????
Frequency	4	32-bit integer	See the description above	0x????????	0x????????
Impedance	4	32-bit integer	See the description above	0x????????	0x????????
NominalVoltage	4	32-bit floating point	See the description above	0x????????	0x????????
Sensibility	4	32-bit floating point	See the description above	0x????????	0x????????

Range	4	32-bit floating point	See the description above	0x????????	0x????????

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x83	0x83	0x83
Exception code	1	8-bit integer	See corresponding chapter	??	??

Function MX370x__TransducerGetTypeInformationEx

Description

Returns the information stored in the database about the type selected by the current TransducerDatabaseCursor.

Parameters:

SelectionIndex : Identifier. Value to use for the transducer type selection in the other SOAP functions.
Name : Name of the transducer type
CalibrationStatus : Calibration status \li 0 : Transducer type is not calibrated
\li 1 : Transducer type is calibrated
Type : Type (0: HB 1: LVDT 2:Knaebel 3:HB-Mahr 4:LVDT-Mahr) ***Frequency :*** Frequency (Hz)
Impedance : Impedance (Ohm)
Veff : Nominal voltage (Vrms)
Sensibility : Sensibility (mv/V/mm)
Range : Range (mm)

Returns:

Possible return value on the remote system (read them with GetLastCommandStatusEx)

- 0 : success

- otherwise : internal error

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Reference number (=register)	2	16-bit integer	1602	0x4206	0x0642
word count	2	16-bit integer	65	0x4100	0x0041

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	132	0x8400	0x0084
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
Byte count	1	8-bit integer	129	0x81	0x81

SelectionIndex	4	32-bit integer	See the description above	0x????????	0x????????
Name	100	8-bit integer array	See the description above	0x??[100]	0x??[100]
CalibrationStatus	1	8-bit integer	See the description above	0x??	0x??
Type	4	32-bit integer	See the description above	0x????????	0x????????
Frequency	4	32-bit integer	See the description above	0x????????	0x????????
Impedance	4	32-bit integer	See the description above	0x????????	0x????????
NominalVoltage	4	32-bit floating point	See the description above	0x????????	0x????????
Sensibility	4	32-bit floating point	See the description above	0x????????	0x????????
Range	4	32-bit floating point	See the description above	0x????????	0x????????

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x83	0x83	0x83
Exception code	1	8-bit integer	See corresponding chapter	??	??

FC16 (write multiple register) Functions

Top

Functions in this group are used to set value on the module.

MXCommon__SetHardwareTriggerFilterTime	Register: **100**
MXCommon__SetHardwareTriggerFilterTimeEx	Register: **11000**
MXCommon__InitAndStartSynchroTimer	Register: **101**
MXCommon__InitAndStartSynchroTimerEx	Register: **11050**
MXCommon__StopAndReleaseSynchroTimer	Register: **102**
MXCommon__StopAndReleaseSynchroTimerEx	Register: **11100**
MXCommon__Reboot	Register: **103**
MXCommon__RebootEx	Register: **11150**
MXCommon__SetCustomerKey	Register: **104**
MXCommon__SetCustomerKeyEx	Register: **11200**
MXCommon__SetFilterChannels	Register: **105**
MXCommon__SetFilterChannelsEx	Register: **11250**
MX370x__TransducerInitAndStartAutoRefresh	Register: **1**
MX370x__TransducerInitAndStartAutoRefreshEx	Register: **1200**
MX370x__TransducerStopAndReleaseAutoRefresh	Register: **2**
MX370x__TransducerStopAndReleaseAutoRefreshEx	Register: **1250**
MX370x__TransducerInitAndStartSequence	Register: **3**
MX370x__TransducerInitAndStartSequenceEx	Register: **1300**
MX370x__TransducerStopAndReleaseSequence	Register: **4**
MX370x__TransducerStopAndReleaseSequenceEx	Register: **1350**

- MX370x__SetTransducerDatabaseCursor Register: **5**
- MX370x__SetTransducerDatabaseCursorEx Register: **1354**
- MX370x__TransducerSetOffset Register: **6**
- MX370x__TransducerSetOffsetEx Register: **1356**

Function MXCommon__SetHardwareTriggerFilterTime

For new application(s) or automate communication it is recommended to use the function MXCommon__SetHardwareTriggerFilterTimeEx.

Description

Sets 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

- [Query frame layout] ***ulFilterTime*** Filter time for the hardware trigger input in 250ns step (max value : 65535).
    - **0**: disable the filter
    - **1**: filter of 250ns
    - **2**: filter of 500ns
    - ...
    - **65535**: filter of 16ms
- [Query frame layout] ***ulOption*** Reserved. Set to 0

Returns

Possible return value on the remote system (read them with GetLastCommandStatus).

- **0** The remote function performed OK
- **-1** Internal system error occurred. See value of syserrno

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
	2		0	0x0000	0x0000

protocol identifier		16-bit integer			
length	2	16-bit integer	16	0x1000	0x0010
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	100	0x6400	0x0064
word count	2	16-bit integer	4	0x0400	0x0004
byte count	2	16-bit integer	8	0x0800	0x0008
ulFilterTime	4	32-bit integer	See the description above	0x????????	0x????????
Reserved	4	32-bit integer	See the description above	0x????????	0x????????

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	100	0x6400	0x0064
word count	2	16-bit integer	4	0x0400	0x0004

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MXCommon__SetHardwareTriggerFilterTimeEx

Description

Sets 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

- [Query frame layout] ***ulFilterTime*** Filter time for the hardware trigger input in 250ns step (max value : 65535).
    - **0**: disable the filter
    - **1**: filter of 250ns
    - **2**: filter of 500ns
    - ...
    - **65535**: filter of 16ms
- [Query frame layout] ***ulOption*** Reserved. Set to 0

Returns

Possible return value on the remote system (read them with GetLastCommandStatusEx).

- **0** The remote function performed OK
- **-1** Internal system error occurred. See value of syserrno

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	15	0x0F00	0x000F
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	11000	0xF82A	0x2AF8
word count	2	16-bit integer	4	0x0400	0x0004
byte count	1	8-bit integer	8	0x08	0x08
ulFilterTime	4	32-bit integer	See the description above	0x????????	0x????????
Reserved	4	32-bit integer	See the description above	0x????????	0x????????

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006

unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	11000	0xF82A	0x2AF8
word count	2	16-bit integer	4	0x0400	0x0004

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MXCommon__InitAndStartSynchroTimer

For new application(s) or automate communication it is recommended to use the function MXCommon__InitAndStartSynchroTimerEx.

