TECHNICAL INFORMATION

Telegram Listing RMS2000

Radar sensor RMS2000

SICK
Sensor Intelligence.

Described product

RMS2000

Manufacturer

SICK AG
Erwin-Sick-Str. 1
79183 Waldkirch

Germany

Legal information

Original document

This document is an original document of SICK AG.

Contents

1 About this document

Please read this chapter carefully before beginning to use the telegram listing.

The document shows how to send telegrams via a terminal program using the SICK protocol CoLa A (ASCII and hexadecimal values, with TCP port 2111 or 2112) or CoLa B (binary/hexadecimal values,) to RMS2000. This comprises the query of the current device state or certain parameter values, how to modify parameter values and the way in which the device confirms or responds to commands/telegrams.

The devices generally support automatic IP address discovery. Default IP address is:

- RMS2000: 192.168.0.1

Subnet mask is 255.255.255.0.

IP ports:

- 2111: CoLa A (fixed)
- 2112: CoLa A (can be switched to CoLa B)

Most parameter changes also require certain user levels. Additionally, commands may change during the product lifecycle and development process with a new firmware.

This telegram listing is based on the following firmware statuses (or newer):

- RMS2000(V1.6.0)

If commands do not seem to work, please verify that your device version supports this functionality, that the minimum required user level has been selected and check on updates of this documentation.

2 Communication format

2.1 Binary telegram (CoLa B)

The binary telegram is the basic protocol of the scanner (CoLa B). All values are in hexadecimal code and grouped into pairs of two digits (= 1 byte). The string consists of four parts: header, data length, data and checksum (CS).

The header indicates with 4 × STX (02 02 02 02) the start of the telegram.

The data length defines the size of the data part (command part) by indicating the number of digit pairs in the third part. The size of the data length itself is 4 bytes, which means that the data part might have a maximum of 16^8 = 4,294,967,295 digit pairs.

The data part comprises the actual command with letters and characters converted to Hex (according to the ASCII chart) and the parameters of either decimal numbers converted to Hex or fixed Hex values with a specific, intrinsic meaning (no conversion). There is always a blank (20) between the command and the parameters, but not between the different parameter values.

The checksum finally serves to verify that the telegram has been transferred correctly. The length of the checksum is 1 byte, CRC8. It is calculated with XOR.

Example: Binary telegram

02 02 02 02	00 00 00 17	73 4D 4E 20 53 65 74 41 63 63 65 73 73 4D 6F 64 65 20 03 F4 72 47 44	B3
Header	Length	Data	CS

Table 1: Example: Binary telegram

This is an example telegram for setting the user level “Authorized Client”:

- Header = 02 02 02 02
- Length = 23 digit pairs (17h)
- Data:
  - 73 4D 4E 20 = sMN = start of Sopas command (and blank)
  - 53 65 74 41 63 63 65 73 73 4D 6F 64 65 20 = Set Access Mode = the actual command for setting the user level (and blank)
  - 03 = fixed Hex value meaning user level “Authorized Client”
  - F4 72 47 44 = fixed Hex value, serving as password for the selected user level “Authorized Client”
- Checksum = B3 from XOR calculation

2.2 ASCII telegram (CoLa A)

The ASCII telegram is an alternative to the binary telegram. Due to the variable string length of ASCII telegrams, the Binary telegram is recommended when using scanners with a PLC.

The ASCII telegram has the advantage that commands can be written in plaintext. The string consists only of two parts: the framing and the data part.

The framing indicates with <STX> and <ETX> the start and stop of each telegram.

The data part comprises the actual command with letters and characters (plaintext), parameter values either in decimal (special indicator required) or in hexadecimal (example: a frequency of 25 Hz = +2500 (decimal) = 09C4 (Hex)) and fixed hexadecimal values with a specific, intrinsic meaning. As leading zeros are being deleted, there is always a blank required between all command parts and parameter parts.

NOTE

The device will comfirm parameter values always in hexadecimal code, regardless of the code sent.

As further alternative within CoLa A, depending on the preferences of the user, all values can be written directly in Hex. This means however a 1:1 conversion of all letters and characters including numbers and fixed hexadecimal values via the ASCII chart.

