Ethernet data protocol

LD-MRS400001
LD-MRS400102
LD-MRS400001S01
LD-MRS400102S01
LD-MRS800001S01

Table of Content

1 Introduction

This document describes how data is received and transmitted from respectively to the LD-MRS via the Ethernet connection.

The addressed firmware version of the LD-MRS is **3.03**.

Addressed scanners are LD-MRS 400xxx and LD-MRS 800xxx sensors or applications using the current API/software versions. In addition, most sensor versions have a SOPAS access port for configuration with the SOPAS ET software. This SOPAS access is partially described in chapter 10 of this document.

2 General information

2.1 Ethernet configuration

The LD-MRS uses default ethernet configurations until changed by the user.

Default IP address: 192.168.0.1
Default subnet mask: 255.255.255.0
Default IP port: 12002

Note: The IP port may not be changed to a value that is identical to the SOPAS IP port (default: 2111).

2.2 Data encoding

2.2.1 Little and Big Endian

On the interface, a mixture of little and big endian encoding is used. While the “Message header”, the frame around the data payload, is encoded using big endian format, the payload itself is always encoded in little endian format.

Be sure to encode/decode correctly!

2.2.2 NTP timestamps

NTP64 timestamps represent the time encoded in 8 bytes. In order to decode NTP64 timestamps, the corresponding 8 bytes need to be interpreted as UINT64:

- The higher 4 bytes are the number of seconds since 1.1.1900 - 0:00:00.
- The lower 4 bytes represent the fractional seconds with a resolution of 2^{32} s.

2.3 Message header

Each message always starts with a message header. To resync just search for the magic word (0xAFFEC0C2).
The message header is encoded in network byte order ("big endian format").

Offset	Bytes	Data header:	datatype	Description
0	4	Magic word (0xAFFEC0C2)	UINT32	The magic word is used for searching the messages and to distinguish between different versions.
4	4	Size of previous messages	UINT32	Helps to navigate backwards through a file. Unused in live data.
8	4	Size of message data	UINT32	Size, in bytes, of the attached message data, without this header.
12	1	Reserved	UINT8	Write as 0.
13	1	DeviceID	UINT8	Unused here; write as 0. In a general datastream, may be filled with a unique ID to identify the device generating the data.
14	2	Data type	UINT16	Specifies the data type within this message.
16	8	NTP time	NTP64	Time when this message was created.
24		Message data	-	Depending on data type.

2.4 Data types

In the message header, the data type of the included message data must be specified. This is a list of the available data types:

Data type	Description
0x2010	Command. Sends a command (e.g. new parameter) to the sensor.
0x2020	Command reply. This is the answer of the LD-MRS to a command (Msg 0x2010).
0x2030	Error/Warning-Message.
0x2202	Scan data. Contains all measurements of a single scan.
0x2221	Object data. Contains all objects tracked by the sensor. This datatype is only available in the S01-versions of the sensor.
0x2805	Movement data (internal use only).
0x2850	Ego motion data. Sends information about current vehicle movement to

Data type	Description
	the sensor.
0x7100	LDMRS SensorInfo structure. Contains additional internal data related to a scan.

3 LD-MRS scan data: Data type 0x2202

Scan data available from LD-MRS. Each scan data block starts with a header followed by the scan point list.

For angle information the unit angle ticks is used. A LD-MRS uses 11520 ticks per rotation (see also Angle ticks per rotation below). Thus the angular resolution is 1/32°. This value is needed to convert angle ticks:

$$\text{angle} = 2\pi \frac{\text{angle ticks}}{\text{angle ticks per rotation}}$$

Note: Scans without stable mirror rotation should be considered as invalid and ignored. The mirror rotation is stable when bit 3 of the Scanner status (“frequency locked”) is set. This is the case when the sensor is starting up, but may also occur during normal operation due to strong sensor movement. These scans are sent on the interface only to provide the header information including the sensor status.

Angles are given in the ISO 8855 / DIN 70000 scanner coordinate system.

Offset	Bytes	Scan header:	datatype	Description
0	2	Scan number	UINT16	The number of this scan. The number will be increased from scan to scan.
2	2	Scanner status	bit field 16 bits	Bit field. See chapter 5.5 for the format description.
4	2	Sync phase offset	UINT16	Phase difference in angle ticks between sync signal and scanner mirror crossing the synchronization angle.
6	8	Scan start time NTP	NTP64	NTP time when the first/last measurement was done.
14	8	Scan end time NTP	NTP64	
22	2	Angle ticks per rotation	UINT16	Number of angle ticks per rotation.
24	2	Start angle	INT16	Start/end angle in angle ticks of this scan.
26	2	End angle	INT16	
28	2	Scan points	UINT16	Number of scan point transmitted in this scan.
30	2	Mounting position yaw angle	INT16	Rotation of the scanner around the axes of the reference coordinate system. All angles are given in angle ticks. Order of translation and rotation is essential: Yaw->Pitch->Roll->Translation.
32	2	Mounting position pitch angle	INT16	
34	2	Mounting position roll angle	INT16	

Offset	Bytes	Scan header:	datatype	Description
36	2	Mounting position x	INT16	Scan data is given in the scanner coordinate system without any transformation. Mounting position of the scanner relative to the reference coordinate system (ISO 8855 / DIN 70000 coordinate system). The origin is located on flat ground under the center of the rear axle. X-axis faces to the vehicle front resp. straight driving direction. Y-axis faces left. The mounting position is needed for ego motion compensation (only available if scanner x-y-plane is almost parallel to the ground). All coordinates are given in centimeters. Order of translation and rotation is essential (Rotation -> Translation). The mounting position is used for ego motion compensation, not to transform scan data but is available for further processing steps.
38	2	Mounting position y	INT16	
40	2	Mounting position z	INT16	
42	2	Processing flags	UINT16	Internal processing flags of the sensor: Bit 0: Ground detection performed (0 = false, 1 = true) Bit 1: Dirt detection performed Bit 2: Rain detection performed Bit 5: Transparency detection performed Bit 6: Horizontal angle offset added **Bit 10: Mirror side: 0=front (for 8-Layer, tilted downward), 1=rear (for 8-layer, tilted upward)** All other flags are reserved for internal use and should not be evaluated.
44		Scan Point List	Scan Point	Array of scan points. See number of scan points above and point information below.

Offset	Bytes	Scan point:	datatype	Description
0	1	Layer	UINT4	Scan layer of this point (zero-based). Encoded in bits 0..3

Offset	Bytes	Scan point:	datatype	Description
				of this byte. Valid values are 0-3. Note that the corresponding mirror side (required for 8-layer sensors) is coded in the processing flags (see above).
		Echo	UINT4	Echo number of this point. 0 is the first measurement of this shot, 1 the second (if available), and so on. Encoded in bits 4..7 of this byte.
1	1	Flags	Bit field (8 bits)	See chapter 3.1
2	2	Horizontal angle	INT16	Angle of this point in angle ticks in the scanner coordinate system
4	2	Radial distance	UINT16	Distance of this point in the scanner coordinate system, in cm
6	2	Echo pulse width	UINT16	Detected width of this echo pulse, in cm
8	2	Reserved	UINT16	-

3.1 Scan point flags

The scan point flags, a bit field of 8 bits for each scanpoint, contain details about the measurement:

Bit	Mask	Name	Description
0	0x01	Transparent	When set, there is at least one more echo (measurement) behind this scan point (B or C echo). Note that multiple measurements in one beam occur under two conditions: 1) when measuring through something transparent, such as glass or dust, or 2) when enough energy was reflected off a surface to trigger another measurement
1	0x02	Atmospheric noise (Clutter)	When set, the scan point is classified as atmospheric noise such as rain, dust, or similar.
2	0x04	Ground	When set, the scan point is classified as ground. Note: This flag is set by the sensor application firmware and is only available if the corresponding processing is active.
3	0x08	Dirt	When set, the scan point is classified as dirt.

