MODBUS interface description

Table of Contents

Table of Contents

Table of Contents

Table of Contents

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.

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	1	8-bit integer	0x03	0x03	0x03

code					
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	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	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
	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__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 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	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
	1			??	??

Exception code		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)
	2			0x0000	0x0000

transaction identifier		16-bit integer	User defined - copied by server - usually 0		
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
MODBUS Function code	1	8-bit integer	0x03	0x03	0x03
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	0x0000	0x0000

			server - usually 0		
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
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__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 MSXE301x__AnalogInputGetAutoRefreshValues

For new application(s) or automate communication it is recommended to use the function MSXE301x__AnalogInputGetAutoRefreshValuesEx.

Description

Returns the values acquired in auto refresh mode.

Parameters:

[Query frame layout] ***Dummy*** parameter

[Response frame layout] ***iReturnValue :***

- 0: means the remote function performed OK
- -1: means an system error occured
- -100: GetAutoRefreshAllValues kernel function error

[Response frame layout] ***ulTimeStampLow :*** number of microseconds since the begin of the second

[Response frame layout] ***ulTimeStampHigh :*** number of seconds since the Epoch

[Response frame layout] ***ulCounterValue :*** Array that contain the counter values

[Response frame layout] ***ulValue :*** Array that contain the channels values

- ulValues [0] : Channel 0 value
- ...
- ulValues [15] : Channel 15 value

Returns:

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

- ***syserrno :*** system-error code (the value of the libc "errno" code) EPERM means a autorefresh acquisition was not started.

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	38	0x2600	0x0026

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	80	0x5000	0x0050
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	76	0x4C00	0x004C
ulTimeStampLow	4	32-bit integer	See the description above	0x????????	0x????????
ulTimeStampHigh	4	32-bit integer	See the description above	0x????????	0x????????
ulCounterValue	4	32-bit integer	See the description above	0x????????	0x????????
ulValue	64	32-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 MSXE301x__AnalogInputGetAutoRefreshValuesEx

Description

Returns the values acquired in auto refresh mode.

Parameters:

[Query frame layout] ***Dummy*** parameter

[Response frame layout] ***iReturnValue :***

- 0: means the remote function performed OK
- -1: means an system error occured
- -100: GetAutoRefreshAllValues kernel function error

[Response frame layout] ***ulTimeStampLow :*** number of microseconds since the begin of the second

[Response frame layout] ***ulTimeStampHigh :*** number of seconds since the Epoch

[Response frame layout] ***ulCounterValue :*** Array that contain the counter values

[Response frame layout] ***ulValue :*** Array that contain the channels values

- ulValues [0] : Channel 0 value
- ...
- ulValues [15] : Channel 15 value

Returns:

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

- ***syserrno :*** system-error code (the value of the libc "errno" code) EPERM means a autorefresh acquisition was not started.

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		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	0x03	0x03	0x03
Reference number (=register)	2	16-bit integer	1000	0xE803	0x03E8
word count	2	16-bit integer	38	0x2600	0x0026

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	79	0x4F00	0x004F
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	76	0x4C	0x4C
ulTimeStampLow	4	32-bit integer	See the description above	0x????????	0x????????
ulTimeStampHigh	4	32-bit integer	See the description above	0x????????	0x????????
ulCounterValue	4	32-bit integer	See the description above	0x????????	0x????????
ulValue	64	32-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 MSXE301x__AnalogInputGetChannelType

For new application(s) or automate communication it is recommended to use the function MSXE301x__AnalogInputGetChannelTypeEx.

Description

Return the type of the analog input channels

Parameters:

[Query frame layout] ***ulOption1*** Reserved

[Response frame layout] ***iReturnValue :***

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

[Response frame layout] ***ulType :*** Array that contain the channels type (0 : voltage, 1 : current)

- ulType [0] : Channel 0 type
- ...
- ulType [15] : Channel 15 type

Returns:

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

- ***syserrno :*** system-error code (the value of the libc "errno" code) EPERM means a autorefresh acquisition was not started.

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
ulOption01	4	32-bit integer	See the description	0x????????	0x????????

			above		
ulType	64	32-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 MSXE301x__AnalogInputGetChannelTypeEx

Description

Return the type of the analog input channels

Parameters:

[Query frame layout] ***ulOption1*** Reserved

[Response frame layout] ***iReturnValue :***

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

[Response frame layout] ***ulType :*** Array that contain the channels type (0 : voltage, 1 : current)

- ulType [0] : Channel 0 type
- ...
- ulType [15] : Channel 15 type

Returns:

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

- ***syserrno :*** system-error code (the value of the libc "errno" code) EPERM means a autorefresh acquisition was not started.

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

ulOption01	4	32-bit integer	See the description above	0x????????	0x????????
ulType	64	32-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	??	??