Example: ASCII telegram

ASCII	<STX>	sMN{SPC}SetAccessMode{SPC}03{SPC}F4724744	<ETX>
Hex	02	73 4D 4E 20 53 65 74 41 63 63 65 73 73 4D 6F 64 65 20 30 33 20 46 34 37 32 34 37 34 34	03
	Start	Data	Stop

Table 2: Example: ASCII telegram

This is again an example telegram for setting the user level “Authorized Client”. As only fixed hexadecimal parameter values are needed, the option to use parameter values in decimal code with special indicator cannot be applied here:

- Framing = <STX> = telegram start = 02 (Hex)
- Data:
  - sMN = start of Sopas command (and blank) = 73 4D 4E 20 (Hex)
  - SetAccessMode = the actual command for setting the user level (and blank) = 53 65 74 41 63 63 65 73 73 4D 6F 64 65 20 (Hex)
  - 03 = fixed Hex value meaning user level “Authorized Client” (and blank) = 30 33 20 (Hex)
  - F4 72 47 44 = fixed Hex value, serving as password for the selected user level “Authorized Client” = 46 34 37 32 34 37 34 34 (Hex)
- Framing = <ETX> = telegram stop = 03 (Hex)

2.3 Variable types

Variable type	Length (byte)	Value range	Sign
Bool_1	1	0 or 1	No
Uint_8	1	0 … 255	No
Int_8	1	-128 … +127	Yes
Uint_16	2	0 … 65,535	No
Int_16	2	-32,768 … +32,767	Yes
Uint_32	4	0 … 4,294,967,295	No
Int_32	4	-2,147,483,648 … +2,147,483,647	Yes
Enum_8	1	Certain values defined in a list of Choices (0 … 255)	No
Enum_16	2	Certain values defined in a list of Choices (0 … 65535)	No
String	Context-dependent	Strings are not terminated in zeroes	
Real		Float nach IEEE754 (see www.h-schmidt.net/FloatConverter/IEEE754de.html)	

Data length is always given in Bytes!

2.4 Command basics

Description	Value ASCII	Value Hex	Value Binary
Start of text	<STX>	02	02 02 02 02 + given length
End of text	<ETX>	03	Calculated checksum
Read	sRN	73 52 4E	
Write	sWN	73 57 4E	
Method	sMN	73 4D 4E	
Event	sEN	73 45 4E	
Answer	sRA	73 52 41	
	sWA	73 57 41	
	sAN	73 41 4E	
	sEA	73 45 41	
	sSN	73 53 4E	
Space	{SPC}	20	20

If values are divided into two parts (e.g. measurement data), they are documented according to LSB 0 (e.g. 00 07), output however is according to MSB (e.g. 07 00).

2.5 Log in: Required user level

Task	Required user level
Change sensor parameters	Authorized Client
Requests or queries (e.g. for measurement data or device state)	None
Manage password	Service

3 Workflows

3.1 Parameterize the devices

1 Log in: sMN SetAccessMode (see 4.1.1, page10)
2 Configure output data contents: sWN TransmitTargets(see4.2.1, page18) and/or sWN TransmitObjects(see4.2.2, page19)
3 Store parameters: sMN mEEwriteall (see 4.1.5, page 15)
4 Log out: sMN Run (see 4.1.6, page 16)
5 Request scan:
sEN LMSradardata (see 4.2.3, page 20)

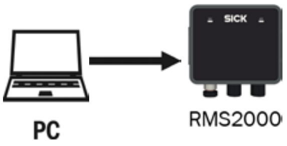
(Device output ...)

More detailed command descriptions can be found in the following part of this document.

4 Telegrams

4.1 Basic Settings

4.1.1 Set Access Mode (Log in)



Telegram structure: sMN SetAccessMode						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Method	String	3	All	sMN	73 4D 4E
Command	User level	String	13	All	SetAccessMode	53 65 74 41 63 63 65 73 73 4D 6F 64 65
User level	Select user level	Int_8	1	All	Maintenance: 02 Authorized client: 03 Service: 04	Maintenance: 02 Authorized client: 03 Service: 04
Password	Hash value for the selected user level	Uint_32	4	All	Maintenance: B21ACE26 Authorized client: F4724744 Service: 81BE23AA	Maintenance: B2 1A CE 26 Authorized client: F4 72 47 44 Service: 81 BE 23 AA

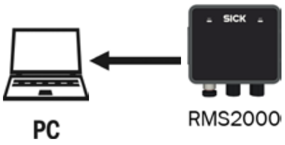
Table 3: Telegram structure: sMN SetAccessMode

Example: sMN SetAccessMode

Log in as “Authorized client” with password “F4724744”.