Bit	Mask	Name	Description
			The main cause is a dirty front screen of the sensor housing.
4	0x10	Threshold switching	This is an internal flag needed for atmospheric noise detection. When set, the scan point was measured in a shot with threshold switching.
5	0x20	Invalid	This is an internal flag needed for atmospheric noise detection. When set, the scan point is considered invalid. Invalid points may occur due to double echos caused by switching thresholds. Note that invalid points will not be sent via Ethernet, so this flag should never appear outside of the sensor.
6	0x40	Near range	This is an internal flag needed for atmospheric noise detection. When set, the scan point was measured with the near range threshold.
7	0x80	Unused	Unused, ignore.

Note that most of the scan point information is needed for internal processing and is subject to change without notice. External applications should only use `Transparent`, `Atmospheric noise`, `Ground` and `Dirt`.

4 LD-MRS object data: Data type 0x2221

Object data available from LD-MRS4xxxxx.S01
Each data block starts with a header followed by the object list. Each object has a list of contour points. The total size of this data type varies with the number of objects and its contour points.

Object header (10 bytes, plus object data):

Offset	Bytes	Object header:	datatype	Description
0	8	Scan start timestamp	NTP64	Time stamp of the first measurement of the scan these objects are updated with.
8	2	Number of objects	UINT16	The number of objects transmitted in this message.
10	(variable)	List of objects	Object	Array of objects.

For each object:

Offset	Bytes	Object: content	datatype	Description
0	2	Object ID	UINT16	ID of this object from tracking.
2	2	Object age	UINT16	Number of scans this object has been tracked for.

Offset	Bytes	Object: content	datatype	Description
4	2	Object prediction age	UINT16	Number of scans this object has currently been predicted for without measurement update. Set to 0 as soon as a measurement update is available.
6	2	Relative timestamp	UINT16	Timestamp of this object relative to the scan start time in ms. The time is based on the object reference point.
8	4	Reference point	Point2D	Depending on tracking this is the tracked object reference point (e.g. center of gravity) in cm. See below for Point2D.
12	4	Reference point sigma	Point2D	Standard deviation of the estimated reference point position in cm.
16	4	Closest point	Point2D	Unfiltered position of the closest object point in cm.
20	4	Bounding box center	Point2D	Center and size in cm of a rectangle in the reference coordinate system containing all object points. See below for Size2D.
24	4	Bounding box size	Size2D	
28	4	Object box center	Point2D	Box center in the reference coordinate system in cm. Box size in cm and orientation in 1/32° in the object coordinate system (box rotated in the direction of the velocity vector; in reference coordinate system).
32	4	Object box size	Size2D	
36	2	Object box orientation	INT16	
38	4	Absolute velocity	Point2D	Velocity of this object in cm/s with ego motion taken into account. This velocity is based on the reference coordinate system which is compensated by the ego motion. Value set to 0x8000 if invalid.
42	4	Absolute velocity sigma	Size2D	Standard deviation of the estimated absolute velocity, in cm/s.
46	4	Relative velocity	Point2D	Velocity of this object in cm/s without ego motion compensation (sensor/vehicle is seen as stationary).
50	2	Reserved	UINT16	reserved

Offset	Bytes	Object: content	datatype	Description
52	2	Reserved	UINT16	reserved
54	2	Reserved	UINT16	reserved
56	2	Number of contour points	UINT16	The number of contour points of this object. 0..0xFFFE: Number of contour points. 0xFFFF: The object does not have current contour information as it is predicted without scan data. In this case, the list of contour points consists of 1 point, the predicted closest point of the object.
58		List of contour points	Point2D	Array of contour points (number of points see above). Units are [cm].

Point2D data structure (4 bytes total):

Offset	Bytes	Point2D:	datatype	Description
0	2	Position x	INT16	X-part/coordinate of this value/point.
2	2	Position y	INT16	Y-part/coordinate of this value/point.

Size2D data structure (4 bytes total):

Offset	Bytes	Size2D	datatype	Description
0	2	Size x	UINT16	X-value/size/width.
2	2	Size y	UINT16	Y-value/size/length.

5 LD-MRS command interface

5.1 Command header

For sending commands to the LD-MRS the data type 0x2010 is used. The command consists of two parts, the command header and the command data. The command data depends on the individual command and is described in the following sections.

Command header (4 bytes, plus command data)

Offset	Bytes	LD-MRS command	datatype	Description
0	2	Command ID	UINT16	See next sections for a detailed list of commands and according options/parameters.
2	2	Reserved	UINT16	Reserved, send as 0.
4	(variable)	Command Data	-	Depending on command. Some commands do not have any command data.

5.2 Command reply

The LD-MRS replies to a command with a dedicated reply message with the datatype 0x2020 (Encoding: Little endian format!).

5.2.1 Successful completion of the command

If the command was successful, the Reply ID (see table below) is identical to the Command ID. Example: Command was 0x0030 → Reply ID = 0x0030.

Offset	Bytes	LD-MRS reply	Content type	Description
0	2	Reply ID	UINT16	If a command succeeded, the reply ID is equal to the corresponding command ID. If a command failed, the reply ID is the command ID + 0x8000. Thus, the most significant bit indicates a failed command.
2	(variable)	Reply data	-	Depending on the corresponding command this reply is related to. May be completely missing for some commands and if a command failed. See detailed command

Offset	Bytes	LD-MRS reply	Content type	Description
				description below.

5.2.2 Command failure

If the command has failed for any reason, the Reply ID is a copy of the Command ID, but with its bit 15 set. Thus, the most significant bit indicates a failed command. Example: Command was 0x0030 and has failed → Reply ID = 0x8030.

Offset	Bytes	LD-MRS reply	Content type	Description
0	2	Reply ID	UINT16	For a failed command, the reply ID is the command ID + 0x8000.
2 ff.	30	(Status message)	n/a	Contents of the GetStatus reply, starting with Byte 2 (Firmware version). See 5.3.2 for details.

5.3 LD-MRS commands and command replies – data types 0x2010/ 0x2020

5.3.1 Reset

Resets the scanner, and restarts it with the saved parameters. All unsaved changes will be lost.
The sensor should be in IDLE mode for at least 1 s when this command is sent. The correct reset sequence is therefore:

- `StopMeasure`
- wait for 1 s
- `Reset`

After the reset, it will take the scanner approximately 20 s before it will be sending scan data again.

Command ID = 0x0000
Command data: none.

Reply: For this command (Reset), no reply is sent.

5.3.2 Get Status

Command ID = 0x0001
Command data: none.

Reply:

Offset	Bytes	LD-MRS reply	Content type	Description
0	2	0x0001	UINT16	ID - Status request
2	2	Firmware version	UINT16	Version of the DSP firmware. See chapter 5.5 for the format description.
4	2	FPGA version	UINT16	Version of the FPGA code. See chapter 5.5 for the format description.
6	2	Scanner status	UINT16	Bit field. See chapter 5.5 for the format description.
8	2	Reserved	UINT16	Internal parameter. DSP firmware software repository index.
10	2	Reserved	UINT16	Internal parameter. Scanner type.
12	2	Temperature	UINT16	Sensor-internal temperature. See chapter 5.5 for the format description.
14	2	Serial number 0	UINT16	These are the first 4 digits of the serial number. Format is YYCW; e.g. 0x1140 = year '11, calendar week 40. In this example, the serial number of the sensor starts with 1140… This value is valid if the lower byte of serial number 2 is 0x01.
16	2	Serial number 1	UINT16	Counter of serial number. These are the lower 16 bits of the serial number. E.g. 0x000A => the serial number of the scanner ends with …0010. Together with the example above, this scanner has the serial number 114000010. This value is valid if the lower byte of serial number 2 is 0x01.
18	2	Serial number 2	UINT16	Upper byte: High-Byte of Counter of serial number (currently unused); Lower byte: Flag for serial number encoding (must be 0x01).
20	6	FPGA time stamp	[3] * UINT16	Date/Time of the FPGA firmware version. See chapter 5.5 for the format description.
26	6	DSP time stamp	[3] * UINT16	Date/Time of the DSP firmware version. See chapter 5.5 for the format description.