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**
- MSXE301x__AnalogInputInitAndStartAutoRefresh Register: **1**
- MSXE301x__AnalogInputInitAndStartAutoRefreshEx Register: **1100**
- MSXE301x__AnalogInputStopAndReleaseAutoRefresh Register: **2**
- MSXE301x__AnalogInputStopAndReleaseAutoRefreshEx Register: **1150**
- MSXE301x__AnalogInputInitAndStartSequence Register: **3**
- MSXE301x__AnalogInputInitAndStartSequenceEx Register: **1200**
- MSXE301x__AnalogInputStopAndReleaseSequence Register: **4**
- MSXE301x__AnalogInputStopAndReleaseSequenceEx Register: **1250**

- MSXE301x__AnalogInputInitAndStartSequenceTrimed Register: **5**
- MSXE301x__AnalogInputInitAndStartSequenceTrimedEx Register: **1254**

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
protocol identifier	2	16-bit integer	0	0x0000	0x0000
length	2	16-bit integer	16	0x1000	0x0010
	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	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	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	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
	2		11000	0xF82A	0x2AF8

Reference number (=register)		16-bit integer			
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
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		0x10	0x10	0x10

MODBUS Function code		8-bit integer			
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
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		0x10	0x10	0x10

MODBUS Function code		8-bit integer			
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

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
	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	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:

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

- 0 : means the remote function performed OK

Response frame layout

- -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	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	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
unit identifier	1	8-bit integer	0 or 1	0x00 or 0x01	0x00 or 0x01
MODBUS	1	8-bit	0x10	0x10	0x10

Function code		integer			
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 - 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	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 MSXE301x__AnalogInputInitAndStartAutoRefresh

For new application(s) or automate communication it is recommended to use the function MSXE301x__AnalogInputInitAndStartAutoRefreshEx.

Description

Starts an autorefresh acquisition using provided configuration.

In the auto refresh mode the measurement value is updated automatically after each acquisition. The analog acquisition is initialised and the values of each channels are stored in memory on the Ethernet E/A module MSX-E301x.

The client reads the data asynchronously to the acquisition via the data socket or a SOAP/MODBUS function. You can define a mask of all channels that should be acquired.

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

- Average value calculation per channel : Each channel is acquired x times to compute an average value for the channel.
- Average value calculation per sequence : All sequences are acquired x times to compute a average value per channel.

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

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

You have the following possibility:

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

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

- One shot
- Sequence

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

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

Parameters:

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

[Query frame layout] ***pulGainArray :*** Define the gain (1 or 2) to use for each channel.

- 1 : +10 V when Unipolar, +/-10V when Bipolar
- 2 : +5 V when Unipolar, +/-5V when Bipolar

Each index of the array correspond to the channel :
sample :

- [0] : Define the gain for the channel 0
- [1] : Define the gain for the channel 1
- ...

[Query frame layout] ***pulPolarityArray :*** Define the polarity (0:Unipolar or 1:Bipolar) to use for each channel Each index of the array correspond to the channel: sample :

- [0] : Define the polarity for the channel 0
- [1] : Define the polarity for the channel 1
- ...

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

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

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

- 0 : not used
- max value : 255

[Query frame layout] ***ulConversionTimeUnit :*** Conversion Time Unit

- 0 : microsecond
- 1 : millisecond
- 2 : second

[Query frame layout] ***ulConversionTime :*** Conversion Time

- range from min 10 to 65535 when the unit is the microsecond
- range from min 1 to 65535 when the unit is the millisecond
- range from min 1 to 65535 when the unit is the second

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

- 0 : trigger disabled
- 1 : Enable Hardware Digital Input Trigger
- 2 : Enable Synchro Trigger

[Query frame layout] ***ulTriggerMode :*** Define the trigger mode

- 1 : One shot trigger
- 2 : Sequence trigger

[Query frame layout] ***ulHardwareTriggerEdge :*** Define the edge of the hardware trigger who generates a trigger action

- 1 : rising edge (Only if hardware trigger selected)
- 2 : falling edge (Only if hardware trigger selected)
- 3 : Both front (Only if hardware trigger selected)

[Query frame layout] ***ulHardwareTriggerCount :*** 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] ***ulByTriggerNbrOfSeqToAcquire :*** Define the number of sequences to acquire by each trigger event

- 0 : continuous mode
- 0 : number of sequence : (1..0xFFFFFFFF)

[Query frame layout] ***ulDataFormat :***

- D0 : Time stamp information
  - 0: no time stamp information
  - 1: time stamp information
- : Data format
  - 0: Digital value
  - 1: Analog value (in V)

[Query frame layout] ***ulOption1 :*** Reserved

[Query frame layout] ***ulOption2 :*** Reserved

[Query frame layout] ***ulOption3 :*** Reserved

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

ReturnValue :

