

Acuity[®]

AccuRange AR500™ Laser Sensor

User's Manual



LLL000501 – Rev. 1.0
For use with AR500™ Laser Position Sensor
August 10, 2012

Acuity
A product line of Schmitt Industries, Inc.
2765 NW Nicolai St.
Portland, OR 97210
www.acuitylaser.com

Limited Use License Agreement

CAREFULLY READ THE FOLLOWING TERMS AND CONDITIONS BEFORE OPENING THE PACKAGE CONTAINING THE PRODUCT AND THE COMPUTER SOFTWARE LICENSED HEREUNDER. CONNECTING POWER TO THE MICROPROCESSOR CONTROL UNIT INDICATES YOUR ACCEPTANCE OF THESE TERMS AND CONDITIONS. IF YOU DO NOT AGREE WITH THE TERMS AND CONDITIONS, PROMPTLY RETURN THE UNIT WITH POWER SEAL INTACT TO THE DEALER FROM WHOM YOU PURCHASED THE PRODUCT WITHIN FIFTEEN DAYS FROM DATE OF PURCHASE AND YOUR PURCHASE PRICE WILL BE REFUNDED BY THE DEALER. IF THE DEALER FAILS TO REFUND YOUR PURCHASE PRICE, CONTACT SCHMITT INDUSTRIES, INC. IMMEDIATELY AT THE ADDRESS SET OUT BELOW CONCERNING RETURN ARRANGEMENTS.

Schmitt Industries, Inc. provides the hardware and computer software program contained in the microprocessor control unit. Schmitt Industries, Inc. has a valuable proprietary interest in such software and related documentation ("Software"), and licenses the use of the Software to you pursuant to the following terms and conditions. You assume responsibility for the selection of the product suited to achieve your intended results, and for the installation, use and results obtained.

License Terms And Conditions

- a. You are granted a non-exclusive, perpetual license to use the Software solely on and in conjunction with the product. You agree that the Software title remains with Schmitt Industries, Inc. at all times.
- b. You and your employees and agents agree to protect the confidentiality of the Software. You may not distribute, disclose, or otherwise make the Software available to any third party, except for a transferee who agrees to be bound by these license terms and conditions. In the event of termination or expiration of this license for any reason whatsoever, the obligation of confidentiality shall survive.
- c. You may not disassemble, decode, translate, copy, reproduce, or modify the Software, except only that a copy may be made for archival or back-up purposes as necessary for use with the product.
- d. You agree to maintain all proprietary notices and marks on the Software.
- e. You may transfer this license if also transferring the product, provided the transferee agrees to comply with all terms and conditions of this license. Upon such transfer, your license will terminate and you agree to destroy all copies of the Software in your possession.

Procedures for Obtaining Warranty Service

1. Contact your Acuity distributor or call Schmitt Industries, Inc. to obtain a return merchandise authorization (RMA) number within the applicable warranty period. Schmitt Industries will not accept any returned product without an RMA number.
2. Ship the product to Schmitt Industries, postage prepaid, together with your bill of sale or other proof of purchase. your name, address, description of the problem(s). Print the RMA number you have obtained on the outside of the package.

This device has been tested for electromagnetic emissions and immunity and has been found to be in compliance with the following directives for class A equipment:

**EN 62500-6-2:2002
EN 55011:2000**

This device complies with part 15 of the FCC Rules. Operation is subject to the following two conditions:

(1) This device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

Note: This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this device in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his or her own expense.

This manual copyright © 2012, Schmitt Industries, Inc.



User's Manual for the
AR500™ Series Laser Sensor
Rev. 1.0

Table of Contents

1.	Introduction	1
1.1	General Overview.....	1
1.2	Definition of Terms	2
1.3	Quick Start Instructions	2
1.3.1	Mounting.....	2
1.3.2	Serial Data Wires	3
1.3.2.1	RS232 serial	3
1.3.2.2	RS485 serial	3
1.3.3	Analog Output Signals.....	3
2.	General Description	4
2.1	Principles of Operation	4
2.2	Mechanical Dimensions	4
2.3	Installation	5
2.4	Laser Safety	5
2.5	Sensor Maintenance	6
2.6	Sensor Service	6
2.7	Sensor Specifications.....	6
3.	Installation and Checkout.....	7
3.1	Mounting.....	7
3.2	Cabling for sensor unit	7
3.2.1	Standalone Cabling	7
3.2.2	Serial Connection to a Host Computer	8
3.3	Power On	8
3.3.1	Serial Communications Check	8
3.3.2	Sensor Output Check	8
4.	Signal and Power Interface	9

4.1	Sensor Cable, Wire Colors and Functions	9
4.2	Power Supply (Red, Brown).....	9
4.3	RS232 / RS485 Serial Comm. (Green, Yellow).....	9
4.4	Analog Output (Blue, Grey).....	10
4.4.1	Current Loop	11
4.4.2	Voltage Output	11
4.5	Logic Outputs (Pink, Grey).....	12
5.	Serial Interface Operation	13
5.1	Communications Protocol and Syntax	13
5.1.1	Request.....	13
5.1.2	Message.....	13
5.1.3	Answer	14
5.1.4	Data Stream	15
6.	Analog Output Operation	16
6.1	Analog Output ON (01h).....	16
6.2	Analog Output Mode (02h bit R).....	16
6.3	Analog Output Working Range (0Ch, 0Dh, 0Eh, 0Fh)	16
7.	Logic Interface(s) Operation (02h bits M1 and M0).....	18
8.	Performance Optimization.....	19
8.1	Baud Rate (04h).....	19
8.2	Laser ON/OFF (00h)	19
8.3	Network Address (03h).....	19
8.4	Zero Point (17h, 18h)	19
8.5	Sampling Mode (02h bit S).....	20
8.6	Sampling Period (08h, 09h).....	20
8.6.1	Output Rate	21
8.7	Integration Time (0Ah, 0Bh)	21
8.8	Results Lock (10h)	21
8.9	Results Averaging Mode (02h bit M).....	22
8.9.1	Averaging Configuration (06h)	22
9.	Demo and Configuration Software	23
9.1	Program Setup	23
9.2	Connecting to the sensor (RS232/RS485).....	23
9.3	Sensor Operation	24
9.4	Display and Archiving of Data	25

9.5	Setting and Saving Sensor Parameters	26
9.5.1	Setting Parameters.....	26
9.5.2	Saving Parameters.....	27
9.5.3	Saving and Writing a Group of Parameters.....	27
9.6	Factory Reset.....	27
10.	Software Development Kit (SDK) Descriptions	28
10.1	Connection to COM-port (RF60x_OpenPort).....	28
10.2	Disconnect from COM-port (RF60x_ClosePort).....	28
10.3	Device identification (RF60x_HelloCmd).....	29
10.4	Reading of parameters (RF60x_ReadParameter)	30
10.5	Saving current parameters in FLASH-memory (RF60x_FlushToFlash).....	31
10.6	Restore default parameters from FLASH-memory (RF60x_RestoreFromFlash)	31
10.7	Latching of the current result (RF60x_LockResult).....	31
10.8	Get Measurement Result (RF60x_Measure)	32
10.9	Start Measurement Stream (RF60X_StartStream)	32
10.10	Stop Measurement stream (RF60x_StopStream).....	33
10.11	Get Measurement Results from Stream (RF60X_GetStreamMeasure).....	33
10.12	Transmission of user data (RF60x_CustomCmd).....	34
10.13	Functions for Operation of sensors connected to FTDI-based USB	34
10.14	Functions for operation of sensors with C Ethernet interface.....	34
10.14.1	Port open for receiving data through Ethernet.....	35
10.14.2	Close port for receiving data through Ethernet.....	35
10.14.3	Getting 168 measurement results from the stream	35
11.	Serial Command Quick Reference.....	38
12.	Examples of communication sessions	41
12.1	Request "Device identification".	41
12.2	Request: "Reading of parameter".....	42
12.3	Request: "Inquiring of result".....	42
12.4	Request "writing sampling regime (trigger sampling)".....	42
12.5	Request: "writing the divider ration"	42
13.	Accessories.....	44
13.1	Protective Enclosure	44
13.2	Spray Guard	45

1. Introduction

This section is a guide to getting started with the AR500 and this manual. The AR500 has a number of configurable parameters, but many applications can use the sensor in its default factory configuration. This manual contains information for a variety of AR500 sensor configurations that can be ordered from Acuity. Your specific AR500 model may not have all interfaces and functions described in this manual.

The recommended order for reading the manual is:

- General Overview – Gives a brief understanding of the sensor operation.
- Operating Guidelines – Provides a few important safety tips.
- Definition of Terms – An aid for proper communication.
- Quick Start Instructions – This should provide the information necessary to connect the sensor and verify its operation, either with a serial terminal program at 9600 baud, or by connecting the current loop or Alarm Output interface.
- General Description – Gives important laser, operation, mechanical, and mounting information.
- Installation and Checkout – Tailor the application. Use the other chapters for reference:
 - Signal and Power Interface – how to hook everything up
 - Serial Interface Operation – modes, formats, bias
 - Analog Output Operation – current loop, voltage, scaling
 - Alarm Output Operation – alarm settings
 - Performance Optimization – Sample Rate, Background Elimination, Exposure control
 - AR500 Command Set – explains all commands for customizing the application

1.1 General Overview

The AR500 is a triangulation sensor that measures distance using a laser beam, a camera, and a microprocessor. A variety of models are specified, each to allow a different measurement range, communications interface, laser power and environmental options. Models vary in range from 5 to 1000 mm. Interface options include RS232, RS485, 4-20mA, 0-10V and Ethernet. Sensors can be optionally ordered with an internal heater or air cooling jacket.

The accuracy is generally specified with a linearity of about +/- 0.15% of the range.

A variety of configuration settings can be selected via the serial port or Ethernet interface (option). The complete list of settings is found in the AR700 Command Set chapter and each setting is discussed in detail in a specific operation chapter.

The Sample Rate can be specified and the sensor has capability above 9400 samples per second. Several other configurable parameters enhance the

performance. Sampling may be turned on and off. It can even be triggered using an input signal wire or a serial command.

After making changes to the configuration, it may be viewed, saved in non-volatile memory, and restored. At power-on the sensor uses the most recently saved configuration settings.

Do not attempt to disassemble the sensor or loosen any screws. Improper disassembly will destroy the optical alignment of the sensor and necessitate factory repairs.

Do not operate the sensor in areas Where: the sensor case is exposed to direct sunlight for extended periods or Where: the air temperature is more than 60°C (140°F) or less than -10°C (-14°F). The optional internal heater or air-cooling jacket may extend these temperature limits

Don't allow fast temperature variations during sensor operation.

Avoid excessive vibration and shocks. The sensor contains securely mounted but precisely aligned optical components.

Do not operate the sensor if the lens is fogged or dirty.

Do not scratch the lenses on the front face of the sensor. Keep the lenses clean with expert optical procedures. The lenses are glass with an anti-reflection coating. Avoid the use of organic cleaning solvents.

Do not touch the lenses with bare fingers. The oils are very difficult to remove.

