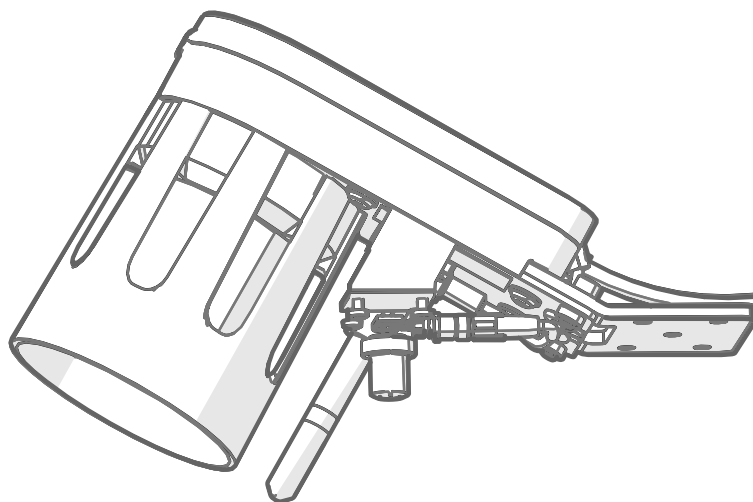


# Interface Description

Vaisala Mobile Detector  
**MD30**



PUBLISHED BY

Vaisala Oyj  
Vanha Nurmijärventie 21, FI-01670 Vantaa, Finland  
P.O. Box 26, FI-00421 Helsinki, Finland  
+358 9 8949 1

Visit our Internet pages at [www.vaisala.com](http://www.vaisala.com).

© Vaisala Oyj 2019

No part of this document may be reproduced, published or publicly displayed in any form or by any means, electronic or mechanical (including photocopying), nor may its contents be modified, translated, adapted, sold or disclosed to a third party without prior written permission of the copyright holder. Translated documents and translated portions of multilingual documents are based on the original English versions. In ambiguous cases, the English versions are applicable, not the translations.

The contents of this document are subject to change without prior notice.

Local rules and regulations may vary and they shall take precedence over the information contained in this document. Vaisala makes no representations on this document's compliance with the local rules and regulations applicable at any given time, and hereby disclaims any and all responsibilities related thereto.

This document does not create any legally binding obligations for Vaisala towards customers or end users. All legally binding obligations and

agreements are included exclusively in the applicable supply contract or the General Conditions of Sale and General Conditions of Service of Vaisala.

This product contains software developed by Vaisala or third parties. Use of the software is governed by license terms and conditions included in the applicable supply contract or, in the absence of separate license terms and conditions, by the General License Conditions of Vaisala Group.

This product may contain open source software (OSS) components. In the event this product contains OSS components, then such OSS is governed by the terms and conditions of the applicable OSS licenses, and you are bound by the terms and conditions of such licenses in connection with your use and distribution of the OSS in this product. Applicable OSS licenses are included in the product itself or provided to you on any other applicable media, depending on each individual product and the product items delivered to you.

## Table of Contents

<b>1. About This Document</b> .....	5
1.1 Version Information.....	5
1.2 Documentation Conventions.....	5
1.3 Trademarks.....	6
<b>2. Vaisala Mobile Detector MD30 Interface</b> .....	7
2.1 Measurement System Overview.....	7
2.1.1 Measurement Sensors in MD30.....	8
2.2 Interface Overview.....	8
2.2.1 Communication Settings.....	9
2.2.2 Supported Data Formats.....	10
<b>3. Observations</b> .....	11
3.1 Data Status Warnings.....	11
3.2 Data Status Erroneous.....	11
3.3 Status Information.....	12
3.4 Error Bits.....	14
3.5 Surface States.....	15
3.6 EN15518 Surface States.....	16
3.7 Data Types.....	16
<b>4. Data Messages</b> .....	18
4.1 Message Format.....	18
4.2 Data Message Example.....	19
4.3 Message IDs.....	21
4.4 GET UNIT ID Message.....	21
4.4.1 GET UNIT ID Request.....	22
4.4.2 GET UNIT ID Response.....	23
4.5 GET FULL PRODUCT INFO Message.....	24
4.5.1 GET FULL PRODUCT INFO Request.....	25
4.5.2 GET FULL PRODUCT INFO Response.....	26
4.6 GET UNIT STATUS Message.....	28
4.6.1 GET UNIT STATUS Request.....	28
4.6.2 GET UNIT STATUS Response.....	29
4.7 SEND DATA Message.....	30
4.7.1 SEND DATA Request.....	32
4.7.2 SEND DATA Response.....	33
4.8 SET REFERENCES Message.....	35
4.8.1 SET REFERENCES Request.....	37
4.8.2 SET REFERENCES Response.....	38
4.9 STOP REFERENCE SETTING Message.....	39
4.9.1 STOP REFERENCE SETTING Request.....	39
4.9.2 STOP REFERENCE SETTING Response.....	40
4.10 SET ROAD COEFFICIENTS Message.....	41
4.10.1 SET ROAD COEFFICIENTS Request.....	42
4.10.2 SET ROAD COEFFICIENTS Response.....	43
4.11 GET PARAMETER Message.....	44
4.11.1 GET PARAMETER Request.....	44
4.11.2 GET PARAMETER Response.....	46
4.12 SET PARAMETER Message.....	47
4.12.1 SET PARAMETER Request.....	48
4.12.2 SET PARAMETER Response.....	49

- 4.13 RESTART UNIT Message..... 50
  - 4.13.1 RESTART UNIT Request..... 50
  - 4.13.2 RESTART UNIT Response..... 51
- 4.14 CRC Error Acknowledgment Message.....52
- 4.15 Parameters..... 53
  
- 5. Flow Control..... 56**
  - 5.1 Flow Control..... 56
  - 5.2 Error Codes..... 56
  - 5.3 CRC Calculation..... 57
  - 5.4 Message and Error Handling.....59
  - 5.5 Multiple Mobile Sensor Units on Same Bus..... 60
  - 5.6 Automatic Data Sending..... 60
  
- Warranty and Product Returns..... 61**
- Technical Support..... 61**
- Recycling.....61**

## List of Tables

Table 1	Document Versions.....	5
Table 2	Interface Functions.....	8
Table 3	Supported Data Formats.....	10
Table 4	Data Status Warnings.....	11
Table 5	Data Status Erroneous.....	11
Table 6	Status Information.....	12
Table 7	Error Messages .....	14
Table 8	Surface States.....	15
Table 9	EN15118 Surface States.....	16
Table 10	Data Types.....	16
Table 11	Request Message Fields.....	18
Table 12	Response Message Fields.....	19
Table 13	Message IDs.....	21
Table 14	GET UNIT ID Message.....	22
Table 15	GET FULL PRODUCT INFO Message.....	24
Table 16	Examples of Key Value Pairs.....	24
Table 17	GET UNIT STATUS Message.....	28
Table 18	SEND DATA Message.....	31
Table 19	SET REFERENCES Message.....	35
Table 20	STOP REFERENCE SETTING Message.....	39
Table 21	SET ROAD COEFFICIENTS Message.....	41
Table 22	GET PARAMETER Message.....	44
Table 23	SET PARAMETER Message.....	48
Table 24	RESTART UNIT Message.....	50
Table 25	Parameters.....	54
Table 26	Error Codes.....	56



# 1. About This Document

## 1.1 Version Information

This document provides a description of the data reporting interface of Vaisala Mobile Detector MD30.

Table 1 Document Versions

Document Code	Date	Description
M212201EN-B	August 2019	For version C of the interface.
M212201EN-A	December 2018	First version. For version B of the interface.

## 1.2 Documentation Conventions



**WARNING! Warning** alerts you to a serious hazard. If you do not read and follow instructions carefully at this point, there is a risk of injury or even death.



**CAUTION! Caution** warns you of a potential hazard. If you do not read and follow instructions carefully at this point, the product could be damaged or important data could be lost.



**Note** highlights important information on using the product.



**Tip** gives information for using the product more efficiently.



Lists tools needed to perform the task.



Indicates that you need to take some notes during the task.

## 1.3 Trademarks

Vaisala® and HUMICAP® are registered trademarks of Vaisala Oyj.

All other product or company names that may be mentioned in this publication are trade names, trademarks, or registered trademarks of their respective owners.



## 2. Vaisala Mobile Detector MD30 Interface

### 2.1 Measurement System Overview

The mobile sensor provides:

- Surface state information, including:
  - Grip
  - Surface state
  - Surface layer thicknesses
- Surface temperature
- Air temperature
- Dew point and frost point
- Relative humidity

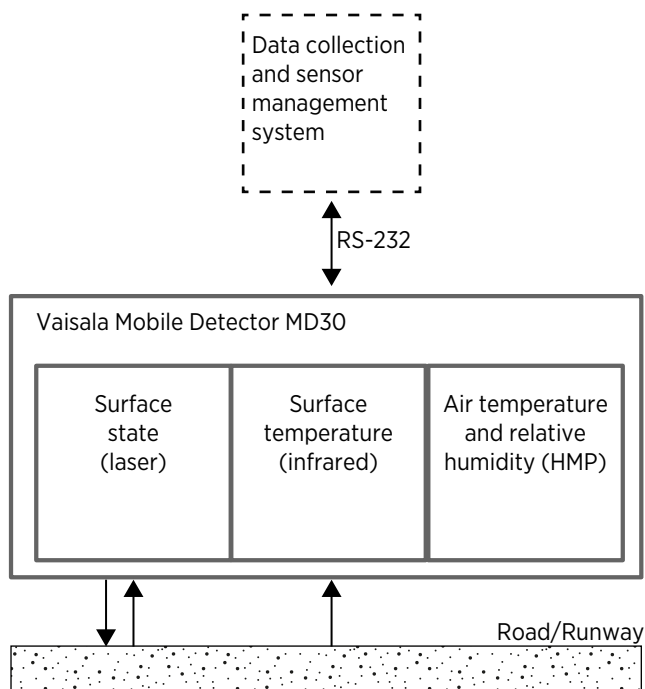


Figure 1 System Overview

The data collection and sensor management system can be a spreader system in a winter maintenance vehicle (for example, snow plow truck), a road monitoring system in a vehicle (for example, patrol car), or any other data collection system.