- 0 : means the remote function performed OK
- -1: means an system error occured
- -2: The channel mask cannot be null
- -3: Channel Mask error
- -4: Gain selection error
- -5: Polarity selection error
- -6: not available average mode
- -7: not available average value
- -8: The minimal converting time is 10 us !
- -9: Not available conversion time unit
- -10: Trigger mode : 2 different mode cannot be simultaneously be activated
- -11: Hardware trigger : front definition error
- -12: Hardware trigger count value not available
- -13: Nbr of sequence to acquire by trigger mode not available
- -14: Data format not available
- -100: Channel initialisation kernel function error
- -101: Conversion time initialisation kernel function error
- -102: Average mode initialisation kernel function error
- -103: hardware trigger initialisation/enable kernel function error
- -104: hardware trigger disable kernel function error
- -105: synchro trigger initialisation/enable kernel function error
- -106: synchro trigger disable kernel function error
- -107 Sequence trigger count initialisation error
- -108: Start auto refresh kernel function error

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

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	192	0xC000	0x00C0
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	92	0x5C00	0x005C
byte count	2	16-bit integer	184	0xB800	0x00B8
ulChannelMask	4	32-bit integer	See the description above	0x????????	0x????????
pulGainArray	64	32-bit integer array	See the description above	0x????????[16]	0x????????[16]
pulPolarityArray	64	32-bit integer array	See the description above	0x????????[16]	0x????????[16]
ulAverageMode	4	32-bit integer	See the description above	0x????????	0x????????
ulAverageValue	4	32-bit integer	See the description above	0x????????	0x????????
ulConversionTime	4	32-bit integer	See the description above	0x????????	0x????????
ulConversionTimeUnit	4	32-bit integer	See the description above	0x????????	0x????????
ulTriggerMask	4	32-bit integer	See the description above	0x????????	0x????????
ulTriggerMode	4	32-bit	See the	0x????????	0x????????

		integer	description above		
ulHardwareTriggerEdge	4	32-bit integer	See the description above	0x????????	0x????????
ulHardwareTriggerCount	4	32-bit integer	See the description above	0x????????	0x????????
ulByTriggerNbrOfSeqToAcquire	4	32-bit integer	See the description above	0x????????	0x????????
ulDataFormat	4	32-bit integer	See the description above	0x????????	0x????????
ulOption1	4	32-bit integer	See the description above	0x????????	0x????????
ulOption2	4	32-bit integer	See the description above	0x????????	0x????????
ulOption3	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	1	0x0100	0x0001
word count	2	16-bit integer	92	0x5C00	0x005C

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 MSXE301x__AnalogInputInitAndStartAutoRefreshEx

Description

Starts an autorefresh acquisition using provided configuration.

In the auto refresh mode the measurement value is updated automatically after each acquisition. The analog acquisition is initialised and the values of each channels are stored in memory on the Ethernet E/A module MSX-E301x.

The client reads the data asynchronously to the acquisition via the data socket or a SOAP/MODBUS function. You can define a mask of all channels that should be acquired.

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

- Average value calculation per channel : Each channel is acquired x times to compute an average value for the channel.
- Average value calculation per sequence : All sequences are acquired x times to compute a average value per channel.

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

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

You have the following possibility:

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

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

- One shot
- Sequence

One shot:

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

Sequence:

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

Parameters:

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

[Query frame layout] ***pulGainArray :*** Define the gain (1 or 2) to use for each channel.

- 1 : +10 V when Unipolar, +/-10V when Bipolar
- 2 : +5 V when Unipolar, +/-5V when Bipolar

Each index of the array correspond to the channel :

sample :

- [0] : Define the gain for the channel 0
- [1] : Define the gain for the channel 1
- ...

[Query frame layout] ***pulPolarityArray :*** Define the polarity (0:Unipolar or 1:Bipolar) to use for each channel Each index of the array correspond to the channel: sample :

- [0] : Define the polarity for the channel 0
- [1] : Define the polarity for the channel 1
- ...

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

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

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

- 0 : not used
- max value : 255

[Query frame layout] ***ulConversionTimeUnit :*** Conversion Time Unit

- 0 : microsecond
- 1 : millisecond
- 2 : second

[Query frame layout] ***ulConversionTime :*** Conversion Time

- range from min 10 to 65535 when the unit is the microsecond
- range from min 1 to 65535 when the unit is the millisecond

- range from min 1 to 65535 when the unit is the second

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

- 0 : trigger disabled
- 1 : Enable Hardware Digital Input Trigger
- 2 : Enable Synchro Trigger

[Query frame layout] ***ulTriggerMode :*** Define the trigger mode

- 1 : One shot trigger
- 2 : Sequence trigger

[Query frame layout] ***ulHardwareTriggerEdge :*** Define the edge of the hardware trigger who generates a trigger action

- 1 : rising edge (Only if hardware trigger selected)
- 2 : falling edge (Only if hardware trigger selected)
- 3 : Both front (Only if hardware trigger selected)

[Query frame layout] ***ulHardwareTriggerCount :*** 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] ***ulByTriggerNbrOfSeqToAcquire :*** Define the number of sequences to acquire by each trigger event

- 0 : continuous mode
- 0 : number of sequence : (1..0xFFFFFFFF)

