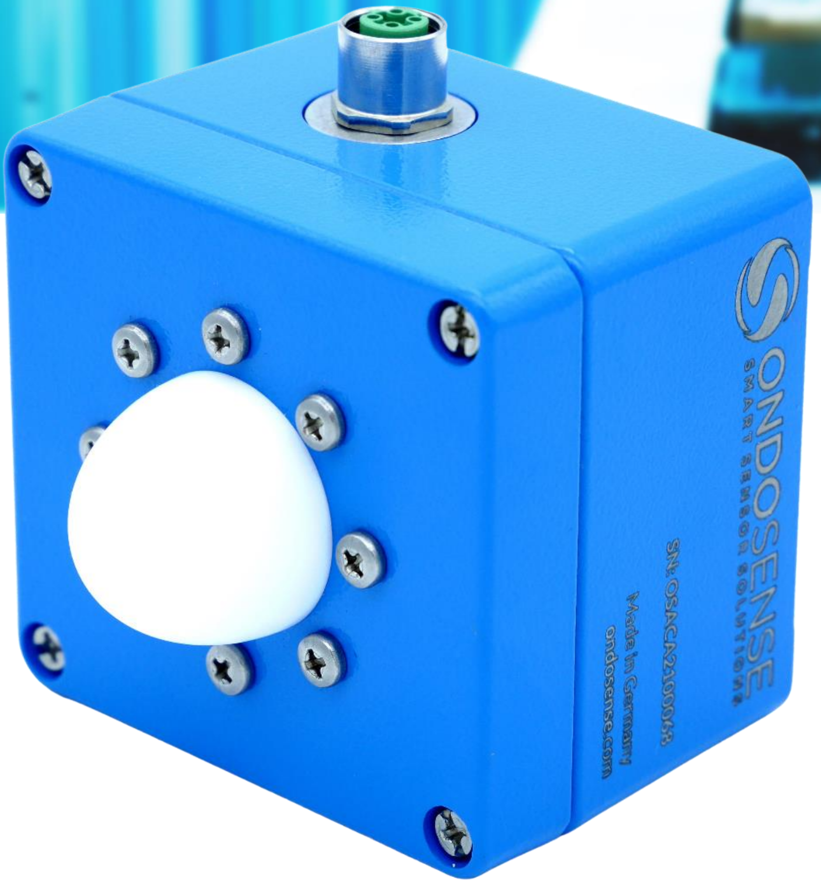
Application Note OS1

RS485 API

Version 3.2.0



ONDOSENSE apex

Contents

Application Note OS1

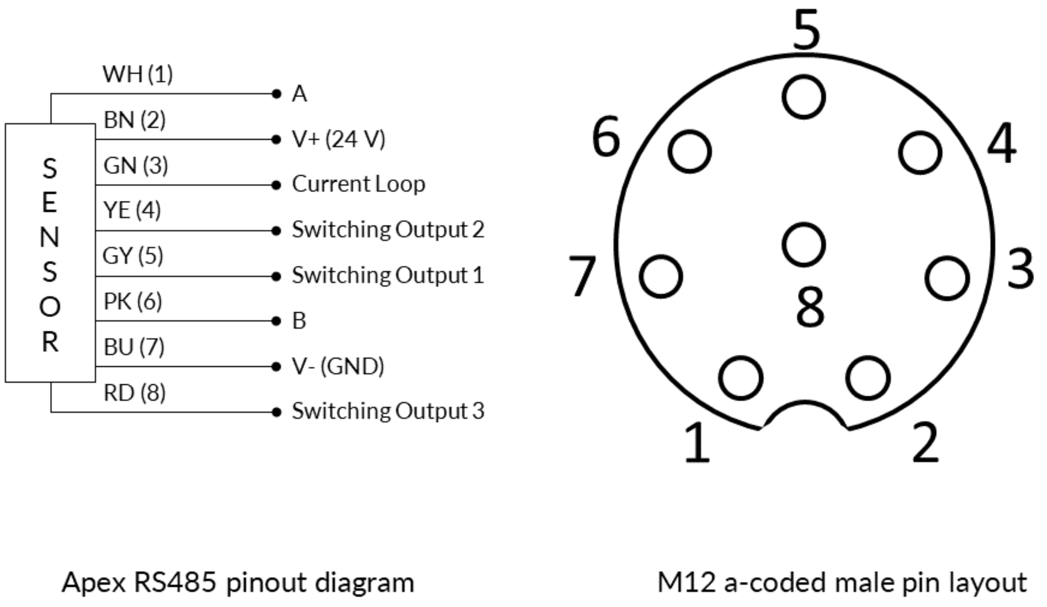
OndoSense *apex*

1 Introduction

The RS485 serial interface of the sensor operates in half-duplex mode (two-wire bus). To ensure an error-free communication, the sensor (slave) only sends data on request from an external controller (master). The sensor interface is internally terminated and needs to be operated using a point-to-point connection with no other devices on the RS485 bus. Supported baud rates range from 9600 baud to 921600 baud (default: 19200), with 1 stop bit, no parity, 8 bit word. The RS485 connection was tested up to 1 MBaud and 30 m of cable.

2 Connection

Figure 1 shows the connections of the sensor. V+ (24 V) and V- (GND) are used for the power supply. The pins A and B are used for RS485 data exchange. These 4 pins are needed for operating the sensor with RS485 communication. The sensor can be connected with an 8-pin a-coded M12 cable. Additional pins are the 3 switching outputs and the current loop.



Apex RS485 pinout diagram

M12 a-coded male pin layout

Figure 1: Connectivity diagram

3 Command Overview

Each command sent to the sensor consists of one byte for the command, followed by a command body. After receiving a command, the sensor will reply with the corresponding status code (see Status Code) and with a response body.

See table below for the commands and the length of the command body. A Body length of “0 Bytes” means that the host must only send the command itself without additional bytes.

Command name	Hex-Code	Body Length	Short description
Read parameter	0x01	1 Byte	Reads a parameter value from the sensor. The body of the command specifies the parameter.
Write parameter	0x02	5 Bytes	Writes a parameter value to the sensor. The body of the command is constructed as: ParameterID (1 Byte) + ParameterValue as Int32 (4 Bytes)