Description

Init and start the synchronisation timer of the module (not already available on all module)

Parameters:

[Query frame layout] ***ulTimeBase:*** Time base of the timer (0 for us, 1 for ms, 2 for s)

[Query frame layout] ***ulReloadValue:*** Timer reload value (0 to 0xFFFF), minimum reload time is 5 us

[Query frame layout] ***ulNbrOfCycle:*** Number of timer cycle

- 0: continuous
- > 0: defined number of cycle

[Query frame layout] ***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

[Query frame layout] ***ulOption01:***Define the source of the trigger

- 0 : Trigger disabled
- 1 : Enable the hardware figital input trigger

[Query frame layout] ***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)

[Query frame layout] ***ulOption03:***Define the number of trigger events before the action occur

- 1 : all trigger event start the action
- max value : 65535

[Query frame layout] ***ulOption04:*** Reserved

Returns:

Possible return value on the remote system (read them with GetLastCommandStatus)

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

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000

length	2	16-bit integer	40	0x2800	0x0028
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	101	0x6500	0x0065
word count	2	16-bit integer	16	0x1000	0x0010
byte count	2	16-bit integer	32	0x2000	0x0020
ulTimeBase	4	32-bit integer	See the description above	0x????????	0x????????
ulReloadValue	4	32-bit integer	See the description above	0x????????	0x????????
ulNbrOfCycle	4	32-bit integer	See the description above	0x????????	0x????????
ulGenerateTriggerMode	4	32-bit integer	See the description above	0x????????	0x????????
ulOption01	4	32-bit integer	See the description above	0x????????	0x????????
ulOption02	4	32-bit integer	See the description above	0x????????	0x????????
ulOption03	4	32-bit integer	See the description above	0x????????	0x????????
ulOption04	4	32-bit integer	See the description above	0x????????	0x????????

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
	2		0	0x0000	0x0000

protocol identifier		16-bit integer			
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	101	0x6500	0x0065
word count	2	16-bit integer	16	0x1000	0x0010

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MXCommon__InitAndStartSynchroTimerEx

Description

Init and start the synchronisation timer of the module (not already available on all module)

Parameters:

[Query frame layout] ***ulTimeBase:*** Time base of the timer (0 for us, 1 for ms, 2 for s)

[Query frame layout] ***ulReloadValue:*** Timer reload value (0 to 0xFFFF), minimum reload time is 5 us

[Query frame layout] ***ulNbrOfCycle:*** Number of timer cycle

- 0: continuous
- > 0: defined number of cycle

[Query frame layout] ***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

[Query frame layout] ***ulOption01:***Define the source of the trigger

- 0 : Trigger disabled
- 1 : Enable the hardware figital input trigger

[Query frame layout] ***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)

[Query frame layout] ***ulOption03:***Define the number of trigger events before the action occur

- 1 : all trigger event start the action
- max value : 65535

[Query frame layout] ***ulOption04:*** Reserved

Returns:

Possible return value on the remote system (read them with GetLastCommandStatusEx)

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

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000

length	2	16-bit integer	39	0x2700	0x0027
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	11050	0x2A2B	0x2B2A
word count	2	16-bit integer	16	0x1000	0x0010
byte count	1	8-bit integer	32	0x20	0x20
ulTimeBase	4	32-bit integer	See the description above	0x????????	0x????????
ulReloadValue	4	32-bit integer	See the description above	0x????????	0x????????
ulNbrOfCycle	4	32-bit integer	See the description above	0x????????	0x????????
ulGenerateTriggerMode	4	32-bit integer	See the description above	0x????????	0x????????
ulOption01	4	32-bit integer	See the description above	0x????????	0x????????
ulOption02	4	32-bit integer	See the description above	0x????????	0x????????
ulOption03	4	32-bit integer	See the description above	0x????????	0x????????
ulOption04	4	32-bit integer	See the description above	0x????????	0x????????

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
	2		0	0x0000	0x0000

protocol identifier		16-bit integer			
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	11050	0x2A2B	0x2B2A
word count	2	16-bit integer	16	0x1000	0x0010

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MXCommon__StopAndReleaseSynchroTimer

For new application(s) or automate communication it is recommended to use the function MXCommon__StopAndReleaseSynchroTimerEx.

Description

stop the synchronisation timer (not already available on all module)

Parameters:

[Query frame layout] ***ulOption01*** : Reserved

Response frame layout

Returns:

Possible return value on the remote system (read them with GetLastCommandStatus)

- 0 : means the remote function performed OK
- -1: means an system error occured
- -100: Start/Stop timer error

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	12	0x0C00	0x000C
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	102	0x6600	0x0066
word count	2	16-bit integer	2	0x0200	0x0002
byte count	2	16-bit integer	4	0x0400	0x0004
ulOption01	4	32-bit integer	See the description above	0x????????	0x????????

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000

protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	102	0x6600	0x0066
word count	2	16-bit integer	2	0x0200	0x0002

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MXCommon__StopAndReleaseSynchroTimerEx

Description

stop the synchronisation timer (not already available on all module)

Parameters:

[Query frame layout] ***ulOption01*** : Reserved

Returns:

Possible return value on the remote system (read them with GetLastCommandStatusEx)

- 0 : means the remote function performed OK
- -1: means an system error occured
- -100: Start/Stop timer error

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	11	0x0B00	0x000B
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	11100	0x5C2B	0x2B5C
word count	2	16-bit integer	2	0x0200	0x0002
byte count	1	8-bit integer	4	0x04	0x04
ulOption01	4	32-bit integer	See the description above	0x????????	0x????????

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000

length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	11100	0x5C2B	0x2B5C
word count	2	16-bit integer	2	0x0200	0x0002

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MXCommon__Reboot

For new application(s) or automate communication it is recommended to use the function MXCommon__RebootEx.

Description

Ask the MSX-E module to reboot

Parameters:

[Query frame layout] ***Dummy*** : Reserved

Returns:

Response frame layout

Possible return value on the remote system (read them with GetLastCommandStatus)

- 0 : means the remote function performed OK
- -1: means an system error occured (probably EPERM)

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	12	0x0C00	0x000C
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	103	0x6700	0x0067
word count	2	16-bit integer	2	0x0200	0x0002
byte count	2	16-bit integer	4	0x0400	0x0004
Dummy	4	32-bit integer	See the description above	0x????????	0x????????

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2		6	0x0600	0x0006

		16-bit integer			
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	103	0x6700	0x0067
word count	2	16-bit integer	2	0x0200	0x0002

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MXCommon__RebootEx

Description

Ask the MSX-E module to reboot

Parameters:

[Query frame layout] ***Dummy*** : Reserved

Returns:

Possible return value on the remote system (read them with GetLastCommandStatusEx)

- 0 : means the remote function performed OK
- -1: means an system error occured (probably EPERM)

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	11	0x0B00	0x000B
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	11150	0x8E2B	0x2B8E
word count	2	16-bit integer	2	0x0200	0x0002
byte count	1	8-bit integer	4	0x04	0x04
Dummy	4	32-bit integer	See the description above	0x????????	0x????????

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
	1		0 or 1		

unit identifier		8-bit integer		0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	11150	0x8E2B	0x2B8E
word count	2	16-bit integer	2	0x0200	0x0002

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MXCommon__SetCustomerKey

For new application(s) or automate communication it is recommended to use the function MXCommon__SetCustomerKeyEx.