[Query frame layout] ***ulDataFormat :***

- D0 : Time stamp information
  - ◊ 0: no time stamp information
  - ◊ 1: time stamp information
- : Data format
  - ◊ 0: Digital value
  - ◊ 1: Analog value (in V)

[Query frame layout] ***ulOption1 :*** Reserved

[Query frame layout] ***ulOption2 :*** Reserved

[Query frame layout] ***ulOption3 :*** Reserved

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

ReturnValue :

- 0 : means the remote function performed OK
- -1: means an system error occured
- -2: The channel mask cannot be null
- -3: Channel Mask error
- -4: Gain selection error
- -5: Polarity selection error
- -6: not available average mode

- -7: not available average value
- -8: The minimal converting time is 10 us !
- -9: Not available conversion time unit
- -10: Trigger mode : 2 different mode cannot be simultaneously be activated
- -11: Hardware trigger : front definition error
- -12: Hardware trigger count value not available
- -13: Nbr of sequence to acquire by trigger mode not available
- -14: Data format not available
- -100: Channel initialisation kernel function error
- -101: Conversion time initialisation kernel function error
- -102: Average mode initialisation kernel function error
- -103: hardware trigger initialisation/enable kernel function error
- -104: hardware trigger disable kernel function error
- -105: synchro trigger initialisation/enable kernel function error
- -106: synchro trigger disable kernel function error
- -107 Sequence trigger count initialisation error
- -108: Start auto refresh kernel function error

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

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	191	0xBF00	0x00BF
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	1100	0x4C04	0x044C
word count	2	16-bit integer	92	0x5C00	0x005C
byte count	1	8-bit integer	184	0xB8	0xB8
ulChannelMask	4	32-bit integer	See the description above	0x????????	0x????????
pulGainArray	64	32-bit integer array	See the description above	0x????????[16]	0x????????[16]

pulPolarityArray	64	32-bit integer array	See the description above	0x????????[16]	0x????????[16]
ulAverageMode	4	32-bit integer	See the description above	0x????????	0x????????
ulAverageValue	4	32-bit integer	See the description above	0x????????	0x????????
ulConversionTime	4	32-bit integer	See the description above	0x????????	0x????????
ulConversionTimeUnit	4	32-bit integer	See the description above	0x????????	0x????????
ulTriggerMask	4	32-bit integer	See the description above	0x????????	0x????????
ulTriggerMode	4	32-bit integer	See the description above	0x????????	0x????????
ulHardwareTriggerEdge	4	32-bit integer	See the description above	0x????????	0x????????
ulHardwareTriggerCount	4	32-bit integer	See the description above	0x????????	0x????????
ulByTriggerNbrOfSeqToAcquire	4	32-bit integer	See the description above	0x????????	0x????????
ulDataFormat	4	32-bit integer	See the description above	0x????????	0x????????
ulOption1	4	32-bit integer	See the description above	0x????????	0x????????
ulOption2	4	32-bit integer	See the description above	0x????????	0x????????
ulOption3	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	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	1100	0x4C04	0x044C
word count	2	16-bit integer	92	0x5C00	0x005C

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 MSXE301x__AnalogInputStopAndReleaseAutoRefresh

For new application(s) or automate communication it is recommended to use the function MSXE301x__AnalogInputStopAndReleaseAutoRefreshEx.

Description

Stops the current autorefresh acquisition.

Must be called before any another call to MSXE301x__AnalogInputInitAndStartAutoRefresh.

Parameters:

- [Query frame layout] ***Dummy*** : 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. EPERM means a autorefresh acquisition was not started

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 code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MSXE301x__AnalogInputStopAndReleaseAutoRefreshEx

Description

Stops the current autorefresh acquisition.

Must be called before any another call to MSXE301x__AnalogInputInitAndStartAutoRefresh.

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. EPERM means a autorefresh acquisition was not started

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	1150	0x7E04	0x047E
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	0x????????	0x????????

			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	1150	0x7E04	0x047E
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 MSXE301x__AnalogInputInitAndStartSequence

For new application(s) or automate communication it is recommended to use the function MSXE301x__AnalogInputInitAndStartSequenceEx.

Description

Initialise and start the analog input sequence acquisition mode

This function is not available for the S7. See function MSXE301x__AnalogInputInitAndStartSequenceTrimed.

A sequence is a list of channels (max 16) that are acquired. It can be any order of the channels in this list.

There are different sequence modes

- Limited number of sequences / Continuous
- With/Without delay

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

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

Without delay
: There is no waiting time between the acquisitions of 2 sequences.

With delay
: A delay between 2 sequences can be configured:
For this there are 2 delay types:
  - ◊ Mode 1: The delay time defines the time between 2 sequence beginnings.
  - ◊ Mode 2: The delay time defines the time between the end of a sequence until the beginning of the next sequence.