Operate only with DC supply voltages up to 36 volts.

1.2 Definition of Terms

Sensor – The complete AR500 measurement device.

Target – The object of measurement. The relative distance from the sensor to the target is measured by the sensor.

Laser, Laser beam – This bright light is emitted from the sensor, reflected from the target, and collected by the camera lens. For the AR500, it is visible (Red or Blue) radiation or in some specially-ordered configurations, infrared.

<Range> – The maximum relative distance measurable by the sensor.

1.3 Quick Start Instructions

This will get the sensor running in its factory default configuration.

Only one output type (Serial or Analog) is needed to indicate sensor operation.

1.3.1 Mounting

Quick suggestion: Lay the sensor on the floor or a table. It may need to be held in place with a clamp or a weight. Orient the laser so that the laser is not obstructed. Use a piece of paper such as a business card to

insert into the beam to use as a measurement target. The laser should be aimed at a target such that the distance from the reference point to the target can be measured.

Mount the sensor in such a way that the case is not twisted or warped. The AR500 can be screwed on using two fastening screws $\phi 3.6 / 06 \times 7$. The fastening screws are not included in the scope of delivery.

Attach the cable(s) 8-pin connector to the plug(s) on the rear of the sensor.

Connect the red (Supply +) and brown (Ground) wires of the sensor cable to a 9 to 36 volt DC power supply (or use the power supply if the sensor came with one).

1.3.2 Serial Data Wires

The serial connection is required to set up a unit for operation. If not using the Acuity Connectivity kit which includes a serial cable, the customer must make their own D-sub 9 serial connector

1.3.2.1 RS232 serial

Connect the RS232 wires to a 9 pin D-SUB male connector that can be plugged into a COM port of a PC (RS232): Gray (Ground) to pin 5, Green (Transmit) to pin 2, and Yellow (Receive) to pin 3. See section 3.2.2.

Use the Demo and Configuration Software (see Section 9) to connect to the sensor via the serial port and get distance measurements.

1.3.2.2 RS485 serial

Connect the RS485 wires to a RS485 adapter connected to a PC COM port Gray (Ground) to pin 5, Green (Data+) to pin 2, and Yellow (Data -) to pin 3. See section 3.2.2.

Follow the same instructions for RS232 serial above.

1.3.3 Analog Output Signals

Quick suggestion: connect a DVM (digital volt meter) to the wires: Gray to Common, Blue to mA input. Type QA. The output is a 4-20mA current loop from 0 to the maximum range. The meter should read near 4 mA when a target is placed in the laser beam near 0 mm and 20 mA near the end of the range

2. General Description

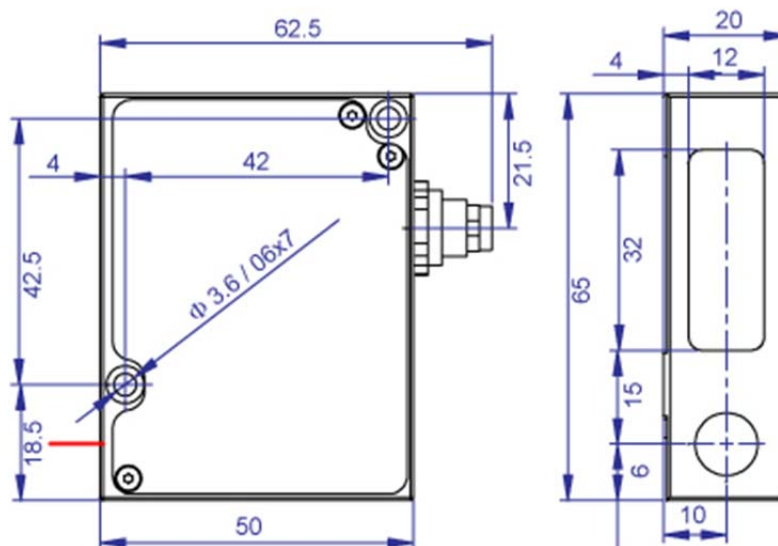
The AR500 is an ultra-compact laser diode-based distance measurement sensor with available ranges covering 5 to 1000 mm. Consult the AR500 data sheet for exact model range availabilities. The accuracy is generally specified with an absolute linearity of +/- 0.15% of the span and a resolution of 0.01% of the span. Linearity will vary depending on sample averaging, temperature, target stability and surface reflectivity of the target surface. The AR500 laser sensors can be ordered with a variety of red, blue and infrared laser diodes and a variety of data interfaces. The sensor can be triggered externally and also has logic outputs to trigger alarms, etc.

2.1 Principles of Operation

The AR500 uses laser triangulation principles to measure distance. The laser beam is projected from the housing's aperture and shines on a target surface, where it creates a small spot. From there, the laser light is scattered in all directions for diffuse surfaces (mirrors reflect the light specularly). A collection lens is located behind a window in the sensor. It collects a portion of the reflected light, which is focused on a CMOS detector array. The linear position of this reflected spot is converted to an electrical signal which is proportional to the target distance relative to the sensor. The position is processed and communicated via serial, analog, digital or Ethernet interfaces.

2.2 Mechanical Dimensions

The following diagram shows the mechanical dimensions for the AR500. The sensor unit has two $\phi 3.6 / 06 \times 7$ holes on the sides. The cable is for power and all communications (serial, analog, trigger, power, etc.). It is a 8-pin connector (Binder series 712). The outer case of the sensor is cast aluminum with anodization for corrosion resistance.

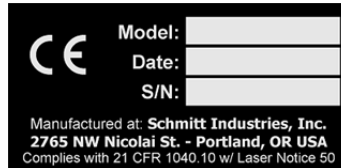


2.3 Installation

The AR500 sensor unit is typically installed by affixing the sensor to a machined bracket with bolts through the two mounting holes on the side of the sensor. Their location is shown in the mechanical drawing in section **Error! Reference source not found..** Note that the zero point is NOT the glass window, but instead the standoff distance minus one half the sensor's span. Most brackets will have adjustment capabilities so the AR500's laser can be aimed in X, Y and Z coordinates.

2.4 Laser Safety

Installers of laser sensors should follow precautions set forth by ANSI Z136.1 Standard for the Safe Use of Lasers or by their local safety oversight organization. The AR500 is a class 1 (eye safe) laser product as stipulated in IEC 60825-1/DIN EN 60825-1:2001-11 and a class 2, 3R or 3B product under FDA 21CFR.



Red Lasers	Blue Lasers	Infrared Laser

Figure 1 AR500 laser safety labels (all available diodes)

2.5 Sensor Maintenance

The AR500 sensor and module require little maintenance from the user. The sensor lens should be kept clean of dust buildup as a part of regular preventative maintenance. Use compressed air to blow dirt off the windows or use delicate tissue wipes. Do not use any organic cleaning solvents on the sensor. If your sensor does not function according to specifications, contact Schmitt Industries, Inc. Do not attempt to loosen any screws or open the sensor housing.

2.6 Sensor Service

The AR500 sensor is not user-serviceable. Refer all service questions to Schmitt Industries, Inc. Do not attempt to loosen any screws or open the sensor housing.

2.7 Sensor Specifications

Go to <http://www.acuitylaser.com/pdf/ar500-data-sheet.pdf>

3. Installation and Checkout

3.1 Mounting

Mount the sensor in such a way that the case not twisted or warped. Do not clamp or squeeze the sensor case excessively. If the case is distorted, the sensitivity and accuracy of the sensor may be affected.

3.2 Cabling for sensor unit

The AR500 sensor has a multipurpose cable with 8 conductors (included). Special-order Ethernet sensors will include a second cable with 4 conductors.

The standard cable is LiYCY (TP) a flexible, overall shielded, PVC twisted-pair data transmission cable for use in flexible and stationary applications under low mechanical stress with free movement without any tensile stress, loads or forced movements in dry, damp and wet conditions. The twisted pair construction reduces interference (crosstalk) within the cable while the tinned copper braid shield offers optimum protection from electrical and electromagnetic interference. Not suggested for outdoor use.

The standard cable length is 2 m in length and longer cable lengths are available. Connection and termination according to the instructions is essential for correct sensor operation. Read the wire descriptions in Section 4.1 for connection information.

Connect the cable's 8-pin connector (Binder series 712, female) to the plug (Binder series 712, male) on the back cover of the AR500 sensor. Be sure to tightly secure the connection for full protection from dust and water.

3.2.1 Standalone Cabling

To use the AR500 sensor unit without a serial connection to a host computer, the only connections necessary are the power and ground wires, the analog output wires, and optionally the alarm output wire connecting to your data display, recording, or control equipment. See Signal and Power Interface (section 4) for wire connections. In its default configuration, the AR500 should stream measurement distances on power-up.

In 4-20mA analog output mode, the best accuracy and linearity for the current loop is obtained with a 500-ohm load to current loop return at the measurement point. To reduce noise, it is recommended to install RC (resistor-capacitor) filter before the measuring instrument. The filter capacitor value is indicated (in section 4.4) for maximum sampling frequency of the sensor (9.4 kHz) and this value increases in proportion to the frequency reduction. An out-of-range current indicates a sensor measurement error.

The alarm output wire can be used to connect to control equipment.

3.2.2 Serial Connection to a Host Computer

The simplest way to connect the AR500 sensor to a PC computer for initial configuration or regular distance measuring is with the use of an Acuity Connectivity Kit. This is a sealed connection box which contains terminal blocks for each wire lead. It also has an AC power supply and a 2m RS232 serial cable for connection to a PC. Without the Acuity connectivity kit, the user must connect a DB9 plug to the cable using the directions below.

RS232: A 9-pin serial D-sub serial connector can be attached to the serial output wires to connect the AR500 directly to an IBM-PC compatible 9-pin serial port.

Wire Color	Function	Binder 712 pin #	DB9 pin #
Gray	Ground	5	5
Green	TxD / Data+	3	2
Yellow	RxD / Data -	4	3

RS485: If your PLC or control system does not support RS485 communications, an RS485 to RS232 adapter must be used to connect the AR500 to an IBM-PC compatible computer. See the wire functionality chart in section 4 for details.

For testing use the included Demo and Configuration software. Refer to section 9.

3.3 Power On

Connect a 15 volt (9 – 36 volts) power supply to the power and ground lines of the sensor cable. See Signal and Power Interface (section 4) for wire connections. Only the power and ground need be connected for operation in addition to the serial interface.

When power is applied the laser beam will be emitted from the round sensor window.

3.3.1 Serial Communications Check

If no information is received over the serial port, check the power supply and serial wire connections. The sensor may be in a configuration that prevents serial communication, such as being set at the wrong baud rate or is in a polling mode.

Use the Demo and Configuration Software to restore the unit to factory defaults. Otherwise, the **04h** command will restore factory defaults to the device.

3.3.2 Sensor Output Check

If the sensor output value is in error, check that the sensor and target are stationary and stable and that the laser beam is hitting the target.

The sensor may need to warm up for 5-10 minutes before reaching full accuracy. Leave it on for a few minutes and re-check the sensor accuracy.

