Inspiring Motion
Since 1988

Release Notes for G-MAS .NET Library 2.1.1.3

Version 2.1.1.3

May 2015 (Ver. 1.000)

www.elmomc.com

Elmo
Motion Control

Notice

This guide is delivered subject to the following conditions and restrictions:

- This guide contains proprietary information belonging to Elmo Motion Control Ltd. Such information is supplied solely for the purpose of assisting users of the Gold Line technology.
- The text and graphics included in this manual are for the purpose of illustration and reference only. The specifications on which they are based are subject to change without notice.
- Information in this document is subject to change without notice.

Elmo Motion Control and the Elmo Motion Control logo are registered trademarks of Elmo Motion Control Ltd.

EtherCAT Conformance Tested. EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.

CANopen compliant. CANopen® is a registered trademark and patented technology, licensed by CAN in Automation (CiA) GmbH, Kontumazgarten 3, DE-90429 Nuremberg, Germany.

Document no. MAN-EASII-RN (Ver. 1.000)

Revision History

Version	Date	Details
Ver. 1.000	May 2015	Initial version

Chapter 1: General

The following release notes highlight the features of the current G-MAS .NET library release (version 2.1.1.3).

This library provides a C# based API for connecting with the G-MAS devices and activating those devices.

This document contains general information and it doesn't replace the library documentation and the example applications also available with the library or on the Elmo web site.

This release is for the use of all customers.

Chapter 2: Communicating with the G-MAS

Means of Communication to the G-MAS

The G-MAS .NET library allows you to communicate with the G-MAS devices using a single communication methods supported by the G-MAS, which is TCP/IP.

Once connected to the G-MAS, you activate it by calling the various Function Block (FB) methods on various communication objects using a connection handle.

Each FB methods returns a success / failure status as well as an error code used to analyze the error received.

The G-MAS also sends events using the UDP protocol. The library can receive those events from the G-MAS and pass them to the client application.

Communication Object

The class used for connecting to the G-MAS is called `MMCConnection`. The connection is established by calling a static method on this class called `ConnectRPC`, which returns a numeric number used as the connection handle for all future calls to this G-MAS.

Library Object Hierarchy

The FBs calling the G-MAS can affect the G-MAS itself, and / or the devices located under the G-MAS network.

The FBs are located in a class hierarchy that represent the affected device/s. This FB and class hierarchy is built accordingly:

- General FBs – affect the G-MAS itself and/or all the devices in its network. Located in the `MMCConnection` class
- FBs affecting a single axis (drive) – located in the `MMCSingleAxis` class
- FBs affecting a group (vector) of axes in the G-MAS – located in the `MMCGroupAxis` class
- FBs affecting a single node (drive or a general device type such as I/O etc.) – located in the `MMCNode` class
- FBs affecting either a single axis or a group of axes – located in the `MMCAxis` class

The `MMCAxis` class is inherited by the classes `MMCNode` and `MMCGroupAxis`. `MMCNode` is inherited by the class `MMCSingleAxis`.

Apart from the above classes, there are specific classes which hold the API to specific subjects such as the `MMCErrorCorr` class which holds the error correction API.

Receiving Events from the G-MAS

The `ConnectRPC` method contains a parameter called `callbackFunc`, and another parameter called `eventMask`.

The `eventMask` parameter is a mask defining which events the G-MAS will send to this connection. If you wish to receive all the possible events, set this mask to the value `0xEFFFFFFF`.

The `callbackFunc` parameter is a delegate method which is called whenever an event is received from the G-MAS.

When an event is arrived from the G-MAS, the library will receive this event and call the `callbackFunc` method with a struct named `MMC_CAN_REPLY_DATA_OUT`. This struct contains the entire event data, including the event type which is described by the `eASYNC_EVENT` enum.

Chapter 3: Changes from Version 2.1.1.2

General

The G-MAS .NET library version 2.1.1.2 was released in August 2014.

Since then the G-MAS was changed dramatically adding many new features that are supported by the library.

This section details the changes divided into subjects.

G-Mas Event Handling

As stated above, the library can receive events from the G-MAS and call a delegate method provided by the client.