5.3.3 SaveConfig

Saves all current parameters permanently.

Command ID = 0x0004
Command data: none.
Reply: The command SaveConfig will be acknowledged by the same command ID without command reply data.

5.3.4 Set Parameter

Sets a parameter to the given value. The change is valid until the next power-off. In order to store the change(s) permanently, use the SaveConfig command after the changes have been made.

Command ID = 0x0010
Command data: see table below

Offset	Bytes	LD-MRS command	datatype	Description
0	4	(header)	n/a	(Command header)
4	2	Parameter index	UINT16	Refer to LD-MRS parameter list (see chapter 5.4).
6	4	Parameter value	UINT32	New value of the parameter. Datatype is according to parameter list. If e.g. a 2 byte value is set, use the first 2 bytes. Fill the remaining 2 bytes with 0.

Reply: The command Set Parameter will be acknowledged by the same command ID without any command reply data.

5.3.5 Get Parameter

Read a single parameter from the LD-MRS.

Command ID = 0x0011
Command data: see table below

Offset	Bytes	LD-MRS command	datatype	Description
0	4	(header)	n/a	(Command header)
4	2	Parameter index	UINT16	Refer to LD-MRS parameter list (see chapter 5.4).

Reply:

Offset	Bytes	LD-MRS reply	Content type	Description
0	2	0x0011	UINT16	ID - Read a single Parameter by its index from the LD-MRS.
2	2	Parameter index	UINT16	Refer to LD-MRS parameter list (see chapter 5.4).
4	4	Parameter	UINT32	Current value of the parameter. Datatype is according to parameter list. If e.g. a 2 byte value is read, read the first 2 bytes.

5.3.6 Reset Default Parameters

Resets all parameters to their factory default values.

Command ID = 0x001A
Command data: none.
Reply: The command Reset Default Parameters will be acknowledged by the same command ID without any command reply data.

5.3.7 Start Measure

Starts scanning.
Also starts the motor if it was not already spinning. This may take approx. 10-20 seconds.

Command ID = 0x0020
Command data: none.
Reply: The command Start Measure will be acknowledged by the same command ID without any command reply data.

5.3.8 Stop Measure

Stops scanning.
Also stops the motor from spinning.

Command ID = 0x0021
Command data: none.
Reply: The command Stop Measure will be acknowledged by the same command ID without any command reply data.

5.3.9 SetNTPTimestampSec

Sets the “seconds”-part of the NTP timestamp.

Command ID = 0x0030
Command data: see below

Offset	Bytes	LD-MRS command:	datatype	Description
0	4	(header)	n/a	(Command header)
4	2	(reserved)	n/a	(unused, send as 0)
6	4	Timestamp	UINT32	Seconds (NTP format). The time will be set in the sensor when the fractional seconds command is received (see 5.3.10).

Reply: The command SetNTPTimestampSec will be acknowledged by the same command ID without any command reply data.

For an example of how to set the NTP timestamp, please refer to chapter 9.2.

5.3.10 SetNTPTimestampFracSec

Sets the “fractional seconds”-part of the NTP timestamp.
Attention: Before this command can be executed, first command "SetNTPTimestampSec” (0x0030) must be sent (see 5.3.9)!

Command ID = 0x0031
Command data: see below

Offset	Bytes	LD-MRS command	datatype	Description
0	4	(header)	n/a	(Command header)
4	2	(reserved)	n/a	(unused, send as 0)
6	4	Timestamp	UINT32	Fractional seconds (NTP format).

Reply: The command SetNTPTimestampFracSec will be acknowledged by the same command ID without any command reply data.

For an example of how to set the NTP timestamp, please refer to chapter 9.2.

5.4 LD-MRS parameter list

This table lists all available LD-MRS parameters. Please refer to the “SetParameter” and “GetParameter” commands for details on reading and setting these parameters.

IP address, subnet mask and standard gateway encode the data as UINT32 value which is built like that: aa.bb.cc.dd = 0xaabbccdd. Due to little endian byte order this value must be sent as 0xddccbbaa.

Bytes	Parameter index	LD-MRS parameter	datatype	Description
4	0x1000	IP address	UINT32	Valid: all
2	0x1001	TCP Port	UINT32	Valid: all
4	0x1002	Subnet Mask	UINT32	Valid: all
4	0x1003	Standard gateway	UINT32	Valid: all
4	0x1010	CAN Base ID	UINT32	Valid: value <= 0x7F0
2	0x1011	CAN Baud Rate	UINT16	in kBaud - next matching value (1000 kBaud, 500 kBaud, 250 kBaud, 125 kBaud) will be used.
2	0x1012	Data Output Flag	16 bit field	Bit true: disable output, false: enable output. 0xFFFF is invalid. bit0: ETH scan data bit1: reserved/internal bit2: ETH object data bit3: ETH vehicle data bit4: ETH errors/warnings bit5: CAN errors/warnings bit6: CAN object data bit7...15: reserved
2	0x1013	maxObjectsViaCAN	UINT16	<= 65 (max. number of objects) limited by tracking and CAN bus capacity.
2	0x1014	ContourPointDensity	UINT16	Valid: < 3 0: closest point only 1: low density 2: high density
2	0x1015	ObjectPriorizationCriterion	UINT16	Valid: < 2 Used to reduce transmitted objects via CAN. Decision which objects are discarded is based on this criterion. 0: Radial 1: Look ahead
2	0x1016	CAN object data options	16 bit field	Valid: all bit 0: 0 = absolute velocities, 1 = relative velocities bit 1: 0 = boxes are object boxes, 1 = boxes are bounding boxes bits 2…15: reserved

Bytes	Parameter index	LD-MRS parameter	datatype	Description
2	0x1017	Minimum Object Age	UINT16	Valid: all Minimum tracking age (number of scans) of an object to be transmitted.
2	0x1018	Maximum Prediction Age	UINT16	Valid: all Maximum prediction age (number of scans) of an object to be transmitted.
2	0x1100	Start angle	INT16	In 1/32°, in the sensor coordinate system. Valid: 1600…-1919. Start angle > end angle!
2	0x1101	End angle	INT16	In 1/32°, in the sensor coordinate system. Valid: 1599…-1920. Start angle > end angle!
2	0x1102	Scan frequency	UINT16	In 1/256 Hz. Valid: 3200 (12.5 Hz) 6400 (25.0 Hz) 12800 (50.0 Hz)
2	0x1103	Sync angle offset	INT14 (!) (16 bits transferred)	In 1/32° in the sensor coordinate system. Valid: -5760…+5759 (-180°…+180°). Bits 14 and 15 are ignored!
2	0x1104	angular resolution type	UINT16	0: Focused 1: Constant 2: FlexRes (user-defined sectors)
2	0x1105	angleTicksPerRotation	UINT16	11520 (read only), constant for LD-MRS
2	0x1108	RangeReduction	UINT16	Available for LDMRS800001.S01 only 0: full sensitivity (default) 1: lower 4 layers reduced 2: upper 4 layers reduced 3: both reduced

Bytes	Parameter index	LD-MRS parameter	datatype	Description
2	0x1109	Upside down mode	UINT16	Available for LDMRS800001.S01 only 0: device not in upside down mode (default) 1: upside down mode active
2	0x110A	Ignore near range	UINT16	Available for LDMRS800001.S01 only 0: do not ignore points in near range (up to 15m) (default) 1: ignore points in near range if 0x1108 is 1
2	0x110B	Sensitivity control active	UITN16	0: not active (default) 1: Sensitivity will be reduced dynamically down to 60% in case of direct sun light.
2	0x1200	SensorMounting_X	INT16	In cm, related to vehicle reference point, rear axle. Order of translation and rotation is essential (Rotation -> Translation).
2	0x1201	SensorMounting_Y	INT16	In cm, related to vehicle reference point, rear axle. Order of translation and rotation is essential (Rotation -> Translation).
2	0x1202	SensorMounting_Z	INT16	In cm, related to vehicle reference point, rear axle. Order of translation and rotation is essential (Rotation -> Translation).
2	0x1203	SensorMounting_Yaw	INT16	In 1/32°, order of translation and rotation is essential (Yaw->Pitch->Roll-> Translation).
2	0x1204	SensorMounting_Pitch	INT16	In 1/32°, order of translation and rotation is essential (Yaw->Pitch->Roll-> Translation).