4. Signal and Power Interface

4.1 Sensor Cable, Wire Colors and Functions

The AR500 sensor includes a multipurpose cable (sensor cable) with solder tail wires. Connection and termination according to the instructions is essential for correct sensor operation. Read the wire descriptions for connection information.

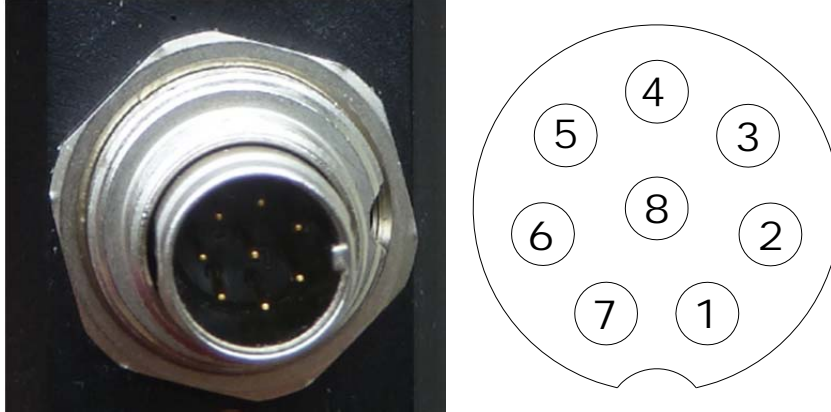


Figure 2 AR500 cable with 8 conductors (Binder 712, #09-0427-80-08)

The tables below shows the wiring on systems ordered without power supplies.

Wire	Pin	Function in All Modes
White	1	Trigger Input
Brown	2	Ground (Power)
Green	3	TxD (Data +)
Yellow	4	RxD (Data -)
Gray	5	Ground (Signal common)
Pink	6	Logic Output (programmable)
Blue	7	Analog output (current loop / voltage)
Red	8	Power, Voltage in

4.2 Power Supply (Red, Brown)

The Brown wire is the Power Supply Common return, also named Ground. It carries the return current for the power supply.

The Red wire is the Power Supply Input to the sensor. The sensor requires +9 - 36 VDC power and consumes 1.5 – 2 Watts of power (< 250mA draw) depending on the sensor's configuration.

Power supplies from 9 VDC to 36 VDC may be used. Higher voltages will result in excessive current drawn by the over-voltage protection circuitry and may cause permanent damage. Voltages less than 9 VDC may result in inaccurate measurement readings or non-functionality.

4.3 RS232 / RS485 Serial Comm. (Green, Yellow)

Your sensor is configured with either RS232 or RS485 communications

See Serial Interface Operation (section) for information on commands and data. The maximum baud rate is 460.8 KBaud for RS232 and 921.6 KBaud for RS485.

RS232: RS232 is normally used for shorter distances of communications and slower data rates. RS232 allows only one transmitter and one receiver per network. A standard 9-pin D-SUB RS232 serial female connector can be built to interface with an IBM or compatible computer using connection the pins below.

Color	Pin on DSUB 9 connector	Function
Green	2	Transmit data from sensor
Yellow	3	Receive data to sensor
Gray	3	Signal ground reference
N / C	1, 4, 6	DCD, DTE, DCE – These three signals can be tied together to satisfy some PC signal requirements for hardware handshake.
N / C	7, 8	CTS, RTS – These two signals can be tied together to satisfy some PC signal requirements for hardware handshake.

RS485: RS485 is normally used for longer distances of communications and faster data rates. Multiple devices can share one line because RS485 is multi-drop. Unless your computer or controller or PLC supports RS485 communications, it may be necessary to use a commercial RS485 to RS232 converter.

4.4 Analog Output (Blue, Grey)

Your sensor is configured with either 4-20mA current loop or 0-10V voltage analog output.

The Grey wire is the return signal for the Analog Output. It is connected to ground inside the sensor and should not be connected to ground outside the sensor. Inadvertently connecting it to ground may cause a reduction in accuracy of the analog output. The analog signal for distance is a 4-20 mA current loop or 0-10V signal.

In Current Loop / Voltage Blue wire delivers a current (or voltage for sensor ordered with Voltage output) proportional to the measured distance. The resolution is characterized by a 16-bit digital-to-analog converter.

4.4.1 Current Loop

The current loop connection scheme is shown in the Figure 3. The value of load resistor should not be higher than 500 Ohms. To reduce noise, it is recommended to install RC (resistor / capacitor) filter before the measuring instrument. The filter capacitor value is indicated for maximum sampling frequency of the sensor (9.4 kHz) and this value increases in proportion to the frequency reduction.

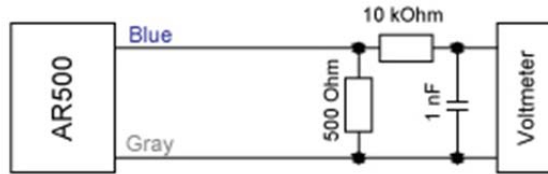


Figure 3 Wiring Diagram for Current Loop output

4.4.2 Voltage Output

The voltage output connection scheme is shown in the. To reduce noise, it is recommended to install an RC (resistor / capacitor) filter before the measuring instrument. The filter capacitor value is indicated for maximum sampling frequency of the sensor (9.4 kHz) and this value increases in proportion to the frequency reduction.

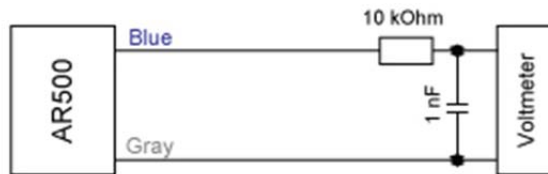


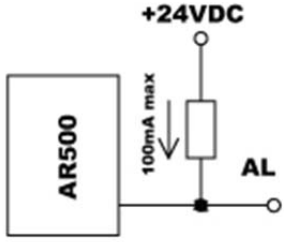
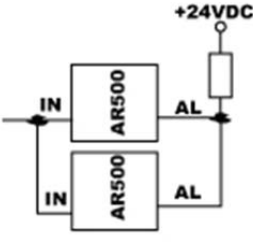
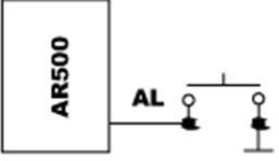
Figure 4 Wiring Diagram for Voltage output

4.5 Logic Outputs (Pink, Grey)

The Pink wire is a multi-purpose logic output / interface that can operate in four distinct modes. The functionality can be configured according to the instructions in section 7 . Functions include distance alarms, multi-sensor synchronization, setting of Zero Point via hardware control and control of the laser ON / OFF state.

See Alarm Output Operation (section 7) for operation options and details.

The Logic Output is an open collector NPN transistor switch to the Ground for Signal (Grey). When the Alarm Output is not active, its output will be high impedance and no current will flow through it. When the Alarm Output is active (On) it can source up to 100mA of current. The voltage on the Alarm wire must not exceed 24 VDC.

Distance Alarms	Multi-sensor Synchronization	Zero point set / Laser On / Off control
		
<p>Figure 5 Wiring Diagram for Distance Alarms</p>	<p>Figure 6 Wiring Diagram for Multi-sensor synchronization</p>	<p>Figure 7 Wiring Diagram for Zero point control and Laser On / Off</p>

5. Serial Interface Operation

This section refers to serial communication protocols for both the Sensor and Module versions of the AR500.

5.1 Communications Protocol and Syntax

Serial port communication is required to configure the AR500 for operation. The easiest way to communicate is by using a PC with an RS232 communication port and a terminal emulation program that uses hexadecimal binary format. The factory default baud rate is 9600 bits/second.

Through these serial interfaces measurement data can be obtained by two methods:

By single requests

- Automatic streaming data
- The serial data transmission byte has the following format:

1 start-bit	8 data bits	1 odd bit	1 stop-bit
-------------	-------------	-----------	------------

The communications protocol is formed by communication sessions (commands), which are only initiated by the 'master' (PC, controller). There are two kinds of sessions with such structures:

1. "request", ["message"] – ["answer"], *square brackets include optional elements*
2. "request" – "data stream" – ["request"].

5.1.1 Request

"Request" (INC) – is a two-byte message, which fully controls the communication session. The 'request' message is the only one of all messages in a session Where: the most significant bit is set at 0, therefore, it serves to synchronize the beginning of the session. In addition, it contains the device address (ADR), code of request (COD) and optional message [MSG].

"Request" format:

Byte 0		Byte 1				[Bytes 2...N]
INC0(7:0)		INC1(7:0)				MSG
0	ADR(6:0)	1	0	0	0	COD(3:0)

5.1.2 Message

"Message" is data burst that can be transmitted by a 'master' in the course of the session.

All messages with a "message" burst contain 1 in the most significant digit. Data in a message are transferred in nibbles (aka: "tetrads"). When a byte is transmitted, its lower tetrad goes first, and then follows the

higher tetrad. When multi-byte values are transferred, the transmission begins with the lower byte. The following is the format of two 'message' data bursts for transmission of byte:

DAT(7:0)									
Byte 0					Byte 1				
1	0	0	0	DAT(3:0)	1	0	0	0	DAT(7:4)

5.1.3 Answer

"Answer" is data burst that can be transmitted by a 'slave' in the course of the session.

All messages with an 'answer' burst contain 1 in the most significant digit. Data in a message are transferred in tetrads. When a byte is transmitted, the lower tetrad goes first, and then follows the higher tetrad. When multi-byte values are transferred, the transmission begins with lower byte.

When 'answer' is transmitted, the message contains:

- SB-bit, characterizes the updating of the result. If SB is equal to "1" this means that the sensor has updated the measurement result in the buffer, if SB is equal to "0" - then non-updated result has been transmitted (subject to sampling period). SB=0 when parameters transmit;
- Two additional bits of cyclic binary batch counter (CNT). Bit values in the batch counter are identical for all transmissions of one batch. The value of batch counter is incremented by the sending of each burst and is used for formation (assembly) of batches or bursts as well as for control of batch losses in receiving data streams.

The following is the format of two 'response' data bursts for transmission of a byte:

DAT(7:0)							
Byte 0				Byte 1			
1	SB	CNT(1:0)	DAT(3:0)	1	SB	CNT(1:0)	DAT(7:4)

All values are given in binary form. The Base distance and range are given in millimeters. The value of the result transmitted by a sensor (D) is so normalized that 4000h (16384) corresponds to a full range of the sensor (S in mm), therefore, the result in millimeters is obtained by the following formula:

$$X = D * S / 4000h \text{ (mm)} \quad (1).$$

On special request (05h), the current result can be latched in the output buffer where it will be stored unchanged up to the moment of arrival of request for data transfer. This request can be sent simultaneously to all sensors in the net in the broadcast mode in order to synchronize data pickup from all sensors.

When working with the parameters, it should be noted that when the power is OFF, the parameter values are stored in nonvolatile FLASH-memory of the sensor. When power is ON, the parameter values are read out to RAM of the sensor. To retain these changes for the next power-up state, a special command for saving current parameter values in the FLASH-memory (04h) must be run.