The data collection system maps the mobile road sensor data to a geographic location.

The communication uses the RS-232 interface.

## 2.1.1 Measurement Sensors in MD30

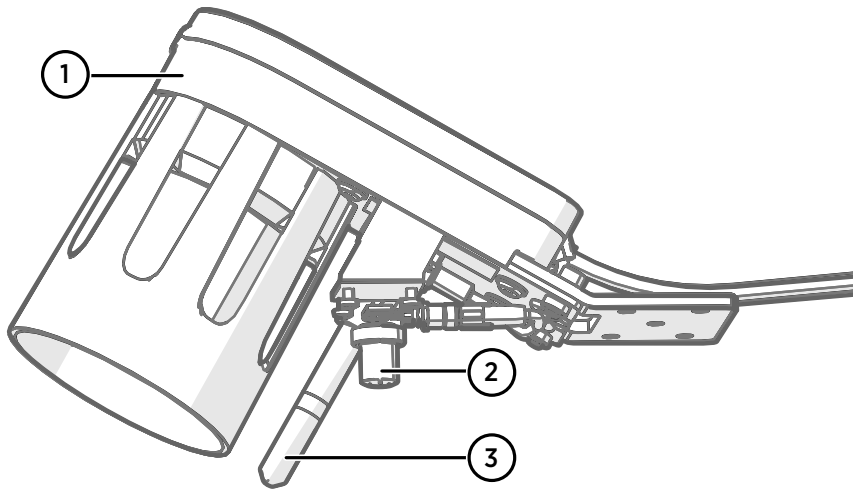


Figure 2 MD30 Measurement Sensors

- 1 Surface state sensor
- 2 Surface temperature sensor, MT10
- 3 Air temperature and humidity sensor, HMP113

MD30 contains 3 sensors:

- Surface state sensor measures the water, ice, and snow layer thicknesses and surface state with 3 lasers.
- Surface temperature sensor MT10 measures the surface temperature with infrared (IR) technology.
- Air temperature and humidity sensor, Vaisala HUMICAP® Humidity and Temperature Probe HMP113, measures the air temperature and relative humidity.

## 2.2 Interface Overview

The mobile sensor unit interface applies a request-response pattern. Every message is acknowledged by the mobile sensor.

Table 2 Interface Functions

Request	Message
Identify the installed unit.	GET UNIT ID GET FULL PRODUCT INFO
Report status and error information.	GET UNIT STATUS SEND DATA

Request	Message
Initiate road condition, temperature, and humidity data reporting continuously at defined interval.	SEND DATA GET PARAMETER SET PARAMETER
Stop continuous data reporting.	SEND DATA GET PARAMETER SET PARAMETER
Report road condition, temperature, and humidity data using polling.	SEND DATA
Set reference values.	SET REFERENCES SET ROAD COEFFICIENTS GET PARAMETER SET PARAMETER STOP REFERENCE SETTING
Set offset to road and air temperature.	GET PARAMETER SET PARAMETER
Configure the serial port speed.	GET PARAMETER SET PARAMETER
Configure units of reported quantities.	GET PARAMETER SET PARAMETER

### More Information

- [GET UNIT ID Message \(page 21\)](#)
- [GET FULL PRODUCT INFO Message \(page 24\)](#)
- [GET UNIT STATUS Message \(page 28\)](#)
- [SEND DATA Message \(page 30\)](#)
- [SET REFERENCES Message \(page 35\)](#)
- [SET ROAD COEFFICIENTS Message \(page 41\)](#)
- [STOP REFERENCE SETTING Message \(page 39\)](#)
- [GET PARAMETER Message \(page 44\)](#)
- [SET PARAMETER Message \(page 47\)](#)
- [RESTART UNIT Message \(page 50\)](#)

## 2.2.1 Communication Settings

The communication uses RS-232 where sensor data is transmitted in binary format. The default settings are:

- 115200 bps
- 8 data bits
- 1 stop bit
- No parity
- No hardware control

## 2.2.2 Supported Data Formats

Data is transmitted in binary format. The following data types are supported.

Table 3 Supported Data Formats

Data Type	Description
f32	Binary 32 IEEE 754 format floating point number
U64/U32/24/16/8	64/32/24/16/8 bit unsigned integer
S32/24/16/8	32/24/16/8 bit signed integer
ASCII	ASCII characters are transmitted as binary values

The byte order is little endian. For example, 4-byte value 0x12345678 is transmitted in the following order: 0x78, 0x56, 0x34, and 0x12.

Bit numbering starts at zero (0) for the last significant bit (LSB).

## 3. Observations

### 3.1 Data Status Warnings

The following bits are set to warn the user that data quality cannot be fully guaranteed and data should be treated with suspicion. The data status warnings are reported in the `SEND DATA` message.

Table 4 Data Status Warnings

Bit	Description
0	Air temperature
1	Relative humidity
2	Dew point temperature
3	Frost point temperature
4	Surface temperature
5	Surface state according to Vaisala classification
6	Surface state according to the EN15518 standard
7	Grip
8	Water layer thickness
9	Ice layer thickness
10	Snow layer thickness
11 ... 15	Reserved for future use

#### More Information

- [SEND DATA Message \(page 30\)](#)

### 3.2 Data Status Erroneous

The following bits are set if data contains errors. The erroneous data status is reported in the `SEND DATA` message.

Table 5 Data Status Erroneous

Bit	Description
0	Air temperature
1	Relative humidity
2	Dew point temperature
3	Frost point temperature

Bit	Description
4	Surface temperature
5	Surface state according to Vaisala classification
6	Surface state according to the EN15518 standard
7	Grip
8	Water layer thickness
9	Ice layer thickness
10	Snow layer thickness
11 ... 15	Reserved for future use

### More Information

- [SEND DATA Message \(page 30\)](#)

## 3.3 Status Information

The status information is used to indicate the current status of the mobile sensor and to issue warnings. The status information is reported in the `SEND DATA` message.

**Table 6** Status Information

Bit <sup>1)</sup>	Message	Value	Description
0	Not ready to measure	0 = Ready 1 = Not ready	Bit 0 is set only when unit is started up. The flag is cleared when the unit has reached fully operational status for the first time. Measurement data may be invalid.  If the condition persists, check error bits.
1	Reference setting ongoing	0 = Not ongoing 1 = Ongoing	If reference setting does not start, check status information bits 10 ... 13 and error bits.
2	Laser temperature change in progress	0 = Not ongoing 1 = Ongoing	Unit operational, but measurement data may be invalid.  Wait for the laser temperature change to finish.  If the condition persists, check error bits.
3	Window contamination warning	0 = OK 1 = Contaminated	Window is getting contaminated. Clean the window.
4	Window heating	0 = OK 1 = Not working	If heating is not working, monitor window contamination.

Bit <sup>1)</sup>	Message	Value	Description
5	Low input voltage detected	0 = Voltage OK 1 = Voltage low	Unit operational, but check the input voltage.
6	High input voltage detected	0 = Voltage OK 1 = Voltage high	
7	High internal temperature detected	0 = Temperature OK 1 = Temperature high	First notification of unit getting too hot.
8	Temperature unit	0 = °C 1 = °F	–
9	Layer thickness unit	0 = mm 1 = inch	–
10	Reference setting interrupted due to laser temperature change	0 = False 1 = True	Repeat the reference setting when laser temperature change has finished.
11	Reference setting interrupted due to hardware error, check parameter 0x56	0 = False 1 = True	Check error bits and parameter 0x56, which contains the reason for the error.
12	Reference setting values are not updated due to poor signal quality	0 = False 1 = True	Excessive variation in road surface type. Find more representative road surface.
13	Reference setting was interrupted by the client	0 = False 1 = True	–
14	Signal levels low, uncertainty in surface layer thickness results	0 = False 1 = True	Unit operational, but measurement data may be invalid. Verify installation and clean window.
15 - 31	Reserved for future use	–	–

1) Bit is shown in MD30 interface response data message, not in RoadAI.



Reference setting cannot be started if status info bit 0, 1, or 2 is set.



Laser temperature change stops the ongoing reference setting.

### More Information

- [SEND DATA Message \(page 30\)](#)

## 3.4 Error Bits

Error bits indicate error situations in the unit or in communication. The erroneous bits are reported in the GET UNIT STATUS and SEND DATA messages.



Error is indicated by setting bit to 1.