The event types sent by the G-MAS can be controlled by setting the G-MAS event mask, which masks in and out the event types.

You can control the event G-MAS event mask using the `SetEventMask`, `ClearEventMask`, `GetEventMask` and `GetEventMaskEnum` methods of the `MMCConnection` class.

The previous G-MAS .NET libraries allow you to receive the G-MAS events using a single notification delegate method for all events, registered in the `ConnectRPC` method of the `MMCConnection` class.

The G-MAS .NET library version 2.1.1.3 allows you to register and unregister the single event notification delegate method using the `RegisterAsyncReplyEventCallback and the UnregisterAsyncReplyEventCallback` methods of the `MMCConnection` class.

This library version also allows you to register to receive event notification on certain G-MAS event types, using several methods with the names `Register<Name>EventCallback` (where "Name" is the name of the event) of the `MMCConnection` class.

For instance: `RegisterTouchProbeEndCallback` etc.

When closing the connection to the G-MAS, in order to avoid memory leaks, all event delegate methods for this connection will be unregistered.

ECAM Tables Support

A class called `MMCCamTable` was added, including methods for setting an ECAM table from memory, or loading an ECAM table file using the `SelectTable` method.

Activating an ECAM table is done by calling the `CamIn` and `CamOut` methods in the `MMCAxis` class. You can check the status of an ECAM process running on a certain axis by calling the `CamStatus` method.

The idea of putting those methods in the `MMCAxis` class was that those methods will be supported by the G-MAS also for a group of axes in the future. Currently they are supported only on a single axis (drive).

G-MAS Error Handling

The G-MAS allows you to determine several error handling policies.

You can manage the G-MAS error policies using the `RegisterErrorPolicy` and `GetErrorPolicy` methods of the `MMCConnection` class.

Some of the errors result in a general G-MAS system error, which can be reset using the `ResetSystemErrors` method of the `MMCConnection` class.

Handling EtherCAT Process Image Variables

The G-MAS allows reading and writing to Process Image (PI) input and output variables mapped for the devices in the EtherCAT network.

The library contains several methods including the use of those PI variables, as detailed below.

On class `MMCNode`:

- GetPIVarInfo, GetPIVarInfoByAlias, GetPIVarsRangeInfo – API for obtaining information on a PI variable
- ReadPIVar, ReadLargePIVar – API for reading the value of a PI variables
- WritePIVar, WriteLargePIVar – API for writing the value of a PI variables

On classes `MMCSingleAxis` and `MMCGroupAxis`:

- WaitUntilConditionFBEx – allows waiting on a condition on a G-MAS parameter or on PI variable value
- WriteGroupParametersEx – allows writing a mixed group of G-Mas parameters an PI variables

On class `MMCConnection`:

- BeginRecordingEx – allows recording also PI variables values
- ConfigPIBulkRead, PerformPIBulkRead – allows you to bulk-read PI variable values in a single operation, which is very performance-efficient if you need to read many values at once

Sending Commands to the Elmo Drives

The G-MAS allows sending commands directly to the devices under its network, by sending SDO commands through the CAN or the EtherCAT networks.

The library implements an API for sending textual commands to the Elmo drives using by sending them SDO Download commands through the network.

This feature allows you to directly set the drives' parameters through the G-MAS network, without having to have a separate physical connection to the drives.

Both reading and writing parameters values are supported (ex. "MO", "AC=100000" etc.).

The `MMCSingleAxis` class methods `SendCmdViaSDO`, `SendIntCmdViaSDO` and `SendFloatCmdViaSDO` support this feature.

Motion with Double Values

Motion commands were added with velocity, acceleration and other parameters entered as double values.

Those commands can be identified by their names which are the same name + the prefix "Ex".

For instance, the methods named `MoveAbsoluteEx` in the `MMCSingleAxis` class receive double velocity, acceleration and other values.

Online Splines Support

The G-MAS now supports activation of Online Spline motion commands.

Online Splines can be changed on the fly, and support some new motion modes such as constant time, constant velocity etc.

You can activate Online Splines motion using the `InitTableEx` method of the `MMCConnection` class.