To issue requests with sizes greater than one byte, set the high byte 'session' then send the low byte request 'session'.

Refer to Section 11 for a list of all requests and parameters and Section 12 for Examples and Clarifications.

5.1.4 Data Stream

'Data stream' is an infinite sequence of data bursts or batches transmitted from 'slave' to 'master', which can be interrupted by a new request. In transmission of the 'data stream' one of the 'slaves' fully holds data transfer channel, therefore, when 'master' produces any new request sent to any address, the data streaming process is stopped. Also, there is a special request to stop data streaming.

6. Analog Output Operation

AR500 sensors can be ordered with either 4-20mA or 0–10 V analog outputs. The analog outputs use the same two wires. Please refer to Section 4.4 for connection details.

The analog output is updated with each sample measured. The analog output will deliver a current which increases linearly from 4 mA (or 0 volts) at the range beginning point to 20 mA (10 volts) at the range end point.

6.1 Analog Output ON (01h)

The analog output can be toggled ON / OFF. The factory default setting is ON (value=1).

The values for code parameter 01h are 1/0 for ON/OFF. If a sensor is not configured with an analog interface, the value will always be set to 0 despite any attempts to change it.

6.2 Analog Output Mode (02h bit R)

The analog output can be operated in two modes, "Window Mode" or "Full Mode". The factory default mode is "Window Mode" Where: the window size is set to the full span of the sensor (its measurement range).

"Window mode" - The value of bit R is 0. This is the default mode. The entire range of the analog output is scaled within the selected window. Outside the window, the analog output is "0".

"Full mode" - The value of bit R is 1. The entire range of the analog output is scaled within the selected window. Outside the selected window, the whole range of the analog output is automatically scaled onto the whole operating range of the sensor.

6.3 Analog Output Working Range (0Ch, 0Dh, 0Eh, 0Fh)

Resolution of the analog output can be increased by adjusting the window size and location within the measurement span. The analog signal will be scaled within this window only.

If the beginning of the range of the analog signal is set at a higher value than the end value of the range, this will change the direction of rise of the analog signal.

0Ch – Low byte for the BEGINNING of the analog output range. Default value is 0. Value range: 0 - 4000h. This code specifies a point within the sensor's span Where: the analog output has a minimum value.

0Dh – High byte for the BEGINNING of the analog output range. Default value is 0. Value range: 0 - 4000h. This code specifies a point within the sensor's span Where: the analog output has a minimum value.

0Eh – Low byte for the END of the analog output range. Default value is 4000h. Value range: 0 - 4000h. This code specifies a point within the sensor's span Where: the analog output has a maximum value.

OFh – High byte for the END of the analog output range. Default value is 4000h. Value range: 0 - 4000h. This code specifies a point within the sensor's span Where: the analog output has a maximum value.

7. Logic Interface(s) Operation (02h bits M1 and M0)

All AR500 sensors include a multi-purpose logic line. See the wiring description in section 4.5.

This line can work in one of the four **modes** defined by the configuration parameter of parameter 02h

Distance Alarm: This is the default logic output. Bits M1 and M0 have values 00. The output value is 0 when the measured distance is out of the range or selected window (see Analog Window in section 6.3).

Synchronization of two or more sensors: Bits M1 and M0 have values 01 to activate this mode. The synchronization mode makes it possible to synchronize measurement sample times of two and more sensors. Synchronization is achieved by tying together the trigger input lines of multiple sensors and also the logic output lines of the sensor. Synchronization is a valuable feature for thickness measurements using opposing laser sensors so that measurements are captured at exactly the same instant.

Hardware Zero Set (Tare function): Bits M1 and M0 have values 10 to activate this mode. When the Logic Interface wire is grounded, the currently measured distance is assigned as the zero point and subsequent measurements are now relative to this location within the span.

Laser Disable: Bits M1 and M0 have values 11 to activate this mode. When the Logic Interface wire is grounded, the laser is ON. When the switch is removed from ground, the laser is OFF. This controls only the laser diode and not power to the sensor itself.

8. Performance Optimization

This section describes how to configure the AR500 sensor for best use in your particular application.

8.1 Baud Rate (04h)

The AR500 automatically begins measuring and outputting distance measurements to the analog and serial lines when powered-up. The default baud rate is 9600 bit/s.

Users may select among several baud rates that will optimize the sensors' speed or accuracy performances over a serial connection.

Parameter 04h has values from 1 – 192 which specifies the data transfer rate in multiples of 2400. The default value is 4 (9600 bit/s).

8.2 Laser ON/OFF (00h)

This serial function toggles the laser ON or OFF. The default setting is ON. Note that there exists a hardware Logic line which can control the laser's state as well. See Laser Disable feature in section 7.

Parameter 00h has values from 1 or 0 for ON or OFF, respectively.

8.3 Network Address (03h)

This parameter defines the network address of the sensor equipped with RS485 interface. The factory default value is 1.

Network data communications protocols assume the presence of a 'master' in the network, which can be a computer or other information-gathering device, and from 1 to 127 'slaves' (AR500 Series sensors) which support the protocol.

Each 'slave' is assigned a unique network identification code – a device address. The address is used to form requests or inquiries all over the network. Each slave receives inquiries containing its unique address as well as '0' address which is broadcast-oriented and can be used for formation of generic commands, for example, for simultaneous latching of values of all sensors and for working with only one sensor (with both RS232 port and RS485 port).

Parameter 03h can be assigned values from 1 – 127. The factory default is 1.

8.4 Zero Point (17h, 18h)

This parameter allows the user to set a zero point within the sensor's measurement span. The factory default value for the zero point is the beginning of the measurement span. Note that users can also set the zero point through hardware controls. See Zero Point in section 7.

Parameter **17h** is the LOW byte of the Zero Point and has values 0 - 4000h with default value 0.

Parameter **18h** is the HIGH byte of the Zero Point and has values 0 - 4000h with default value 0.

8.5 Sampling Mode (02h bit S)

The AR500 has two sampling modes, Time or Trigger Sampling. The factory default setting is Time Sampling.

The sensor must be in Data Stream Mode.

Time Sampling: The value of bit S is 0. This is the factory default configuration. When selected, the sensor automatically transmits the measurement result via serial interface in accordance with selected sampling period.

Trigger Sampling: The value of bit S is 1. When selected, the sensor transmits the measurement result when an external synchronization input (see section 4.5) is switched. This takes the division factor set into account. See Sampling Period section 8.6.

8.6 Sampling Period (08h, 09h)

If the *Time* sampling mode is selected, the 'sampling period' parameter determines the time interval over which the sensor will automatically transmit the measurement result. The time interval value is set in increments of 0.01 ms. For example, for the parameter value equal to 100, data are transmitted through bit-serial interface with a period of $0.01 \times 100 = 1$ ms.

If the *Trigger* sampling mode is selected, the 'sampling period' parameter determines the division factor for the external synchronization input. For example, for the parameter value equal to 100, data are transmitted through bit-serial interface when each 100th synchronizing pulse arrives at trigger input of the sensor.

08h – Low byte for the sampling period. **09h** – High byte of sampling period. Default value is 500. Value ranges:

- In *Time* sampling mode: 10 to 65535. The time interval in increments of 0.01 ms with which sensor automatically communicates of results on streaming request (priority of sampling = 0)
- In *Trigger* sampling mode: 1 to 65535, divider ratio of trigger input with which sensor automatically communicates of result on streaming request (priority of sampling = 1)

Note 1: It should be noted that the 'sampling mode' and 'sampling period' parameters control only the transmission of data. The sensor operation algorithm is built so that measurements are taken at a maximum possible rate determined by the integration time period, the measurement results are sent to buffer and stored therein until a new result arrives. The above-mentioned parameters determine the method of the read-out of the result from the buffer.

Note 2: If the bit-serial interface is used to receive the result, the time required for data transmission at a selected data transmission rate should be

taken into account in the case Where: small sampling period intervals are used. If the transmission time exceeds the sampling period, it is *this* time that will determine the data transmission rate.

8.6.1 Output Rate

The sensor's output rate (OR) depends on the selected Baud rate (BR) of serial interface and is calculated by the following formula:

$$OR = 1 / (44/BR + 1 * 10^{-5}) \text{ Hz.}$$

For example, for BR=460800 b/s, Output Rate = 9.4 kHz

8.7 Integration Time (0Ah, 0Bh)

The intensity of the reflected laser radiation varies with target surface characteristics (color, texture, etc.) and ambient lighting conditions. The laser's output power and the integration time (like camera shutter time) of the CMOS detector array are automatically adjusted to achieve maximum measurement accuracy.

The Integration Time parameter specifies a maximum allowable integration time limit. If the radiation intensity received by the sensor is so small that no reasonable result is obtained within the time of integration equal to the limiting value, the sensor transmits a zero value.

0Ah – Low byte for maximum integration time.

0Bh – High byte for maximum integration time.

Default value is 3200 μ s. Value ranges:

2 to 65535. This value specifies the limiting integration time for the CMOS array in increments of 1mks.

The measurement frequency depends on the integration time of the receiving array. Maximum frequency (9.4 kHz) is achieved for the integration time $\leq 106 \mu$ s (minimum possible integration time is 0.1 μ s). As the integration time exceeds 106 μ s, the resulting update time increases proportionally.

Increasing this parameter may improve the sensor's ability to measure to low-reflecting targets at the cost of decreased speed and possible increased effects of ambient light on measurement accuracy. Decreasing the parameter's value increases measurement frequency, but may decrease measurement accuracy.

8.8 Results Lock (10h)

If the sensor does not "see" the target surface or if a new measurement cannot be received, a zero value is transferred. The Results Lock parameter sets a time limit during which the sensor transfers the last valid result instead of a zero value. The factory default value is 5 ms.

10h has value ranges from 0 to 255 (factory default value 1). It specifies the time interval in increments of 5 ms.

8.9 Results Averaging Mode (02h bit M)

This parameter defines one of the two methods of averaging of measurement results implemented directly in the sensor: Averaging over a Number of Results or Time Averaging. The factory default setting is for Averaging over a Number of Results.

Parameter 02h bit M can have two values, 0 or 1, representing one of two modes:

Averaging over a Number of Results – Bit M=0 (factory default). When selected, a sliding average is calculated and transmitted to the sensor's outputs

Time averaging – Bit M=1. When selected, the measurement results are averaged over a time interval

8.9.1 Averaging Configuration (06h)

Depending on the selected averaging mode (Number or Time), this parameter controls either the number of results to be averaged or the time for averaging. The factory default value is 1 result (no averaging).

Averaging can reduce noise or occasional spikes in the output of the sensor caused by inaccurate readings in dynamic applications.

Averaging over a number of results does not affect the data update in the sensor output buffer. However, in case of Time Averaging, data in the output buffer are updated at a rate equal to the averaging period.

06h is either the number of samples or time (in seconds) for averaging. Values for this parameter range from 1 to 127. The default value is 1.