CoLa A	ASCII	<STX>sMN{SPC}SetAccessMode{SPC}03{SPC}F4724744<ETX>
	Hex	02 73 4D 4E 20 53 65 74 41 63 63 65 73 73 4D 6F 64 65 20 30 33 20 46 34 37 32 34 37 34 34 03
CoLa B	Binary	02 02 02 02 00 00 00 17 73 4D 4E 20 53 65 74 41 63 63 65 73 73 4D 6F 64 65 20 03 F4 72 47 44 B3

Table 4: Example: sMN SetAccessMode



Telegram structure: sAN SetAccessMode						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Answer	String	3	All	sAN	73 41 4E
Command	User level	String	13	All	SetAccessMode	53 65 74 41 63 63 65 73 73 4D 6F 64 65
Change user level	Changed level	Bool_1	1	All	Error: 0 Success: 1	Error: 00 Success: 01

Table 5: Telegram structure: sAN SetAccessMode

Example for RMS2000: sAN SetAccessMode

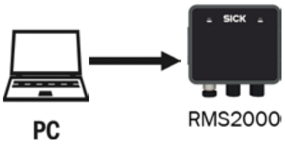
CoLa A	ASCII	<STX>sAN{SPC}SetAccessMode{SPC}1<ETX>
	Hex	02 73 41 4E 20 53 65 74 41 63 63 65 73 73 4D 6F 64 65 20 31 03
CoLa B	Binary	02 02 02 02 00 00 00 13 73 41 4E 20 53 65 74 41 63 63 65 73 73 4D 6F 64 65 20 01 38

Table 6: Example: sAN SetAccessMode

4.1.2 Load factory defaults

NOTE

The Factory-Reset (Load factory defaults) deletes the entire parametrization of the device.All parameters, settings and system applications will be set to default.



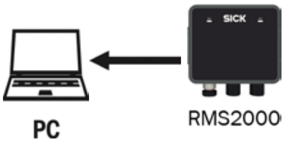
Telegram structure: sMN mSCloadfacdef (Authorized client)						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Method	String	3	All	sMN	Not possible
Command	Load factory defaults	String	13	All	mSCloadfacdef	Not possible

Table 7: Telegram structure: sMN mSCloadfacdef

Example: sMN mSCloadfacdef

CoLa A	ASCII	<STX>sMN{SPC}mSCloadfacdef<ETX>
	Hex	02 73 4D 4E 20 6D 53 43 6C 6F 61 64 66 61 63 64 65 66 03
CoLa B	Binary	Not possible

Table 8: Example: sMN mSCloadfacdef



Telegram structure: sAN mSCloadfacdef						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Answer	String	3	All	sAN	Not possible
Command	Load factory defaults	String	13	All	mSCloadfacdef	Not possible

Table 9: Telegram structure: sAN mSCloadfacdef

Example: sAN mSCloadfacdef

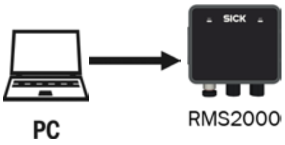
CoLa A	ASCII	<STX>sAN{SPC}mSCloadfacdef<ETX>
	Hex	02 73 41 4E 20 6D 53 43 6C 6F 61 64 66 61 63 64 65 66 03
CoLa B	Binary	Not possible

Table 10: Example: sAN mSCloadfacdef

4.1.3 Load application defaults

NOTE

The Application-Reset (Load application defaults) deletes all the user parametrization. Other parameters like Interface settings remain unaffected.