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

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

You have the following possibility:

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

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

- One shot
- Sequence

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

Sequence:

After the software start the module is waiting for the trigger signal and acquires x sequences (also adjustable) and then wait again.
Initialise and start the analog input sequence acquisition mode

Parameters:

[Query frame layout] ***ulNbrOfChannel :*** nbr of channel in the sequence

[Query frame layout] ***pulChannelList :*** list of the channel who compose the sequence

[Query frame layout] ***pulGainArray :*** Define the gain (1 or 2) to use for each channel.

- ◊ 1 : +10 V when Unipolar, +/-10V when Bipolar
- ◊ 2 : +5 V when Unipolar, +/-5V when Bipolar Each index of the array correspond to the channel :
  sample :
- ◊ [0] : Define the gain for the channel 0
- ◊ [1] : Define the gain for the channel 1
- ◊ ...

[Query frame layout] ***pulPolarityArray :*** Define the polarity (0:Unipolar or 1:Bipolar) to use for each channel
Each index of the array correspond to the channel :
sample :

- ◊ [0] : Define the polarity for the channel 0
- ◊ [1] : Define the polarity for the channel 1
- ◊ ...

[Query frame layout] ***ulConversionTime :*** Conversion Time (min 10 us or 40 us)

[Query frame layout] ***ulConversionTimeUnit :*** Conversion Time Unit

- ◊ 0 : us
- ◊ 1 : ms

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

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

[Query frame layout] ***ulNbrMaxSequenceToTransfer :*** Max nbr of sequence to acquire before a data transfer : (1,65535)

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

- ◊ 0 : delay not used
- ◊ 1 : mode 1
- ◊ 2 : mode 2

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

0: us
1: ms
2: s

[Query frame layout] ***ulDelayValue :*** Delay Value : (1..0xFFFF)

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

- ◊ 0 : trigger disabled
- ◊ 1 : Enable Hardware Digital Input Trigger
- ◊ 2 : Enable Synchro Trigger
- ◊ 3 : Enable both Hardware and Synchro Trigger

[Query frame layout] ***ulTriggerMode :*** Define the trigger mode

- ◊ 1 : One shot trigger
- ◊ 2 : Sequence trigger

[Query frame layout] ***ulHardwareTriggerEdge :*** Define the edge of the hardware trigger who generate a trigger action

- ◊ 1 : rising front (Only if hardware trigger selected)
- ◊ 2 : falling front (Only if hardware trigger selected)
- ◊ 3 : Both front (Only if hardware trigger selected)

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

- ◊ 0 : continuous mode
- ◊ 0 : number of sequence : (1..0xFFFFFFFF)

[Query frame layout] ***ulDataFormat :*** 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 : 32/16 bit digital value
  - · 1 : 32-bit analog value (in V)
- ◊ D3 : 16-Bit digital value
  - · 0 : 32-bit digital/analog value
  - · 1 : 16-bit digital value.

  Only a pair number of acquisition channels can be selected.

[Query frame layout] ***ulOption1 :*** 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: The nbr of channel in the sequence cannot be null
◊ -3: Channel index selection error
◊ -4: Gain selection error
◊ -5: Polarity selection error
◊ -6: The minimal converting time is 10 us !
◊ -7: Not available conversion time unit
◊ -8: Delay mode selection not available
◊ -9: Delay time unit selection not available
◊ -10: Delay value not available
◊ -11: Trigger mode : 2 different mode cannot be simultaneously be activated
◊ -12: Hardware trigger : edge definition error
◊ -13: Hardware trigger count value not available
◊ -14: Nbr of sequence to acquire by trigger mode not available
◊ -15: Data format not available
◊ -100: Channel initialisation kernel function error
◊ -101: Conversion time initialisation kernel function error
◊ -102: Delay initialisation kernel function error
◊ -103: hardware trigger initialisation/enable kernel function error
◊ -104: hardware trigger disable kernel function error
◊ -105: synchro trigger initialisation/enable kernel function error
◊ -106: synchro trigger disable kernel function error
◊ -107 Sequence trigger count initialisation error
◊ -108: Sequence initialisation kernel function error
◊ -109: Start sequence kernel function error

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

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	260	0x0401	0x0104
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	126	0x7E00	0x007E
byte count	2	16-bit integer	252	0xFC00	0x00FC

ulNbrOfChannel	4	32-bit integer	See the description above	0x????????	0x????????
pulChannelList	64	32-bit integer array	See the description above	0x????????[16]	0x????????[16]
pulGainArray	64	32-bit integer array	See the description above	0x????????[16]	0x????????[16]
pulPolarityArray	64	32-bit integer array	See the description above	0x????????[16]	0x????????[16]
ulConversionTime	4	32-bit integer	See the description above	0x????????	0x????????
ulConversionTimeUnit	4	32-bit integer	See the description above	0x????????	0x????????
ulNbrOfSequence	4	32-bit integer	See the description above	0x????????	0x????????
ulNbrMaxSequenceToTransfer	4	32-bit integer	See the description above	0x????????	0x????????
ulDelayMode	4	32-bit integer	See the description above	0x????????	0x????????
ulDelayTimeUnit	4	32-bit integer	See the description above	0x????????	0x????????
ulDelayValue	4	32-bit integer	See the description above	0x????????	0x????????
ulTriggerMask	4	32-bit integer	See the description above	0x????????	0x????????
ulTriggerMode	4	32-bit integer	See the description above	0x????????	0x????????
ulHardwareTriggerEdge	4	32-bit integer	See the description above	0x????????	0x????????
ulHardwareTriggerCount	4	32-bit integer	See the description above	0x????????	0x????????
ulByTriggerNbrOfSeqToAcquire	4	32-bit integer	See the description above	0x????????	0x????????
ulDataFormat	4			0x????????	0x????????