Table 7 Error Messages

Bit <sup>1)</sup>	Message	Probable Cause	Actions
0	Surface temperature sensor error	Cables may be loose, damaged, or disconnected.	Check the cables and connectors.  If the problem persists, replace the mobile sensor.
1	Air temperature error		
2	Relative humidity error		
3	Window contamination alarm	Mobile sensor window is heavily contaminated.	Clean the window.
4	Laser status error	-	Restart the mobile sensor.
5	Laser heating error		
6	Excessive ambient light detected	Sunlight is reflected from road surface to mobile sensor.	Move the vehicle or reposition mobile sensor.
7	Receiver error	-	Restart the mobile sensor.
8	Signal level out of range, gain adjustment limit reached	-	Check the installation height and angle of the mobile sensor.
9	Received signals contain too much noise	-	Check that the mobile sensor is firmly attached to the vehicle.
10	Optical measurement data timeout	-	Restart the mobile sensor.
11	Low input voltage	Incorrect operating voltage.	Check the operating voltage.
12	High input voltage		
13	Flash failure status	-	Restart the mobile sensor.
14	Internal temperature too high	Mobile sensor overheated.	Disconnect the mobile sensor from power supply.
15	Reference status: 0 = OK 1 = Invalid or not set	-	Verify the installation with the reference plate and adapt the mobile sensor to road surface types.
16	Factory calibration status: 0 = OK 1 = Not calibrated	-	Return the mobile sensor to Vaisala.



Bit <sup>1)</sup>	Message	Probable Cause	Actions
17-31	Reserved for future use	-	-

1) Bit is shown in MD30 interface response data message, not in RoadAI.



The reference setting cannot be started or is interrupted if an error is detected and indicated by bits 3 ... 14, or 16.

#### More Information

- [GET UNIT STATUS Message \(page 28\)](#)
- [SEND DATA Message \(page 30\)](#)
- [SET REFERENCES Message \(page 35\)](#)

## 3.5 Surface States

The following table shows the surface state values. Surface states are reported in the SEND DATA message.

Table 8 Surface States

Value	Description
0	Error
1	Dry
2	Moist
3	Wet
4	-
5	Frost
6	Snow
7	Ice
8	-
9	Slushy
10	Streaming water
11	Slippery
12	Ice watch

#### More Information

- [SEND DATA Message \(page 30\)](#)

## 3.6 EN15518 Surface States

The following table shows the surface state values according to EN15518. EN15518 surface states are reported in the `SEND DATA` message.

Table 9 EN15118 Surface States

Value	Description
0	Error
1	Dry
2	Moist
3	Wet
4	Wet and chemical
10	Streaming water
11	Slippery

### More Information

- [SEND DATA Message \(page 30\)](#)

## 3.7 Data Types

The following data types and ranges are available. The data is reported in the `SEND DATA` message.



See the Vaisala Mobile Detector MD30 datasheet at [vaisala.com](https://www.vaisala.com) for up-to-date values.

Table 10 Data Types

Data Type	Format	Description
Grip	f32	Grip value Range: 0.09 ... 0.82 Missing data: NaN
Water layer thickness	f32	Water layer thickness Range: 0 ... 5 mm (0 ... 0.20 in) Missing data: NaN
Ice layer thickness	f32	Ice layer thickness Range: 0 ... 2 mm (0 ... 0.08 in) Missing data: NaN

Data Type	Format	Description
Snow layer thickness	f32	Snow layer thickness Range: 0 ... 10 mm (0 ... 0.39 in) Missing data: NaN
Surface temperature	f32	Measured road surface temperature Range: -40 ... +60 °C or -40 ... +140 °F Missing data: NaN
Air temperature	f32	Measured air temperature Range: -40 ... 60 °C or -40 ... +140 °F Missing data: NaN
Dew point and frost point	f32	Frost and dew point temperature Range: -40 ... +60 °C or -40 ... +140 °F Missing data: NaN Frost point is sent if the surface temperature is below 0 °C (+32 °F)
Relative humidity	f32	Measured relative humidity Range: 0 ... 100 %RH Missing data: NaN

#### More Information

- [SEND DATA Message \(page 30\)](#)

## 4. Data Messages

### 4.1 Message Format

The following tables lists the data fields that are used in the request and response messages.

The byte sizes are calculated for binary format.

The messages, including the CRC error acknowledgment message, contain the listed fields. Exception is the data field which is not always included.

Error messages do not carry any message data. The message ID and message number are copied from the request message and the error code is set according to the detected error. An exception is the CRC error acknowledgment message, where the message receiver ID, message ID, and message number are set to 0.

Offset information indicates the position of the hexadecimal value in the message. The message start marker has offset 0.



The user must wait for the message to be acknowledged or for a communication timeout before sending subsequent messages.

Table 11 Request Message Fields

Field	Size in Bytes	Description	Value
Message start	1	Message start marker, static value	0xab
Message sender ID	1	User	0 = Default
Message receiver ID	1	MD30	1 = Default
Message ID	1	Valid message ID	<a href="#">Message IDs (page 21)</a>
Message number	1	Message number is copied from the request message to the response message and can be used for tracking which request message was acknowledged. If tracking is not needed, can be kept as zero (0).	0 = Not used
Data length	2	Data length in bytes; depends on message type Data length can be zero (0).	0 ... 65535
Data	0 ... 65535	Message data bytes	
CRC	2	CRC checksum CRC is calculated for the whole message, excluding the message start marker and the CRC bytes	<a href="#">CRC Calculation (page 57)</a>

Table 12 Response Message Fields

Field	Size in Bytes	Description	Value
Message start	1	Message start marker, static value	0xab
Message sender ID	1	MD30	1= Default
Message receiver ID	1	User	0 = Default
Message ID	1	Valid message ID	<a href="#">Message IDs (page 21)</a>
Message number	1	Message number is copied from the request message. Exception is the <b>SEND DATA</b> response where the message number of the first message is copied from the request or set to zero in case on automatic sending, and after the first message, the message number is incremented in each response.	0 = Not used
Data length	2	Data length in bytes; depends on message type The data length covers the interface version, error code, and response data. The data length is always at least 2 bytes because interface version and error code are always included.	2 ... 65535
Interface version	1	Version of protocol in use by MD30 Hexadecimal value of the ASCII character, representing the revision	A = 0x41, B = 0x42, and so on
Error code	1	Error code according to detected error If no errors, set to 0	<a href="#">Error Codes (page 56)</a>
Data	2 ... 65535	Message data bytes	
CRC	2	CRC checksum CRC is calculated for the whole message, excluding the message start marker and the CRC bytes	<a href="#">CRC Calculation (page 57)</a>

**More Information**

- [CRC Error Acknowledgment Message \(page 52\)](#)
- [Error Codes \(page 56\)](#)
- [Message and Error Handling \(page 59\)](#)

## 4.2 Data Message Example

The transferred binary data is in hex format.

**Request Message**

The format of the request data message is the following:

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <data> <CRC>
```

The following is an example of a single SEND DATA request that is sent to the mobile sensor in hex format.

```
0xab 0x00 0x01 0x20 0x0e 0x02 0x00 0x00 0x00 0x97 0x9e
```

The example request contains the following information:

- Unit ID: 1
- Client ID: 0
- Message ID: 0x20
- Message number: 14
- Length: 2
- Data interval: 0
- CRC: 0x9e97

## Response Message

The format of the response data message is the following:

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <interface version number> <error code> <data>
<CRC>
```

The following is an example of a single SEND DATA response that is sent from the mobile sensor in hex format.

```
0xab 0x01 0x00 0x20 0x0e 0x36 0x00 0x43 0x00 0xd7 0x08 0x00 0x00 0x00 0x00
0x8f 0xc2 0xbf 0x41 0x29 0x5c 0x45 0x42 0xfb 0x52 0x4b 0x41 0xfb 0x52 0x4b
0x41 0x08 0xd7 0x02 0x42 0x01 0x01 0x85 0xeb 0x51 0x3f 0x00 0x00 0x00 0x00
0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x53 0xe8
```

The example response contains the following information:

- Unit ID: 1
- Client ID: 0
- Message ID: 0x20
- Message number: 14
- Interface version: C
- Length: 54
- Error code: 0
- Count: 2263
- Warning: 0

- Error: 0
- Air temperature: 23.9699
- RH: 49.3400
- Dew point: 12.7077
- Frost point: 12.7077
- Surface temperature: 32.7099
- Surface state: 1
- EN15518: 1
- Grip: 0.8199
- Water: 0.00
- Ice: 0.00
- Snow: 0.00
- Status: 0
- Error: 0
- CRC: 0xe853

## 4.3 Message IDs

The message ID is used to identify the message type. It is part of the request and response messages.

Table 13 Message IDs

Message ID	Message
0x00	CRC ERROR ACKNOWLEDGMENT
0x10	GET UNIT ID
0x11	GET FULL PRODUCT INFO
0x12	GET UNIT STATUS
0x20	SEND DATA
0x30	SET REFERENCES
0x31	SET ROAD COEFFICIENTS
0x32	STOP REFERENCE SETTING
0x40	GET PARAMETER
0x41	SET PARAMETER
0x50	RESTART UNIT

## 4.4 GET UNIT ID Message

GET UNIT ID is used for requesting the ID of the mobile sensor. The ID is the serial number.