Bytes	Parameter index	LD-MRS parameter	datatype	Description
2	0x1205	SensorMounting_Roll	INT16	In 1/32°, order of translation and rotation is essential (Yaw->Pitch->Roll-> Translation).
2	0x1206	VehicleFrontToFrontAxle	UINT16	valid: all; in cm
2	0x1207	FrontAxleToRearAxle	UINT16	valid: all; in cm
2	0x1208	RearAxleToVehicleRear	UINT16	valid: all; in cm
2	0x1209	VehicleWidth	UINT16	Width of the vehicle; valid: all; in cm.
2	0x120A	steerRatioType	UINT16	Reserved, internal.
4	0x120C	SteerRatioPoly0	Float32	Reserved, internal. Valid: all
4	0x120D	SteerRatioPoly1	Float32	Reserved, internal. Valid: all
4	0x120E	SteerRatioPoly2	Float32	Reserved, internal. Valid: all
4	0x120F	SteerRatioPoly3	Float32	Reserved, internal. Valid: all
2	0x1210	Vehicle Motion Data Flags	16 bit field	Reserved, internal. Bit 0: Vehicle Motion data expected: 1=true, 0=false Bits 1 to 15: reserved
2	0x2208	EnableSensorInfo	UINT16	Enable flag for “SensorInfo”-data. Default is 0 (disabled). When set to 1, the sensor sends SensorInfo data with each scan. Please refer to chapter 8 for details. This setting is non-persistent and has to be re-activated after each reboot.
2	0x3302	BeamTilt	CompressedRadian	Beam tilt angle. For standard devices, this angle is 0. For 8-layer devices, this is the tilt of the scan center in 0° (forward) direction - typically either 1.2° or 1.6°. For a description of the data format, see chapter 5.4.1.

Bytes	Parameter index	LD-MRS parameter	datatype	Description
4	0x3500	Timemeter	UINT32	Current timemeter value, in [minutes]. This is the overall power-on-time of the sensor since production. This parameter is read-only.
2	0x3600	Enable APD control	UINT16	LDRMS400001.S09 only Enables adaptive sensitivity control. If disabled, sensor works at maximum sensitivity. Valid values are 1 (Enabled, DEFAULT) or 0 (Disabled). This parameter is not persistent; it will not be saved in the flash memory and is set to default (=enabled) at reboot. This parameter is available only in DSP firmware version 2.2.09. See parameter 0x110B since FW V3.03
2	0x4000	NumSectors	UINT16	FlexRes feature: Number of angle sectors. This defines the number of valid sectors, set with the following parameters. The sensor checks the validity of the sectors before applying the configuration. In case of an error, read parameter 0x7000 for further information.
2	0x4001	StartAngle, Sector 1	INT16	In 1/32°, in the sensor coordinate system. Valid: 1600…-1919. This is the start angle of the first sector. The angular resolution for this sector is set with parameter 0x4009.
2	0x4002	StartAngle, Sector 2	INT16	In 1/32°, in the sensor coordinate system. Valid: 1600…-1919. This is the start angle of the second sector. The angular resolution for this sector is set with parameter 0x400A.

Bytes	Parameter index	LD-MRS parameter	datatype	Description
2	0x4003	StartAngle, Sector 3	INT16	…see above.
2	0x4004	StartAngle, Sector 4	INT16	…see above.
2	0x4005	StartAngle, Sector 5	INT16	…see above.
2	0x4006	StartAngle, Sector 6	INT16	…see above.
2	0x4007	StartAngle, Sector 7	INT16	…see above.
2	0x4008	StartAngle, Sector 8	INT16	…see above.
2	0x4009	Angular resolution, Sector 1	INT16	Angular resolution of first sector. Valid values are 32 (1.0°), 16 (0.5°), 8 (0.25°) and 4 (0.125°).
2	0x400A	Angular resolution, Sector 2	INT16	Angular resolution of second sector.
2	0x400B	Angular resolution, Sector 3	INT16	…see above.
2	0x400C	Angular resolution, Sector 4	INT16	…see above.
2	0x400D	Angular resolution, Sector 5	INT16	…see above.
2	0x400E	Angular resolution, Sector 6	INT16	…see above.
2	0x400F	Angular resolution, Sector 7	INT16	…see above.
2	0x4010	Angular resolution, Sector 8	INT16	…see above.

Bytes	Parameter index	LD-MRS parameter	datatype	Description
4	0x7000	Detailed error code for FlexRes	UINT32	Detailed error information for FlexRes feature. When parameter setting of 0x4000 fails, reading this parameter will give the reason. 0x006C: The number of shots per scan is higher than 440. 0x006D: The sectors with a resolution of 0.125° sum up to more than 20°. 0x006E: The scan frequency is not 12.5Hz. 0x006F: The start angles of the sectors decrease not strictly monotone. 0x0070: Could not set FlexRes parameter because the sensor is not idle and in flex res mode. 0x0071: The resolution of one sector is not 4,8,16 or 32 (0.125°, 0.25°, 0.5°, 1°) 0x0072: The number of sectors is larger than 8

5.4.1 Coding of datatype “CompressedRadian”

This datatype is sometimes used to transport angle information.
To convert an angle to CompressedRadian, do this:

- Normalize the angle to the interval [-pi, +pi).
- Multiply the normalized value by 10000
- Round the result, and convert the result to INT16

The resulting INT16 value is the angle with a resolution of "tenth of mrad" and its range extends from -31416 to +31416 (which is the rounded and quantized form of Pi, 3.14159).
Do decode the value, use the algorithm above backwards.

5.5 Coding of LD-MRS Status / Version information

Name	Datatype	Bytes	Description
Firmware version	UINT16	2	Examples: 0x1230 = version 1.23.0; 0x3011 = version 3.01.1
FPGA version	(see DSP firmware version)	(see DSP firmware version)	(see DSP firmware version)
Scanner status	UINT16	2	Bit field, with the following meaning for every bit: Bit 15 ...6: reserved / internal Bit 5: phase locked (Motor control) Bit 4: external sync signal available Bit 3: frequency locked (Mirror rotation is stable) Bit 2: reserved / internal Bit 1: laser on Bit 0: motor on
Temperature	UINT16	2	If value is <= 0x7FFF: T[°C] = - (value - 579.2364) / 3.63 else T[°C] = invalid. Example: Sensor value reads 0x017D = 381. T[°C] = - (381 - 579.2364) / 3.63 = 54.6
FPGA time stamp	3*UINT16	6	Format is YYYY MMDD hhmm. Example: 0x2010 0x1104 0x0921 = Nov. 04, 2010, 9:21 h
DSP time stamp	(see FPGA time stamp)	(see FPGA time stamp)	(see FPGA time stamp)

6 LD-MRS errors and warnings - Data type 0x2030

As soon as the LD-MRS detects an error or wants to emit a warning, this message is sent. Errors and warning bits are reset after sending this message.
This message will be sent periodically as long as errors of warnings persist.
The data is encoded in little endian format!