Description

Permit to set the Customer key

Parameters:

> [Query frame layout] ***bKey*** : Customer key (only writable on the module) [32 bytes containing a AES key]
>
> [Query frame layout] ***bPublicKey*** : IV (Initialisation vector) for the AES cryptography [16 bytes containing a AES key]

Returns:

Possible return value on the remote system (read them with GetLastCommandStatus)

- 0 : means the remote function performed OK
- -1: means an system error occured (probably EPERM)

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	56	0x3800	0x0038
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	104	0x6800	0x0068
word count	2	16-bit integer	24	0x1800	0x0018
byte count	2	16-bit integer	48	0x3000	0x0030
bKey	32	8-bit integer array	See the description above	0x??[32]	0x??[32]
bPublicKey	16	8-bit integer array	See the description above	0x??[16]	0x??[16]

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server -	0x0000	0x0000

			usually 0		
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	104	0x6800	0x0068
word count	2	16-bit integer	24	0x1800	0x0018

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MXCommon__SetCustomerKeyEx

Description

Permit to set the Customer key

Parameters:

> [Query frame layout] ***bKey*** : Customer key (only writable on the module) [32 bytes containing a AES key]

[Query frame layout] ***bPublicKey*** : IV (Initialisation vector) for the AES cryptography [16 bytes containing a AES key]

Returns:

Possible return value on the remote system (read them with GetLastCommandStatusEx)

- 0 : means the remote function performed OK
- -1: means an system error occured (probably EPERM)

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	55	0x3700	0x0037
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	11200	0xC02B	0x2BC0
word count	2	16-bit integer	24	0x1800	0x0018
byte count	1	8-bit integer	48	0x30	0x30
bKey	32	8-bit integer array	See the description above	0x??[32]	0x??[32]
bPublicKey	16	8-bit integer array	See the description above	0x??[16]	0x??[16]

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined	0x0000	0x0000

			- copied by server - usually 0		
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	11200	0xC02B	0x2BC0
word count	2	16-bit integer	24	0x1800	0x0018

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MXCommon__SetFilterChannels

For new application(s) or automate communication it is recommended to use the function MXCommon__SetFilterChannelsEx.

Description

Permit to set a filter per channel

Parameters:

[Query frame layout] ***ChannelList*** : Each index of the array is representing a channel. To set a filter on a channel, enter the filter ID. By default the value ist 0 (No filter).

Returns:

Possible return value on the remote system (read them with GetLastCommandStatus)

- 0 : means the remote function performed OK
- -1: means a system error occurred (probably EPERM)

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	24	0x1800	0x0018
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	105	0x6900	0x0069
word count	2	16-bit integer	8	0x0800	0x0008
byte count	2	16-bit integer	16	0x1000	0x0010
ChannelList	16	8-bit integer array	See the description above	0x??[16]	0x??[16]

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	105	0x6900	0x0069
word count	2	16-bit integer	8	0x0800	0x0008

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MXCommon__SetFilterChannelsEx

Description

Permit to set a filter per channel

Parameters:

> [Query frame layout] ***ChannelList*** : Each index of the array is representing a channel. To set a filter on a channel, enter the filter ID. By default the value ist 0 (No filter).

Returns:

> **Possible return value on the remote system (read them with GetLastCommandStatusEx)**
>
> - 0 : means the remote function performed OK
> - -1: means a system error occurred (probably EPERM)

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	23	0x1700	0x0017
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	11250	0xF22B	0x2BF2
word count	2	16-bit integer	8	0x0800	0x0008
byte count	1	8-bit integer	16	0x10	0x10
ChannelList	16	8-bit integer array	See the description above	0x??[16]	0x??[16]

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	11250	0xF22B	0x2BF2
word count	2	16-bit integer	8	0x0800	0x0008

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MX370x__TransducerInitAndStartAutoRefresh

For new application(s) or automate communication it is recommended to use the function MX370x__TransducerInitAndStartAutoRefreshEx.

Description

Initialise and start the transducer auto refresh acquisition mode

Parameters:

[Query frame layout] ***TransducerSelection:*** Transducer type selection

[Query frame layout] ***ChannelMask:*** Mask of the channel to acquire by the auto refresh (1 bit = 1 Channel)

[Query frame layout] ***AverageMode:*** Set the average mode :

- ♦ 0: not used
- ♦ 1: average per Sequence
- ♦ 2: average per channel

[Query frame layout] **AverageValue:** Set the average value (only used, when average is used)

- ♦ 0: average not used
- ♦ max value: 255

[Query frame layout] **DivisionFactor:** Division factor (min : 5, max 255)

[Query frame layout] **TriggerAction:** Trigger action :

- ♦ ***Hardware Trigger Start D0 - D7***

  Bit 3,2,1,0 : Define the trigger mode
  - ◊ 0000 : Trigger disabled
  - ◊ 0001 : One shot trigger : After the software start, the module is waiting for a trigger signal to start the acquisition. After this the trigger signal is ignored.
  - ◊ 0010 : Sequence trigger : After the software start the module is waiting for the trigger signal and acquires x sequences (also adjustable) and then wait again.

  Bit 7,6 : define the active front (Only if hardware trigger selected)
  - ◊ 01 : rising front (Only if hardware trigger selected)
  - ◊ 10 : falling front (Only if hardware trigger selected)
  - ◊ 11 : Both front (Only if hardware trigger selected)

- ♦ ***Synchronisation Trigger Start : D8-D15***

  Bit 11,10,9,8 : Define the trigger mode
  - ◊ 0000 : trigger disabled
  - ◊ 0001 : One shot trigger : After the software start, the module is waiting for a trigger signal to start the acquisition. After this the trigger signal is ignored.
  - ◊ 0010 : Sequence trigger : After the software start the module is waiting for the trigger signal and acquires x sequences (also adjustable) and then wait again.

- ***Hardware Trigger Stop D16 - D19***

  The hardware trigger stop can only be activated when :
  - ◊ The hardware trigger start is not used.
  - ◊ The hardware trigger start is used in one shot mode.

The stop of the acquisition is really do at the end of a sequence acquisition(to avoid that the acquisition is stop in the middle of a sequence).

Bit 16 : Define the trigger stop is enable or not
- ◊ 0 : Stop trigger disabled
- ◊ 1 : Stop trigger enabled.

Bit 18,17 : define the active front (Only if hardware trigger stop selected)
- ◊ 01 : rising front (Only if hardware trigger stop selected)
- ◊ 10 : falling front (Only if hardware trigger stop selected)
- ◊ 11 : Both front (Only if hardware trigger stop selected)

Bit 19 : define if the hardware trigger stop use the ulHardwareTriggerCount (Only if hardware trigger stop selected)
- ◊ 0 : ulHardwareTriggerCount not used : First hardware trigger stop will stop the acquisition
- ◊ 1 : ulHardwareTriggerCount is used : The ulHardwareTriggerCount hardware trigger will stop the acquisition

[Query frame layout] **HardwareTriggerCount:** Define the number of trigger events before the action occur

- 0 or 1: all trigger event start the action
- max value: 65535

[Query frame layout] **HardwareTriggerFilterTime:** filter time for the hardware trigger

- in multiplier from 250 ns step
- max value: 65535

[Query frame layout] **ByTriggerNbrOfSeqToAcquire:** define the number of sequences to acquire by each trigger event

[Query frame layout] **Option1:** Data format option

- 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)

[Query frame layout] **Option2:** Reserved

[Query frame layout] **Option3:** Reserved

[Query frame layout] **Option4:** Reserved

Returns:

Possible return value on the remote system (read them with GetLastCommandStatus)

- 0: success
- -1: means an system error occured
- -2: Transducer selection error
- -3: Channel mask error : can not be null
- -4: Channel mask error
- -5: Average mode error
- -6: Average value error
- -7: Division factor error
- -8: Incorrect value for Hardware Trigger Mode
- -9: Incorrect value for Hardware Trigger front
- -10: Incorrect value for Synchro Trigger Mode
- -11: Incorrect value for Hardware Trigger count
- -12: Incorrect value for Hardware Trigger filter time
- -13: Incorrect value for "trigger number of sequences to acquire"
- -14: Wrong data format parameter (ulOption1)
- -15: A value for Hardware Trigger front was defined but Hardware Trigger Mode is not set
- -16: Cannot use both triggers at the same time
- -17: Incorrect value for the hardware trigger stop front
- -18: Hardware trigger stop can not be used by this configuration of hardware trigger start
- -100: TransducerInit kernel function error
- -101: InitConvertTimeDivisionFactor kernel function error
- -102: SetAutoRefreshAverageValue kernel function error
- -103: InitDigitalInputFilter kernel function error
- -104: InitEnableDisableHardwareTrigger kernel function error
- -105: SynchroTrigger Init/Enable/Disable kernel function error
- -106: SetTriggerSequenceCount kernel function error
- -107: StartAutoRefresh kernel function error