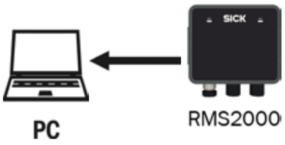
Telegram structure: sMN mSCloadappdef (Authorized client)						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Method	String	3	All	sMN	Not possible
Command	Load application defaults	String	13	All	mSCloadappdef	Not possible

Table 11: Telegram structure: sMN mSCloadappdef

Example: sMN mSCloadappdef

CoLa A	ASCII	<STX>sMN{SPC}mSCloadappdef<ETX>
	Hex	02 73 4D 4E 20 6D 53 43 6C 6F 61 64 61 70 70 64 65 66 03
CoLa B	Binary	Not possible

Table 12: Example: sMN mSCloadappdef



Telegram structure: sAN mSCloadappdef						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Answer	String	3	All	sAN	Not possible
Command	Load application defaults	String	13	All	mSCloadappdef	Not possible

Table 13: Telegram structure: sAN mSCloadappdef

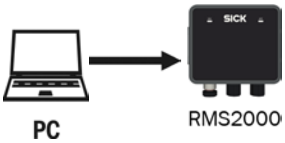
Example: sAN mSCloadappdef

CoLa A	ASCII	<STX>sAN{SPC}mSCloadappdef<ETX>
	Hex	02 73 41 4E 20 6D 53 43 6C 6F 61 64 61 70 70 64 65 66 03
CoLa B	Binary	Not possible

Table 14: Example: sAN mSCloadappdef

4.1.4 Reboot device

This command includes saving all parameters.



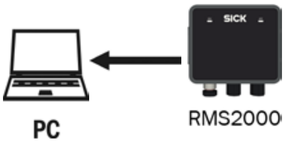
Telegram structure: sMN mSCreboot (Authorized client)						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Method	String	3	All	sMN	73 4D 4E
Command	Reboot device	String	9	All	mSCreboot	6D 53 43 72 65 62 6F 6F 74

Table 15: Telegram structure: sMN mSCreboot

Example: sMN mSCreboot

CoLa A	ASCII	<STX>sMN{SPC}mSCreboot<ETX>
	Hex	02 73 4D 4E 20 6D 53 43 72 65 62 6F 6F 74 03
CoLa B	Binary	02 02 02 02 00 00 00 0D 73 4D 4E 20 6D 53 43 72 65 62 6F 6F 74 2C

Table 16: Example: sMN mSCreboot



Telegram structure: sAN mSCreboot						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Answer	String	3	All	sAN	73 41 4E
Command	Reboot device	String	9	All	mSCreboot	6D 53 43 72 65 62 6F 6F 74

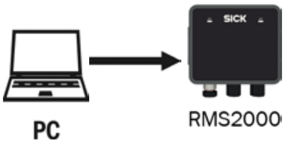
Table 17: Telegram structure: sAN mSCreboot

Example: sAN mSCreboot

CoLa A	ASCII	<STX>sAN{SPC}mSCreboot<ETX>
	Hex	02 73 41 4E 20 6D 53 43 72 65 62 6F 6F 74 03
CoLa B	Binary	02 02 02 02 00 00 00 0E 73 41 4E 20 6D 53 43 72 65 62 6F 6F 74 00

Table 18: Example: sAN mSCreboot

4.1.5 Save parameters permanently



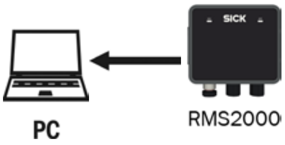
Telegram structure: sMN mEEwriteall (Authorized client)						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Method	String	3	All	sMN	73 4D 4E
Command	Store parameters permanently	String	11	All	mEEwriteall	6D 45 45 77 72 69 74 65 61 6C 6C

Table 19: Telegram structure: sMN mEEwriteall

Example: sMN mEEwriteall

CoLa A	ASCII	<STX>sMN{SPC}mEEwriteall<ETX>
	Hex	02 73 4D 4E 20 6D 45 45 77 72 69 74 65 61 6C 6C 03
CoLa B	Binary	02 02 02 02 00 00 00 0F 73 4D 4E 20 6D 45 45 77 72 69 74 65 61 6C 6C 21

Table 20: Example: sMN mEEwriteall



Telegram structure: sAN mEEwriteall						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Answer	String	3	All	sAN	73 41 4E
Command	Store parameters permanently	String	11	All	mEEwriteall	6D 45 45 77 72 69 74 65 61 6C 6C
Status code	Accepted when value is 1	Bool_1	1	All	Error: 0 Success: 1	Error: 00 Success: 01