Measurement	0x03	0 Bytes	Requests a single or multiple measurements of the type of measurement set by the result data selector parameter
Factory reset	0xFF	5 Bytes	Resets all parameters to its default values. The body must be equal to: `0x52 0x45 0x53 0x45 0x54` (This is `RESET` in ASCII)
Autoset amplifier	0x07	0 Bytes	Sets the amplifier values automatically
Background calibration	0x0D	0 Bytes	Records background data. Data will be subtracted for further measurements (the calibration stays until it is removed by the remove background calibration command)
Remove background calibration	0x0E	0 Bytes	Deletes recorded background data
Save parameters	0x0F	0 Bytes	Saves all current parameters (except result data selector and baud rate)
Read parameter minimum	0x10	1 Byte	Reads the minimal allowed value of a parameter. The body of the command specifies the parameter.
Read parameter maximum	0x11	1 Byte	Reads the maximal allowed value of a parameter. The body of the command specifies the parameter.
Restart high precision [1] mode	0x19	0 Bytes	Manually restart the high precision mode (resets the high precision distance to zero).

[1] This command is only available for some OndoSense product variants.

4 Status Code

Status code	Hex-Code	Short description
Success	1 (0x01)	Command was successfully executed
Success weak signal	2 (0x02)	The measurement was successful, but the reflected signal amplitude is too low for consistent target detection (check amplification parameters or target position)
Error	-1 (0xFF)	An error has occurred
Command error	-2 (0xFE)	Unknown command
Parameter error	-3 (0xFD)	Unknown parameter
Range error	-4 (0xFC)	The value set for this parameter is not within the allowed range of values
Forbidden error	-5 (0xFB)	This parameter is protected and cannot be changed
No target error	-6 (0xFA)	The sensor has not detected a target
Target lost	-7 (0xF9)	Only applicable to high precision mode: The sensor lost the target.
Calculation error	-8 (0xF8)	An error occurred in the distance calculation module.