Table 14 GET UNIT ID Message

<b>Message ID</b>	0x10		
<b>Data in Request</b>	None		
<b>Data in Response</b>	<b>Offset</b>	<b>Data</b>	<b>Size in Bytes</b>
	9	Serial number	8
		<b>Data Length</b>	8

## 4.4.1 GET UNIT ID Request

### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
CRC	7 ... 8	2

### Example

```
0xab 0x00 0x01 0x10 0x05 0x00 0x00 0x16 0x54
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x00	0
Message receiver ID	0x01	1
Message ID	0x10	0x10
Message number	0x05	5
Data length	0x00 0x00	0
CRC	0x16 0x54	-



## 4.4.2 GET UNIT ID Response

### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <interface version number> <error code>
<serial number> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
Interface version	7	1
Error code	8	1
Serial number	9 ... 16	8
CRC	17 ... 18	2

### Example

```
0xab 0x01 0x00 0x10 0x05 0x0a 0x00 0x43 0x00 0x50 0x31 0x38 0x33 0x30 0x30
0x30 0x32 0x32 0x8a
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x01	1
Message receiver ID	0x00	0
Message ID	0x10	0x10
Message number	0x05	5
Data length	0x0a 0x00	10
Interface version	0x43	C
Error code	0x00	0
Serial number	0x50 0x31 0x38 0x33 0x30 0x30 0x30 0x32	P1830002

Field	Bytes	Value
CRC	0x32 0x8a	-

## 4.5 GET FULL PRODUCT INFO Message

GET FULL PRODUCT INFO is used for requesting full product information.

Table 15 GET FULL PRODUCT INFO Message

<b>Message ID</b>	0x11		
<b>Data in Request</b>	None		
<b>Data in Response</b>	<b>Offset</b>	<b>Data</b>	<b>Size in Bytes</b>
	9	Number of pairs	1
	10	Pair 1 key length	1
	11	Pair 1 key	max. 255 characters
	11 + pair 1 key length	Pair 1 value length	1
	11 + pair 1 key length + 1	Pair 1 value	max. 255 characters
	...	Next key value pair containing: <ul style="list-style-type: none"> <li>• Key length</li> <li>• Key value</li> <li>• Value length</li> <li>• Value</li> </ul>	...
	<b>Data Length</b>	4 ... 65526	

The message must contain:

- Product name
- Product serial number
- Software version
- Serial numbers of replaceable sensors

Also other information can be included.

The full product information is presented as key value pairs. Each key value pair contains:

- Key length presented with 1 byte
- Key value, a set of characters
- Value length presented with 1 byte
- Value, a set of characters

Table 16 Examples of Key Value Pairs

Key	Value
Product name	MD30

Key	Value
Serial number	P1830002
Software version	0.2.0
MT10 ID	700572D61114B1C2
HMP serial number	P2130779

## 4.5.1 GET FULL PRODUCT INFO Request

### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
CRC	7 ... 8	2

### Example

```
0xab 0x00 0x01 0x11 0x06 0x00 0x00 0xf2 0x7b
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x00	0
Message receiver ID	0x01	1
Message ID	0x11	0x11
Message number	0x06	6
Data length	0x00 0x00	0
CRC	0xf2 0x7b	-

## 4.5.2 GET FULL PRODUCT INFO Response

### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <interface version number> <error code> <data length> <number
of pairs and key-value pairs and values> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
Interface version	7	1
Error code	8	1
Data: number of pairs and key-value pairs and values	9 ... 121	111
CRC	122 ... 123	2

### Example

For example, MD30 has 2 external sensors: HMP113 and MT10, so the full product information contains 5 key value pairs.

```
0xab 0x01 0x00 0x11 0x06 0x71 0x00 0x43 0x00 0x05 0x0c 0x50 0x72 0x6f 0x64
0x75 0x63 0x74 0x20 0x4e 0x61 0x6d 0x65 0x04 0x4d 0x44 0x33 0x30 0x0d0x53 0x65
0x72 0x69 0x61 0x6c 0x20 0x4e 0x75 0x6d 0x62 0x65 0x72 0x08 0x50 0x31 0x38
0x33 0x30 0x30 0x30 0x32 0x0a 0x53 0x57 0x20 0x56 0x65 0x72 0x730x69 0x6f 0x6e
0x05 0x30 0x2e 0x39 0x2e 0x30 0x07 0x4d 0x54 0x31 0x30 0x20 0x49 0x44 0x10
0x37 0x30 0x30 0x35 0x37 0x32 0x44 0x36 0x31 0x31 0x31 0x340x42 0x31 0x43 0x32
0x11 0x48 0x4d 0x50 0x20 0x53 0x65 0x72 0x69 0x61 0x6c 0x20 0x4e 0x75 0x6d
0x62 0x65 0x72 0x08 0x50 0x32 0x31 0x33 0x30 0x37 0x370x39 0x41 0x80
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x01	1
Message receiver ID	0x00	0

Field	Bytes	Value
Message ID	0x11	0x11
Message number	0x06	6
Data length	0x71 0x00	113
Interface version	0x43	C
Error code	0x00	0
Data: number of pairs and key-value pairs and values		
Number of pairs	0x05	5
Pair 1 key length	0x0c	12
Pair 1 key	0x50 0x72 0x6f 0x64 0x75 0x63 0x74 0x20 0x4e 0x61 0x6d 0x65	Product name
Pair 1 value length	0x04	4
Pair 1 value	0x4d 0x44 0x33 0x30	MD30
Pair 2 key length	0x0d	13
Pair 2 value	0x53 0x65 0x72 0x69 0x61 0x6c 0x20 0x4e 0x75 0x6d 0x62 0x65 0x72	Serial number
Pair 2 value length	0x08	8
Pair 2 value	0x50 0x31 0x38 0x33 0x30 0x30 0x30 0x32	P1830002
Pair 3 key length	0x0a	10
Pair 3 key	0x53 0x57 0x20 0x56 0x65 0x72 0x73 0x69 0x6f 0x6e	SW version
Pair 3 value length	0x05	5
Pair 3 value	0x30 0x2e 0x39 0x2e 0x30	0.9.0
Pair 4 key length	0x07	7
Pair 4 key	0x4d 0x54 0x31 0x30 0x20 0x49 0x44	MT10 ID
Pair 4 value length	0x10	16
Pair 4 value	0x37 0x30 0x30 0x35 0x37 0x32 0x44 0x36 0x31 0x31 0x31 0x34 0x42 0x31 0x43 0x32	700572D61114B1C2
Pair 5 key length	0x11	17
Pair 5 key	0x48 0x4d 0x50 0x20 0x53 0x65 0x72 0x69 0x61 0x6c 0x20 0x4e 0x75 0x6d 0x62 0x65 0x72	HMP serial number
Pair 5 value length	0x08	8
Pair 5 value	0x50 0x32 0x31 0x33 0x30 0x37 0x37 0x39	P2130779

Field	Bytes	Value
CRC	0x41 0x80	-

The example response contains the following information:

- Unit ID: 1
- Client ID: 0
- Message ID: 0x11
- Message number: 6
- Length: 113
- Interface version: C
- Error code: 0
- Product name: MD30
- Serial number: P1830002
- SW version: 0.9.0
- MT10 ID: 700572D6114B1C2
- HMP serial number: P2130779

## 4.6 GET UNIT STATUS Message

GET UNIT STATUS is used for requesting status and error information from the mobile sensor.

Table 17 GET UNIT STATUS Message

<b>Message ID</b>	0x12		
<b>Data in Request</b>	None		
<b>Data in Response</b>	<b>Offset</b>	<b>Data</b>	<b>Size in Bytes</b>
	9	Status info	4
	13	Error bits	4
		<b>Data Length</b>	8

### More Information

- [Status Information \(page 12\)](#)
- [Error Bits \(page 14\)](#)

### 4.6.1 GET UNIT STATUS Request

#### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
CRC	7 ... 8	2

### Example

```
0xab 0x00 0x01 0x12 0x0d 0x00 0x00 0xdf 0x10
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x00	0
Message receiver ID	0x01	1
Message ID	0x12	0x12
Message number	0x0d	13
Data length	0x00 0x00	0
CRC	0xdf 0x10	-

## 4.6.2 GET UNIT STATUS Response

### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>  
<message number> <data length> <interface version number> <error code>  
<status info> <error bits> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1

Field	Offset	Size in Bytes
Message number	4	1
Data length	5 ... 6	2
Interface version	7	1
Error code	8	1
Status info	9 ... 12	4
Error bits	13 ... 16	4
CRC	17 ... 18	2

**Example**

```
0xab 0x01 0x00 0x12 0x0d 0x0a 0x00 0x43 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x00 0x18 0x67
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x01	1
Message receiver ID	0x00	0
Message ID	0x12	0x12
Message number	0x0d	13
Data length	0x0a 0x00	10
Interface version	0x43	C
Error code	0x00	0
Status info	0x00 0x00 0x00 0x00	0x00000000
Error bits	0x00 0x00 0x00 0x00	0x00000000
CRC	0x18 0x67	-

## 4.7 SEND DATA Message

SEND DATA is used for requesting the default data set from the mobile sensor unit. The message can be used for the following purposes:

- Report status and error information.
- Initiate road condition, temperature, and humidity data reporting continuously at defined interval.
- Stop continuous data reporting.
- Report road condition, temperature, and humidity data using polling.



Table 18 SEND DATA Message

<b>Message ID</b>	0x20		
<b>Data in Request</b>	<b>Offset</b>	<b>Data</b>	<b>Size in Bytes</b>
	7	Data interval as u16	2
		<b>Data Length</b>	2
<b>Data in Response</b>	<b>Offset</b>	<b>Data</b>	<b>Size in Bytes</b>
	9	Data analyze count	2
	11	Data status warning	2
	13	Data status error	2
	15	Air temperature	4
	19	Relative humidity	4
	23	Dew point temperature	4
	27	Frost point temperature	4
	31	Surface temperature	4
	35	Surface state	1
	36	EN15518 surface state	1
	37	Grip	4
	41	Water layer thickness	4
	45	Ice layer thickness	4
	49	Snow layer thickness	4
	53	Unit status info	4
	57	Unit error bits	4
	<b>Data Length</b>	52	

### Data Interval

- If the data interval is 0, only one set of requested data is sent back.
- If the data interval is 25 ... 5000 ms, the unit starts to send measurement data using the given interval. Data sending continues until data polling request with interval 0 is sent, or unit is restarted.