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000

length	2	16-bit integer	56	0x3800	0x0038
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	1	0x0100	0x0001
word count	2	16-bit integer	24	0x1800	0x0018
byte count	2	16-bit integer	48	0x3000	0x0030
TransducerSelection	4	32-bit integer	See the description above	0x????????	0x????????
ChannelMask	4	32-bit integer	See the description above	0x????????	0x????????
AverageMode	4	32-bit integer	See the description above	0x????????	0x????????
AverageValue	4	32-bit integer	See the description above	0x????????	0x????????
DivisionFactor	4	32-bit integer	See the description above	0x????????	0x????????
TriggerAction	4	32-bit integer	See the description above	0x????????	0x????????
HardwareTriggerCount	2	16-bit integer	See the description above	0x????	0x????
HardwareTriggerFilterTime	2	16-bit integer	See the description above	0x????	0x????
ByTriggerNbrOfSeqToAcquire	4	32-bit integer	See the description above	0x????????	0x????????
Option1	4	32-bit integer	See the description above	0x????????	0x????????
Option2	4	32-bit integer	See the description above	0x????????	0x????????
Option3	4	32-bit integer	See the description above	0x????????	0x????????
Option4	4			0x????????	0x????????

		32-bit integer	See the description above		

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	1	0x0100	0x0001
word count	2	16-bit integer	24	0x1800	0x0018

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception	1	8-bit	See	0x??	0x??

code		integer	corresponding chapter		

Function MX370x__TransducerInitAndStartAutoRefreshEx

Description

Initialise and start the transducer auto refresh acquisition mode

Parameters:

[Query frame layout] ***TransducerSelection:*** Transducer type selection

[Query frame layout] ***ChannelMask:*** Mask of the channel to acquire by the auto refresh (1 bit = 1 Channel)

[Query frame layout] ***AverageMode:*** Set the average mode :

- 0: not used
- 1: average per Sequence
- 2: average per channel

[Query frame layout] **AverageValue:** Set the average value (only used, when average is used)

- 0: average not used
- max value: 255

[Query frame layout] **DivisionFactor:** Division factor (min : 5, max 255)

[Query frame layout] **TriggerAction:** Trigger action :

- ***Hardware Trigger Start D0 - D7***
  Bit 3,2,1,0 : Define the trigger mode
  - 0000 : Trigger disabled
  - 0001 : One shot trigger : After the software start, the module is waiting for a trigger signal to start the acquisition. After this the trigger signal is ignored.
  - 0010 : Sequence trigger : After the software start the module is waiting for the trigger signal and acquires x sequences (also adjustable) and then wait again.

  Bit 7,6 : define the active front (Only if hardware trigger selected)
  - 01 : rising front (Only if hardware trigger selected)
  - 10 : falling front (Only if hardware trigger selected)
  - 11 : Both front (Only if hardware trigger selected)

- ***Synchronisation Trigger Start : D8-D15***
  Bit 11,10,9,8 : Define the trigger mode
  - 0000 : trigger disabled
  - 0001 : One shot trigger : After the software start, the module is waiting for a trigger signal to start the acquisition. After this the trigger signal is ignored.
  - 0010 : Sequence trigger : After the software start the module is waiting for the trigger signal and acquires x sequences (also adjustable) and then wait again.

- ***Hardware Trigger Stop D16 - D19***

  The hardware trigger stop can only be activated when :
  - ◊ The hardware trigger start is not used.
  - ◊ The hardware trigger start is used in one shot mode.

The stop of the acquisition is really do at the end of a sequence acquisition(to avoid that the acquisition is stop in the middle of a sequence).

Bit 16 : Define the trigger stop is enable or not
- ◊ 0 : Stop trigger disabled
- ◊ 1 : Stop trigger enabled.

Bit 18,17 : define the active front (Only if hardware trigger stop selected)
- ◊ 01 : rising front (Only if hardware trigger stop selected)
- ◊ 10 : falling front (Only if hardware trigger stop selected)
- ◊ 11 : Both front (Only if hardware trigger stop selected)

Bit 19 : define if the hardware trigger stop use the ulHardwareTriggerCount (Only if hardware trigger stop selected)
- ◊ 0 : ulHardwareTriggerCount not used : First hardware trigger stop will stop the acquisition
- ◊ 1 : ulHardwareTriggerCount is used : The ulHardwareTriggerCount hardware trigger will stop the acquisition

[Query frame layout] **HardwareTriggerCount:** Define the number of trigger events before the action occur

- 0 or 1: all trigger event start the action
- max value: 65535

[Query frame layout] **HardwareTriggerFilterTime:** filter time for the hardware trigger

- in multiplier from 250 ns step
- max value: 65535

[Query frame layout] **ByTriggerNbrOfSeqToAcquire:** define the number of sequences to acquire by each trigger event

[Query frame layout] **Option1:** Data format option

- 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)

[Query frame layout] **Option2:** Reserved

[Query frame layout] **Option3:** Reserved

[Query frame layout] **Option4:** Reserved

Returns:

Possible return value on the remote system (read them with GetLastCommandStatusEx)

- 0: success
- -1: means an system error occured
- -2: Transducer selection error
- -3: Channel mask error : can not be null
- -4: Channel mask error
- -5: Average mode error
- -6: Average value error
- -7: Division factor error
- -8: Incorrect value for Hardware Trigger Mode
- -9: Incorrect value for Hardware Trigger front
- -10: Incorrect value for Synchro Trigger Mode
- -11: Incorrect value for Hardware Trigger count
- -12: Incorrect value for Hardware Trigger filter time
- -13: Incorrect value for "trigger number of sequences to acquire"
- -14: Wrong data format parameter (ulOption1)
- -15: A value for Hardware Trigger front was defined but Hardware Trigger Mode is not set
- -16: Cannot use both triggers at the same time
- -17: Incorrect value for the hardware trigger stop front
- -18: Hardware trigger stop can not be used by this configuration of hardware trigger start
- -100: TransducerInit kernel function error
- -101: InitConvertTimeDivisionFactor kernel function error
- -102: SetAutoRefreshAverageValue kernel function error
- -103: InitDigitalInputFilter kernel function error
- -104: InitEnableDisableHardwareTrigger kernel function error
- -105: SynchroTrigger Init/Enable/Disable kernel function error
- -106: SetTriggerSequenceCount kernel function error
- -107: StartAutoRefresh kernel function error

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	55	0x3700	0x0037

unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	1200	0xB004	0x04B0
word count	2	16-bit integer	24	0x1800	0x0018
byte count	1	8-bit integer	48	0x30	0x30
TransducerSelection	4	32-bit integer	See the description above	0x????????	0x????????
ChannelMask	4	32-bit integer	See the description above	0x????????	0x????????
AverageMode	4	32-bit integer	See the description above	0x????????	0x????????
AverageValue	4	32-bit integer	See the description above	0x????????	0x????????
DivisionFactor	4	32-bit integer	See the description above	0x????????	0x????????
TriggerAction	4	32-bit integer	See the description above	0x????????	0x????????
HardwareTriggerCount	2	16-bit integer	See the description above	0x????	0x????
HardwareTriggerFilterTime	2	16-bit integer	See the description above	0x????	0x????
ByTriggerNbrOfSeqToAcquire	4	32-bit integer	See the description above	0x????????	0x????????
Option1	4	32-bit integer	See the description above	0x????????	0x????????
Option2	4	32-bit integer	See the description above	0x????????	0x????????
Option3	4	32-bit integer	See the description above	0x????????	0x????????
Option4	4	32-bit integer	See the description above	0x????????	0x????????

