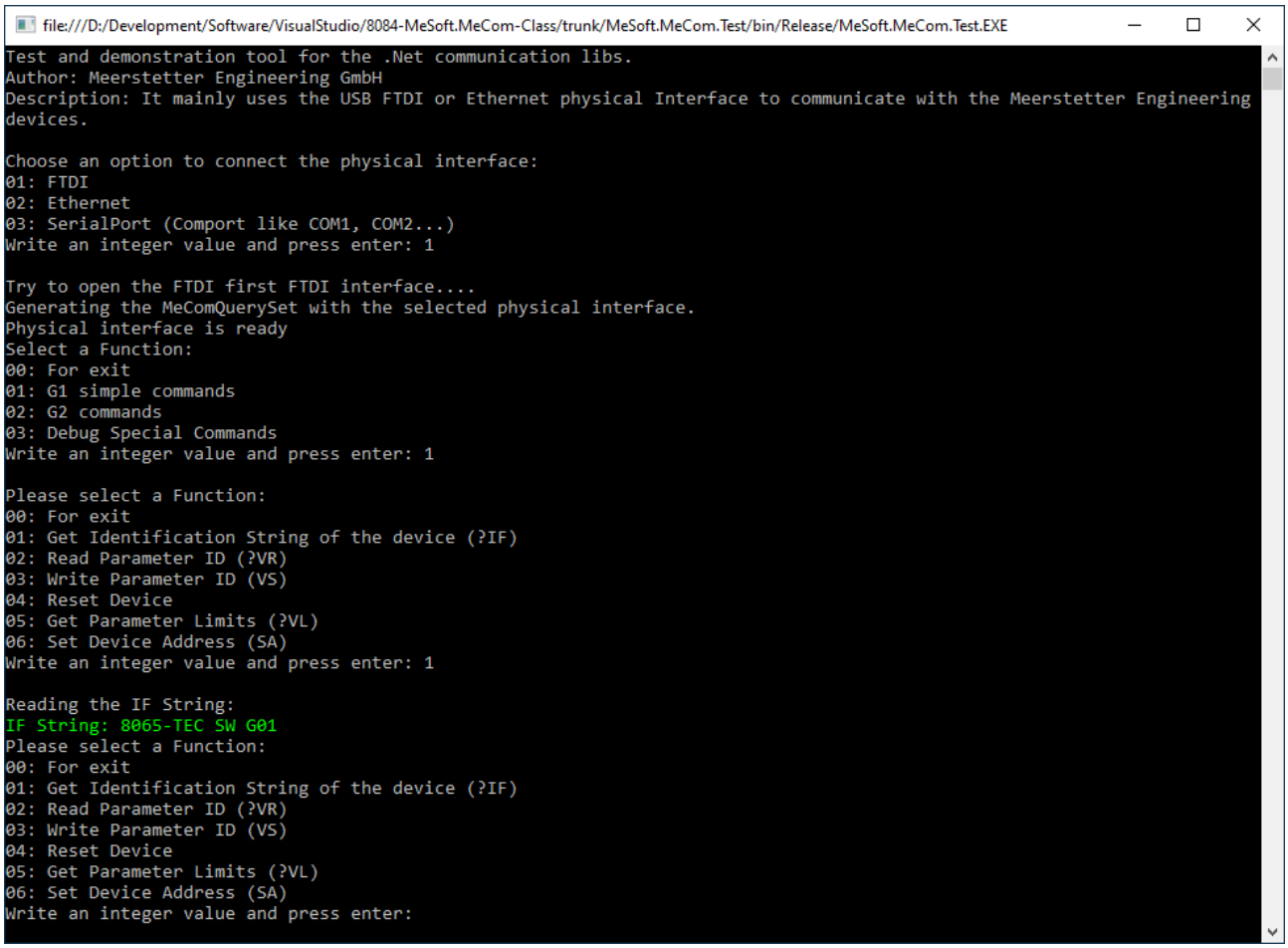
MeComAPI for .NET

Communication API for
Meerstetter Engineering Devices

The package consists of a C# Code Library and sample Applications

The Application exemplifies the control of Meerstetter Engineering Devices over a USB FTDI, Ethernet Interface, or a simple Serial Port

This package does also contain the MeCom Device Server.



Demo Application

1 General Description

1.1 General

- The MeComAPI for .NET provides C# code to fully control Meerstetter Engineering devices.
- Please take a look at the inline XML comments for more details about how the API works and how it can be used.
- The included Test project exemplifies how the API can be used and integrated in a real-world application. It implements most of the API functions that are available.
- The user will only need to call some simple functions to set or read parameters.
- The MeComAPI for .NET does everything that is necessary to have a reliable communication interface:
    - Implements nearly all functions used to communicate with the devices.
    - CRC calculations and checks.
    - Sequence Number monitoring.
    - Data resend on timeout and error management.
- To create a trace the MeSoft.Core library is also linked to the projects. In the standard configuration the whole serial communication is being traced to the trace file, which is located next to the .exe file. By changing the settings in the "appsettings.json" file, it is also possible to disable the full trace out.