### More Information

- [Data Status Warnings \(page 11\)](#)
- [Data Status Erroneous \(page 11\)](#)
- [Status Information \(page 12\)](#)
- [Error Bits \(page 14\)](#)
- [Surface States \(page 15\)](#)
- [EN15518 Surface States \(page 16\)](#)
- [Data Types \(page 16\)](#)
- [Automatic Data Sending \(page 60\)](#)

## 4.7.1 SEND DATA Request

### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <data interval> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
Data interval	7 ... 8	2
CRC	9 ... 10	2

### Example

```
0xab 0x00 0x01 0x20 0x0e 0x02 0x00 0x00 0x00 0x97 0x9e
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x00	0
Message receiver ID	0x01	1
Message ID	0x20	0x20
Message number	0x0e	14

Field	Bytes	Value
Data length	0x02 0x00	0
Data interval	0x00 0x00	0
CRC	0x97 0x9e	-

## 4.7.2 SEND DATA Response

### Format

```

<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <interface version number> <error code>
<data analyze count> <data status warning> <data status error>
<air temperature> <relative humidity> <dew point temperature>
<frost point temperature> <surface temperature> <surface state>
<EN15518 surface state> <grip> <water layer thickness> <ice layer thickness>
<snow layer thickness> <unit status info> <unit error bits> <CRC>

```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
Interface version	7	1
Error code	8	1
Data analyze count	9 ... 10	2
Data status warning	11 ... 12	2
Data status error	13 ... 14	2
Air temperature	15 ... 18	4
Relative humidity	19 ... 22	4
Dew point temperature	23 ... 26	4
Frost point temperature	27 ... 30	4
Surface temperature	31 ... 34	4
Surface state	35	1
EN15518 surface state	36	1

Field	Offset	Size in Bytes
Grip	37 ... 40	4
Water layer thickness	41 ... 44	4
Ice layer thickness	45 ... 48	4
Snow layer thickness	49 ... 52	4
Unit status info	53 ... 56	4
Unit error bits	57	4
CRC	58 ... 59	2

**Example**

```

0xab 0x01 0x00 0x20 0x0e 0x36 0x00 0x43 0x00 0xd7 0x08 0x00 0x00 0x00 0x00
0x8f 0xc2 0xbf 0x41 0x29 0x5c 0x45 0x42 0xfb 0x52 0x4b 0x41 0xfb 0x52 0x4b
0x41 0x08 0xd7 0x02 0x42 0x01 0x01 0x85 0xeb 0x51 0x3f 0x00 0x00 0x00 0x00
0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x53 0xe8
    
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x01	1
Message receiver ID	0x00	0
Message ID	0x20	0x20
Message number	0x0e	14
Data length	0x36 0x00	54
Interface version	0x43	C
Error code	0x00	0
Data analyze count	0xd7 0x08	2263
Data status warning	0x00 0x00	0
Data status error	0x00 0x00	0
Air temperature	0x8f 0xc2 0xbf 0x41	23.9699
Relative humidity	0x29 0x5c 0x45 0x42	49.3400
Dew point temperature	0xfb 0x52 0x4b 0x41	12.7077
Frost point temperature	0xfb 0x52 0x4b 0x41	12.7077
Surface temperature	0x08 0xd7 0x02 0x42	32.7099
Surface state	0x01	1 = Dry

Field	Bytes	Value
EN15518 surface state	0x01	1 = Dry
Grip	0x85 0xeb 0x51 0x3f	0.8199
Water layer thickness	0x00 0x00 0x00 0x00	0.0
Ice layer thickness	0x00 0x00 0x00 0x00	0.0
Snow layer thickness	0x00 0x00 0x00 0x00	0.0
Unit status info	0x00 0x00 0x00 0x00	0
Unit error bits	0x00 0x00 0x00 0x00	0
CRC	0x53 0xe8	-

The data analyze count starts from 0 and is incremented at the maximum measurement rate supported by the mobile road sensor. The counter goes to 0 after reaching the maximum value

When continuous sending is initiated, the message number of the request message is used in the first data message and then incremented by 1 every time a new message is generated and sent.

## 4.8 SET REFERENCES Message

SET REFERENCES is used for updating reference values and road type specific coefficients. Coefficients are applied on reference values to adjust references values for given road surface type.

Table 19 SET REFERENCES Message

Message ID	0x30		
Data in Request	Offset	Data	Size in Bytes
	7	Surface type: 0 = Plate 1 = Road	1
		Data Length	1
Data in Response	Offset	Data	Size in Bytes
	9	Operation success/fail: 0 = Fail 1 = Success	1
	10	Status info	4
	14	Error bits	4
		Data Length	9

**SET REFERENCES** initiates data collection for the reference values setting. After collecting enough data, new reference setting values are calculated and taken into use.

Updated values depend on the given surface type:

- If the surface is reference plate and the surface type is **PLATE**, the reference values in parameters **0x50**, **0x51**, and **0x52** are updated.
- If reference setting is done using a road surface and the surface type is **ROAD**, road coefficients in parameters **0x53**, **0x54**, and **0x55** are updated.

Data collection takes a minimum of 25 seconds. The reference setting requires 1000 suitable samples. If the road surface is not static, meaning that the vehicle is moving, data collection can take longer than 25 seconds.

If initiation fails, the error condition is indicated with the status info and error bits that are included in the response message. The status and error information that is included in the response represent the conditions that were used for checking whether reference setting can be started or not.

The ongoing reference setting is indicated in the status data and does not affect the measurements. The user can check the progress by requesting status or data messages.

Reference value setting initiation fails if:

- Device is not ready to measure.
- Reference setting is already ongoing.
- Laser temperature change is ongoing.  
If the laser temperature change takes place during the reference setting, the sample collection is stopped and references are not updated. This is indicated in the status info. Temperature change is part of normal operation and reference setting can be initiated as soon as the temperature change is completed.
- Error condition that affects the measurements is detected.

If any of the error conditions that prevent the start of reference setting are detected during sample collection, sample collection is stopped and references are not updated. This is indicated in the status info. Error reason is stored to parameter **0x56**.

The sample quality is checked after all samples have been collected. If the quality criteria is not met, references are not updated. Failure is indicated in the status info.

When the mobile sensor receives the **SET REFERENCES** message, it clears all 3 status info bits that indicate failed reference setting and parameter **0x56** is reset.

The parameter update after data collection period can interfere with the ongoing data transfer.

An example of a use case where reference values have already been set using reference plate:

1. Client sends the **SET REFERENCES** message with the parameter **ROAD**.
2. Initiation succeeds. The unit collects data for 25 seconds. The **reference setting ongoing** bit is set in status info bits.
3. The client can check from the status info bits if data collection is ongoing. Both **GET UNIT STATUS** response and measurement data contain status info bits.
4. All data is collected. Unit clears the **reference setting ongoing** bit. Road type coefficient values are updated and taken into use.

**More Information**

- [Status Information \(page 12\)](#)
- [Error Bits \(page 14\)](#)

**4.8.1 SET REFERENCES Request****Format**

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <surface type> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
Surface type	7	1
CRC	8 ... 9	2

**Example**

```
0xab 0x00 0x01 0x30 0x0f 0x01 0x00 0x01 0x7f 0x4b
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x00	0
Message receiver ID	0x01	1
Message ID	0x30	0x10
Message number	0x0f	15
Data length	0x01 0x00	1
Surface type	0x01	1 = Road
CRC	0x7f 0x4b	-

## 4.8.2 SET REFERENCES Response

### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <interface version number> <error code>
<operation success/fail> <status info> <error bits> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
Interface version	7	1
Error code	8	1
Operation success/fail	9	1
Status info	10	4
Error bits	14	4
CRC	18 ... 19	2

### Example

```
0xab 0x01 0x00 0x30 0x0f 0x0b 0x00 0x43 0x00 0x01 0x00 0x00 0x00 0x00 0x00
0x00 0x00 0x00 0x0e 0x8c
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x01	1
Message receiver ID	0x00	0
Message ID	0x30	0x30
Message number	0x0f	15
Data length	0x0b 0x00	11
Interface version	0x43	C
Error code	0x00	0



Field	Bytes	Value
Operation success/fail	0x01	1 = Success
Status info	0x00 0x00 0x00 0x00	0
Error bits	0x00 0x00 0x00 0x00	0
CRC	0x0e 0x8c	-

## 4.9 STOP REFERENCE SETTING Message

STOP REFERENCE SETTING is used for interrupting reference data collection.

The reference values and road type specific coefficients are not changed when the reference setting is interrupted. Interrupted reference data collection is indicated in the status info. The flag is cleared when reference setting starts.