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	1200	0xB004	0x04B0
word count	2	16-bit integer	24	0x1800	0x0018

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MX370x__TransducerStopAndReleaseAutoRefresh

For new application(s) or automate communication it is recommended to use the function MX370x__TransducerStopAndReleaseAutoRefreshEx.

Description

Stop and release the transducer auto refresh acquisition mode

Parameters:

[Query frame layout] ***Dummy:*** Is not used

Returns:

Possible return value on the remote system (read them with GetLastCommandStatus)

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

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	12	0x0C00	0x000C
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	2	0x0200	0x0002
word count	2	16-bit integer	2	0x0200	0x0002
byte count	2	16-bit integer	4	0x0400	0x0004

Dummy	4	32-bit integer	See the description above	0x????????	0x????????

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	2	0x0200	0x0002
word count	2	16-bit integer	2	0x0200	0x0002

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception	1	8-bit	See	0x??	0x??

code		integer	corresponding chapter		

Function MX370x__TransducerStopAndReleaseAutoRefreshEx

Description

Stop and release the transducer auto refresh acquisition mode

Parameters:

[Query frame layout] ***Dummy:*** Is not used

Returns:

Possible return value on the remote system (read them with GetLastCommandStatusEx)

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

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	11	0x0B00	0x000B
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	1250	0xE204	0x04E2
word count	2	16-bit integer	2	0x0200	0x0002
byte count	1	8-bit integer	4	0x04	0x04

Dummy	4	32-bit integer	See the description above	0x????????	0x????????

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	1250	0xE204	0x04E2
word count	2	16-bit integer	2	0x0200	0x0002

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception	1	8-bit	See	0x??	0x??

code		integer	corresponding chapter		

Function MX370x__TransducerInitAndStartSequence

For new application(s) or automate communication it is recommended to use the function MX370x__TransducerInitAndStartSequenceEx.

Description

Initialise and start the transducer sequence acquisition mode

Parameters:

[Query frame layout] **TransducerSelection :** Transducer type selection

[Query frame layout] **NbrOfChannel :** Number of channel in the sequence

[Query frame layout] **ChannelList :** List of the channel index (0 to MaxChannel-1) who compose the sequence

[Query frame layout] **DivisionFactor :** Division factor (min: 5, max: 255)

[Query frame layout] **NbrOfSequence :** Number of sequence to acquire :

- 0 : continuous mode
- > 0 : number of sequence

[Query frame layout] **NbrMaxSequenceToTransfer :** This parameter defined the minimal number of sequences to acquire between each send of data by the modul.

Warning : They are two possibilities that the number of sequences sent doesn't reach the minimal number:

- By the end of the acquisition.
- If the memory capicity is not big enough.

[Query frame layout] **DelayMode :** Delay Mode :

- ADDIDATA_DELAY_NOT_USED 0 : Delay is not used.
- ADDIDATA_DELAY_MODE1_USED 1 : The delay time defines the time between 2 sequence beginnings.
- ADDIDATA_DELAY_MODE2_USED 2 : The delay time defines the time between the end of a sequence until the beginning of the next sequence.

[Query frame layout] **DelayTimeUnit :** Selection of the delay time unit

- 0: ms
- 1: s

[Query frame layout] **DelayValue :** Delay Value (max value: 65535)

[Query frame layout] **TriggerAction :** Trigger action :

- ***Hardware Trigger Start D0 - D7***

  Bit 3,2,1,0 : Define the trigger mode
    - ◊ 0000 : Trigger disabled
    - ◊ 0001 : One shot trigger : After the software start, the module is waiting for a trigger signal to start the acquisition. After this the trigger signal is ignored.
    - ◊ 0010 : Sequence trigger : After the software start the module is waiting for the trigger signal and acquires x sequences (also adjustable) and then wait again.

  Bit 7,6 : define the active front (Only if hardware trigger selected)
    - ◊ 01 : rising front (Only if hardware trigger selected)
    - ◊ 10 : falling front (Only if hardware trigger selected)
    - ◊ 11 : Both front (Only if hardware trigger selected)

- ***Synchronisation Trigger Start : D8-D15***

  Bit 11,10,9,8 : Define the trigger mode
    - ◊ 0000 : trigger disabled
    - ◊ 0001 : One shot trigger : After the software start, the module is waiting for a trigger signal to start the acquisition. After this the trigger signal is ignored.
    - ◊ 0010 : Sequence trigger : After the software start the module is waiting for the trigger signal and acquires x sequences (also adjustable) and then wait again.

- ***Hardware Trigger Stop D16 - D19***

  The hardware trigger stop can only be activated when :
    - ◊ The hardware trigger start is not used.
    - ◊ The hardware trigger start is used in one shot mode.

The stop of the acquisition is really do at the end of a sequence acquisition(to avoid that the acquisition is stop in the middle of a sequence).

Bit 16 : Define the trigger stop is enable or not
- ◊ 0 : Stop trigger disabled
- ◊ 1 : Stop trigger enabled.

Bit 18,17 : define the active front (Only if hardware trigger stop selected)
- ◊ 01 : rising front (Only if hardware trigger stop selected)
- ◊ 10 : falling front (Only if hardware trigger stop selected)
- ◊ 11 : Both front (Only if hardware trigger stop selected)

Bit 19 : define if the hardware trigger stop use the ulHardwareTriggerCount (Only if hardware trigger stop selected)
- ◊ 0 : ulHardwareTriggerCount not used : First hardware trigger stop will stop the acquisition
- ◊ 1 : ulHardwareTriggerCount is used : The ulHardwareTriggerCount hardware trigger will stop the acquisition

[Query frame layout] **HardwareTriggerCount :** Define the number of trigger events before the trigger action occur

- 0 or 1 : all trigger event start the trigger action
- max value : 65535

[Query frame layout] **HardwareTriggerFilterTime :** Filter time for the hardware trigger (= multiplier from 250 ns step)

- max value : 65535

[Query frame layout] **ByTriggerNbrOfSeqToAcquire :** define the number of sequences to acquire by each trigger event

[Query frame layout] **Option1 :** Data format option

- 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 : receive a relative Time Stamp (first acquisition => time stamp=0) instead of absolute time stamp
  - 0 : No relative time stamp information
  - 1 : Relative time stamp information

[Query frame layout] **Option2 :** Reserved

[Query frame layout] **Option3 :** Reserved

[Query frame layout] **Option4 :** Reserved

Returns:

Possible return value on the remote system (read them with GetLastCommandStatus)