9. Demo and Configuration Software

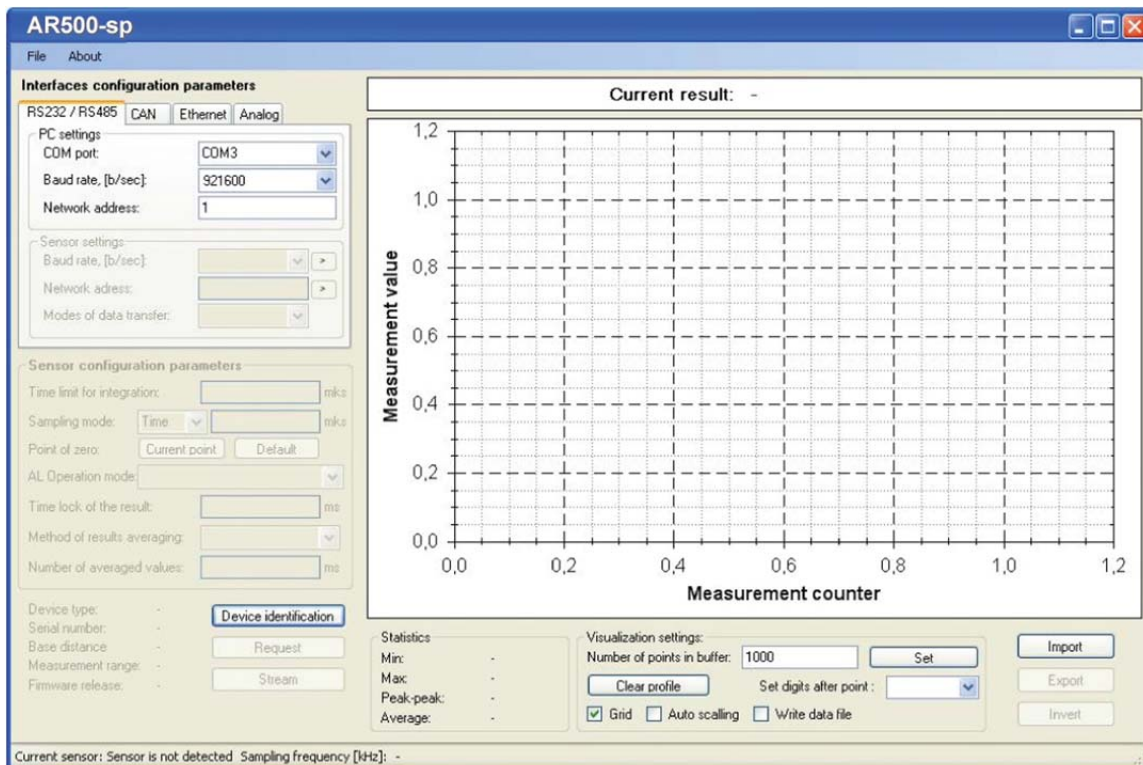
The AR500-SP software is intended for making simple serial (or Ethernet, if ordered) connections to the AR500 for demonstration purposes and configuration of the many sensor parameters. It is also possible to archive measurement data to a file using this software.

9.1 Program Setup

Start file AR500setup.exe and follow the instructions for the installation wizard.

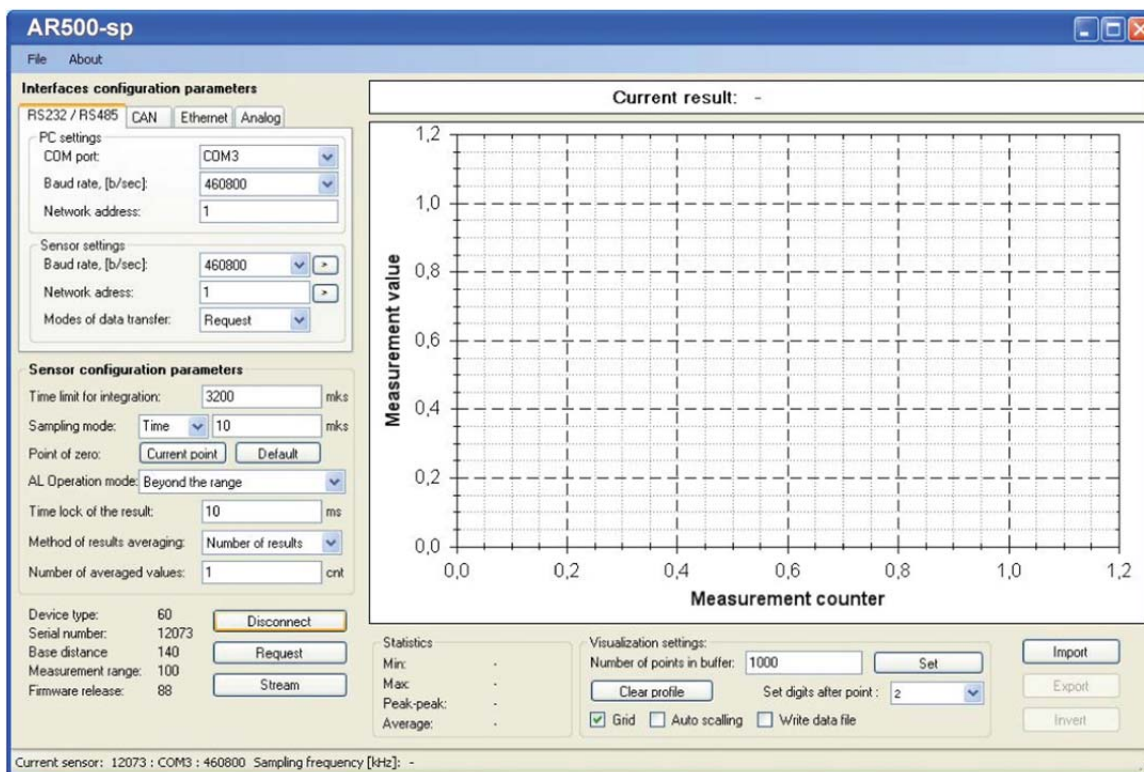
9.2 Connecting to the sensor (RS232/RS485)

Once the program is started, the pop-up window appears:

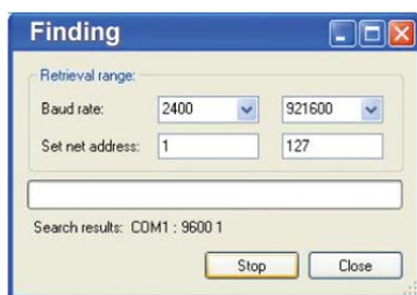
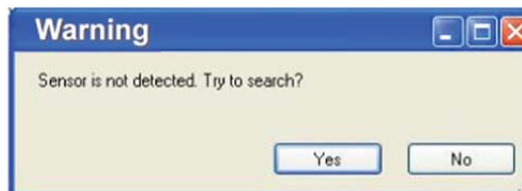


In the RS232/RS485 tab, select the COM-port Where: the sensor is connected. If using a USB to RS232 adapter, look at your computer's hardware profile to determine the assigned COM port. Select the Baud rate noting that the factory default is set to 9600 baud. Select the sensor network address, if necessary. Press the **Device identification** button.

If the selected parameters correspond to the parameters of the sensor interface, the program will identify the sensor, read and display its configuration parameters:



If connection a connection cannot be established, a prompt will appear asking to make an automatic search for the sensor.



To start search, press the **Yes** button and a configuration window will appear Where: it will be necessary to set the search range for the baud rate and the network addresses. When set, press the **Search** button.

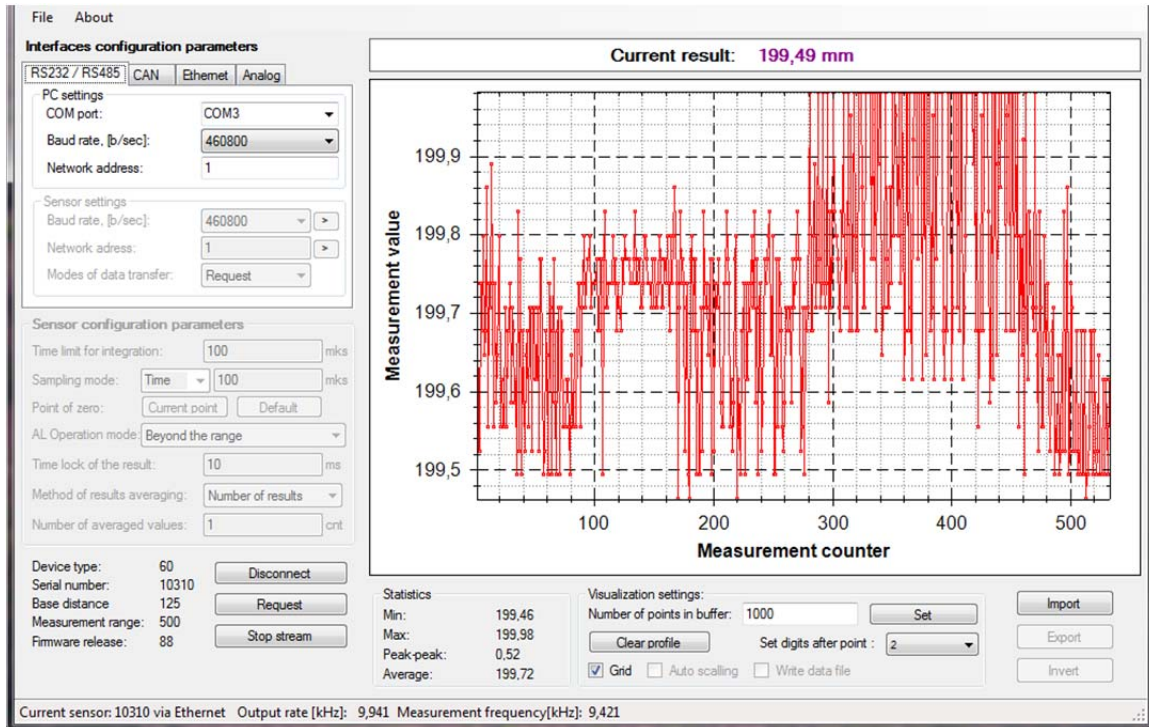
The program will perform automatic search of the sensor by searching over possible rates, network addresses and COM-ports of PC.

9.3 Sensor Operation

Once the sensor is successfully identified, check its operability by placing a target in the path of the laser and within the sensor's working range. Press the **Request** button to obtain a single measurement on the *Current Result* indicator at the top of the screen.

Note: The pressing of the **Request** button executes 06h request

Pressing the **Stream** button will switch the sensor to the data stream transmission mode (**07h** request code).



Move the target object and observe changes in the distance readings. The status line in the lower part of the window will show current data transmission and refreshing rates. Press the **Stop stream** button to stop data transmission (08h request code)

9.4 Display and Archiving of Data

Measurement results are displayed in digitally and in chart form while stored in the PC memory. The number of points displayed along the X axis can be set in the Number of points in buffer window. Y axis scaling can be set by the Auto scaling function.