1.2 Documents and Versions

This project shows the C# code implementation of the following specification documents:

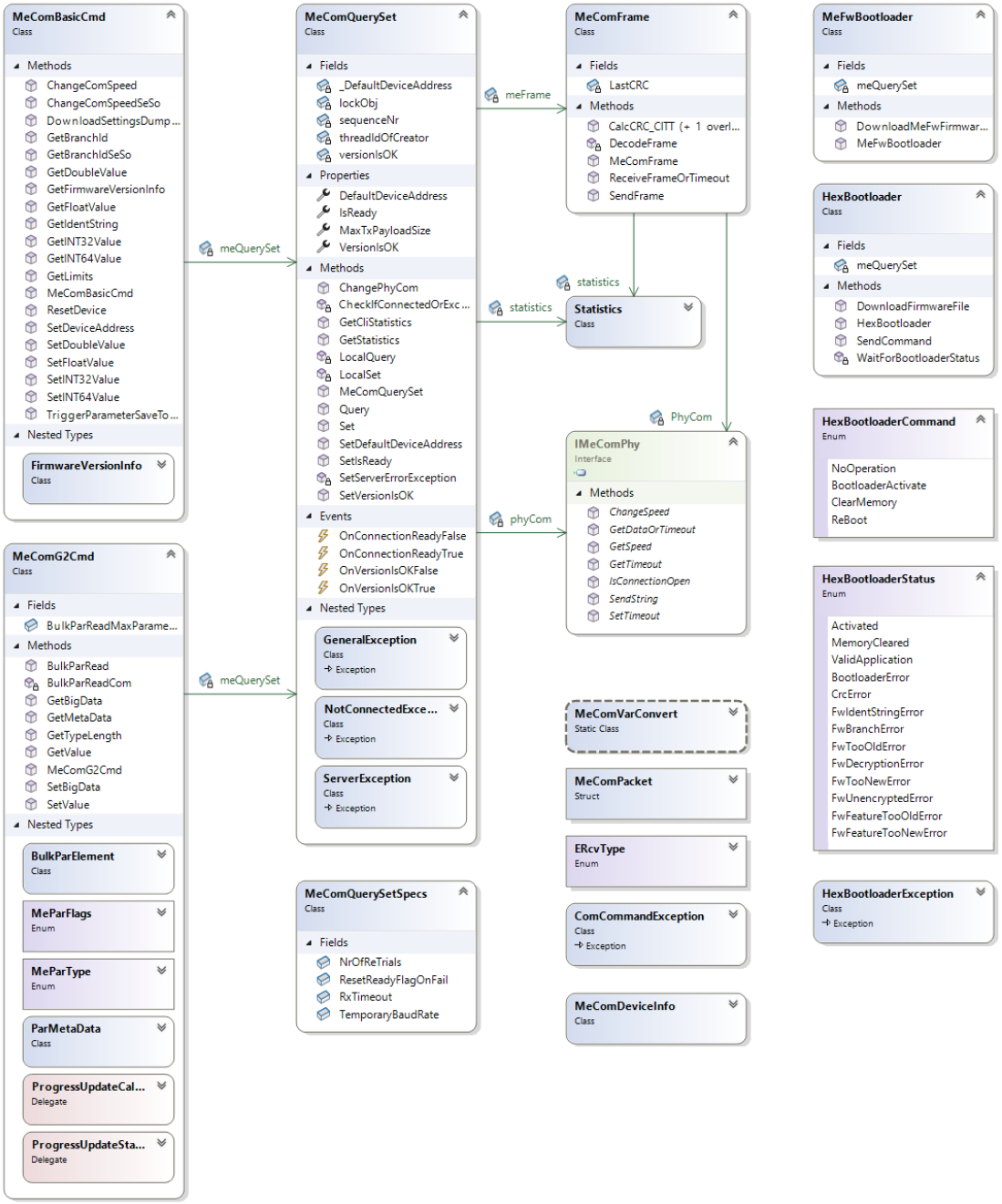
- MeCom Protocol Specification 5117
- Implements most functions of the Communication Specification documents:
    - LDD Communication Protocol 5130
    - TEC Controller Communication Protocol 5136
    - LTC Communication Protocol 5199
    - LDD-130x Communication Protocol 5260
    - LDD-1321 Communication Protocol 5294
- The solution contains also the Device Server. Documentation: 5325

2 Class Diagrams of the MeComAPI for .NET

2.1 MeSoft.MeCom.Core

The core part of the library provides the main functions to communicate with the devices.