- 0 : success
- -1: means an system error occured
- -2: Tranducer selection error
- -3: Number of channel error
- -4: Channel array selection error
- -5: Division factor error
- -6: Incorrect value for Hardware Trigger Mode
- -7: Incorrect value for Hardware Trigger Front
- -8: Incorrect value for Synchro Trigger Mode
- -9: Incorrect value for Hardware Trigger Count
- -10: Incorrect value for Hardware Trigger filter time
- -11: Incorrect value for "trigger number of sequences to acquire"

- -12: Delay Mode selection error
- -13: Delay time unit selection error
- -14: Delay value
- -15: Wrong data format parameter (ulOption1)
- -16: A value for Hardware Trigger front was defined but Hardware Trigger Mode is not set
- -17: Cannot use both triggers at the same time
- -18: Incorrect value for the hardware trigger stop front
- -19: Hardware trigger stop can not be used by this configuration of hardware trigger start
- -100: TransducerInit kernel function error
- -101: InitConvertTimeDivisionFactor kernel function error
- -102: InitEnableDisableSequenceDelay kernel function error
- -103: InitDigitalInputFilter kernel function error
- -104: InitEnableDisableHardwareTrigger kernel function error
- -105: InitEnableSynchroTrigger kernel function error
- -106: DisableSynchroTrigger kernel function error
- -107: SetTriggerSequenceCount kernel function error
- -108: InitSequence kernel function error
- -109: StartStopSequence kernel function error

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	136	0x8800	0x0088
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	3	0x0300	0x0003
word count	2	16-bit integer	64	0x4000	0x0040
byte count	2	16-bit integer	128	0x8000	0x0080
TransducerSelection	4	32-bit integer	See the description above	0x????????	0x????????
NbrOfChannel	4	32-bit integer	See the description above	0x????????	0x????????
ChannelList	64			0x????????[16]	0x????????[16]

		32-bit integer array	See the description above		
DivisionFactor	4	32-bit integer	See the description above	0x????????	0x????????
NbrOfSequence	4	32-bit integer	See the description above	0x????????	0x????????
NbrMaxSequenceToTransfer	4	32-bit integer	See the description above	0x????????	0x????????
DelayMode	4	32-bit integer	See the description above	0x????????	0x????????
DelayTimeUnit	4	32-bit integer	See the description above	0x????????	0x????????
DelayValue	4	32-bit integer	See the description above	0x????????	0x????????
TriggerAction	4	32-bit integer	See the description above	0x????????	0x????????
HardwareTriggerCount	4	32-bit integer	See the description above	0x????????	0x????????
HardwareTriggerFilterTime	4	32-bit integer	See the description above	0x????????	0x????????
ByTriggerNbrOfSeqToAcquire	4	32-bit integer	See the description above	0x????????	0x????????
Option1	4	32-bit integer	See the description above	0x????????	0x????????
Option2	4	32-bit integer	See the description above	0x????????	0x????????
Option3	4	32-bit integer	See the description above	0x????????	0x????????
Option4	4	32-bit integer	See the description above	0x????????	0x????????

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	3	0x0300	0x0003
word count	2	16-bit integer	64	0x4000	0x0040

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MX370x__TransducerInitAndStartSequenceEx

Description

Initialise and start the transducer sequence acquisition mode

Parameters:

[Query frame layout] **TransducerSelection :** Transducer type selection

[Query frame layout] **NbrOfChannel :** Number of channel in the sequence

[Query frame layout] **ChannelList :** List of the channel index (0 to MaxChannel-1) who compose the sequence

[Query frame layout] **DivisionFactor :** Division factor (min: 5, max: 255)

[Query frame layout] **NbrOfSequence :** Number of sequence to acquire :

- 0 : continuous mode
- > 0 : number of sequence

[Query frame layout] **NbrMaxSequenceToTransfer :** This parameter defined the minimal number of sequences to acquire between each send of data by the modul.

Warning : They are two possibilities that the number of sequences sent doesn't reach the minimal number:

- By the end of the acquisition.
- If the memory capicity is not big enough.

[Query frame layout] **DelayMode :** Delay Mode :

- ADDIDATA_DELAY_NOT_USED 0 : Delay is not used.
- ADDIDATA_DELAY_MODE1_USED 1 : The delay time defines the time between 2 sequence beginnings.
- ADDIDATA_DELAY_MODE2_USED 2 : The delay time defines the time between the end of a sequence until the beginning of the next sequence.

[Query frame layout] **DelayTimeUnit :** Selection of the delay time unit

- 0: ms
- 1: s

[Query frame layout] **DelayValue :** Delay Value (max value: 65535)

[Query frame layout] **TriggerAction :** Trigger action :

- ***Hardware Trigger Start D0 - D7***
  Bit 3,2,1,0 : Define the trigger mode
    - 0000 : Trigger disabled
    - 0001 : One shot trigger : After the software start, the module is waiting for a trigger signal to start the acquisition. After this the trigger signal is ignored.
    - 0010 : Sequence trigger : After the software start the module is waiting for the trigger signal and acquires x sequences (also adjustable) and then wait again.

Bit 7,6 : define the active front (Only if hardware trigger selected)
- ◊ 01 : rising front (Only if hardware trigger selected)
- ◊ 10 : falling front (Only if hardware trigger selected)
- ◊ 11 : Both front (Only if hardware trigger selected)

- ♦ ***Synchronisation Trigger Start : D8-D15***

Bit 11,10,9,8 : Define the trigger mode
- ◊ 0000 : trigger disabled
- ◊ 0001 : One shot trigger : After the software start, the module is waiting for a trigger signal to start the acquisition. After this the trigger signal is ignored.
- ◊ 0010 : Sequence trigger : After the software start the module is waiting for the trigger signal and acquires x sequences (also adjustable) and then wait again.

- ♦ ***Hardware Trigger Stop D16 - D19***

The hardware trigger stop can only be activated when :
- ◊ The hardware trigger start is not used.
- ◊ The hardware trigger start is used in one shot mode.

The stop of the acquisition is really do at the end of a sequence acquisition(to avoid that the acquisition is stop in the middle of a sequence).

Bit 16 : Define the trigger stop is enable or not
- ◊ 0 : Stop trigger disabled
- ◊ 1 : Stop trigger enabled.

Bit 18,17 : define the active front (Only if hardware trigger stop selected)
- ◊ 01 : rising front (Only if hardware trigger stop selected)
- ◊ 10 : falling front (Only if hardware trigger stop selected)
- ◊ 11 : Both front (Only if hardware trigger stop selected)

Bit 19 : define if the hardware trigger stop use the ulHardwareTriggerCount (Only if hardware trigger stop selected)
- ◊ 0 : ulHardwareTriggerCount not used : First hardware trigger stop will stop the acquisition
- ◊ 1 : ulHardwareTriggerCount is used : The ulHardwareTriggerCount hardware trigger will stop the acquisition

[Query frame layout] **HardwareTriggerCount :** Define the number of trigger events before the trigger action occur

- ♦ 0 or 1 : all trigger event start the trigger action
- ♦ max value : 65535

[Query frame layout] **HardwareTriggerFilterTime :** Filter time for the hardware trigger (= multiplier from 250 ns step)

- ♦ max value : 65535

[Query frame layout] **ByTriggerNbrOfSeqToAcquire :** define the number of sequences to acquire by each trigger event

[Query frame layout] **Option1 :** Data format option

- 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 : receive a relative Time Stamp (first acquisition => time stamp=0) instead of absolute time stamp
  - 0 : No relative time stamp information
  - 1 : Relative time stamp information