Table 20 STOP REFERENCE SETTING Message

<b>Message ID</b>	0x32
<b>Data in Request</b>	None
<b>Data in Response</b>	None

### 4.9.1 STOP REFERENCE SETTING Request

#### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
CRC	7 ... 8	2

## Example

```
0xab 0x00 0x01 0x32 0x10 0x00 0x00 0xa3 0x26
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x00	0
Message receiver ID	0x01	1
Message ID	0x32	0x32
Message number	0x10	16
Data length	0x00 0x00	0
CRC	0xa3 0x26	-

## 4.9.2 STOP REFERENCE SETTING Response

### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>  
<message number> <data length> <interface version number> <error code> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
Interface version	7	1
Error code	8	1
CRC	9 ... 10	2

### Example

```
0xab 0x01 0x00 0x32 0x10 0x02 0x00 0x43 0x00 0x8c 0x63
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x01	1
Message receiver ID	0x00	0
Message ID	0x32	0x32
Message number	0x10	16
Data length	0x02 0x00	2
Interface version	0x43	C
Error code	0x00	0 = Stop reference setting acknowledged
CRC	0x8c 0x63	-

## 4.10 SET ROAD COEFFICIENTS Message

SET ROAD COEFFICIENTS is used for updating all road coefficient values for parameters 0x53, 0x54, and 0x55, and taking new values into use immediately.

The parameter update can interfere with ongoing data transfer.

If the road coefficient values are not known, the **SET REFERENCES ROAD** message must be used instead of the SET ROAD COEFFICIENTS message.

Table 21 SET ROAD COEFFICIENTS Message

Message ID	0x31		
<b>Data in Request</b>	<b>Offset</b>	<b>Data</b>	<b>Size in Bytes</b>
	7	Road coefficient, laser 1	4
	11	Road coefficient, laser 2	4
	15	Road coefficient, laser 3	4
		<b>Data Length</b>	12
<b>Data in Response</b>	<b>Offset</b>	<b>Data</b>	<b>Size in Bytes</b>
	9	Operation success/fail: 0 = Fail 1 = Success	1
		<b>Data Length</b>	1



The response times can be longer when parameter values are changed. The extra time is needed to write parameters into the permanent memory before sending the response.

## 4.10.1 SET ROAD COEFFICIENTS Request

### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <road coefficient for laser 1>
<road coefficient for laser 2> <road coefficient for laser 3> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
Road coefficient for laser 1	7 ... 10	4
Road coefficient for laser 2	11 ... 14	4
Road coefficient for laser 3	15 ... 18	4
CRC	19 ... 20	2

### Example

```
0xab 0x00 0x01 0x31 0x11 0x0c 0x00 0x00 0x00 0x80 0x3f 0x00 0x00 0x00 0x40
0x00 0x00 0x40 0x40 0xc9 0xb2
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x00	0
Message receiver ID	0x01	1
Message ID	0x31	0x31
Message number	0x11	17
Data length	0x0c 0x00	12
Road coefficient for laser 1	0x00 0x00 0x80 0x3f	1
Road coefficient for laser 2	0x00 0x00 0x00 0x40	2
Road coefficient for laser 3	0x00 0x00 0x40 0x40	3
CRC	0xc9 0xb2	-

## 4.10.2 SET ROAD COEFFICIENTS Response

### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <interface version number> <error code>
<operation success/fail> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
Interface version	7	1
Error code	8	1
Operation success/fail	9	1
CRC	10 ... 11	2

### Example

```
0xab 0x01 0x00 0x31 0x11 0x03 0x00 0x43 0x00 0x01 0x97 0xf7
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x01	1
Message receiver ID	0x00	0
Message ID	0x31	0x31
Message number	0x11	17
Data length	0x03 0x00	3
Interface version	0x43	C
Error code	0x00	0
Operation success/fail	0x01	1 = Success
CRC	0x97 0xf7	-

## 4.11 GET PARAMETER Message

GET PARAMETER returns the value of the given parameter. The data length in the response is determined by its type.

Table 22 GET PARAMETER Message

<b>Message ID</b>	0x40		
<b>Data in Request</b>	<b>Offset</b>	<b>Data</b>	<b>Size in Bytes</b>
	7	Parameter ID	2
		<b>Data Length</b>	2
<b>Data in Response</b>	<b>Offset</b>	<b>Data</b>	<b>Size in Bytes</b>
	9	Parameter ID	2
	11	Data value	1...4
		<b>Data Length</b>	3...6

### More Information

- ▶ [Parameters \(page 53\)](#)

### 4.11.1 GET PARAMETER Request

#### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <parameter ID> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
Parameter ID	7 ... 8	2
CRC	9 ... 10	2

#### Example 1

The requested parameter is 0x13, mobile sensor ID.

```
0xab 0x00 0x01 0x40 0x12 0x02 0x00 0x13 0x00 0xde 0x18
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x00	0
Message receiver ID	0x01	1
Message ID	0x40	0x40
Message number	0x12	18
Data length	0x02 0x00	2
Parameter ID	0x13 0x00	0x13
CRC	0xde 0x18	-

### Example 2

The requested parameter is **0x41**, air temperature offset correction.

```
0xab 0x00 0x01 0x40 0x13 0x02 0x00 0x41 0x00 0x52 0xda
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x00	0
Message receiver ID	0x01	1
Message ID	0x40	0x40
Message number	0x13	19
Data length	0x02 0x00	2
Parameter ID	0x41 0x00	0x41 = air temperature offset correction
CRC	0x52 0xda	-

## 4.11.2 GET PARAMETER Response

### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <interface version number> <error code>
<parameter ID> <sensor ID> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 - 6	2
Interface version	7	1
Error code	8	1
Parameter ID	9 ... 10	2
Sensor ID	11	1
CRC	12 ... 13	2

### Example 1

The response returns the mobile sensor ID that was requested with parameter 0x13.

```
0xab 0x01 0x00 0x40 0x12 0x05 0x00 0x43 0x00 0x13 0x00 0x01 0x82 0x6d
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x01	1
Message receiver ID	0x00	0
Message ID	0x40	0x40
Message number	0x12	18
Data length	0x05 0x00	5
Interface version	0x43	C
Error code	0x00	0



Field	Bytes	Value
Parameter ID	0x13 0x00	0x13 = Sensor ID
Sensor ID	0x01	1
CRC	0x82 0x6d	-

### Example 2

The response returns the air temperature offset correction that was requested with parameter **0x41**.

```
0xab 0x01 0x00 0x40 0x13 0x08 0x00 0x43 0x00 0x41 0x00 0x00 0x00 0x00 0x00
0xd2 0x79
```

The example response contains the following information:

- Unit ID: 1
- Client ID: 0
- Message ID: 0x40
- Message number: 19
- Length: 8
- Interface version: C
- Error code: 0
- Parameter ID: 0x41
- Value: 0.0 (air temperature offset)

## 4.12 SET PARAMETER Message

SET PARAMETER is used for setting a given parameter to a given value.

The message can be used for the following purposes:

- Stop continuous data reporting.
- Set reference values.
- Set offset to road and air temperature.
- Configure the serial port speed.
- Configure units of reported quantities.



Some of the parameters are read-only. An attempt to set a value to a read-only parameter causes an **INVALID DATA** error.



Some of the parameters require a restart for the changes to take effect. The reported parameter value is reported immediately.



The response times can be longer when parameter values are changed. The extra time is needed to write parameters into the permanent memory before sending the response.

Table 23 SET PARAMETER Message

<b>Message ID</b>	0x41		
<b>Data in Request</b>	<b>Offset</b>	<b>Data</b>	<b>Size in Bytes</b>
	7	Parameter ID	2
	9	Parameter value	1...4
		<b>Data Length</b>	3...6
<b>Data in Response</b>	None		

#### More Information

- ▶ [Parameters \(page 53\)](#)
- ▶ [Message and Error Handling \(page 59\)](#)

## 4.12.1 SET PARAMETER Request

### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <parameter ID> <parameter value> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
Parameter ID	7 ... 8	2
Parameter value	9 ... 12	4
CRC	13 ... 14	2

### Example

The configured parameter is 0x41, air temperature offset correction.

```
0xab 0x00 0x01 0x41 0x14 0x06 0x00 0x41 0x00 0x00 0x00 0x40 0x3f 0xf5 0xeb
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x00	0
Message receiver ID	0x01	1
Message ID	0x41	0x41
Message number	0x14	20
Data length	0x06 0x00	6
Parameter ID	0x41 0x00	0x41 = air temperature offset
Parameter value	0x00 0x00 0x40 0x3f	0.75
CRC	0xf5 0xeb	-

## 4.12.2 SET PARAMETER Response

### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>  
<message number> <data length> <interface version number> <error code> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
Interface version	7	1
Error code	8	1
CRC	9 ... 10	2

### Example

```
0xab 0x01 0x00 0x41 0x14 0x02 0x00 0x43 0x00 0xf6 0x61
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x01	1
Message receiver ID	0x00	0
Message ID	0x41	0x41
Message number	0x14	20
Data length	0x02 0x00	2
Interface version	0x43	C
Error code	0x00	0
CRC	0xf6 0x61	-

## 4.13 RESTART UNIT Message

RESTART UNIT is used for initiating a software restart after an acknowledgment has been sent.

First the unit acknowledges the restart message, and restarts the software after that.

Table 24 RESTART UNIT Message

<b>Message ID</b>	0x50
<b>Data in Request</b>	None
<b>Data in Response</b>	None

### More Information

- [Parameters \(page 53\)](#)

### 4.13.1 RESTART UNIT Request

#### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1

Field	Offset	Size in Bytes
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
CRC	7 ... 8	2

### Example

```
0xab 0x00 0x01 0x50 0x15 0x00 0x00 0xe9 0x79
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x00	0
Message receiver ID	0x01	1
Message ID	0x50	0x50
Message number	0x15	21
Data length	0x00 0x00	0
CRC	0xe9 0x79	-

## 4.13.2 RESTART UNIT Response

### Format

```
<message start> <message sender ID> <message receiver ID> <message ID>  
<message number> <data length> <interface version number> <error code> <CRC>
```

Field	Offset	Size in Bytes
Message start	0	1
Message sender ID	1	1
Message receiver ID	2	1
Message ID	3	1
Message number	4	1
Data length	5 ... 6	2
Interface version	7	1
Error code	8	1

Field	Offset	Size in Bytes
CRC	9 ... 10	2

### Example

```
0xab 0x01 0x00 0x50 0x15 0x02 0x00 0x43 0x00 0x83 0x94
```

Field	Bytes	Value
Message start	0xab	0xab
Message sender ID	0x01	1
Message receiver ID	0x00	0
Message ID	0x50	0x50
Message number	0x15	21
Data length	0x02 0x00	2
Interface version	0x43	C
Error code	0x00	0 = Restart acknowledged
CRC	0x83 0x94	-

## 4.14 CRC Error Acknowledgment Message

MD30 sends an CRC error acknowledgment message when it detects an error in the CRC checksum.