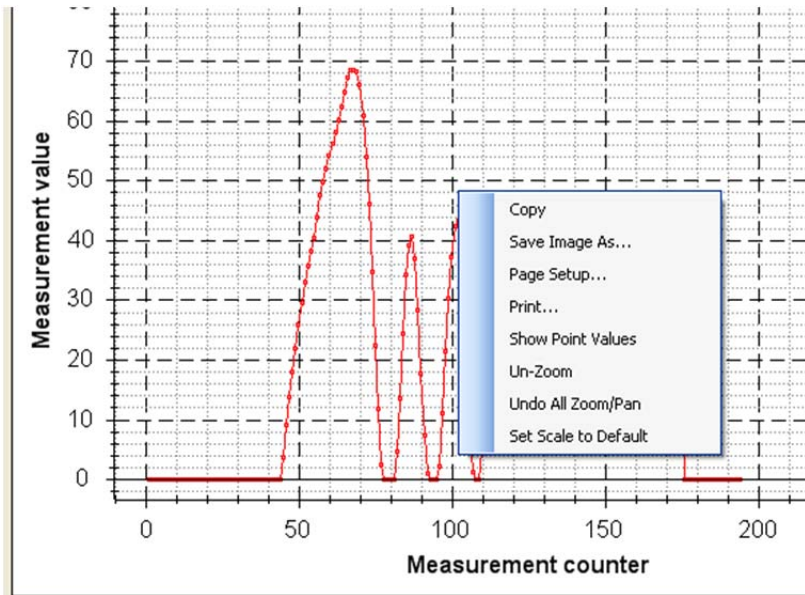
One can toggle ON/OFF the scaling grid by checking the **Grid** checkbox.

One can select the number of significant digits to the measurement result in the **Set digits after point** pull-down menu.

To archive received data to a file, select the **Write data file** checkbox.

Note: the number of points displayed on the chart depends on PC speed and becomes smaller in proportion to the data transmission rate. After the stream is stopped by using the **Stop Stream** button, the graph will display all received data.

To manipulate the chart image, right mouse click on the chart to reveal the corresponding menu:



Additionally, one can manipulate the chart image, press the mouse wheel for movement or rotating the mouse wheel for zooming capabilities.

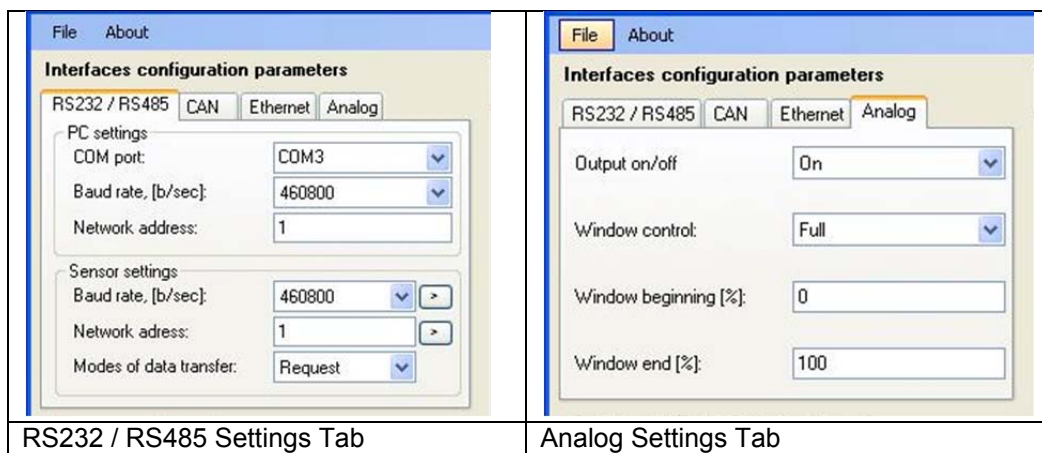
To save measurement data to a file, press the **Export** button. The program will offer saving of data in two possible formats: internal or Excel.

To scan or look at previously saved data, press the **Import** button and select the required file.


9.5 Setting and Saving Sensor Parameters

9.5.1 Setting Parameters

Configuring is only accomplished through RS232 or RS485 interfaces. Setting of parameters for all (optional) interfaces can be done using the respective tabs on the Interfaces configuration parameters panel. Setting of all configuration parameters of the sensor is possible with the help of the respective panel (Sensor configuration parameters):



9.5.2 Saving Parameters

After setting one or more parameters as required, users must *write* them into the sensor memory. Write parameter by clicking **File > Write parameters** or by clicking the  button.

Be sure to perform testing of the sensor operation with the new parameters. Once satisfied, it is necessary to store the new parameters in *nonvolatile memory* by clicking **File > Write to flash**. This procedure saves the parameters so that upon powering-up in the future will cause the sensor to default to your new configuration and not factory default configuration.

9.5.3 Saving and Writing a Group of Parameters

Parameters of the sensor can be saved to a file on your computer. This is done by selecting **File > Write parameters set** and saving the file in the window offered.

To open a group of parameters from a file, select **File > Sensor parameters sets** and select the file required. This function is convenient for writing identical parameters to several sensors. After loading the parameters follow the instructions above for saving the parameters to non-volatile memory.

9.6 Factory Reset

To restore the sensor's factory default parameters, use **File > Restore defaults**. This recalls the factory settings to the interface. It will also be necessary to save the parameters as described above.

10. Software Development Kit (SDK) Descriptions

The AR500 Laser sensor is supplied together with an SDK consisting of:

- dynamic library RF60x.dll
- file for static linking of DLL to project RF60x.lib,
- Definition file RF60x.h.

The SDK allows user to develop his / her own software applications without going into details of the sensor communications protocol.

10.1 Connection to COM-port (RF60x_OpenPort)

The function RF60x_OpenPort opens COM-port with specified symbolic name, fills in the pointer to the device descriptor and returns the operation result.

```
BOOL RF60x_OpenPort(  
    LPCSTR    IpPort_Name,  
    DWORD     dwSpeed,  
    HANDLE *  IpHandle  
);
```

Parameters:

IpPort_Name – name of COM-port (e.g., "COM1:"), full syntax for COM-port name specification see in MSDN, function CreateFile;

dwSpeed – operation speed through COM-port. The parameter is identical to field BaudRate in DCB structure described in MSDN;

IpHandle – pointer to the device descriptor;

Returned value:

If COM-port fails to be opened and adjusted, the function will return FALSE, otherwise if COM-port was opened and adjusted successfully the function will return TRUE. More detailed information about returned errors can be obtained using API function GetLastError described in MSDN.

10.2 Disconnect from COM-port (RF60x_ClosePort)

The function RF60x_ClosePort closes COM-port and returns the operation result:

```
BOOL RF60x_ClosePort(  
    HANDLE    hHandle  
);
```

Parameters:

hHandle – descriptor of the device obtained from function RF60x_OpenPort or CreateFile;

Returned value:

If COM-port fails to be closed, the function will return FALSE, otherwise if COM-port was closed successfully, the function will return TRUE.

10.3 Device identification (RF60x_HelloCmd)

The function RF60x_HelloCmd makes identification of RF60x according to net address and fills RF60xHELLOANSWER structure:

```
typedef struct _RF60x_HELLO_ANSWER_ {
    BYTE    bDeviceType;
    BYTE    bcDeviceModification;
    WORD    wDeviceSerial;
    WORD    wDeviceMaxDistance;
    WORD    wDeviceRange;
```

Where:

- bDeviceType – one byte value, which shows type of the device (for RF60x this value is equal 60) (type BYTE);
- bDeviceModification – one byte value, which shows firmware release (type BYTE);
- wDeviceSerial – two byte value, which contains serial number of the device (type WORD);
- wDeviceMaxDistance – two byte value, which contains the base distance of RF60X sensor (type WORD);
- wDeviceRange – two byte value, which contains the measurement range of RF60X sensor (type WORD).

The function RF60x_HelloCmd:

```
BOOL RF60x_HelloCmd (
    HANDLE          hCOM,
    BYTE            bAddress,
    LPRF60xHELLOANSWER lpHelloAnswer
);
```

Parameters:

- hCOM* – descriptor of the device obtained from function RF60x_OpenPort or CreateFile;
- bAddress* - device address;
- lpHelloAnswer* - pointer to the RF60xHELLOANSWER structure.

Returned value:

If the device does not respond to identification request, the function returns FALSE, otherwise the function returns TRUE and fills variable RF60xHELLOANSWER structure.

10.4 Reading of parameters (RF60x_ReadParameter)

The function RF60x_ReadParameter reads internal parameters of the RF603 sensor and returns the current value to the parameters address:

```

BOOL RF60x_ReadParameter (
    HANDLE          hCOM,
    BYTE           bAddress,
    WORD           wParameter,
    DWORD *       lpdwValue
);
    
```

Parameters:

hCOM – descriptor of the device obtained from function RF60x_OpenPort, or CreateFile;

bAddress - address of the device;

wParameter - number of parameter, see Table below,

Parameter	Description
RF60x_PARAMETER_POWER_STATE	Power status of sensor
RF60x_PARAMETER_ANALOG_OUT	Connection of analog output
RF60x_PARAMETER_SAMPLE_AND_SYNC	Control of sampling and synchronization
RF60x_PARAMETER_NETWORK_ADDRESS	Network address
RF60x_PARAMETER_BAUDRATE	Data transmission rate through serial port
RF60x_PARAMETER_AVERAGE_COUNT	Number of averaged values
RF60x_PARAMETER_SAMPLING_PERIOD	Sampling period
RF60x_PARAMETER_ACCUMULATION_TIME	Maximum accumulation time
RF60x_PARAMETER_BEGIN_ANALOG_RANGE	Beginning of analog output range
RF60x_PARAMETER_END_ANALOG_RANGE	End of analog output range
RF60x_PARAMETER_RESULT_DELAY_TIME	Result delay time
RF60x_PARAMETER_ZERO_POINT_VALUE	Zero point value
RF60x_PARAMETER_CAN_SPEED	Data transmission rate through CAN interface
RF60x_PARAMETER_CAN_STANDARD_ID	CAN standard identifier
RF60x_PARAMETER_CAN_EXTENDED_ID	Specifies CAN extended identifier
RF60x_PARAMETER_CAN_ID	CAN interface identifier

lpdwValue - pointer to WORD-type variable Where: current parameter value will be saved.

Returned value:

If the device does not respond to parameter reading request, the function returns FALSE, otherwise the function returns TRUE and fills variable *lpdwValue*.

10.5 Saving current parameters in FLASH-memory (RF60x_FlushToFlash)

Function RF60x_FlushToFlash saves all parameters in the FLASH-memory of the RF603 sensor:

```
BOOL RF60x_FlushToFlash(  
    HANDLE hCOM,  
    BYTE bAddress  
);
```

Parameters:

hCOM – descriptor of the device obtained from function RF60x_OpenPort or CreateFile;

bAddress - address of the device.

Returned value:

If the device does not respond to request to save all parameters in the FLASH-memory, the function returns FALSE, otherwise, if record confirm is obtained from the sensor, the function returns TRUE.

