LD-MRS Software Examples

Release Notes and Customer Documentation

v2014-12-18

Contents

1 Overview

The LD-MRS SoftwareExamples is the source code of an example program that can be compiled under Linux. It is written in the C++ programming language. It is intended to demonstrates the usage of different functions of the LD-MRS laserscanner for developers who want to use the LD-MRS ethernet interface.

Additional code-specific documentation is provided in the `Readme.txt` file in the project.

1.1 Disclaimer

The source code is provided “as is”, without any warranty.
All of the source code may be used in other projects. However, the source code provided here is intended to demonstrate the sensor functions in a way that is easily understandable. It is neither intended to show the “best” solution, nor is it proven to be free of errors.

2 Source code structure

2.1 Important files

The subdirectory `src` contains all of the source code. Important files are:

- main.cpp: Main project file, contains the selection of the demo application.
- devices/LD_MRS.cpp: Basic device driver for the LD-MRS. Depending on the desired function, it uses the native interface (LuxBase.cpp) or the SOPAS interface (sopas/LdmrsSopasLayer.cpp) for communication with the sensor. Also, application-specific settings must be made here.
- devices/LuxBase.cpp: Contains all of the communication via the native interface.
- sopas/LdmrsSopasLayer.cpp: Contains all of the communication via the SOPAS interface.

2.2 Software structure

Generally, a manager object is created. Then, one (or more) devices are added as data sources, and one (or more) applications are added as data users.
The manager handles all of the data traffic and connects the devices with the applications.

2.3 Building the project

In order to build the project, follow the instructions in the file `Readme.txt` which is included in the project.

3 Applications

In order to simplify the source code structure, all functions and parameters are hard-coded into the program.
To use the following applications, typically two settings have to be made inside the source code:

1. Activate the desired application. This is done by commenting in / commenting out the desired application in the file `main.cpp`.
2. Set the correct sensor options. This is done by making the appropriate settings in the device file `LD-MRS.cpp`. For example, if the FieldApp application is selected, the SOPAS interface must be enabled (`m_weWantFieldData = true`).

3.1 Application "MrsApp"

The MRS-App connects to an MRS, reads its configuration and receives all incoming data.

3.2 Application "SectorChangeApp"

The SectorChangeApp demonstrates the usage of the FlexRes ("Flexible resolution") feature. It defines several sectors with different scan resolutions and configures the sensor accordingly.
This application works only with firmware that supports FlexRes.

Supported firmware: v3.02 (or newer)

3.3 Application "FieldApp"

The FieldApp demonstrates the usage of the SOPAS interface. It removes all fields and eval cases, then creates a new field and matching eval case. It also demonstrates logging in and out, and flashing of the configuration.

Note that this application will only work with the field eval versions of the LD-MRS (LD-MRS400001, LD-MRS400102 HD).

3.4 Application "NtpTimeApp"

This application demonstrates the setting of a system time in the sensor. It waits for 2 seconds, then sets the sensor-internal system time to January 1st, 2000.

3.5 Application " ScanpointCoordinateApp"

The ScanpointCoordinateApp demonstrates the use of scanpoint coordinates and allows to verify the coordinate calculation (done in LuxBase.cpp).
It also demonstrates how to distinguish between 4- and 8-layer-scanners and their mirror sides.
In the course of this application, the LuxBase was adapted to support scanner firmware v3.03 (e.g. to recognize the new SensorInfo data structure).

4 Version history

Version	Changes
v2014-03-04	Initial version.
v2014-12-18	ScanpointCoordinateApp added. Also extended the calculation of scanpoint coordinates in LuxBase.cpp to the algorithm used in SICK API reference to support both 4- and 8-layer-scanners. Added the datatype 0x7100 (SensorInfo) to be recognized, although not decoded.