Offset	Bytes	LD-MRS error/warning registers:	datatype	Description
0	2	Error register 1	bit field 16 bits	See below
2	2	Error register 2	bit field 16 bits	
4	2	Warning register 1	bit field 16 bits	
6	2	Warning register 2	bit field 16 bits	
8	2	reserved	bit field 16 bits	
10	2	reserved	bit field 16 bits	
12	2	reserved	bit field 16 bits	
14	2	reserved	bit field 16 bits	

6.1 Error register 1

This 16-bit register contains error flags from the FPGA of the scanner.

Bit	Comment
0	contact support
1	contact support
2	scan buffer transmitted incompletely, decrease scan resolution/frequency/range; contact support
3	Scan buffer overflow , decrease scan resolution/frequency/range; contact support
4	contact support
5 - 7	Reserved
8 - 9	Bit 8: APD Under Temperature, provide heating. This bit is set when the internal temperature drops below -40°C. Bit 9: APD Over Temperature, provide cooling. This bit is set when the internal temperature rises above +125°C. Bit 8 and 9: APD Temperature Sensor defect, contact support
10	contact support
11	contact support
12	contact support
13	contact support
14 -15	Reserved

6.2 Error register 2

This 16-bit register contains error flags from the processor of the scanner.

Bit	Comment
0	Internal communication error. The DSP did not receive any scan data from the FPGA. This error typically occurs if the sensor cannot measure correctly. If the error persists, contact support.
1	Internal communication error. The DSP could not communicate correctly with the FPGA via the control interface. If the error persists, contact support.
2	Internal communication error. The DSP did not receive valid scan data from the FPGA for more than 500 ms. This error may be caused by the current operating state (e.g. failure of the motor control due to heavy vibration), or by an internal defect. If the error persists, contact support.
3	contact support
4	Incorrect configuration data, load correct configuration values.
5	Configuration contains incorrect parameters, load correct configuration values
6	Data processing timeout, decrease scan resolution or scan frequency
7	contact support
8	Incoming CAN message lost. This is an overflow of the input buffer, and at least one incoming CAN message was lost. This error may indicate that the sensor is flooded with incoming messages. Reduce the number of messages.
9	reserved
10	Severe deviation (> 10%) from expected scan frequency. This may indicate motor trouble. If error persists, contact support. This bit is available in firmware v2.2.09 and 3.02 or newer.
11	Motor blocked. No rotation of the internal mirror was detected, and automatic restart has failed. This may indicate motor trouble. If error persists, contact support. This bit is available in firmware v2.2.09 and 3.02 or newer.
12 - 15	reserved

6.3 Warning register 1

This 16-bit register contains warning flags from the FPGA of the scanner.

Bit	Comment
0 - 2	reserved
3	Warning of insufficient temperature (“Low temperature”). This bit is set when the internal temperature drops below -30°C.
4	Warning of exceeding temperature (“High temperature”). This bit is set when the internal temperature rises above +115°C.
5 - 6	reserved
7	Check syncronisation- and scan frequency. This bit is set if the scanner detects an external sync-signal, but is unable to synchronize the motor to this signal. Possible causes are: 1) The sync signal is in the wrong frequency range (e.g. sensor is set to 12.5 Hz, but sync signal is 25 Hz) 2) The sync signal is unstable, e.g. has a high jitter
8 - 11	reserved
12	Startpulse of first Laser was not detected. This is an internal warning from the measurement system. If this warning persists, contact support.
13	Startpulse of second Laser was not detected. This is an internal warning from the measurement system. If this warning persists, contact support.
14-15	reserved

6.4 Warning register 2

This 16-bit register contains warning flags from the processor of the scanner.

Bit	Comment
0	CAN interface blocked. This is a general failure warning of the CAN interface. This warning occurs when data cannot be sent or received via CAN (e.g. because it is not connected), but is configured to be sent or received. A typical cause would be that vehicle motion data is requested (see parameter 0x1210, bit 0), but CAN is not connected.
1	Ethernet interface blocked, check Ethernet connection
2	reserved
3	contact support
4	Check Ethernet data

Bit	Comment
5	Incorrect or forbidden command received, check command
6	Memory access failure, restart LD-MRS, contact support
7 - 14	Internal warning states: Bit 7: Segement overflow Bit 8: EgoMotion Bit 9: MountingPosition Bit 10: calculated frequency Bit 11: no ntp time Bit 12: no time sync pps Bit 13: no time sync command Bit 14: no time sync
15	Slight deviation from expected scan frequency (5..10%). This may occur during operation due to vibration or sensor movement, but may also indicate motor trouble. This bit is available in Firmware v2.2.09 and 3.02 or newer.

7 Movement data

7.1 Vehicle Data - Data type 0x2805

When “ETH vehicle data” is requested in the Data Output Flags, the sensor will periodically send the output of its vehicle model in the Movement data message.
As the message format is internal only, this message should be ignored by the users.

7.2 Ego Motion data: Data type 0x2850

This data can be sent to the sensor to inform it about the movement of the sensor (resp. the vehicle it is mounted onto). Supplying this information may enhance the quality of object tracking. Data update rate must be at least the scan frequency of the LD-MRS. Better is twice or more the sensors frequency. If the data is older than 240 ms it is treated as invalid and all values will be set to 0. The warning 0x0100 will be sent in the warning register 2.

Offset	Bytes	Name	datatype	Description
0	2	Version	UINT16	This protocol description refers to version 1.
2	2	Velocity	INT16	Velocity, in 0.01 m/s. Two’s complement. Forward: positive values. If unknown, set to 0.
4	2	unused	n/a	unused, set to 0x0000.
6	2	Steering wheel angle	INT16	Steering wheel angle in 0.001 rad. Two’s complement. To the left: positive values. If unknown, set to 0.
8	2	Yaw rate	INT16	Yaw rate in 0.0001 rad/s. Two’s complement. To the left: positive values. If unknown, set to 0.

For an example of how to send this data, please refer to chapter 9.3.

8 LDMRS SensorInfo structure - Data type 0x7100

This message contains additional information about the scanner and is related to a scan. If available, it will be automatically sent before the scan. Synchronisation with the scan can be done by comparing the scan numbers.
This LDMRS SensorInfo message is currently available in firmware version 2.2.09 and 3.02 (and newer) only.

Encoded in little endian format.
Version 1 has the following structure:

Offset	Bytes	Scan header:	datatype	Description
0	2	Version	UINT16	Version of this sensor info structure (here: 1)
2	2	Related scan number	UINT16	The scan number of the scan that this info is referring to. This is the same scan number as in the scan data structure.
4	2	Error Register FPGA	UINT16	Error register 1
6	2	Error Register FPGA	UINT16	Error register 2
8	2	Warning Register FPGA	UINT16	Warning register 1
10	2	Warning Register DSP	UINT16	Warning register 2
12	2	Temperature in Celsius	INT16	Temperature at the APD. 0x7FFF is invalid.
14	2	APD-Voltage U_{apd}	UINT16	Total receiver voltage of the current scan, in [Volts]. $U_{apd} = U_{apd,temp} - U_{apd,reduction}$. If this value is 0xFFFF it is invalid.
16	2	APD-Voltage reduction	UINT16	Receiver voltage reduction, in [Volts]. "$U_{apd,reduction}$". Was subtracted from original value Uapd,temp. 0xFFFF is invalid.
18	4	Rotation duration in microseconds	UINT32	Time from the last scan to the current scan, in [µs]. If this value is 0xFFFFFFFF it is invalid.
22	4	Operating hours	UINT32	Total operating hours of this device. Unit is [hours]. If this value is 0xFFFFFFFF it is invalid.