5 Parameter

Parameter name	Hex-Code	Allowed values	Default value	Unit
Result data selector[1]	0x41	1: IQ-Data 2: Spectrum 4: Peak list 8: Peak 16: Distance 64: Distance list 128: Ramp Count 256: Temperature 512: High precision distance	16 (Distance)	-
Measurement rate	0x43	1 – max[2]	max	Hz
Minimal distance[3]	0x44	Variant specific	Variant specific	mm
Maximal distance[3]	0x45	Variant specific	Variant specific	mm
Number of integrations (Raw data)	0x46	1 – 10000	1	-
Number of integrations (Spectrum data)	0x47	1 – 10000	1	-
Radar profile selector[4]	0x48	2: Long Range - Fast moving objects 3: Close Range (up to 5 m) 5: Long Range - Slow moving objects 16: Maximum Accuracy (up to 6 m)	2 (Long Range - Fast moving objects)	-
Baud rate	0x49	9600 – 921600	19200	Baud
High precision distance threshold[5]	0x83	1 – max(uint32)	25	$\frac{1}{2}\lambda$
High precision timeout[5]	0x84	0 – max(uint32)	5000	ms
Pre amplifier gain Q-channel	0x70	0 – 255	127	-
Pre amplifier gain I-channel	0x71	0 – 255	127	-
ADC amplifier gain Q-channel	0x72	0 – 255	127	-
ADC amplifier gain I-channel	0x73	0 – 255	127	-
Rx delay	0x90	0 – 100000	5000	μs
Threshold Sensitivity	0x91	0 – max(int32)	10	-
Threshold Offset	0x82	0 – max(int32)	0	-
Distance Offset	0xED	(-min. distance) – max. distance	0	mm
Exponential moving average time	0x96	0 – max(int32)	0	ms

Continuation of page 4

Parameter name	Hex-Code	Allowed values	Default value	Unit
Outlier maximum time	0x97	0 – max(int32)	0	ms
Outlier maximum distance	0x98	0 – max(int32)	0	ms
Outlier maximum speed	0x99	0 – max(int32)	0	μm/s
Peak sorting method	0x92	0: Distance 1: Amplitude 2: Normalized amplitude 3: Distance backwards 4: Amplitude backwards 5: Normalized amplitude backwards	1 (Amplitude)	-
Peak index	0x93	0 – 4	0	-
Serial number (read only)	0xF0	1 – max(uint32)	-	-

[1] You can also select multiple data types at the same time, such as distance and spectrum. Then the value for the result data selector results in 16 + 2 =18 (0b00010010).

[2] Maximum measurement rate depends on OndoSense product variants.

[3] Please note for which measuring range your sensor has been configured and calibrated (see order). The distance can be set beyond this measuring range, but then the specified accuracy cannot be guaranteed.

[4] Available profiles depend on OndoSense product variants.

[5] This parameter is only available for some OndoSense product variants.

5.1 Hardware interface configuration

The following parameters allow configuration of the switching outputs and the current loop.