		32-bit integer	See the description above		
ulOption1	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	126	0x7E00	0x007E

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	1	8-bit	0x90	0x90	0x90

Function code		integer			
Exception code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MSXE301x__AnalogInputInitAndStartSequenceEx

Description

Initialise and start the analog input sequence acquisition mode

This function is not available for the S7. See function MSXE301x__AnalogInputInitAndStartSequenceTrimed.

A sequence is a list of channels (max 16) that are acquired. It can be any order of the channels in this list.

There are different sequence modes

- Limited number of sequences / Continuous
- With/Without delay

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

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

Without delay
: There is no waiting time between the acquisitions of 2 sequences.

With delay
: A delay between 2 sequences can be configured:
For this there are 2 delay types:
  · Mode 1: The delay time defines the time between 2 sequence beginnings.
  · Mode 2: The delay time defines the time between the end of a sequence until the beginning of the next sequence.

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

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

You have the following possibility:

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

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

- One shot
- Sequence

One shot:

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

Sequence:

After the software start the module is waiting for the trigger signal and acquires x sequences (also adjustable) and then wait again.
Initialise and start the analog input sequence acquisition mode

Parameters:

[Query frame layout] ***ulNbrOfChannel :*** nbr of channel in the sequence

[Query frame layout] ***pulChannelList :*** list of the channel who compose the sequence

[Query frame layout] ***pulGainArray :*** Define the gain (1 or 2) to use for each channel.

- 1 : +10 V when Unipolar, +/-10V when Bipolar
- 2 : +5 V when Unipolar, +/-5V when Bipolar Each index of the array correspond to the channel :
  sample :
- [0] : Define the gain for the channel 0
- [1] : Define the gain for the channel 1
- ...

[Query frame layout] ***pulPolarityArray :*** Define the polarity (0:Unipolar or 1:Bipolar) to use for each channel
Each index of the array correspond to the channel :
sample :

- [0] : Define the polarity for the channel 0
- [1] : Define the polarity for the channel 1
- ...

[Query frame layout] ***ulConversionTime :*** Conversion Time (min 10 us or 40 us)

[Query frame layout] ***ulConversionTimeUnit :*** Conversion Time Unit

- 0 : us
- 1 : ms

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

- 0 : continuous mode
- 0 : number of sequence

[Query frame layout] ***ulNbrMaxSequenceToTransfer :*** Max nbr of sequence to acquire before a data transfer : (1,65535)

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

- 0 : delay not used
- 1 : mode 1
- 2 : mode 2

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

0: us
1: ms
2: s

[Query frame layout] ***ulDelayValue :*** Delay Value : (1..0xFFFF)

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

- 0 : trigger disabled
- 1 : Enable Hardware Digital Input Trigger
- 2 : Enable Synchro Trigger
- 3 : Enable both Hardware and Synchro Trigger

[Query frame layout] ***ulTriggerMode :*** Define the trigger mode

- 1 : One shot trigger
- 2 : Sequence trigger

[Query frame layout] ***ulHardwareTriggerEdge :*** Define the edge of the hardware trigger who generate a trigger action

- 1 : rising front (Only if hardware trigger selected)
- 2 : falling front (Only if hardware trigger selected)
- 3 : Both front (Only if hardware trigger selected)

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

- 0 : continuous mode
- 0 : number of sequence : (1..0xFFFFFFFF)

[Query frame layout] ***ulDataFormat :*** 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 : 32/16 bit digital value
  - 1 : 32-bit analog value (in V)
- D3 : 16-Bit digital value
  - 0 : 32-bit digital/analog value
  - 1 : 16-bit digital value.

  Only a pair number of acquisition channels can be selected.