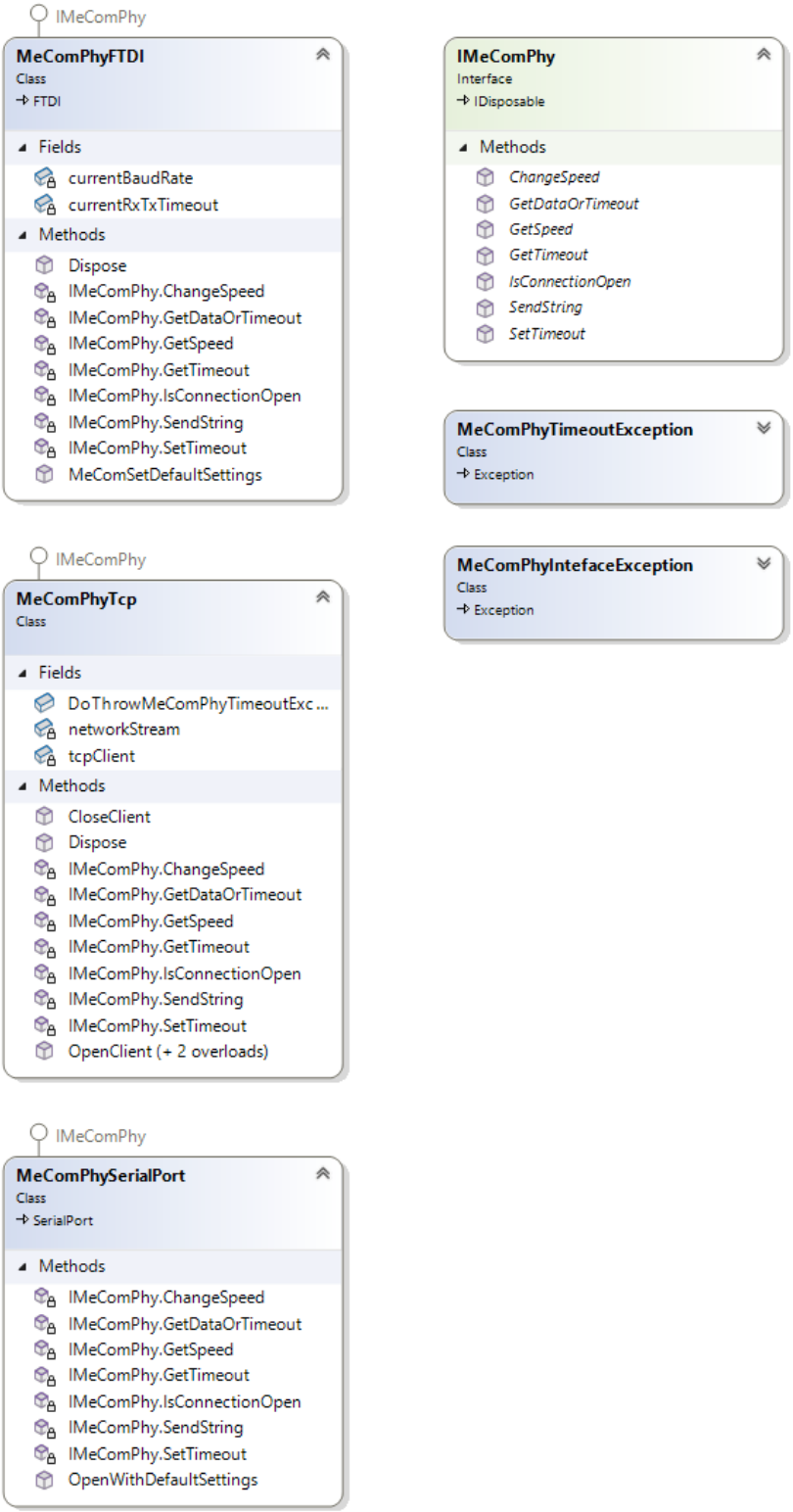
- The classes MeComBasicCmd and MeComG2Cmd provide the top functions to be used.
- The MeComQuerySet represents the layer that is responsible that the data is transferred correctly. It does automatically re-query if no data returns or the returned data is corrupted. It does also handle multi-threading queries to the same interface.
- The MeComFrame creates and decodes the effective package and transfers it to the physical layer.
- The Statistics part provides some statistics about the communication.



2.2 MeSoft.MeCom.PhyWrapper

The PhyWrapper part is responsible for connecting the physical layer.

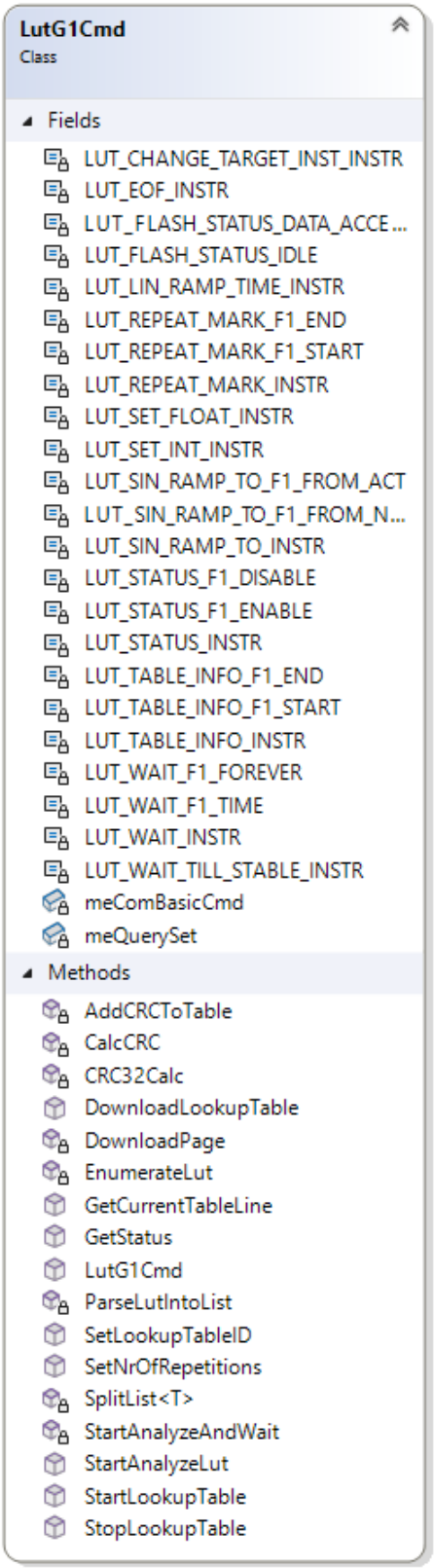
The project is easily extendable for other physical interfaces. For example, RS232 / RS485 interfaces. Please contact us directly for more information.