The CRC error acknowledgment message does not contain data. The message receiver ID, message ID, and message number are set to 0.

To disable CRC error acknowledgments, use parameter `0x11`.

### Example of Request Message with CRC Error

```
0xab 0x00 0x01 0x10 0x00 0x00 0x00 0x00 0x00
```

The example request is a `GET UNIT ID` message with a checksum of 0.

## Format of CRC Error Acknowledgment Message

```
<message start> <message sender ID> <message receiver ID> <message ID>
<message number> <data length> <interface version number> <error code> <CRC>
```

Message Field	Offset	Size in Bytes	Bytes	Value
Message start	0	1	0xab	0xab
Message sender ID	1	1	0x01	1
Message receiver ID	2	1	0x00	0
Message ID	3	1	0x00	0x00
Message number	6	1	0x00	0
Data length	4 - 5	2	0x02 0x00	2
Interface version	7	1	0x43	C
Error code	8	1	0x01	1 = CRC error
CRC			0x3b 0xd3	-

### Example

```
0xab 0x01 0x00 0x00 0x00 0x02 0x00 0x43 0x01 0x3b 0xd3
```

The example response contains the following information:

- Unit ID: 1
- Client ID: 0
- Message ID: 0x00
- Message number: 0
- Length: 2
- Interface version: C
- Error code: 1 (CRC error)

### More Information

- ▶ [Error Codes \(page 56\)](#)

## 4.15 Parameters

The following table lists the parameter IDs and values.

The parameter value is sent in the request and response messages in the format given in the table.

The parameters are read and configured using the `GET PARAMETER` and `SET PARAMETER` messages.

Table 25 Parameters

Data ID	Read/Write	Description	Format	Value
0x10	R/W	Baud rate (bps). Defines the serial port speed. When writing to parameters, the parameter value is updated immediately, but the changes take effect after a restart. Communication works with old values until software is restarted. Baud rate change requires restart for the changes to take effect.	u8	0 = 9600 1 = 19200 2 = 38400 3 = 57600 4 = 115200
0x11	R	Sends acknowledgment about CRC checksum error.	u8	0 = No 1 = Yes
0x12	R	Latest error code. Read-only, not stored to non-volatile memory.	u8	For possible values, see <a href="#">Error Codes (page 56)</a>
0x13	R/W	Mobile sensor unit ID that is used in the header. When writing to parameters, the parameter value is updated immediately, but the changes take effect after a restart. Communication works with old values until software is restarted.	u8	1 = Default Values 0xFE and 0xFF are not allowed. See <a href="#">Multiple Mobile Sensor Units on Same Bus (page 60)</a> .
0x14	R/W	Receiver ID when automatic data sending is started. ID value is used as receiver ID only when automatic data sending is enabled.	u8	0 = Default
0x20	R/W	Data sending interval from MD30 to system (SEND DATA). Data interval in milliseconds (ms). See <a href="#">Automatic Data Sending (page 60)</a> .	u16	Range 25 ... 5000 0 = Automatic sending off (default)
0x21	R/W	Continuous sending started automatically after start-up. See <a href="#">Automatic Data Sending (page 60)</a> .	u8	0 = Not enabled (default) 1 = Enabled
0x30	R/W	Temperature unit. Changing temperature unit updates temperature offset correction parameters 0x40 and 0x41.	u8	0 = °C (default) 1 = °F
0x31	R/W	Layer thickness unit.	u8	0 = mm (default) 1 = inch



Data ID	Read/Write	Description	Format	Value
0x40	R/W	Road surface temperature offset correction. Set and read using temperature unit selected with parameter 0x30.	f32	0 = Default
0x41	R/W	Air temperature offset correction. Set and read using unit selected with parameter 0x30.	f32	0 = Default
0x50	R/W	Reference setting value of laser 1. Value change requires restart for the changes to take effect.	f32	1.0 = Default Value must be greater than 0
0x51	R/W	Reference setting value of laser 2. Value change requires restart for the changes to take effect.	f32	1.0 = Default Value must be greater than 0
0x52	R/W	Reference setting value of laser 3. Value change requires restart for the changes to take effect.	f32	1.0 = Default Value must be greater than 0
0x53	R/W	Reference setting coefficient of laser 1. Value change requires restart for the changes to take effect.	f32	1.0 = Default Value must be greater than 0
0x54	R/W	Reference setting coefficient of laser 2. Value change requires restart for the changes to take effect.	f32	1.0 = Default Value must be greater than 0
0x55	R/W	Reference setting coefficient of laser 3. Value change requires restart for the changes to take effect.	f32	1.0 = Default Value must be greater than 0
0x56	R	Error condition causing reference setting to be interrupted. Read-only, not stored to non-volatile memory. Error code is reset when reference setting is started.	u32	

When the SET PARAMETER message updates reference related parameters 0x50 - 0x55, changes take effect after the unit is restarted.

Receiver ID 0xFF can be used in cases where the assigned mobile sensor ID is not known to the client. The assigned ID is in the response message.

#### More Information

- ▶ [GET PARAMETER Message \(page 44\)](#)
- ▶ [SET PARAMETER Message \(page 47\)](#)
- ▶ [RESTART UNIT Message \(page 50\)](#)

# 5. Flow Control

## 5.1 Flow Control

A generic receiver performs the following checks to the received data:

- Message start marker is located to find the beginning of the message.
- Data length is recorded and the recorded number of data bytes is received or timeout occurs.
- Checksum of the message is checked. Messages with a checksum error are ignored.
- Receiver ID must match the assigned unit ID.
- Message ID must be valid.
- Data length must comply with the message description.
- Data contents must comply with the message data description.

These rules apply to both requests and responses.

The client design should take it into account that if a mobile sensor is commanded to send continuously data (`SEND DATA`), the response to a command is not necessarily the first message received after issuing the command. If the mobile sensor is about to send the data message at the same time when client is sending its command, the client may receive the data message before the acknowledgement. For example:

1. Mobile road sensor is set to transmit `SEND DATA` response messages at 1 Hz.
2. Client sends the `GET STATUS` message.
3. Mobile road sensor starts sending `SEND DATA` at the same time with the client.
4. Client receives the `SEND DATA` message.
5. Client receives the `GET STATUS` response.

## 5.2 Error Codes

The error codes are used in the response message.

Table 26 Error Codes

Error Code	Description	Explanation
0	No error	-
1	CRC checksum error	CRC checksum check fails.
2	Invalid message ID	Unit receives an unknown message ID.
3	Invalid length	Data length does not comply with the message type. For example, a data length of 0 in a message which always contains data triggers this error.
4	Invalid data	Data contained in the message is invalid. For example, an attempt to set an invalid port speed or to set value to a read-only parameter.

**More Information**

- ▶ [CRC Error Acknowledgment Message \(page 52\)](#)

## 5.3 CRC Calculation

The cyclic redundancy check (CRC) that is used in the messages is CRC-16 CCITT-FALSE.