Offset	Bytes	Scan header:	datatype	Description
26	2	Info bitfield	UINT16	Bit0: Result of blindness detection (1=Scanner blind) Bit1: Noise reduction active (adaptive sensitivity reduction, see parameter 0x3600 for details) Bit2..15: unused
28	2	Range estimation	UINT16	Estimated view range, in [percent] (0..100). 0%: equal to blind 100%: full range of view. If the value is > 100, it is invalid.

9 Examples

9.1 Setting a parameter

This example shows how to set the IP address via Ethernet to 10.152.36.200.

Offset	Bytes	Message header **Big endian byte order!**	datatype	Content
0	4	Magic word	UINT32	0xAFFEC0C2
4	4	Size of previous message	UINT32	Not mandatory. Set e.g. to 0: 0x00000000
8	4	Size of this message	UINT32	0x000000XX
12	1	Reserved	UINT8	0x00
13	1	Device ID	UINT8	Not used. Set e.g. to 7: 0x07
14	2	Data type: LD-MRS command	UINT16	0x2010
16	8	NTP timestamp	UINT64	Not mandatory. Set e.g. to 0: 0x0000000000000000
Offset	Bytes	Message data **Little endian byte order!**	datatype	Content
24	2	Command ID: Set parameter	UINT16	0x0010 (send encoded as 0x1000)
26	2	Reserved	UINT16	0x0000
28	2	Parameter index: IP address	UINT16	0x1000 (send encoded as 0x0010)
30	4	Parameter data (here: 10.152.36.200)	UINT32	0x0A9824C8 (send encoded as 0xC824980A)
34				

The data sent to the scanner is (all hex, total 34 bytes):

```
AF FE C0 C2  00 00 00 00  00 00 00 0A  00  00  20 10  00 00 00 00
00 00 00 00  10 00  00 00  00 10  C8 24 98 0A
```

The sensor reply is (total 26 bytes):

```
AF FE C0 C2  00 00 00 00  00 00 00 02  00  00  20 20  00 00 00 00
00 00 00 00  10 00
```


9.2 Setting an NTP timestamp

To set an NTP timestamp, two commands must be set. This is described in chapter 5.3.9 and 5.3.10.
In this example, we want to set the date to (approximately) January 1st, 2000, 0:00h. In seconds, this is the value 3155670000 (seconds since 1.1.1900, in HEX format: BC17B3F0).

First, the seconds are sent to the sensor (34 bytes in total):

```
AF FE C0 C2 00 00 00 00 00 00 00 0A 00 01 20 10 00 00 00 00 00 00
00 00 30 00 03 00 00 00 F0 B3 17 BC
```

The sensor reply is (total 26 Bytes):

```
AF FE C0 C2 00 00 00 00 00 00 00 02 00 00 20 20 D6 C0 27 8F 19 56
AC 98 30 00
```

Next, the fractional seconds are sent to the sensor (34 bytes in total):

```
AF FE C0 C2 00 00 00 00 00 00 00 0A 00 01 20 10 00 00 00 00 00 00
00 00 31 00 03 00 00 00 00 00 00 00
```

The sensor reply is (total 26 Bytes):

```
AF FE C0 C2 00 00 00 00 00 00 00 02 00 00 20 20 BC 17 B3 F0 00 00
AB CC 31 00
```

9.3 Sending ego motion data

This example shows how to send ego motion data (0x2850) to the sensor via Ethernet.

Offset	Bytes	Message header **Big endian byte order!**	datatype	Content
0	4	Magic word	UINT32	0xAFFEC0C2
4	4	Size of previous message	UINT32	Not mandatory. Set e.g. to 0: 0x00000000
8	4	Size of this message	UINT32	0x000000XX
12	1	Reserved	UINT8	0x00
13	1	Device ID	UINT8	Not used. Set e.g. to 7: 0x07
14	2	Data type: LD-MRS ego motion	UINT16	0x2850
16	8	NTP timestamp	UINT64	Not mandatory. Set e.g. to 0: 0x0000000000000000
Offset	Bytes	Message data **Little endian byte order!**	datatype	Content
24	2	Version	UINT16	0x0001 (send encoded as 0x0100)

26	2	Velocity (here: 10 m/s = 1000 [1/100 m/s])	INT16	0x03E8 (send encoded as 0xE803)
28	2	0x0000	n/a	0x0000
30	2	Steering wheel angle (here: 0)	INT16	0x0000
32	2	Yaw rate (here: -10 deg/s = -1745 [1/10000 rad/s])	INT16	0xF92F (send encoded as 0x2FF9)
34				

The data sent to the scanner is (all hex, total 34 bytes):

```
AF FE C0 C2  00 00 00 00  00 00 00 0A  00  00  28 50  00 00 00 00
00 00 00 00  01 00  03 E8  00 00  00 00  2F F9
```

10 SOPAS interface

The LD-MRS running field evaluation have a SOPAS interface. This interface is similar to the SOPAS interface of other SICK laserscanners, but supports only a subset of the instructions.
IP port: 2111
Protocol: CoLa-B (binary format, fixed)
For a general description of the CoLa-B interface, its data frame format and the coding of the datatypes, please refer to the appropriate CoLa documentation. Please note that all SOPAS communication takes place with “big endian” byte order.

An example project (C++) that covers both the Port 12002 and the SOPAS interface of the LD-MRS is available.

10.1 Access to functions and data

Generally, possible actions fall into the categories of functions (called “methods” in SOPAS) and data (called “variables” in SOPAS). Examples of methods are Login/Logout, StartMeasure, Reset etc. Examples for variables are scanner setup (frequency, angles, …), field configuration, eval cases and so on.

In order to call a method, an “invoke method” command, with appropriate parameters, is sent to the sensor.
In order to either read or write data, a “read variable” or “write variable” command, with appropriate parameters, is sent to the sensor.

SOPAS supports access control to avoid accidental writing of settings. In order to write variables, the access level must be raised to an appropriate level. In case of the variables described here, this is the “Authorised client” level. In the following, this is

referred to as “Login”. After the writing, the access level should be set back to normal (“Logout”).

10.2 Reading the basic scanner configuration

The basic parameters of the scanner can be obtained by reading the SOPAS variable `aScanConfig` (index 0x0e).
The returned `ScanConfig` structure for this variable is:

Offset	Bytes	Name	Datatype	Description
0	1	ScanFrequency	UINT8	Scan frequency: 0 = 12.5 Hz, 1 = 25 Hz, 2 = 50 Hz
1	2	Length	UINT16	Length of the following data. Following bytes are valid only if length > 0.
3	2	AngleResolution	INT16	Coded in 1/32 degrees.
5	2	StartAngle	INT16	Coded in 1/32 degrees.
7	2	EndAngle	INT16	Coded in 1/32 degrees.

10.3 Reading or writing the fields

Each of the 16 possible fields can be read or written with its own SOPAS variable `field000` (index 0x003d) to `index_var_field015` (index 0x004c). The length and sequence of this data structure is not fixed, but depends on the internal variables.
In order to find out how many fields are configured, all 16 fields have to be read and evaluated. It is not possible to read the number of valid fields from the sensor directly.
In order to clear a field in the sensor, set both the `FieldSegmented` and `FieldRectangular` parameters to 0 and write the field.

Offset	Bytes	Name	Datatype	Description
0	4	DistScaleFactor	float	Scale factor, in [mm per step].
4	4	DistScaleOffset	float	Common offset for all distances, in [mm].
8	4	AngleScaleFactor	UINT32	[1/32 deg].
12	4	AngleScaleOffset	UINT32	[1/32 deg].
16	1	FieldTypeIntern	UINT8	Type of this field. 0 = Radial (not supported), 1 = Rectangle, 2 = Segmented, 3 = Dynamic (not supported)
17	1	FieldNumber	UINT8	Number of this field (1..x).
18	2	FieldSegmented	UINT16	If this marker is 1, a “Segmented Field” structure follows now.