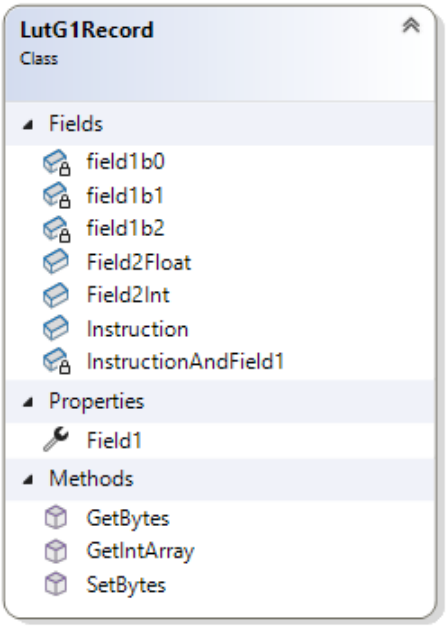


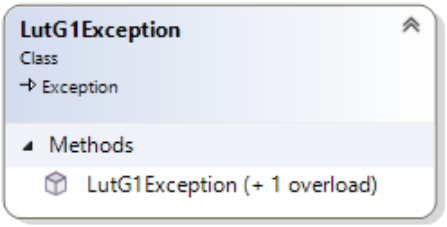
2.3 MeSoft.MeCom.TEC

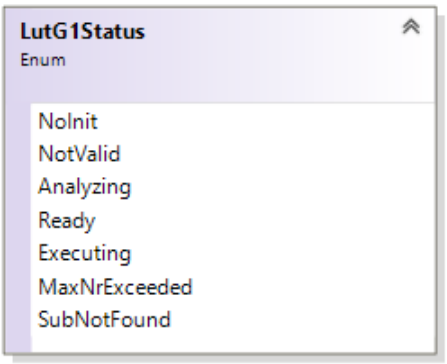
The TEC part contains all functionality that is specific to TEC Controller devices only.