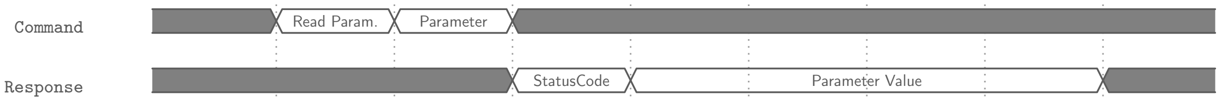
Parameter name	Hex-Code	Allowed values	Default value	Unit
Switching Output 1 enabled	0xC0	0 – 1	1 (true)	-
Switching Output 2 enabled	0xC1	0 – 1	1 (true)	-
Switching Output 3 enabled	0xC2	0 – 1	1 (true)	-
Switching Output 1 polarity	0xD0	0 – 1	1 (Active high)	-
Switching Output 2 polarity	0xD1	0 – 1	1 (Active high)	-
Switching Output 3 polarity	0xD2	0 – 1	1 (Active high)	-
Switching Output 1 value activate	0xD4	0 – max(int32)	200000	μm
Switching Output 2 value activate	0xD5	0 – max(int32)	200000	μm
Switching Output 3 value activate	0xD6	0 – max(int32)	200000	μm
Switching Output 1 value deactivate	0xD8	0 – max(int32)	500000	μm
Switching Output 2 value deactivate	0xD9	0 – max(int32)	500000	μm
Switching Output 3 value deactivate	0xDA	0 – max(int32)	500000	μm
Switching Output 1 hysteresis width	0xDC	0 – max(int32)	10000	μm
Switching Output 2 hysteresis width	0xE4	0 – max(int32)	10000	μm
Switching Output 3 hysteresis width	0xE5	0 – max(int32)	10000	μm
Switching Output 1 response delay time	0xDD	0 – max(int32)	0	μs
Switching Output 2 response delay time	0xE7	0 – max(int32)	0	μs
Switching Output 3 response delay time	0xE8	0 – max(int32)	0	μs
Switching Output 1 response release time	0xDE	0 – max(int32)	0	μs
Switching Output 2 response release time	0xEA	0 – max(int32)	0	μs
Switching Output 3 response release time	0xEB	0 – max(int32)	0	μs
Switching Output 1 selection IO[1]	0xDF	0 – 1	1 (Input)	-
Switching Output 2 selection IO[1]	0xE0	0 – 1	1 (Input)	-
Switching Output 3 selection IO[1]	0xE1	0 – 1	1 (Input)	-
Current loop min distance (4 mA)	0xB0	0 – max(int32)	Sensor specific	mm
Current loop max distance (20 mA)	0xB1	0 – max(int32)	Sensor specific	mm
Current loop error mode[2]	0xB2	0: Low (3.6 mA) 1: Preserve	0: Low (3.6 mA)	-

[1] Setting the IO selection to the value 1, configures the sensor to use the output lines as "inputs". Currently there is no function associated to such an input but in order to protect the device from wrong wiring, the default value is "inputs".

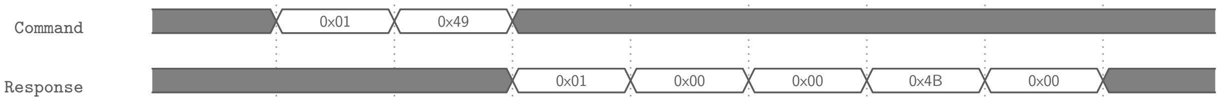
[2] Defines the behaviour of the current loop interface when no target is detected or another error occurs in the sensor. The "Low" setting leads to anb output of 3.6 mA in the error case With the "preserve" setting, the last valid measurement is used on the current loop.

5.2 Read Parameter

Structure to read a parameter

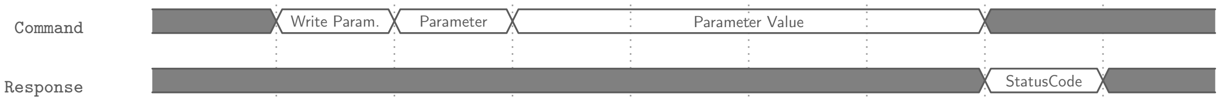


Example: Read value of parameter baudrate (19.200)

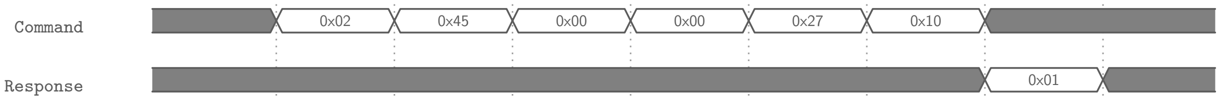


5.3 Write Parameter

Structure to set a parameter



Example: Write value 10.000 [mm] to parameter maximal distance



6 Measurement Data

The parameter result data selector defines which data is serialized when a measurement command is sent to the sensor (see chapter Parameter)
When multiple measurement types are selected via the result data selector, the data output occurs in the order as specified by the index in the following table.