The following is a C code implementation.

```

u16 calculateCrc(u16  bufSize, u8 * buf)
{
  u16 i;
  u16 index;
  u16 crc  = 0xFFFF;
  for (i = 0; i <  bufSize; i++)
  {
    index = (crc  >> 8) ^ buf[i];
    crc = (crc  << 8) ^ crcTable[index & 0xFF];
  }
  return crc;
}
#define  SIZE_CRC_TABLE  (256)
const u16  crcTable[SIZE_CRC_TABLE] =
{
  0x0000, 0x1021, 0x2042, 0x3063, 0x4084, 0x50A5, 0x60C6, 0x70E7, 0x8108,
  0x9129, 0xA14A, 0xB16B, 0xC18C, 0xD1AD, 0xE1CE, 0xF1EF, 0x1231, 0x0210,
  0x3273, 0x2252, 0x52B5, 0x4294, 0x72F7, 0x62D6, 0x9339, 0x8318, 0xB37B,
  0xA35A, 0xD3BD, 0xC39C, 0xF3FF, 0xE3DE, 0x2462, 0x3443, 0x0420, 0x1401,
  0x64E6, 0x74C7, 0x44A4, 0x5485, 0xA56A, 0xB54B, 0x8528, 0x9509, 0xE5EE,
  0xF5CF, 0xC5AC, 0xD58D, 0x3653, 0x2672, 0x1611, 0x0630, 0x76D7, 0x66F6,
  0x5695, 0x46B4, 0xB75B, 0xA77A, 0x9719, 0x8738, 0xF7DF, 0xE7FE, 0xD79D,
  0xC7BC, 0x48C4, 0x58E5, 0x6886, 0x78A7, 0x0840, 0x1861, 0x2802, 0x3823,
  0xC9CC, 0xD9ED, 0xE98E, 0xF9AF, 0x8948, 0x9969, 0xA90A, 0xB92B, 0x5AF5,
  0x4AD4, 0x7AB7, 0x6A96, 0x1A71, 0x0A50, 0x3A33, 0x2A12, 0xDBFD, 0xCBDC,
  0xFBBF, 0xEB9E, 0x9B79, 0x8B58, 0xBB3B, 0xAB1A, 0x6CA6, 0x7C87, 0x4CE4,
  0x5CC5, 0x2C22, 0x3C03, 0x0C60, 0x1C41, 0xEDAE, 0xFD8F, 0xCDEC, 0xDDCD,
  0xAD2A, 0xBD0B, 0x8D68, 0x9D49, 0x7E97, 0x6EB6, 0x5ED5, 0x4EF4, 0x3E13,
  0x2E32, 0x1E51, 0x0E70, 0xFF9F, 0xEFBE, 0xDFDD, 0xCFFC, 0xBF1B, 0xAF3A,
  0x9F59, 0x8F78, 0x9188, 0x81A9, 0xB1CA, 0xA1EB, 0xD10C, 0xC12D, 0xF14E,
  0xE16F, 0x1080, 0x00A1, 0x30C2, 0x20E3, 0x5004, 0x4025, 0x7046, 0x6067,
  0x83B9, 0x9398, 0xA3FB, 0xB3DA, 0xC33D, 0xD31C, 0xE37F, 0xF35E, 0x02B1,
  0x1290, 0x22F3, 0x32D2, 0x4235, 0x5214, 0x6277, 0x7256, 0xB5EA, 0xA5CB,
  0x95A8, 0x8589, 0xF56E, 0xE54F, 0xD52C, 0xC50D, 0x34E2, 0x24C3, 0x14A0,
  0x0481, 0x7466, 0x6447, 0x5424, 0x4405, 0xA7DB, 0xB7FA, 0x8799, 0x97B8,
  0xE75F, 0xF77E, 0xC71D, 0xD73C, 0x26D3, 0x36F2, 0x0691, 0x16B0, 0x6657,
  0x7676, 0x4615, 0x5634, 0xD94C, 0xC96D, 0xF90E, 0xE92F, 0x99C8, 0x89E9,
  0xB98A, 0xA9AB, 0x5844, 0x4865, 0x7806, 0x6827, 0x18C0, 0x08E1, 0x3882,
  0x28A3, 0xCB7D, 0xDB5C, 0xEB3F, 0xFB1E, 0x8BF9, 0x9BD8, 0xABBB, 0xBB9A,
  0x4A75, 0x5A54, 0x6A37, 0x7A16, 0x0AF1, 0x1AD0, 0x2AB3, 0x3A92, 0xFD2E,
  0xED0F, 0xDD6C, 0xCD4D, 0xBDAA, 0xAD8B, 0x9DE8, 0x8DC9, 0x7C26, 0x6C07,
  0x5C64, 0x4C45, 0x3CA2, 0x2C83, 0x1CE0, 0x0CC1, 0xEF1F, 0xFF3E, 0xCF5D,
  0xDF7C, 0xAF9B, 0xBFBA, 0x8FD9, 0x9FF8, 0x6E17, 0x7E36, 0x4E55, 0x5E74,
  0x2E93, 0x3EB2, 0x0ED1, 0x1EF0
};

```

For example, the checksum of the following 10-byte sequence is 0xc241:

```
0x0, 0x1, 0x2, 0x3, 0x4, 0x5, 0x6, 0x7, 0x8, 0x9
```

## 5.4 Message and Error Handling

The mobile sensor records the time when the message handling starts. The mobile sensor expects to receive a complete message within the given time, which is known as data discarding period. If a checksum error is detected, all received data is discarded until data discarding period has passed. If message reception is not ready by the end of data discarding period, the receiver is reset and the received data bytes are ignored. The unit is ready to receive the next message after the response message sending has started or the data discarding period has passed.

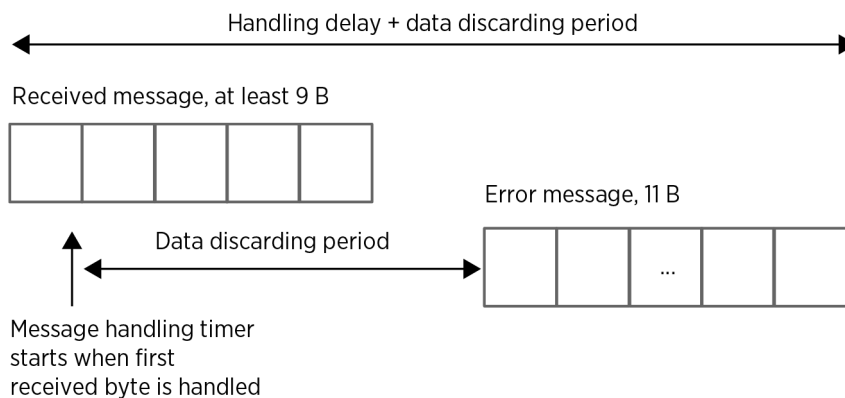


Figure 3 Error Handling

### Response Time

In case a checksum error is acknowledged, the receiver waits the data discarding period of 20 ms before producing a response message. The data discarding period is included in the total response time. Disabling the checksum error acknowledgment does not affect the total time. Incomplete messages and time-outs have the same total time, but are not acknowledged.

The mobile sensor is expected to start data handling and to produce a response message in 500 ms after data reception has started. This is the total response time, including message handling delays and the data discarding period. This time does not include the time it takes to physically transmit and receive the message at a given data rate. Ongoing continuous data sending may be interrupted for the duration of the message handling.

Response times of up to 2.5 seconds can be expected from messages that update several parameters. The `SET ROAD COEFFICIENTS` message has the longest response time. Ongoing continuous data sending is interrupted for the duration of the message handling. Also the `SET REFERENCES` message updates several parameters. The response is sent within 500 ms, but the parameter update takes place after the data collection period. During the parameter update, continuous data sending frequency cannot be guaranteed. Assigned data sending frequency is assumed as soon as the parameter update is over.

## Buffer Overflow

Buffer overflow occurs if the mobile sensor receives a message that does not fit into the data buffer that is allocated for data reception. In case of buffer overflow, the receiver is reset and no acknowledgment is sent.

## 5.5 Multiple Mobile Sensor Units on Same Bus

The protocol has been designed to be used in a point-to-point manner over RS-232, but it has features which allow having more than one unit on the same bus. Parameter **0x13** can be used for allocating individual IDs to each unit. The mobile road sensor uses the ID to check whether the message is addressed to it and whether to handle it. Messages to other units are ignored and the unit is ready to receive the next message right away.

If there are use cases where there is more than one mobile road sensor unit on the same bus, each unit must have a unique ID. Also the device controlling the mobile sensor units must have a unique ID.

If there is more than one mobile road sensor unit on the same bus, the CRC error acknowledgment must be disabled from parameter **0x11**.

Currently the protocol does not support broadcast messages that do not require acknowledgment.

- Receiver ID **0xFF** is reserved for messages that the mobile sensor unit receives and acknowledges regardless of the assigned unit ID. **0xFF** cannot be used as a unit ID.
- Receiver ID **0xFE** is reserved for future use and cannot be used as a unit ID.

## 5.6 Automatic Data Sending

The mobile sensor can be configured to start sending data automatically after boot. Parameters **0x20** and **0x21** control the automatic data sending of the **SEND DATA** message.

Parameters are checked when the unit is started. If sending is enabled and the interval is not zero, the data sending starts.

The automatic data sending interval cannot be adjusted while the unit is sending data. Changes to the parameter values take effect when the unit is restarted.

**SEND DATA** can be used for stopping data sending or changing the interval. Note that the parameters controlling the automatic sending are not affected. If the parameter controlling the interval is not updated, automatic sending with the original interval resumes when the unit is restarted.

The receiver ID that is used in automatic sending is defined by parameter **0x14**. The parameter is common to all data messages.

### More Information

- [SEND DATA Message \(page 30\)](#)

# Warranty and Product Returns

For standard warranty terms and conditions, see [www.vaisala.com/warranty](http://www.vaisala.com/warranty).

Please observe that any such warranty may not be valid in case of damage due to normal wear and tear, exceptional operating conditions, negligent handling or installation, or unauthorized modifications. Please see the applicable supply contract or Conditions of Sale for details of the warranty for each product.

If the product is faulty, these steps help to speed up the return process and avoid extra costs.

1. Read the warranty information.
2. Contact Vaisala technical support and request a Return Material Authorization (RMA) and shipping instructions.



Always request the RMA before returning any faulty material.  
Provide the failure report as requested.

# Technical Support



Contact Vaisala technical support at [helpdesk@vaisala.com](mailto:helpdesk@vaisala.com). Provide at least the following supporting information:

- Product name, model, and serial number
- Name and location of the installation site
- Name and contact information of a technical person who can provide further information on the problem

For more information, see [www.vaisala.com/support](http://www.vaisala.com/support).

# Recycling



Recycle all applicable material.



Follow the statutory regulations for disposing of the product and packaging.







**VAISALA**

[www.vaisala.com](http://www.vaisala.com)