2.3.1 Lookup Table

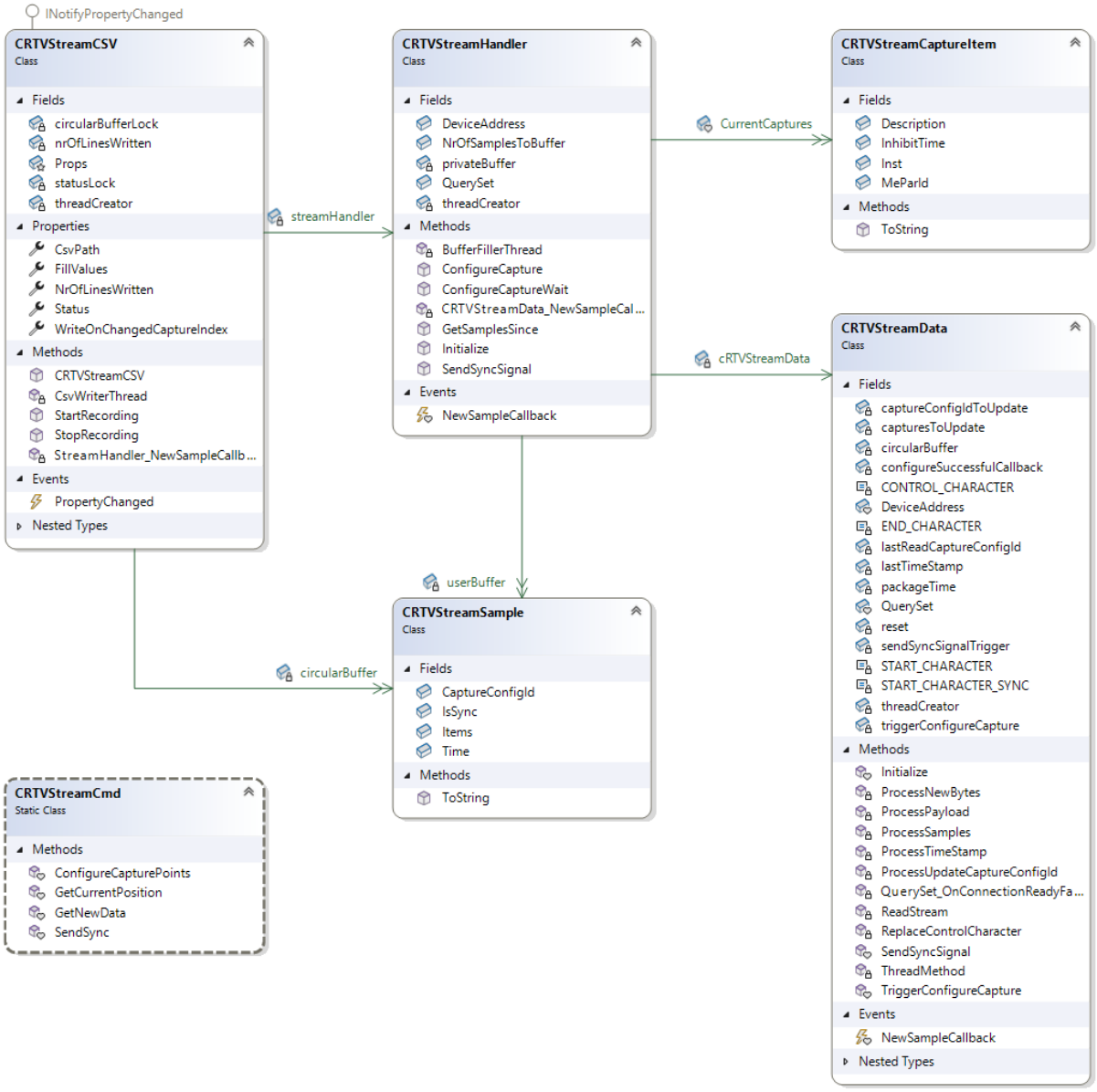




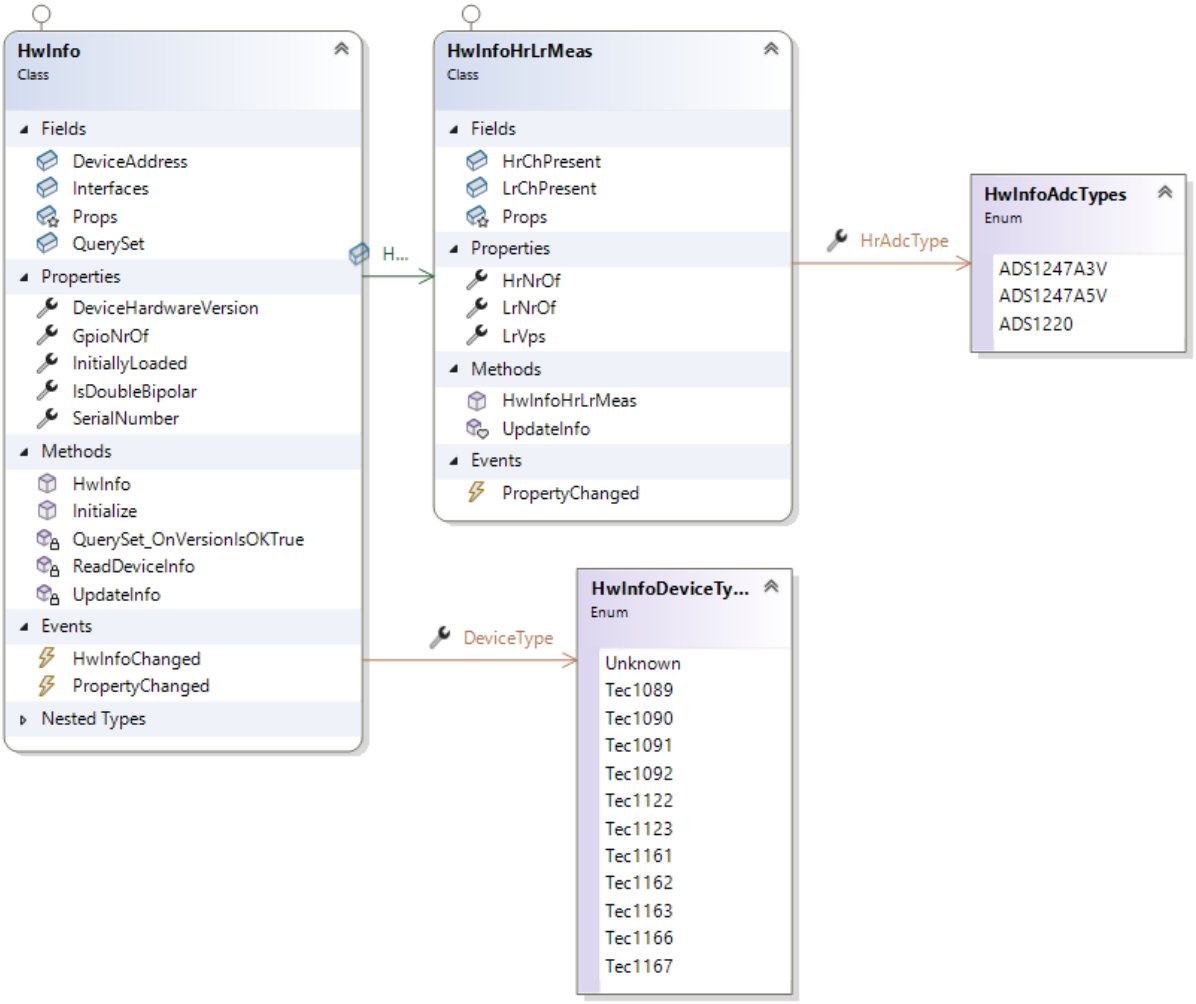




2.3.2 CRTVStream (Real Time Data)



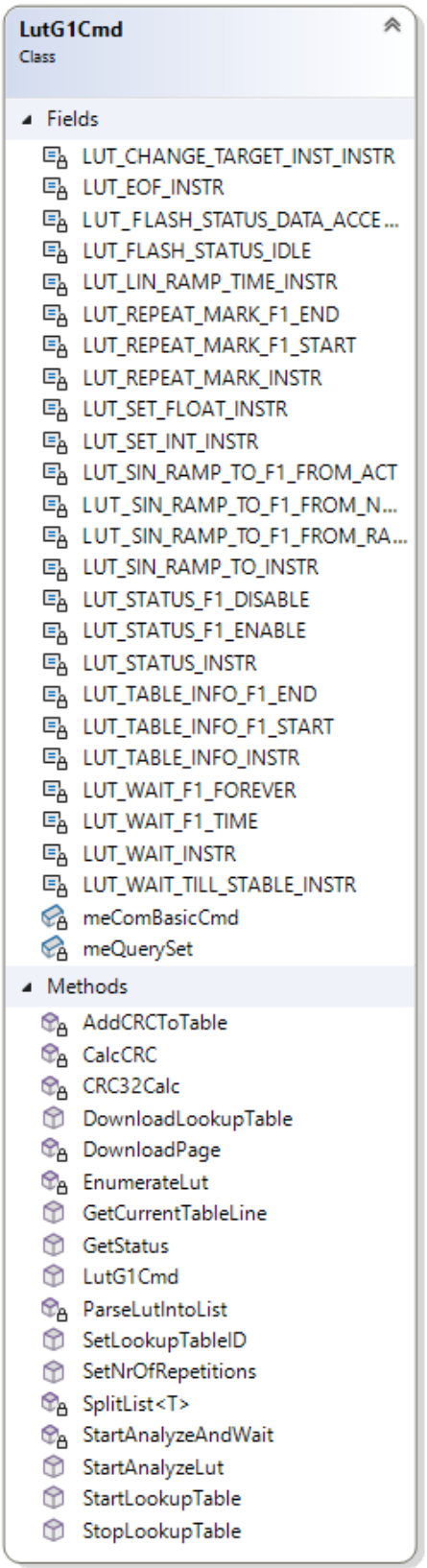
2.3.3 HwInfoProvider

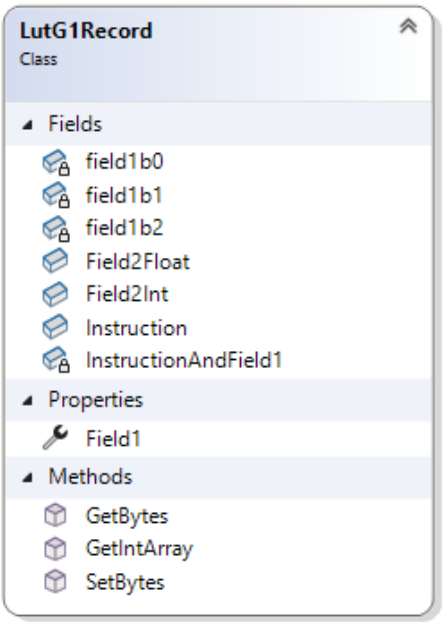


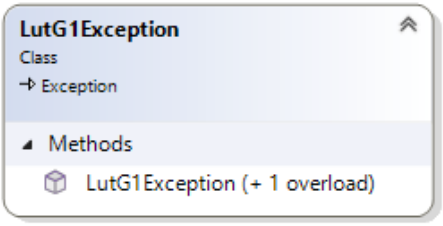
2.4 MeSoft.MeCom.LDD

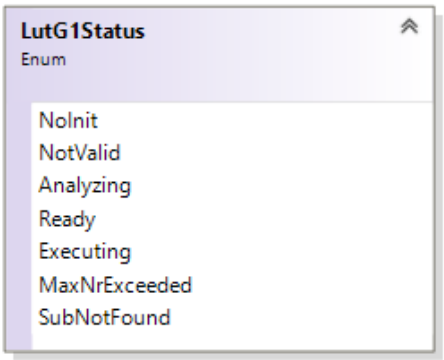
The LDD part contains functions that only concern LDD devices.

Currently LDD-1121, LDD-1124, LDD-1125, LDD-1301, LDD-1303, LDD-1321 devices are supported.