Offset	Bytes	Name	Datatype	Description
<< A Segmented Field structure follows here, if the above parameter is “1” >>				
x	2	FieldRectangular	UINT16	If this marker is 1, a “Rectangular Field” structure follows now.
<< A Rectangular Field structure follows here, if the above parameter is “1” >>				
x	2	FieldRadial	UINT16	Must be 0 as the MRS does not support radial fields.
x+2	2	FieldDynamic	UINT16	Must be 0 as the MRS does not support dynamic fields.
x+4	2	VersionNumber	UINT16	Version number of this field.
x+6	2	FieldNameLength	UINT16	Length of the following string, in [bytes].
x+8	FieldNameLength	FieldName	String	Name of this field. Maximum length is 32 chars.
y	2	CommentLength	UINT16	Length of the following string, in [bytes].
y+2	CommentLength	Comment	String	Comment to this field, as a string. Maximum length is 128 chars.
z	1	EnableLayerFilter	Bool	Flag if layer filter is enabled. Only internal use.
z+1	2	LayerBitField	UINT16	Included layers in this field. Bit 0 = layer 1, bit 1 = layer 2, bit 2 = layer 3 and bit 3 = layer 4. Only for internal use.

10.3.1 Segmented fields

Segmented fields are essentially polygons in which the evaluation takes place. Each of the points of a segmented field structure define an angle (scan beam angle) and a start and end distance (start and end of the evaluation, in between is the field). Either of the start and end angles can also be invalid, so that only a start- or end-distance is defined.

10.3.1.1 Data structures

Segmented Field structure:

Offset	Bytes	Name	Datatype	Description
s	2	arrayLength	UINT16	Number of following points (Segmented field point structures).

Segmented field point structure:

Offset	Bytes	Name	Datatype	Description
p	2	AngleIndex_u	UINT16	Angle of the point, in [1/32°]
p+2	2	StartDist_u	UINT16	StartDist, in [mm].

Offset	Bytes	Name	Datatype	Description
				0xFFFF is invalid. For decoding see below.
p+4	2	EndDist_u	UINT16	EndDist, in [mm]. 0xFFFF is invalid. For decoding see below.

10.3.1.2 Decoding the data

To decode the received field points into a polygon, use the following algorithm:

- For each point:
  - If StartDist_u is valid:
    `StartDist_m = (StartDist_u / 1000.0) * DistScaleFactor + DistScaleOffset / 1000.0`
  - If EndDist_u is valid:
    `EndDist_m = (EndDist_u / 1000.0) * DistScaleFactor + DistScaleOffset / 1000.0`
  - `Angle_deg = (angleIdx * AngleScaleFactor + AngleScaleOffset) / 32.0`

StartDist_m and EndDist_m are now available in [m], but only if they are valid. The angle is available in [°].
To convert points from polar coordinates (dist / angle) to cartesian coordinates (x / y), use the standard transform `x = dist * cos(angle)`, `y = dist * sin(angle)`.
Now build a polygon by adding the points clockwise, starting with the first valid endpoint and adding all other valid endpoints. If there are no valid endpoints, add at least a point in the origin (0.0, 0.0). Then, add the valid starting points the other way round, starting with the last point, and add all other valid starting points up to the first startpoint.
All field points have the coordinate unit [m].

10.3.2 Rectangular fields

Rectangular fields are rectangular areas in which the evaluation takes place. Rectangular fields are defined by a reference point, a width, a length and a rotation angle (around the reference point).

10.3.2.1 Data structures

Offset	Bytes	Name	Datatype	Description
r	4	RefPointAngle	INT32	Angle of the reference point, in [1/32°]. Valid values are -1600 .. 1920.
r+4	2	RefPointDist	UINT16	Distance of reference point to the origin.
r+6	4	RotAngle	INT32	Rotation angle of the rectangle, in [1/32°]. Valid values are -5759 .. 5760.

Offset	Bytes	Name	Datatype	Description
r+10	4	Length	UINT32	Length of the rectangle (x-direction), in [mm]
r+14	4	Width	UINT32	Width of the rectangle (y-direction), in [mm]

10.3.2.2 Decoding the data

To convert the RefPointAngle to degrees, divide by 32:

```
RefPointAngle_deg = RefPointAngle / 32.0
```

To convert the RefPointDist to meters:

```
RefPointDist_m = (RefPointDist * DistScaleFactor +
                  DistScaleOffset) / 1000.0
```

To convert the RotAngle to degrees, divide by 32:

```
RotAngle_deg = RotAngle / 32.0
```

To convert the Length and Width to m, divide by 1000:

```
Length_m = Length / 1000.0
Width_m = Width / 1000.0
```

Then, build a clockwise polygon of the edge points.

10.3.3 Example

To read the field #0, send the command (total length 14 bytes):

```
02 02 02 02  00 00 00 05  73 52 49  00 3D  55
```

The MRS reply depends on the field configuration. Here is a valid example (length= 68 bytes):

```
02 02 02 02  00 00 00 3B  73 52 41  00 3D  41 20 00  00 00 00 00
00 00 00 00 08 FF FF F8 80 02 01 00 01 00 02 00 93 FF FF 0C 76 01
50 FF FF 11 4D 00 00 00 00 00 00 00 01 00 06 46 65 6C 64 2D 31 00
00 00 00  98
```

10.4 Reading or writing the EvalCases

All configured eval cases can be obtained by reading the SOPAS variable `aEvalCaseParam` (index 0x004e). This variable contains a list of the up to 16 eval cases.

Offset	Bytes	Name	Datatype	Description
0	2	Number of Eval Cases	UINT16	Number of following eval cases (0..16).
2	x	EvalCase(s)		

10.4.1 EvalCase structure

Each of the eval cases consists of this structure.

Offset	Bytes	Name	Datatype	Description
0	2	Version Number	UINT16	Version of this structure. In case of the MRS, it is 1.
2	1	Case Number	UINT8	Number of this EvalCase (1..16). Must not be 0.
3	1	Evaluation strategy	UINT8	Evaluation strategy: 0 = BLANKING, 1 = PIXEL, 2 = CONTOUR, 3 = IO_MAPPING, 4 = FOCALPOSITION
4	1	Result negation	Bool	Negation of the eval result (0=off, 1=on)
5	4	Response Time	UINT32	Response time to an event, in [ms]
9	4	Response Time Extended	UINT32	Response time in case of a detected “Manipulation prevention” incident, in [ms].
13	1	Output number	UINT8	The assigned digital output (via external CAN module).
14	1	Logical inputs	UINT8	The assigned logical inputs. Bits 0 and 1 are for input 1, Bits 2 and 3 for input 2: 0 = Inactive, 1 = Low-active, 2 = High-active, 3 = invalid
15	1	Hardware inputs	UINT8	The assigned hardware inputs. As the MRS does not have hardware inputs, this value is 0.
16	1	Dist Dependent	Bool	If “true”, field evaluation works distance dependent.
17	2	Max Radial Corridor	UINT16	Max radial corridor, in [mm]. Only active if distance dependency is active
19	1	Manipulation prevention	Bool	If true, “Manipulation prevention” is enabled.
20	2	Blanking Size	UINT16	This is the minimum object size, in [mm], if strategies “Blanking” or “Contour” are selected.
22	2	Min Field Exp	UINT16	Minimum radial distance between field start and end point, in [mm]. Only used when strategy “Contour” is selected.
24	1	Field number	UINT8	Number of the assigned field.
25	1	Filter type	UINT8	Associated filter. For the MRS, this value is always 0.