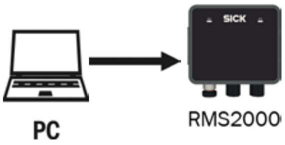
Table 21: Telegram structure: sAN mEEwriteall

Example: sAN mEEwriteall

CoLa A	ASCII	<STX>sAN{SPC}mEEwriteall{SPC}1<ETX>
	Hex	02 73 41 4E 20 6D 45 45 77 72 69 74 65 61 6C 6C 20 31 03
CoLa B	Binary	02 02 02 02 00 00 00 11 73 41 4E 20 6D 45 45 77 72 69 74 65 61 6C 6C 20 01 0C

Table 22: Example: sAN mEEwriteall

4.1.6 Set to run (Log out)



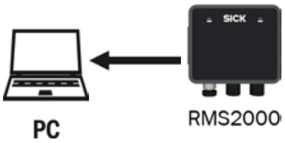
Telegram structure: sMN Run						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Method	String	3	All	sMN	73 4D 4E
Command	Start the device	String	3	All	Run	52 75 6E

Table 23: Telegram structure: sMN Run

Example: sMN Run

CoLa A	ASCII	<STX>sMN{SPC}Run<ETX>
	Hex	02 73 4D 4E 20 52 75 6E 03
CoLa B	Binary	02 02 02 02 00 00 00 07 73 4D 4E 20 52 75 6E 19

Table 24: Example: sMN Run



Telegram structure: sAN Run						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Answer	String	3	All	sAN	73 41 4E
Command	Start the device	String	3	All	Run	52 75 6E
Status code	Accepted when value is 1	Bool_1	1	All	Error: 0 Success: 1	Error: 00 Success: 01

Table 25: Telegram structure: sAN Run

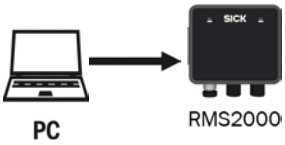
Example: sAN Run

CoLa A	ASCII	<STX>sAN{SPC}Run{SPC}1<ETX>
	Hex	02 73 41 4E 20 52 75 6E 20 31 03
CoLa B	Binary	02 02 02 02 00 00 00 09 73 41 4E 20 52 75 6E 20 01 34

Table 26: Example: sAN Run

4.2 Measurement output telegram

4.2.1 Enable/disable target data output in the data content



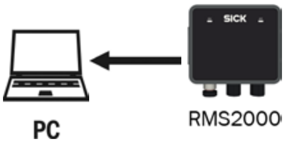
Telegram structure: sWN TransmitTargets						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Event	String	3	All	sWN	73 57 4E
Command	Data telegram	String	11	All	TransmitTargets	54 72 61 6E 73 6D 69 74 54 61 72 67 65 74 73
Measurement	Start/stop	Enum_8	1	All	Stop: 0 Start: 1	Stop: 00 Start: 01

Table 27: Telegram structure: sWN TransmitTargets

Example: sWN TransmitTargets

CoLa A	ASCII	<STX>sWN{SPC}TransmitTargets{SPC}1<ETX>
	Hex	02 73 57 4E 20 54 72 61 6E 73 6D 69 74 54 61 72 67 65 74 73 20 31 03
CoLa B	Binary	02 02 02 02 00 00 00 15 73 57 4E 20 54 72 61 6E 73 6D 69 74 54 61 72 67 65 74 73 20 01 03

Table 28: Example: sWN TransmitTargets



Telegram structure: sWA TransmitTargets						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Answer	String	3	All	sWA	73 57 41
Command	Data telegram	String	11	All	TransmitTargets	54 72 61 6E 73 6D 69 74 54 61 72 67 65 74 73

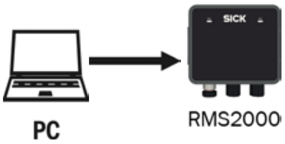
Table 29: Telegram structure: sWA TransmitTargets

Example: Confirmation of sWA TransmitTargets

CoLa A	ASCII	<STX>sWA{SPC}TransmitTargets<ETX>
	Hex	02 73 57 41 20 54 72 61 6E 73 6D 69 74 54 61 72 67 65 74 73 03
CoLa B	Binary	02 02 02 02 00 00 00 14 73 57 41 20 54 72 61 6E 73 6D 69 74 54 61 72 67 65 74 73 20 0D