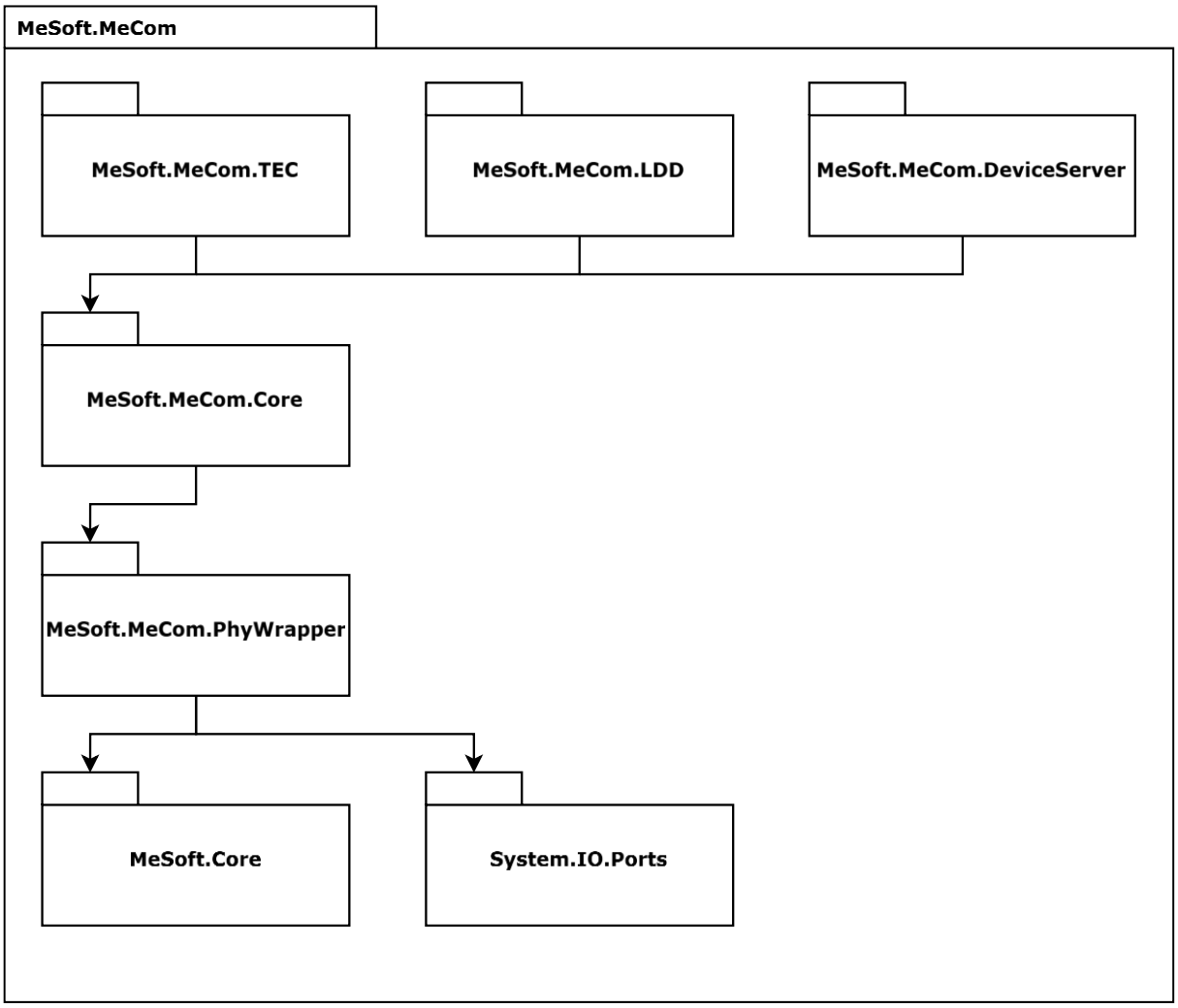






3 Package Diagram

In the following package diagram, you can see how the individual packages are connected and what external dependencies they have.



The current release contains also a project named **MeSoft.MeCom.GeneralApi**. This project will contain code for shared applications between LDD and TEC. This is currently under development.

4 Compatibility

As of v2.00 of the MeComAPI for .NET project it is now targeting .NET Standard 2.0 instead of previously .NET Framework 4.6. This means that the libraries are now compatible with .NET Framework 4.6.1 and later as well as .NET/.NET Core 2.0 and later[1]. This change was primarily made because of the announcement by Microsoft to no longer continue development on .NET Framework and instead focus on the development of ".NET" (previously called ".NET Core"). Furthermore, this change also allows the use of the libraries on other platforms (namely Linux and macOS) than just Windows (FTDI communication interface is still limited to Windows).

4.1 Runtime requirement

.NET Framework was always shipped together with the Windows operating system and there rarely was any need to manually install specific versions of it. However, this is no longer the case with .NET/.NET Core, as they are not shipped together with Windows.

If the application, you are building is targeting .NET/.NET Core instead of .NET Framework your users will have to install the runtime manually to be able to run your application. Alternatively, you can also ship the runtime together with your application. You can find downloads of the runtime on the following page by Microsoft: Download .NET (microsoft.com).

Further information about how .NET applications can be published can be found on the following page: .NET application publishing overview (microsoft.com).

[1] .NET Standard version compatibility | Microsoft

5 Compilation

5.1 Building

If you want to build the libraries or the example applications from source, you can do so using the official Visual Studio project.

To build the example applications we recommend using the included publish profiles by navigating to "Build > Publish Selection" within Visual Studio while having the respective application project selected.

5.2 Packaging

By default, the MeCom libraries are deployed as NuGet packages as of v2.00 and can be downloaded from the official nuget.org package repository (see also NuGet.org – MeerstetterEngineering).

Unfortunately, because the "MeSoft.Core" library makes use of the "Microsoft.Extensions.Configuration" package to optionally load the contents of "appsettings.json" configuration files the MeSoft.MeCom libraries are not entirely self-contained and you need to also import the external library files (*.dll) if you do not want to use NuGet Package Manager packages into your project.

Alternatively, you can also make use of the Costura.Fody or ILRepack packages, which allow you to compile the projects and their dependencies into single-file libraries.