[Query frame layout] **Option2 :** Reserved

[Query frame layout] **Option3 :** Reserved

[Query frame layout] **Option4 :** Reserved

Returns:

Possible return value on the remote system (read them with GetLastCommandStatusEx)

- 0 : success
- -1: means an system error occured
- -2: Tranducer selection error
- -3: Number of channel error
- -4: Channel array selection error
- -5: Division factor error
- -6: Incorrect value for Hardware Trigger Mode
- -7: Incorrect value for Hardware Trigger Front
- -8: Incorrect value for Synchro Trigger Mode
- -9: Incorrect value for Hardware Trigger Count
- -10: Incorrect value for Hardware Trigger filter time
- -11: Incorrect value for "trigger number of sequences to acquire"
- -12: Delay Mode selection error
- -13: Delay time unit selection error
- -14: Delay value
- -15: Wrong data format parameter (ulOption1)
- -16: A value for Hardware Trigger front was defined but Hardware Trigger Mode is not set
- -17: Cannot use both triggers at the same time
- -18: Incorrect value for the hardware trigger stop front

- -19: Hardware trigger stop can not be used by this configuration of hardware trigger start
- -100: TransducerInit kernel function error
- -101: InitConvertTimeDivisionFactor kernel function error
- -102: InitEnableDisableSequenceDelay kernel function error
- -103: InitDigitalInputFilter kernel function error
- -104: InitEnableDisableHardwareTrigger kernel function error
- -105: InitEnableSynchroTrigger kernel function error
- -106: DisableSynchroTrigger kernel function error
- -107: SetTriggerSequenceCount kernel function error
- -108: InitSequence kernel function error
- -109: StartStopSequence kernel function error

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	135	0x8700	0x0087
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	1300	0x1405	0x0514
word count	2	16-bit integer	64	0x4000	0x0040
byte count	1	8-bit integer	128	0x80	0x80
TransducerSelection	4	32-bit integer	See the description above	0x????????	0x????????
NbrOfChannel	4	32-bit integer	See the description above	0x????????	0x????????
ChannelList	64	32-bit integer array	See the description above	0x????????[16]	0x????????[16]
DivisionFactor	4	32-bit integer	See the description above	0x????????	0x????????
NbrOfSequence	4	32-bit integer	See the description	0x????????	0x????????

			above		
NbrMaxSequenceToTransfer	4	32-bit integer	See the description above	0x????????	0x????????
DelayMode	4	32-bit integer	See the description above	0x????????	0x????????
DelayTimeUnit	4	32-bit integer	See the description above	0x????????	0x????????
DelayValue	4	32-bit integer	See the description above	0x????????	0x????????
TriggerAction	4	32-bit integer	See the description above	0x????????	0x????????
HardwareTriggerCount	4	32-bit integer	See the description above	0x????????	0x????????
HardwareTriggerFilterTime	4	32-bit integer	See the description above	0x????????	0x????????
ByTriggerNbrOfSeqToAcquire	4	32-bit integer	See the description above	0x????????	0x????????
Option1	4	32-bit integer	See the description above	0x????????	0x????????
Option2	4	32-bit integer	See the description above	0x????????	0x????????
Option3	4	32-bit integer	See the description above	0x????????	0x????????
Option4	4	32-bit integer	See the description above	0x????????	0x????????

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000

protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	1300	0x1405	0x0514
word count	2	16-bit integer	64	0x4000	0x0040

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MX370x__TransducerStopAndReleaseSequence

For new application(s) or automate communication it is recommended to use the function MX370x__TransducerStopAndReleaseSequenceEx.

Description

Stop and release the transducer sequence acquisition mode

Parameters:

[Query frame layout] ***Dummy:*** Is not used

Returns:

Possible return value on the remote system (read them with GetLastCommandStatus)

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

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	12	0x0C00	0x000C
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	4	0x0400	0x0004
word count	2	16-bit integer	2	0x0200	0x0002
byte count	2	16-bit integer	4	0x0400	0x0004
Dummy	4	32-bit integer	See the description above	0x????????	0x????????

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000

protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	4	0x0400	0x0004
word count	2	16-bit integer	2	0x0200	0x0002

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MX370x__TransducerStopAndReleaseSequenceEx

Description

Stop and release the transducer sequence acquisition mode

Parameters:

[Query frame layout] ***Dummy:*** Is not used

Returns:

Possible return value on the remote system (read them with GetLastCommandStatusEx)

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

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	11	0x0B00	0x000B
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	1350	0x4605	0x0546
word count	2	16-bit integer	2	0x0200	0x0002
byte count	1	8-bit integer	4	0x04	0x04
Dummy	4	32-bit integer	See the description above	0x????????	0x????????

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000

length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	1350	0x4605	0x0546
word count	2	16-bit integer	2	0x0200	0x0002

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MX370x__SetTransducerDatabaseCursor

For new application(s) or automate communication it is recommended to use the function MX370x__SetTransducerDatabaseCursorEx.

Description

Change the active transducer database cursor

Parameters:

[Query frame layout] ***TransducerDatabaseCursor:*** New cursor value. This is an integer from 0 .. (NumberOfTransducerTypes-1)

Returns:

Possible return value on the remote system (read them with GetLastCommandStatus)

- 0: success
- -1: otherwise : internal error

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	12	0x0C00	0x000C
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	5	0x0500	0x0005
word count	2	16-bit integer	2	0x0200	0x0002
byte count	2	16-bit integer	4	0x0400	0x0004
TransducerDatabaseCursor	4	32-bit integer	See the description above	0x????????	0x????????

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2		6	0x0600	0x0006

		16-bit integer			
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	5	0x0500	0x0005
word count	2	16-bit integer	2	0x0200	0x0002

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MX370x__SetTransducerDatabaseCursorEx

Description

Change the active transducer database cursor

Parameters:

> [Query frame layout] ***TransducerDatabaseCursor:*** New cursor value. This is an integer from 0 .. (NumberOfTransducerTypes-1)

Returns:

> **Possible return value on the remote system (read them with GetLastCommandStatusEx)**

- 0: success
- -1: otherwise : internal error

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	11	0x0B00	0x000B
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	1354	0x4A05	0x054A
word count	2	16-bit integer	2	0x0200	0x0002
byte count	1	8-bit integer	4	0x04	0x04
TransducerDatabaseCursor	4	32-bit integer	See the description above	0x????????	0x????????

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01

MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	1354	0x4A05	0x054A
word count	2	16-bit integer	2	0x0200	0x0002

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MX370x__TransducerSetOffset

For new application(s) or automate communication it is recommended to use the function MX370x__TransducerSetOffsetEx.

Description

Set the offset for each transducer.

Set / Reset an offset on transducer channels.

- This function permits to set an offset (reference point) to the measured value.
- To disable (reset) a channel offset, set the corresponding channel value to 0.0.

Example: To set a reference point to a transducer in a particular position:

- Reset the offset by setting all channel offset to 0 (pdOffsetArray).

- Run a sequence with the transducer at the position you want to be 0 (reference point). Save the acquired values to put them into pdOffsetArray.
- Stop the acquisition.
- Run MX370x__TransducerSetOffset function to set the offset with the pdOffsetArray previously saved.
- In the next sequence, position will be 0.

For more information see SetOffset sample.

Parameters:

[Query frame layout] ***fOffsetArrayPointer:*** table pointer with each offsets for transducers.