Table 30: Example: Confirmation of sWA TransmitTargets

4.2.2 Enable/disable object data output in the data content



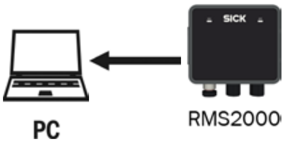
Telegram structure: sWN TransmitObjects						
Telegram part	Description	Variable	Length	Sensor	Values CoLa A (ASCII)	Values CoLa B (Binary)
Command type	Event	String	3	All	sWN	73 57 4E
Command	Data telegram	String	11	All	TransmitObjects	54 72 61 6E 73 6D 69 74 4F 62 6A 65 63 74 73
Measurement	Start/stop	Enum_8	1	All	Stop: 0 Start: 1	Stop: 00 Start: 01

Table 31: Telegram structure: sWN TransmitObjects

Example: sWN TransmitObjects

CoLa A	ASCII	<STX>sWN{SPC}TransmitObjects{SPC}1<ETX>
	Hex	02 73 57 4E 20 54 72 61 6E 73 6D 69 74 54 61 72 67 65 74 73 20 31 03
CoLa B	Binary	02 02 02 02 00 00 00 15 73 57 4E 20 54 72 61 6E 73 6D 69 74 4F 62 6A 65 63 74 73 20 01 07

Table 32: Example: sWN TransmitObjects



Telegram structure: sEA TransmitObjects						
Telegram part	Description	Variable	Length	Sensor	Values CoLa A (ASCII)	Values CoLa B (Binary)
Command type	Answer	String	3	All	sWA	73 57 41
Command	Data telegram	String	11	All	TransmitObjects	54 72 61 6E 73 6D 69 74 4F 62 6A 65 63 74 73

Table 33: Telegram structure: sWA TransmitObjects

Example: Confirmation of sWA TransmitObjects

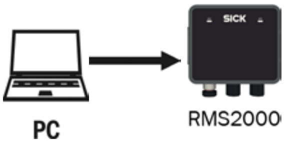
CoLa A	ASCII	<STX>sWA{SPC}TransmitObjects<ETX>
	Hex	02 73 57 41 20 54 72 61 6E 73 6D 69 74 4F 62 6A 65 63 74 73 03
CoLa B	Binary	02 02 02 02 00 00 00 14 73 57 41 20 54 72 61 6E 73 6D 69 74 4F 62 6A 65 63 74 73 20 09

Table 34: Example: Confirmation of sWA TransmitObjects

4.2.3 Send data permanently

NOTE

After changing the parameters, there will be no data telegram or answer from the devices for up to 30 seconds.The same applies when the device is powering up or rebooting.



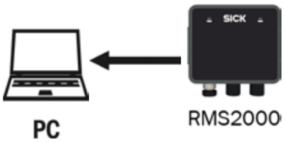
Telegram structure: sEN LMDradardata						
Telegram part	Description	Variable	Length	Sensor	Values CoLa A (ASCII)	Values CoLa B (Binary)
Command type	Event	String	3	All	sEN	73 45 4E
Command	Data telegram	String	11	All	LMDradardata	4C 4D 44 72 61 64 61 72 64 61 74 61
Measurement	Start/stop	Enum_8	1	All	Stop: 0 Start: 1	Stop: 00 Start: 01

Table 35: Telegram structure: sEN LMDradardata

Example: sEN LMDradardata

CoLa A	ASCII	<STX>sEN{SPC}LMDradardata{SPC}1<ETX>
	Hex	02 73 45 4E 20 4C 4D 44 72 61 64 61 72 64 61 74 61 20 31 03
CoLa B	Binary	02 02 02 02 00 00 00 12 73 45 4E 20 4C 4D 44 72 61 64 61 72 64 61 74 61 20 01 48

Table 36: Example: sEN LMDradardata



Telegram structure: sEA LMDradardata						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Answer	String	3	All	sEA	73 45 41
Command	Data telegram	String	11	All	LMDradardata	4C 4D 44 72 61 64 61 72 64 61 74 61
Measurement	Start/stop	Enum_8	1	All	Stop: 0 Start: 1	Stop: 00 Start: 01

Table 37: Telegram structure: sEA LMDradardata

Example: Confirmation of sEA LMDradardata

CoLa A	ASCII	<STX>sEA{SPC}LMDradardata{SPC}1<ETX>
	Hex	02 73 45 41 20 4C 4D 44 72 61 64 61 72 64 61 74 61 20 31 03
CoLa B	Binary	02 02 02 02 00 00 00 12 73 45 41 20 4C 4D 44 72 61 64 61 72 64 61 74 61 20 01 47

Table 38: Example: Confirmation of sEA LMDradardata

Telegram stream

The answer to the telegram will be followed by the LMDradardata:

NOTE

Leading zeros of a value will not be displayed in ASCII.