10.6 Restore default parameters from FLASH-memory (RF60x_RestoreFromFlash)

The function RF60x_RestoreFromFlash restores all parameter values in the FLASH by default:

```
BOOL RF60x_RestoreFromFlash(  
    HANDLE hCOM,  
    BYTE bAddress  
);
```

Parameters:

hCOM – descriptor of the device obtained from function RF60x_OpenPort or CreateFile;

bAddress - address of the device.

Returned value:

If the device does not respond to request to restore all parameters in the FLASH-memory, the function returns FALSE, otherwise, if restore confirm is obtained from the sensor, the function returns TRUE.

10.7 Latching of the current result (RF60x_LockResult)

The function RF60x_LockResult latches current measurement result till next calling of the function RF60x_LockResult:

```
BOOL RF60x_LockResult(  
    HANDLE hCOM,  
    BYTE bAddress  
);
```

Parameters:

hCOM – descriptor of the device obtained from function RF60x_OpenPort or CreateFile;

bAddress - address of the device.

Returned value:

If the device does not respond to result-latching request, the function returns FALSE, otherwise the function returns TRUE.

10.8 Get Measurement Result (RF60x_Measure)

The function RF60x_Measure reads current measurement value from the RF603 sensor. The result value (D) transmitted by the sensor is normalized in such a way as the value of 4000h (16384) corresponds to full range of the sensor (S в мм), the result in mm is obtained by the following formula:
 $X = D * S / 4000h$ (mm) :

```
BOOL RF60x_Measure(  
    HANDLE hCOM,  
    BYTE bAddress,  
    USHORT * lpusValue  
);
```

Parameters:

hCOM – descriptor of the device obtained from function RF60x_OpenPort or CreateFile;

bAddress - address of the device.

lpusValue - pointer to USHORT/WORD-type variable containing the result D.

Returned value:

If the device does not respond to result request, the function returns FALSE, otherwise, if the restore confirm is obtained from the sensor, the function returns TRUE.

10.9 Start Measurement Stream (RF60X_StartStream)

The function RF60x_StartStream switches RF603 sensor to the mode Where: continuous transmission of measurement results takes place:

```
BOOL RF60x_StartStream(  
    HANDLE hCOM,  
    BYTE bAddress  
);
```

Parameters:

hCOM – descriptor of the device obtained from function RF60x_OpenPort or CreateFile;

bAddress - address of the device.

Returned value:

If the device fails to be switched to continuous measurement transmission mode, the function returns FALSE, otherwise the function returns TRUE.

10.10 Stop Measurement stream (RF60x_StopStream)

The function RF60x_StopStream switches the sensor from continuous measurement transmission mode to the "request-response" mode:

```
BOOL RF60x_StopStream(  
    HANDLE hCOM,  
    BYTE bAddress  
);
```

Parameters:

hCOM – descriptor of the device obtained from function RF60x_OpenPort or CreateFile;

bAddress - address of the device.

Returned value:

If the device fails to be stopped in the continuous data transmission mode, the function returns FALSE, otherwise the function returns TRUE

10.11 Get Measurement Results from Stream (RF60X_GetStreamMeasure)

The function RF60x_GetStreamMeasure reads data from the COM-port input buffer which are received from RF603 sensor after successful execution of the RF60x_StartStream function. The data arrive in the buffer at a rate specified in the RF603 sensor parameters. Since depth of the input buffer is limited to 1024 bytes, it is preferable to read data with periodicity equal to that specified in the RF603 sensor parameters. The parameter *lpusValue* is identical to the parameter *lpusValue* in the RF60x_Measure function.

```
BOOL RF60x_GetStreamMeasure(  
    HANDLE hCOM,  
    USHORT * lpusValue  
);
```

Parameters:

hCOM – descriptor of the device obtained from function RF60x_OpenPort or CreateFile;

lpusValue - pointer to USHORT/WORD-type variable containing the result D.

Returned value:

If there are no data in the buffer, the function returns FALSE, otherwise the function returns TRUE and fills the value *lpusValue*.

10.12 Transmission of user data (RF60x_CustomCmd)

The function RF60x_CustomCmd is used for transmission and/or reception of data from in RF603 sensor.

```
BOOL RF60x_CustomCmd(  
    HANDLE          hCOM,  
    char *          pcInData,  
    DWORD           dwInSize,  
    char *          pcOutData,  
    DWORD *         pdwOutSize  
);
```

Parameters:

hCOM – descriptor of the device obtained from function RF60x_OpenPort or CreateFile;

pcInData - pointer to data array which will be transmitted to RF603 sensor. If no data need to be transmitted, *pcInData* must be NULL and *dwInSize* must be 0.

dwInSize - size of transmitted data. If no data need to be transmitted, this parameter must be 0.

pcOutData - pointer to data array Where: data received from RF603 will be saved. If no data need to be received, *pcOutData* must be NULL.

pdwOutSize - pointer to the variable containing size of data to be received. If no data need to be received, this parameter must be NULL. After successful receipt of data, the amount of read bytes will be recorded to the variable Where: this parameter points to.

Returned value:

If transmission or reception of bytes fails, the function returns FALSE, otherwise the function returns TRUE

10.13 Functions for Operation of sensors connected to FTDI-based USB

To work with FTDI-based USB devices, this library supports functions operating through D2XX library of FTDI. Performance of the functions is identical to that of the functions used for operation through serial port, the main difference being the presence of FTDI_ prefix in the function name, for example: "getting result" function for serial port is RF60x_Measure while for FTDI USB devices it is RF60x_FTDI_Measure.

10.14 Functions for operation of sensors with C Ethernet interface

These functions lets receive data from the sensor in the stream regime (UDP protocol is used) and get results synchronized by time or trigger

10.14.1 Port open for receiving data through Ethernet

The function `RF60x_Ethernet_OpenPort` opens net port, fills in the pointer to the device descriptor and returns the operation result:

```
BOOL RF60x_Ethernet_OpenPort (  
    HANDLE * lpHandle  
);
```

Parameters:

lpHandle - pointer to the device descriptor;

Returned value:

If net-port fails to be opened and/or adjusted, the function will return FALSE, otherwise if net-port was opened and adjusted successfully the function will return TRUE. More detailed information about returned errors can be obtained using API function `WSAGetLastError` described in MSDN.

10.14.2 Close port for receiving data through Ethernet

The function `RF60x_Ethernet_ClosePort` closes net port and returns the result of operation:

```
BOOL RF60x_Ethernet_ClosePort(  
    HANDLE hHandle  
);
```

Parameters:

hHandle - descriptor of the device obtained from function `RF60x_Ethernet_OpenPort`;

Returned value:

If net-port fails to be closed, the function will return FALSE, otherwise if net-port was closed successfully, the function will return TRUE

10.14.3 Getting 168 measurement results from the stream

The function `RF603_Ethernet_GetStreamMeasure` gets from the sensor 168 measurement results and fills in the next structure:

```
typedef struct _RF60x_UDP_PACKET_  
{  
    RF60xUDPVALUE rf60xVal Array[168];  
    WORD wDeviceSerial;  
    WORD wDeviceBaseDistance;  
    WORD wDeviceMeasureRange;  
    BYTE bPackCount;  
    DWORD dwReserved;  
    BYTE bPacketControl Summ;  
} RF60xUDPPACKET, *LPRF60xUDPPACKET;
```

Where:

rf60xValArray -168 structures (measurements)
 RF60xUDPVALUE, which contain measurement and status word;

wDeviceSerial -two byte value which contains serial number of the device (type WORD);

wDeviceBaseDistance -two byte value which contains base distance of the device (type WORD);

wDeviceMeasureRange -two byte value which contains measurement range of the device (type WORD);

bPackCount -one byte value which contains packet counter value (type BYTE);

dwReserved -four byte reserved value

bPacketControlSumm -one byte value which contains checksum value (type BYTE).

The structure RF60xUDPVALUE:

```
typedef struct _RF60x_UDP_VALUE_ {
    WORD          wMeasure;
    BYTE          bStatus;
} RF60xUDPVALUE, *LPRF60xUDPVALUE;
```

Where:

wMeasure -pointer to variable USHORT/WORD which contains the result D;

bStatusonebyte -value which contains measurement status (type BYTE);

As for function RF60x_Measure, the value of the result transmitted by a sensor (D) is so normalized that 4000h (16384) corresponds to a full range of the sensor (S in mm), therefore, the result in millimeters is obtained by the following formula: $X=D*S/4000h$ (mm).

The function RF603_Ethernet_GetStreamMeasure:

```
BOOL RF603_Ethernet_GetStreamMeasure (
    HANDLE          hHandle,
    LPRF60xUDPPACKET lprf60xUDPPacket
);
```

Parameters:

hHandle - device descriptor, obtained from function RF60x_Ethernet_OpenPort;

lprf60xUDPPacket - pointer to the structure RF60xUDPPACKET, which contains result D.

Returned value:

If there are no data in the buffer, the function returns FALSE, otherwise the function returns TRUE and fills the structure lprf60xUDPPacket.

11. Serial Command Quick Reference

Request code	Description	Message (size in bytes)	Answer (size in bytes)
1h	Device identification	—	–device type (1) –firmware release (1) –serial number (2) –base distance (2) –range (2)
2h	Reading of parameter	- code of parameter (1)	- value of parameter (1)
3h	Writing of parameter	- code of parameter (1) - value of parameter (1)	—
4h	Storing current parameters to FLASH-memory	- constant AAh(1)	- constant AAh(1)
4h	Recovery of parameter default values in FLASH-memory	- constant 69h (1)	- constant 69h (1)
5h	Latching of current result	—	—
6h	Inquiring of result	—	- result (2)
7h	Inquiring of a stream of results	—	- stream of results (2)
8h	Stop data streaming	—	—

Parameter Code	Description	Values
00h	Sensor ON	1 — laser is ON, measurements are taken (default state); 0 — laser is OFF, sensor in power save mode
01h	Analog output ON	1/0 — analog output is ON/OFF; if a sensor has no analog output, this bit will remain in 0 despite all attempts of writing 1 into it.
02h	Averaging, sampling and AL output control	x,x,M,C,M1,M0,R,S – control byte which determines averaging mode – bit M, CAN interface mode - bit C, logical output mode - bit M1, analog output mode - bit R, and sampling mode - bit S; bites x – do not use; bit M: 0 — quantity sampling mode (by default); 1 — time sampling mode bit C: 0 – request mode of CAN interface (by default); 1 – Synchronization mode of CAN interface. bit M1 and M0: 00 – out of the range indication (by default); 01 – mutual synchronization mode.