A Change History

Date of change	Doc	API Version	Change / Reason
12 July 2016	A	1.10	• Initial Release • Tested with TEC Firmware Version 1.70 • LTC-114x Firmware Version 0.30
8 Nov 2016	B	1.20	• Add: Option to connect to a SerialPort like COM1, COM2, … • Bug: SetIsReady(true) added in the demo application. Without this it is not possible to access the com functions from a different thread.
26 Jan 2017	C	1.30	• The MeSoft.Core library has been added. This is used to output a trace. In the standard configuration, the demo application writes the whole serial communication to a trace.txt file next to the .exe file. By changing the trace level, it is also possible to disable the full printout. • A lot of the Get functions used out parameters to return the queried values. These out parameters have been replaced by the normal return parameters. This may be a breaking change for your application if you just update. • An example to get the data from the LTC-1141 has been added. • The functions do not expect an answer if you send to the address 255. This is a new definition for broadcasting. Till now, only the TEC-Family Firmware does support it in the version 3.00.

29 June 2018	D	1.40	• Add: ChangeComSpeed command (CS)
			• Big Data Commands (VB / ?VB):
			  ○ Add: BYTE Type
			  ○ Bug: SetBigData failed when no feedback function was specified.
			  ○ Add: Send and receive ASCII strings. Type is called LATIN1.
			• Add: MeSoft.MeCom.SCPI Project. This is a project which might be used as interface to our TEC Controllers, but it is not yet sure if we want to continue this project. It was used for a customer who wanted to control our TEC Controller from an Excel VBA script.
			• Add: Set device address command (SA)

14 April 2019	E	1.45	• Add: DownloadSettingsDumpFile • Add: Message if a package must be re-sent • Bug: SCPI did not start the internal thread • Bug: GetMetaData did not support Latin1 and Byte types • Add: LTC-1141 special test command (reset and check status while) • Add: GetBigData ProgressUpdateCallback and expectedNrOfElements • Bug: MeComG2Cmd SetBigData nr of package calculation wrong (caused package loss in conjunction with LTC-1141 Lookup Table download).
06 October 2020	F	1.47	• Add: Lookup table download functionality for TEC controllers.

Date of change	Doc	API Version	Change / Reason

02 March 2021	G	1.48	• Add: Download Hex firmware file functionality in MeSoft.MeCom.Core.Bootloader namespace.
			• Add: MaintenanceCommands class in MeSoft.MeCom.Test project with an example to download a Hex firmware file on a device.
			• Add: The following Lookup Table commands for TEC Controllers:
			  o Lookup Table Stop
			  o Lookup Table Status: Get Current Table Line
			  o Lookup Table ID Selection
			  o Set Nr Of Repetitions
			• Mod: Moved Download MeFw firmware file function from MeComG2Cmd class to MeFwBootloader class in the MeSoft.MeCom.Core.Bootloader namespace.

31 March 2021	H	1.49	• Fix: Lookup Table "SET_INT" command was not parsed correctly.
28 April 2023	I	2.00	• Mod: Migrate the libraries to .NET Standard 2.0 and the programs to .NET 6.0. • Add: MIT license to the project. • Add: Package diagram, compatibility, and compilation section in the documentation.
07 July 2023	I	2.01	• Add: Example application publish profiles that target Linux and macOS. • Add: 64-Bit parameter handling. • Bug: SetBigData busy timeout did not throw after timeout.
10 August 2023	J	2.02	• Fix: Update FTDI .NET wrapper to support .NET 5 and higher.
23 September 2024	K	2.10	• Add: Trigger Save to Flash (SP) command. When a parameter is changed it is now only ever changed in the volatile memory of the device until a SP command is sent to the device. Before this change parameter changes were periodically saved to the flash. • Mod: IMeComPhy interface now implements IDisposable so that the connection to a device should always be closed when the application shuts down. • Mod: SCPI program can now also be used as a command-line interface (CLI). • Add: LDD-112x Lookup Table control functions. • Add: CRTVStream classes for TEC Controllers, which can be used to subscribe to parameters and log their data down to within a 10-millisecond interval. See TEC-Family User Manual for more details regarding this feature. • Add: HwInfo class for TEC Controllers that can be used to query certain device-specific information about a connected device. • Mod: Increase TCP send and receive timeouts from 500ms to 5s. • Mod: Increase SetBigData command timeout from 500ms to 3s. • Add: GetFirmwareVersionInfo function to real older firmware version information that still use the ?VI command.

Date of change	Doc	API Version	Change / Reason
17 March 2026	L	3.04	• **Mod: MeSoft.MeCom.LDD112x renamed to MeSoft.MeCom.LDD** • Update Class Diagrams • Add: Lookup Table functions for LDD-130x and LDD-1321. • Bug: TEC: Lookup Table: Change Download Page Timeout from 500ms to 5s, because the flash erase can take longer. • Add: Meta Data: MeParFlags: RamOnly flag. • Bug: TEC: CRTVStream: The User Buffer was filled the wrong way. • Bug: CRTVStream: Handle ConfigureCapturePoints.Count = 0 correctly. • Bug: TEC: HwInfoProvider: Define default values. • Add: Device Server to the solution. • Add: MeComQuerySetSpecs: Provides the ability to change some com settings just for one query or set command. Usually needed for Change Speed commands or Bootloader commands. • Add: Core: Wait before return after a broadcast command. • Add: MeComPhyTcp: DoThrowMeComPhyTimeoutException option, in case there is just something like an Ethernet to Serial Port device connected. • Add: IMeComPhy.IsConnectionOpen • Add: ME-Device-Server-Error message detection • Add: MeComDeviceInfo Class • Add: MeSoft.MeCom.GeneralApi. (Currently under development)