Index	Result data	Format
1	IQ-Data	Count (2 Bytes) + Count × Interleaved (I-Data (1 Byte) + Q-Data (1 Byte))
2	Spectrum	Count (2 Bytes) + Max. Frequency [Hz] (4 Bytes) + Frequency Interval [Hz] (4 Bytes) + Amplitude (4 Bytes) + Count × Magnitude Spectrum Bin (1 Byte) + Count × Spectrum Threshold (1 Byte)
3	Peak List	Count (1 Byte) + Index (1 Byte) + Count × (Frequency $[10^{-2}$ Hz] (4 Bytes) + Phase[1] (2 Bytes) + Amplitude (4 Bytes))
4	Peak	Frequency $[10^{-2}$ Hz] (4 Bytes) + Phase† (2 Bytes) + Amplitude (4 Bytes)
5	Distance List	Count (1 Byte) + Index (1 Byte) + Count × Distance in micrometer (4 Bytes as UINT32)
6	Distance	Distance in micrometer (4 Bytes)
7	Measurement Count	(Cyclic) Number of RF-ramps since sensor power-up (4 Bytes)
8	Temperature	Internal temperature in centi-Degree Celsius (2 Bytes) + External temperature (2 Bytes *currently not supported*)
9	High Precision Distance[2]	(Cyclic) Target lost counter (1 Byte) + High precision mode distance in micrometer (4 Bytes as INT32)

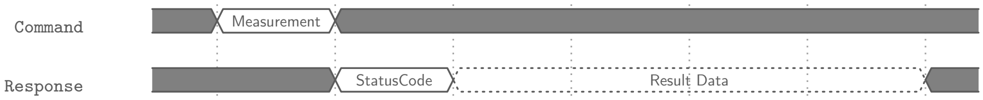
[1] The phase is serialized as UINT16. To obtain the phase in rad, apply the following formula: $\phi_{\text{rad}} = \phi_{UINT16} \times \frac{2\pi}{\text{UINT16MAX}} - \pi$.
[2] This data selector value is only available for some OndoSense product variants

Notes:

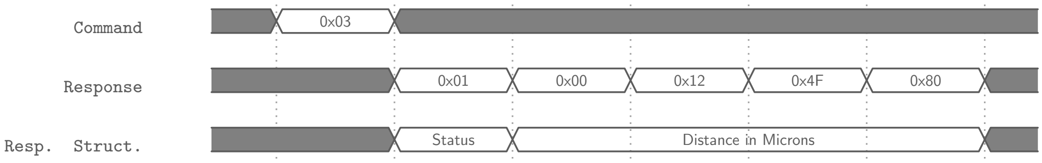
- When reading multiple types of data simultaneously (e.g., distance and spectrum), it's necessary to read the status byte before each type of data. Reading the status only once may result in misinterpretation of every data type beyond the first one.
- When the status of a given result data set is negative (i.e. an error code, see chapter Status Code), the result data is *not* serialized.

6.1 Get Measurement Data

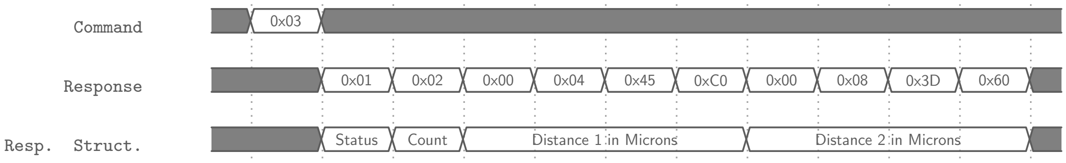
Structure for getting Measurement data



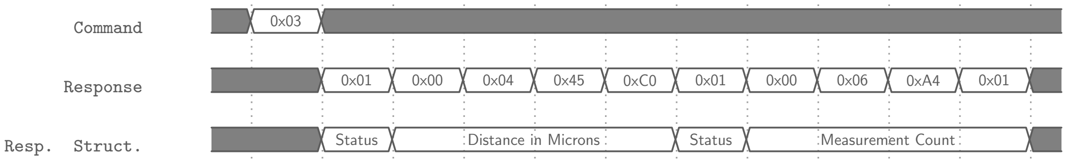
Example: Measurement of distance (value: 120cm)



Example: Measurement of distance list with 2 peaks (28 cm, 54 cm)



Example: Measurement of distance and measurement count, i.e. result data selector = (144 = 16 + 128 = 0b10010000)



7 Pre-Setup

Connect the sensor to the ConfigBox and configure it via OndoNet. For more information on how to use the sensor in OndoNet, click on the operation manual in OndoNet.
Once the sensor is configured for the measurement task, remove the sensor from the ConfigBox. Connect the master device and the sensor by cable, connecting the 4 required pins of the power supply and the RS485 communications (see pinout diagram in chapter Connection). The following commands need to be executed for the sensor to output data.