		10 – hardware zero set mode 11 – laser turn OFF/ON bit R: 0 – window mode (default); 1 – full range. bit S: 0 – time sampling (default) 1 – trigger sampling.
03h	Network address	1...127 (default — 1)
04h	Rate of data transfer through serial port	1...192, (default — 4) specifies data transfer rate in increments of 2400 baud; e.g., 4 means the rate of $4 \times 2400 = 9600$ baud. (NOTE: max baud rate = 460800)
05h	Reserved	
06h	Number of averaged values	1...128, (default — 1)
07h	Reserved	
08h	Lower byte of the sampling period	1) 10...65535, (default — 500) the time interval in increments of 0.01 ms with which sensor automatically communicates of results on streaming request (priority of sampling = 0); 2) 1...65535, (default — 500) divider ratio of trigger input with which sensor automatically communicates of result on streaming request (priority of sampling = 1)
09h	Higher byte of the sampling period	
0Ah	Lower byte of maximum integration time	2...65535, (default — 200) specifies the limiting time of integration by CMOS-array in increments of 1mks
0Bh	Higher byte of maximum integration time	
0Ch	Lower byte for the beginning of analog output range	0...4000h, (default — 0) specifies a point within the absolute range of transducer where the analog output has a minimum value
0Dh	Higher byte for the beginning of analog output range	
0Eh	Lower byte for the end of analog output range	0...4000h, (default — 4000h)) specifies a point within the absolute range of transducer where the analog output has a maximum value
0Fh	Higher byte for the end of analog output range	
10h	Time lock of result	0...255, specifies of time interval in increments of 5 mc
11...16h	Reserved	
17h	Lower zero point	0...4000h, (default — 0) specifies beginning of absolute coordinate system.
18h	Higher byte zero point	
19...1Ch	Reserved	
20h	Data transfer rate via CAN interface	10...200, (by default — 25) specifies data transmission rate in increments of 5 000 baud, for example, the value of 50 gives the rate of $50 \times 5\ 000 = 250\ 000$ baud.
22h	Low byte of standard identifier	0...7FFh, (by default — 7FFh) specifies standard CAN identifier
23h	High byte of standard identifier	
24h	0th byte of extended identifier	0...1FFFFFFFh, (by default — 1FFFFFFFh) specifies extended CAN identifier CAN
25h	1 st byte of standard identifier	

26h	2 nd byte of extended identifier	
27h	3 rd byte of standard identifier	
28h	CAN interface identifier	1 — extended identifier; 0 — standard identifier .
29h	CAN interface ON/OFF	1 — CAN interface ON; 0 — CAN interface OFF.
6Ch	0th byte of Destination IP Address	by default — FFFFFFFFh = 255.255.255.255
6Dh	1 st byte of Destination IP Address	
6Eh	2 nd byte of Destination IP Address	
6Fh	3 rd byte of Destination IP Address	
70h	0th byte of Gateway IP Address	by default — C0A80001h = 192.168.0.1
71h	1 st byte of Gateway IP Address	
72h	2 nd byte of Gateway IP Address	
73h	3 rd byte of Gateway IP Address	
74h	0th byte of Subnet Mask	by default — FFFFFFF0h = 255.255.255.0
75h	1 st byte of Subnet Mask	
76h	2 nd byte of Subnet Mask	
77h	3 rd byte of Subnet Mask	
78h	0th byte of Source IP Address	by default — C0A80003h = 192.168.0.3
79h	1 st byte of Source IP Address	
7Ah	2 nd byte of Source IP Address	
7Bh	3 rd byte of Source IP Address	
88h	ETHERNET interface ON/OFF	0 — ETHERNET interface OFF; 1 — ETHERNET interface ON (UDP protocol)

12. Examples of communication sessions

See section 5.1 for descriptions of protocols and section 11 for the codes themselves.

12.1 Request "Device identification".

Condition: device address — 1, request code — 1h, device type — 61, firmware release — 88 (58h), serial number — 0402 (0192h), base distance — 80mm (0050h), measurement range — 50mm (0032h), packet number — 1.

The request format:

Byte 0				Byte 1				[Bytes 2...N]
INC0(7:0)				INC1(7:0)				MSG
0	ADR(6:0)			1	0	0	0	COD(3:0)

Request from "Master"

Byte 0				Byte 1											
INC0(7:0)				INC1(7:0)											
0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1
01h				1h											

The following is the format of two 'answer' data bursts for transmission (transmission bytes) of one data byte DAT(7:0):

DAT(7:0)							
Byte 0				Byte 1			
1	0	CNT(1:0)	DAT(3:0)	1	0	CNT(1:0)	DAT(7:4)

Answer of "Slave": Note all response transmission bytes have the same high order nibble (in this case 9h) to show that these are all part of one message. They call this identifier the burst number (CNT). This nibble has values of 9h, Ah, Bh, or Ch that identify the transmission bytes as part of a message.

Device type:

DAT(7:0)													
Byte 0				Byte 1									
1	0	0	1	0	0	0	1	1	0	0	1	1	0
91h				96h									

Firmware release:

DAT(7:0)													
Byte 0				Byte 1									
1	0	0	1	1	0	0	0	1	0	0	1	0	1
98h				95h									

Serial Number:

DAT(7:0)															
Byte 0				Byte 1											
1	0	0	1	0	0	1	0	1	0	0	1	1	0	0	1
92h				99h											
DAT(7:0)															
Byte 0				Byte 1											
1	0	0	1	0	0	0	1	1	0	0	0	0	0		
91h				90h											

Base distance:

DAT(7:0)															
Byte 0								Byte 1							
1	0	0	1	0	0	0	0	1	0	0	1	0	1	0	1
90h								95h							
DAT(7:0)															
Byte 0								Byte 1							
1	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0
90h								90h							

Measurement range:

DAT(7:0)															
Byte 0								Byte 1							
1	0	0	1	0	0	1	0	1	0	0	1	0	0	1	1
92h								93h							
DAT(7:0)															
Byte 0								Byte 1							
1	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0
90h								90h							

Note: as burst number =1, then CNT=1

12.2 Request: "Reading of parameter".

Condition: device address – 1, request code – 02h, code of parameter – 05h, value of parameter – 04h, packet number – 2.

Request ("Master") – 01h;82h;

Message ("Master") – 85h, 80h;

Answer ("Slave") – A4h, A0h

12.3 Request: "Inquiring of result"

Condition: device address – 1, result – 02A5h, packet number – 3.

Request ("Master") – 01h;86h;

Answer ("Slave") – B5h, BAh, B2h, B0h

Measured distance (mm) (for example, range of the sensor= 50 mm):

$$X=677(02A5h)*50/16384 = 2.066 \text{ mm}$$

12.4 Request "writing sampling regime (trigger sampling)"

Condition: device address – 1, request code – 03h, code of parameter – 02h, value of parameter – 01h.

Request ("Master") – 01h, 83h;

Message ("Master") – 82h, 80h, 81h, 80h;

12.5 Request: "writing the divider ration"

Condition: divider ration – 1234=3039h, device address – 1, request code – 03h, code of parameter – 09h (first or higher byte), value of parameter – 30h

Request ("Master") – 01h, 83h

Message ("Master") – 89h, 80h, 80h, 83h

and for lower byte, code of parameter – 08h, value of parameter – 39h

Request ("Master") – 01h, 83h

Message ("Master") – 88h, 80h, 89h, 83h

13. Accessories

13.1 Protective Enclosure

Acuity offers an air-cooled, protective housing for the AR500. It is designed to be used in harsher environments of higher ambient temperature or airborne particles. The temperature of compressed air at the sensor input must be <math><25^{\circ}\text{C}</math> and it must be filtered of oil / dirt and passed through a desiccator to remove moisture. The maximum allowable ambient operating temperature for the sensor in the protective enclosure is

The sensor is factory calibrated directly in the housing with its engineer glass windows. If the sensor is removed from the housing, it's linearity will be affected adversely.

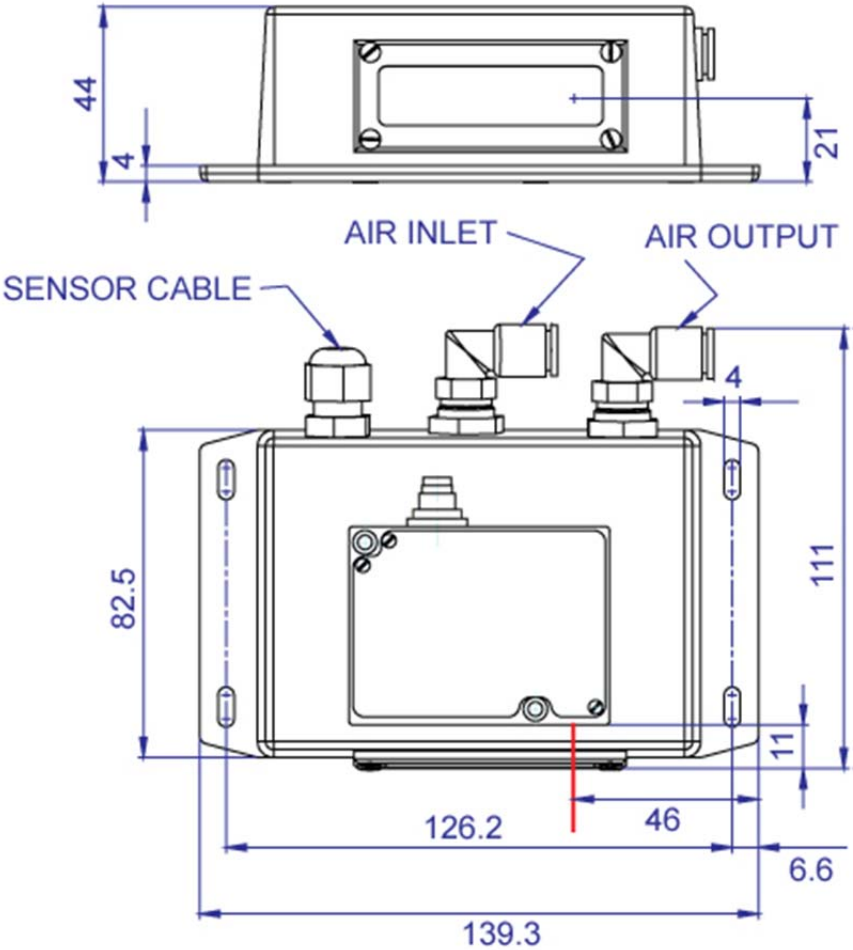


Figure 8 AR500 Protective Enclosure Schematic

13.2 Spray Guard

The optional spray guard is designed to minimize the amount of dirt or liquid spray from reaching the optical windows of the sensor.

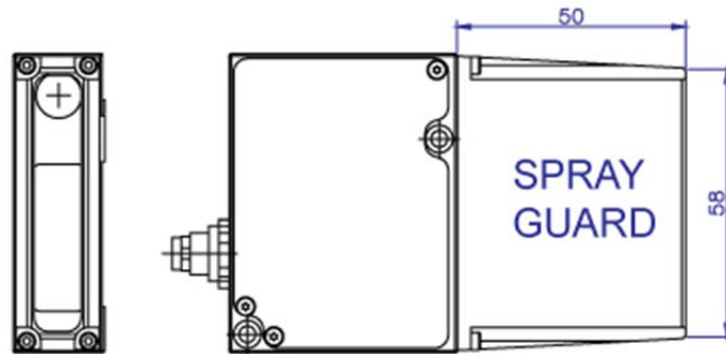


Figure 9 Spray Guard