[Query frame layout] ***ulOption1 :*** 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: The nbr of channel in the sequence cannot be null
- -3: Channel index selection error
- -4: Gain selection error
- -5: Polarity selection error
- -6: The minimal converting time is 10 us !
- -7: Not available conversion time unit
- -8: Delay mode selection not available
- -9: Delay time unit selection not available
- -10: Delay value not available
- -11: Trigger mode : 2 different mode cannot be simultaneously be activated
- -12: Hardware trigger : edge definition error
- -13: Hardware trigger count value not available
- -14: Nbr of sequence to acquire by trigger mode not available
- -15: Data format not available
- -100: Channel initialisation kernel function error
- -101: Conversion time initialisation kernel function error
- -102: Delay initialisation kernel function error
- -103: hardware trigger initialisation/enable kernel function error
- -104: hardware trigger disable kernel function error
- -105: synchro trigger initialisation/enable kernel function error
- -106: synchro trigger disable kernel function error
- -107 Sequence trigger count initialisation error
- -108: Sequence initialisation kernel function error
- -109: Start sequence kernel function error

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

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	259	0x0301	0x0103
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	126	0x7E00	0x007E
byte count	1	8-bit integer	252	0xFC	0xFC
ulNbrOfChannel	4	32-bit integer	See the description above	0x????????	0x????????
pulChannelList	64	32-bit integer array	See the description above	0x????????[16]	0x????????[16]
pulGainArray	64	32-bit integer array	See the description above	0x????????[16]	0x????????[16]
pulPolarityArray	64	32-bit integer array	See the description above	0x????????[16]	0x????????[16]
ulConversionTime	4	32-bit integer	See the description above	0x????????	0x????????
ulConversionTimeUnit	4	32-bit integer	See the description above	0x????????	0x????????
ulNbrOfSequence	4	32-bit integer	See the description above	0x????????	0x????????
ulNbrMaxSequenceToTransfer	4	32-bit integer	See the description above	0x????????	0x????????
ulDelayMode	4	32-bit integer	See the description above	0x????????	0x????????
ulDelayTimeUnit	4	32-bit integer	See the description above	0x????????	0x????????
ulDelayValue	4	32-bit integer	See the description above	0x????????	0x????????
ulTriggerMask	4	32-bit integer	See the description above	0x????????	0x????????
ulTriggerMode	4	32-bit integer	See the description above	0x????????	0x????????
ulHardwareTriggerEdge	4	32-bit integer	See the description above	0x????????	0x????????
ulHardwareTriggerCount	4	32-bit integer	See the description above	0x????????	0x????????

ulByTriggerNbrOfSeqToAcquire	4	32-bit integer	See the description above	0x????????	0x????????
ulDataFormat	4	32-bit integer	See the description above	0x????????	0x????????
ulOption1	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	126	0x7E00	0x007E

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 MSXE301x__AnalogInputStopAndReleaseSequence

For new application(s) or automate communication it is recommended to use the function MSXE301x__AnalogInputStopAndReleaseSequenceEx.

Description

Stop and release the analog input sequence acquisition mode [Query frame layout] _ : no input parameter [Response frame layout] Response :
iReturnValue :

Stops the current sequence acquisition.

Must be called before any another call to MSXE301x__AnalogInputInitAndStartSequence.

Parameters:

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

Returns:

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

· 0: means the remote function performed OK
· -100: Stop sequence kernel function error
· -101: Release sequence kernel function error
· -1: means an system error occured.

syserrno : system-error code (the value of the libc "errno" code) EPERM means a sequence acquisition was not started

Query 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	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 MSXE301x__AnalogInputStopAndReleaseSequenceEx

Description

Stop and release the analog input sequence acquisition mode [Query frame layout] _ : no input parameter [Response frame layout] Response :
iReturnValue :

Stops the current sequence acquisition.

Must be called before any another call to MSXE301x__AnalogInputInitAndStartSequence.

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
· -100: Stop sequence kernel function error
· -101: Release sequence kernel function error
· -1: means an system error occured.

syserrno : system-error code (the value of the libc "errno" code) EPERM means a sequence acquisition was not started

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
	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	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 code	1	8-bit integer	See corresponding chapter	0x??	0x??

Function MSXE301x__AnalogInputInitAndStartSequenceTrimed

For new application(s) or automate communication it is recommended to use the function MSXE301x__AnalogInputInitAndStartSequenceTrimedEx.

Description

Initialise and start the analog input sequence acquisition mode

This function performs the same as MSXE301x__AnalogInputInitAndStartSequence, but has parameters with smaller types that allows it to be compliant with Schneider Electric MODBUS standard.

Available from OS revision 5492.

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	92	0x5C00	0x005C
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	42	0x2A00	0x002A
byte count	2	16-bit integer	84	0x5400	0x0054
ulNbrOfChannel	1	8-bit integer	See the description above	0x??	0x??
pulChannelList	16	8-bit integer array	See the description above	0x??[16]	0x??[16]
pulGainArray	16	8-bit integer array	See the description above	0x??[16]	0x??[16]
pulPolarityArray	16	8-bit integer array	See the description above	0x??[16]	0x??[16]
ulConversionTime	4	32-bit integer	See the description above	0x????????	0x????????
ulConversionTimeUnit	1	8-bit integer	See the description above	0x??	0x??
ulNbrOfSequence	4	32-bit integer	See the description above	0x????????	0x????????
ulNbrMaxSequenceToTransfer	4	32-bit integer	See the description	0x????????	0x????????