1. Open the serial interface of the master device with the default baud rate of 19200. If you want a faster connection change the baud rate via the write parameter command.

2. If you want to receive any other result type than distance values (default), the result data selector parameter has to be set accordingly (see chapter Parameter). The sensor will respond with success status if the write parameter command has been successfully executed. Otherwise, see chapter Status Code to identify the status code of the response byte.

3. Execute the measurement command and receive the answer. The formatting of the measurement results is shown in chapter Measurement Data.

Note: As soon as the sensor is disconnected from the power supply, the baud rate and the result data selector are reset to default.

8 Test Program

The following python 3 program provides a simple way of interfacing the apex RS485 sensor on a PC using a USB-to-RS485 adapter. Please note however, that depending on the manufacturer, these adapters can add padding bytes to the received message, which has to be considered when specifying the desired number of bytes to receive. The program shown below is only a code example. Under https://ondosense.com/**Setup** you can find in the download area an example python code file to download and test.

```
############################################################
# OS APEX RS485 API Example (Python 3)
############################################################

import serial

# Make sure that you have installed pyserial package for Python 3:
# pip install pyserial

# User specific settings (may need to be changed)
COM_PORT_RADAR = 'COM0'

# Commands
READ_PARAMETER_CMD = 0x01
WRITE_PARAMETER_CMD = 0x02
GET_MEASUREMENT_CMD = 0x03

# Parameter
PARAMETER_RESULT_DATA_SELECTOR = int(0x41)
PARAMETER_BAUDRATE = int(0x49)

# Status codes
SUCCESS = 1
API_ERROR = -1

# Open serial port
ser = serial.Serial(port=COM_PORT_RADAR, baudrate=19200,
                    bytesize=8, parity='N', stopbits=1, timeout=1)

# Set higher baudrate if needed (default 19200)
ser.write([WRITE_PARAMETER_CMD, PARAMETER_BAUDRATE, 0, 1, 194, 0])
# 0*256^3 + 1*256^2 + 194*256 + 0 = 115200
response = ser.read(1)
if response[0] != SUCCESS:
    print('Set␣baudrate␣failed')  # further error handling

# Reconfigure serial port in case baudrate was changed
ser.close()
ser = serial.Serial(port=COM_PORT_RADAR, baudrate=115200,
                    bytesize=8, parity='N', stopbits=1, timeout=1)

# Set result data selector to receive distance values (distance is default)
ser.write([WRITE_PARAMETER_CMD, PARAMETER_RESULT_DATA_SELECTOR, 0, 0, 0, 16])
# 0b00010000 = 16
```

```
response = ser.read(1)
if response[0] != SUCCESS:
    print('Set␣Result␣Data␣Selector␣failed')  # further error handling

# Endless loop
while True:
    ser.flushInput()
    # Request distance value
    ser.write([GET_MEASUREMENT_CMD])
    # Capture response
    response = ser.read(5)
    if response[0] != SUCCESS:
        print('Measurement␣error␣received')
        continue  # check error types and further error handling

    # Determine measured distance value from uint32 response and convert from micron to meter
    distance = (response[1] * pow(256, 3) + response[2] * pow(256, 2)
                + response[3] * 256 + response[4]) / 1e6
    print(distance, '␣m')  # Display measured distance value
```

9 Change log

Version number	Date	Description	Affected chapters
3.2.0	05.02.2024	Add description for (high precision) target lost status code. Add status code “Calculation error” Threshold offset can be set over the API. Add description of spectrum integration parameter. Unify Baud rate max value throughout the document. Add parameters for advanced filtering (exponential average on distance, outlier detection and distance offset). Add table for hardware interface configuration parameters. Adapt and correct allowed values for minimal distance, maximal distance, high precision distance threshold and peak index.	4, 5, 4
3.0.0	23.11.2023	Adapt serialization of high precision distance (int32 representing high precision distance in microns instead of float for meters)	6