Returns:

Possible return value on the remote system (read them with GetLastCommandStatus)

- 0: success
- -1: otherwise : internal error
- -2: driver status error, acquisition is running
- -100: transducerSetOffset kernel function error

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	72	0x4800	0x0048
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	6	0x0600	0x0006
word count	2	16-bit integer	32	0x2000	0x0020
byte count	2	16-bit integer	64	0x4000	0x0040
fOffsetArrayPointer	64	32-bit floating point array	See the description above	0x????????[16]	0x????????[16]

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	6	0x0600	0x0006
word count	2	16-bit integer	32	0x2000	0x0020

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MX370x__TransducerSetOffsetEx

Description

Set the offset for each transducer.

Set / Reset an offset on transducer channels.

- This function permits to set an offset (reference point) to the measured value.
- To disable (reset) a channel offset, set the corresponding channel value to 0.0.

Example: To set a reference point to a transducer in a particular position:

- Reset the offset by setting all channel offset to 0 (pdOffsetArray).
- Run a sequence with the transducer at the position you want to be 0 (reference point). Save the acquired values to put them into pdOffsetArray.
- Stop the acquisition.
- Run MX370x__TransducerSetOffset function to set the offset with the pdOffsetArray previously saved.
- In the next sequence, position will be 0.

For more information see SetOffset sample.

Parameters:

[Query frame layout] ***fOffsetArrayPointer:*** table pointer with each offsets for transducers.

Returns:

Possible return value on the remote system (read them with GetLastCommandStatusEx)

- 0: success
- -1: otherwise : internal error
- -2: driver status error, acquisition is running
- -100: transducerSetOffset kernel function error

Query frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	71	0x4700	0x0047

unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	1356	0x4C05	0x054C
word count	2	16-bit integer	32	0x2000	0x0020
byte count	1	8-bit integer	64	0x40	0x40
fOffsetArrayPointer	64	32-bit floating point array	See the description above	0x????????[16]	0x????????[16]

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	6	0x0600	0x0006
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x10	0x10	0x10
Reference number (=register)	2	16-bit integer	1356	0x4C05	0x054C
word count	2	16-bit integer	32	0x2000	0x0020

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Intel)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server -	0x0000	0x0000

			usually 0		
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x90	0x90	0x90
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

FC23 (read/write registers) Functions

Top

Functions in this group are used to read/write values on the module.
This functions permits to call a write (FC16) and then a read(FC3) function in one command.

Query frame layout

Field	**Size (Bytes)**	**Type**	**Value**	**little endian (Motorola)**	**big endian (Motorola)**
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	Depends to the FC16 function called	?	?
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x17	0x17	0x17
Reference number for read (=register)	2	16-bit integer	FC3 reference	?	?
Word count for read	2	16-bit integer	See the corresponding FC3 function	?	?
Reference number for write (=register)	2	16-bit integer	FC16 reference	?	?
Word count for write	2	16-bit integer	See the corresponding FC16 function	?	?
Byte count	1	8-bit integer	(= 2xWord count for write)	?	?
Register values	?	?	See the corresponding FC16 function	?	?

Response frame layout

Field	Size (Bytes)	Type	Value	little endian (Motorola)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	Depends to the FC3 function called	?	?
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x17	0x17	0x17
Byte count	1	8-bit integer	(= 2x word count for read)	?	?
Register values	?	?	See the corresponding FC3 function	?	?

Exception frame layout

Field	Size (Bytes)	Type	Value	little endian (Motorola)	big endian (Motorola)
transaction identifier	2	16-bit integer	User defined - copied by server - usually 0	0x0000	0x0000
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	3	0x0300	0x0003
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS Function code	1	8-bit integer	0x97	0x97	0x97
Exception code	1	8-bit integer	See corresponding chapter	??	??

Exception code description

Top

Name	Value	Description
MODBUS_ILLEGAL_FUNCTION	0x1	function code is not allowable action for the slave
MODBUS_ILLEGAL_DATA_ADDRESS	0x2	data address received in query is not allowable
MODBUS_ILLEGAL_DATA_VALUE	0x3	incorrect value int the query data field or the length is incorrect
MODBUS_ILLEGAL_DATA_RESPONSE_LENGTH	0x4	the request as framed would generate a response whose size exceeds the available MODBUS datasize.
MODBUS_ACKNOWLEDGE	0x5	specialized use in conjunction with programming commands
MODBUS_DSLAVE_DEVICE_BUSY	0x6	specialized use in conjunction with programming commands
MODBUS_NEGATIVE_ACKNOWLEDGE	0x07	specialized use in conjunction with programming commands
MODBUS_MEMORY_PARITY_ERROR	0x08	the extended file area failed to pass a consistency check
MODBUS_REMOTE_EXECUTION_ERROR	0x09	the remote function performed incorrectly (use function GetLastCommandStatus to know why)
MODBUS_GATEWAY_PATH_UNAVAILABLE	0x0A	used with modbus plus gateway
MODBUS_GATEWAY_TARGET_DEVICE_FAILED_TO_RESPOND	0x0B	used with modbus plus gateway

Siemens Step 7 compatibility information (AWL/SDF code)

Top

Due to limitations of the S7 platform, some names of function and parameter have been shortened in the AWL and S7 code. This table summarizes the changes against the standard version as described above.

Function/Parameter	Renamed as
MXCommon__GetModuleType	GetModuleType
MXCommon__GetTime	GetTime
MXCommon__TestCustomerID	TestCustomerID
MX370x__getNumberOfChannels	370x_GetNbrOfChannels
MX370x__TransducerGetAutoRefreshValues	370x_GetAutoRefVal
MX370x__TransducerGetNbrOfType	370x_GetNbrOfType
MX370x__GetTransducerDatabaseCursor	370x_GetDataBaseCursor
TransducerDatabaseCursor	TransducerDBCursor
MX370x__TransducerGetTypeInformation	370x_GetTypeInfo
Type	TransducerType
MXCommon__SetHardwareTriggerFilterTime	SetHwTrigFiltTime
MXCommon__InitAndStartSynchroTimer	InitStartSyncTimer
MXCommon__StopAndReleaseSynchroTimer	StopRelSyncTimer
MXCommon__Reboot	Reboot
MXCommon__SetCustomerKey	SetCustomerKey
MXCommon__SetFilterChannels	SetFilterChannels
MX370x__TransducerInitAndStartAutoRefresh	370x_InitStartAutoRef
HardwareTriggerCount	HwTrigCount
HardwareTriggerFilterTime	HwTrigFilterTime
ByTriggerNbrOfSeqToAcquire	ByTrigNbrOfSeqToAcq
MX370x__TransducerStopAndReleaseAutoRefresh	370x_StopRelAutoRef
MX370x__TransducerInitAndStartSequence	370x_InitStartSeq
HardwareTriggerCount	HwTrigCount
NbrMaxSequenceToTransfer	NbrMaxSeqToTransfer
HardwareTriggerFilterTime	HwTrigFilterTime
ByTriggerNbrOfSeqToAcquire	ByTrigNbrOfSeqToAcq
MX370x__TransducerStopAndReleaseSequence	370x_StopRelSeq
MX370x__SetTransducerDatabaseCursor	370x_SetDataBaseCursor
TransducerDatabaseCursor	TransducerDBCursor
MX370x__TransducerSetOffset	370xTrsducerSetOff
fOffsetArrayPointer	TransducerSetOffset