			above		
ulDelayMode	2	16-bit integer	See the description above	0x????	0x????
ulDelayTimeUnit	1	8-bit integer	See the description above	0x??	0x??
ulDelayValue	2	16-bit integer	See the description above	0x????	0x????
ulTriggerMask	1	8-bit integer	See the description above	0x??	0x??
ulTriggerMode	1	8-bit integer	See the description above	0x??	0x??
ulHardwareTriggerEdge	1	8-bit integer	See the description above	0x??	0x??
ulHardwareTriggerCount	2	16-bit integer	See the description above	0x????	0x????
ulByTriggerNbrOfSeqToAcquire	4	32-bit integer	See the description above	0x????????	0x????????
ulDataFormat	4	32-bit integer	See the description above	0x????????	0x????????
ulOption1	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	5	0x0500	0x0005
word count	2	16-bit integer	42	0x2A00	0x002A

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 MSXE301x__AnalogInputInitAndStartSequenceTrimedE

Description

Initialise and start the analog input sequence acquisition mode

This function performs the same as MSXE301x__AnalogInputInitAndStartSequence, but has parameters with smaller types that allows it to be compliant with Schneider Electric MODBUS standard.

Available from OS revision 5492.

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	91	0x5B00	0x005B
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	1254	0xE604	0x04E6
word count	2	16-bit integer	42	0x2A00	0x002A
byte count	1	8-bit integer	84	0x54	0x54
ulNbrOfChannel	1	8-bit integer	See the description above	0x??	0x??
pulChannelList	16	8-bit integer array	See the description above	0x??[16]	0x??[16]
pulGainArray	16	8-bit integer array	See the description above	0x??[16]	0x??[16]
pulPolarityArray	16	8-bit integer array	See the description above	0x??[16]	0x??[16]
ulConversionTime	4	32-bit integer	See the description above	0x????????	0x????????
ulConversionTimeUnit	1	8-bit integer	See the description above	0x??	0x??
ulNbrOfSequence	4	32-bit integer	See the description above	0x????????	0x????????
ulNbrMaxSequenceToTransfer	4	32-bit integer	See the description above	0x????????	0x????????
ulDelayMode	2	16-bit integer	See the description above	0x????	0x????
ulDelayTimeUnit	1	8-bit integer	See the description above	0x??	0x??

ulDelayValue	2	16-bit integer	See the description above	0x????	0x????
ulTriggerMask	1	8-bit integer	See the description above	0x??	0x??
ulTriggerMode	1	8-bit integer	See the description above	0x??	0x??
ulHardwareTriggerEdge	1	8-bit integer	See the description above	0x??	0x??
ulHardwareTriggerCount	2	16-bit integer	See the description above	0x????	0x????
ulByTriggerNbrOfSeqToAcquire	4	32-bit integer	See the description above	0x????????	0x????????
ulDataFormat	4	32-bit integer	See the description above	0x????????	0x????????
ulOption1	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	1254	0xE604	0x04E6
word count	2		42	0x2A00	0x002A

		16-bit integer			

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??

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 t slave
MODBUS_ILLEGAL_DATA_ADDRESS	0x2	data address received query is not allowabl
MODBUS_ILLEGAL_DATA_VALUE	0x3	incorrect value int th query data field or th 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 comma
MODBUS_DSLAVE_DEVICE_BUSY	0x6	specialized use in conjunction with programming comma
MODBUS_NEGATIVE_ACKNOWLEDGE	0x07	specialized use in conjunction with programming comma
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 GetLastCommandSta to know why)
MODBUS_GATEWAY_PATH_UNAVAILABLE	0x0A	used with modbus pl gateway
MODBUS_GATEWAY_TARGET_DEVICE_FAILED_TO_RESPOND	0x0B	used with modbus pl 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
MSXE301x__AnalogInputGetAutoRefreshValues	301x_AIGetAutoRefVal
MSXE301x__AnalogInputGetChannelType	301x_AIGetChannelType
MXCommon__SetHardwareTriggerFilterTime	SetHwTrigFiltTime
MXCommon__InitAndStartSynchroTimer	InitStartSyncTimer
MXCommon__StopAndReleaseSynchroTimer	StopRelSyncTimer
MXCommon__Reboot	Reboot
MXCommon__SetCustomerKey	SetCustomerKey
MXCommon__SetFilterChannels	SetFilterChannels
MSXE301x__AnalogInputInitAndStartAutoRefresh	301x_AIIniStaAutoRef
ulByTriggerNbrOfSeqToAcquire	ulByTrigNbrOfSeqToAcq
MSXE301x__AnalogInputStopAndReleaseAutoRefresh	301x_AIStopRelAutoRef
MSXE301x__AnalogInputInitAndStartSequence	*not available*
MSXE301x__AnalogInputStopAndReleaseSequence	301x_AIStopRelSeq
MSXE301x__AnalogInputInitAndStartSequenceTrimed	301x_AIInitStartSeq
ulNbrMaxSequenceToTransfer	ulNbrMaxSeqToTransf
ulByTriggerNbrOfSeqToAcquire	ulByTrigNbrOfSeqToAcq