Offset	Bytes	Name	Datatype	Description
26	2	EvalCase Name Length	UINT16	Length of the following string, in [bytes].
28	EvalCaseNameLength	EvalCase Name	String	Name of this field. Maximum length is 32 chars.
x	2	CommentLength	UINT16	Length of the following string, in [bytes].
x+2	CommentLength	Comment	String	Comment to this EvalCase. Maximum length is 128 chars.

10.4.2 Example

To write the EvalCases, log in and then write the variable. Here is a valid example in which 1 eval case is written (totel length 56 bytes):

```
02 02 02 02  00 00 00 2F  73 57 49  00 4E  00 01 00 01 01 00 00
00 00 01 F4 00 00 01 F4 00 00 38 00 07 D0 00 00 C8 00 00 01 00 00
0A 44 65 6D 6F 43 61 73 65 2D 31 00 00  05
```

The response from the MRS (success) is (length 14 bytes):

```
02 02 02 02  00 00 00 05  73 57 41  00 4E  2B
```

In case of an error, the MRS would respond with a “FA” message.

10.5 Reading eval case results

The status of the field evaluation cases (short: eval cases) can be obtained by reading the SOPAS variable `aEvalCaseResult` (index 0x0029).
In addition, it is possible to receive this variable whenever status of an eval case changes. To do this, the application must subscribe to the variable by registering to the event with the variable index.
If the application no longer requires the events, it can unsubscribe from the variable.

The returned `evalCaseResult` structure for the variable is:

Offset	Bytes	Name	Datatype	Description
0	2	NumCases	UINT16	Number of following eval cases
2	…	EvalCaseResult 1..x		The eval cases

Eval Case Result:

Offset	Bytes	Name	Datatype	Description
0	2	versionNo	UINT16	Version number
2	1	Number	UINT8	Number of this eval case

3	4	SysCount	UINT32	(ignore)
7	4	DistScaleFactor	float	(ignore)
11	4	DistScaleOffset	float	(ignore)
15	4	AngleScaleFactor	UINT32	(ignore)
19	4	AngleScaleOffset	INT32	(ignore)
23	1	CaseResult	UINT8	The current status of the eval case: 1 = Don't care, 2, 3 = Low (not triggered) 4 = Detecting (not yet triggered) 5 = invalid, 6, 7 = High (triggered) all else: invalid.
24	2	FieldInfringementLength	UINT16	Length of the following string (0..x chars)
26..x	FieldInfringementLength	FieldInfringementString	string	Text
x	2	CaseNameLength	UINT16	Length of the following string (0..x chars)
x	CaseNameLength	CaseName	string	Name of the EvalCase. Can be 0..32 chars.
x	2	CommentStringLength	UINT16	Length of the following string (0..x chars)
x	CommentStringLength	CommentString	string	Text
x	2	NumTimeBlocks	UINT16	Number of following time blocks (see below)
x	11* NumTimeBlocks	TimeBlock 1..n	TimeBlock (see below)	

Time Block:

Offset	Bytes	Name	Datatype	Description
0	2	Year	UINT16	Year
2	1	Month	UINT8	Month, 1..12
3	1	Day	UINT8	Day, 1..31
4	1	Hour	UINT8	Hour, 0..23
5	1	Minute	UINT8	Minute, 0..59
6	1	Second	UINT8	Second, 0..59
7	4	Microsecond	UINT32	Microsecond, 0..999999

10.6 Login

To allow variable writes, a login must be performed. Typically, the level needs to be at least “Authorized client”. For the login, the method `SetAccessMode` (index 0x0000) has to be invoked.

Parameters for method `SetAccessMode`:

Offset	Bytes	Name	Datatype	Description
0	1	Level	UINT8	Desired access level. For “Authorized client”, this is 3.
1	4	Password hash	UINT32	Password hash value. For AuthorizedClient, this is 0xF4724744

10.6.1 Sensor answer to Login

The sensor answers with an “AI”, the index (0x0000) and a boolean flag (1 byte). If this flag is 0 = false, the method was not successful, otherwise it was successful.

10.6.2 Example

Login command, sent to MRS (all in hex format, total length 19 bytes):

```
02 02 02 02  00 00 00 0A  73 4D 49  00 00  03  F4 72 47 44  F1
```

Response of the MRS (length 15 bytes):

```
02 02 02 02  00 00 00 06  73 41 49  00 00  01  7A
```

10.7 Logout

Logout leaves the access level. The command sequence is identical to the Login procedure, but the access level is 1, and any password hash value can be used.

10.8 Scan data

The SOPAS interface also offers access to the scan data of the sensor. This is intended to display a scan data reference while configuring fields.
For applications that are only using the scan data, it is recommended to use the native MRS interface described in chapter 3.
Scan data can be requested by registering to the event `ScanDataMonitor` (index 0x0011). The sensor then sends an event answer with new scans whenever they are available.

Payload:

Offset	Bytes	Name	Datatype	Description
0	2	Version Number	UINT16	This description covers the version

Offset	Bytes	Name	Datatype	Description
				number 1.
2	2	Logical Number	UINT16	Unused. The MRS does not provide this data.
4	4	Serial Number	UINT32	Unused. The MRS does not provide this data.
8	2	Status Bits	UINT16	Unused. The MRS does not provide this data.
10	2	TelegramCount	UINT16	Rolling counter for data telegrams.
12	2	Scan Count	UINT16	Rolling counter for the scan number. Is increased by 1 with every scan taken by the sensor.
14	4	SystemTimeScan	UINT32	Time of the scan capture, in [us], since power-on of the sensor.
18	4	SystemTimeTransmit	UINT32	Time of the scan transmission, in [us], since power-on of the sensor.
22	2	InputState	UINT16	Unused. The MRS does not provide this data.
24	2	OutputState	UINT16	Unused. The MRS does not provide this data.
26	2	(reserved)	UINT16	Unused.
28	4	ScanFrequency	UINT32	Scan frequency, in [1/256 Hz].
32	4	MeasurementFrequency	UINT32	Unused. The MRS does not provide this data.
36	2	NumOfEncoders	UINT16	Always 0. The MRS has no encoder inputs.
38	2	NumOf16BitData Channels	UINT16	Number of 16-bit data arrays. Always 12 (4 layers x 3 echos).
<< 12 DataChannel structures follow here >>				

The rest of the message can be ignored as the MRS does not provide any more information.

10.8.1 DataChannel structure

The MRS has 4 scan layers, and in each layer, each measurement can provide up to 3 range results (“echos”). Therefore, the scan data is transmitted in 12 DataChannel structures, each covering 1 echo in 1 layer.

Offset	Bytes	Name	Datatype	Description
0	6	ContentType	string	A string which describes the following contents: DISTA1 = Layer 1, Echo 1 DISTB1 = Layer 2, Echo 1 DISTC1 = Layer 3, Echo 1 DISTD1 = Layer 4, Echo 1 … DISTD3 = Layer 4, Echo 3.

Offset	Bytes	Name	Datatype	Description
6	4	ScaleFactor	float	Factor by which the following range values can be brought to [mm] scale. Here: 10.
10	4	ScaleOffset	float	Offset for all range values. Here: 0.
14	4	StartAngle	INT32	Start angle of the following range data, in [1/10000 deg].
18	2	AngularResolution	UINT16	Angular resolution between the range values, in [1/10000 deg].
20	2	NumOfRangeData	UINT16	Number of following range values.
..	..	RangeValue	UINT16	..

At the end of this structure there are 0..x range values (as defined by `NumOfRangeData`). Each value is a UINT16 value. For each value, the distance [m] is:

```
dist = (RangeValue * scaleFactor) / 1000.0
```

and the horizontal angle (in [deg] is:

```
hAngle = ((startAngle + d * angularResolution) / 10000.0)
```

with d = number of this range value (0 .. (NumOfRangeData-1)).