Telegram structure: sSN LMDradardata							
Telegram part		Description	Variable	Length	Sensor /Data	Values CoLa A (ASCII)	Values CoLa B (Binary)
Command type		Read	String	3	All	sSN	73 53 4E
Command		Data telegram	String	11	RMS2000	LMDradardata	4C 4D 44 73 63 61 6E 64 61 74 61
Version number		For detecting format changes by the version. (may change due to firmware changes)	Uint_16	2	All	0000h … FFFFh	00 00 … FF FF
Device	Device number	Defined with SOPAS	Uint_16	2	All	0000h … FFFFh	00 00 … FF FF
	Serial number	Defined in factory	Uint_32	4	All	00000000h … FFFFFFFFh BC614Eh: 12345678d	00 00 00 00 … FF FF FF FF
	Device status	(See values column)	Uint_8	2 × 1	All	Ok: 00 00 Error: 00 01	00 00 00 01
Status info	Telegram counter	Number of measurement telegrams finished in the scanner and given to the interface. [1]	Uint_16	2	All	0000h … FFFFh	00 00 … FF FF
	Scan counter	Number of scans which were created in the device; counts how many scans were really done.	Uint_16	2	All	0000h … FFFFh	00 00 … FF FF
	Time since start up in µs	Counting the time since power up the device; starting with 0	Uint_32	4	All	00000000h … FFFFFFFFh	00 00 00 00 … FF FF FF FF

[1] Does not count how many telegrams were really given out; is relevant if not all scans are delivered from the scan core.

	Time of transmission in µs	Time in µs when the complete scan is transmitted to the buffer for data output; starting with 0 at scanner bootup.	Uint_32	4	All	00000000h … FFFFFFFFh	00 00 00 00 … FF FF FF FF
	Status of digital inputs	Low byte represents input 1.	Uint_8	2 × 1	All	All inputs low: 00 00 All inputs high: 00 FF Always 00 00	00 00 00 FF 00 00
	Status of digital outputs	Low byte represents output 1.	Uint_8	2 × 1	All	All outputs low: 00 00 All outputs high: 00 FF Always 00 00	All outputs low: 00 00 All outputs high: 00 FF
	CycleDuration	Actual radar measurement cycles time including detection and tracking, <= 50ms	Uint_16	2	RMS2000	Example: B400 Decimal: 46080 microsecond	Example: B4 00
Reserved		Reserved	Uint_16	2	RMS2000	0	00
Amount of encoder (Not in use)			Enum_16	2	RMS2000 always 1	0 … 3 If 0, then next two values are missing.	00 … 03
Values	Encoder position (Not in use)	Info in ticks	Uint_32	4	RMS2000 always 0	00000000h … FFFFFFFFh	00 00 00 00 … FF FF FF FF
	Encoder speed (Not in use)	Ticks/mm	Uint_16	2	RMS2000 always 0	0000h … FFFFh	00 00 … FF FF

RMS2000 Target data (16 bit and 8 bit) will be sent in one telegram. (from “sSN LMDradardata” to the end)

RMS2000 Object data (16 bit and 8 bit) will be sent in one telegram. (from “sSN LMDradardata” to the end)

The differences are maily in 16 bit channels, 8 bit channels and the corresponding scale factors and offsets.

For RMS2000 Target data:

16 bit channels:

- DIST1:Radial distance (unit: mm)
- AZMT1:Azimuth angle (unit: °)
- VRAD1:Radial speed (unit: m/s)
- AMPL1:Amplitude (unit: dB)

8 bit channel:

- MODE1:Internal usage only

For RMS2000 Object data:

16 bit channels:

- P3DX1: Distance in x-direction (unit: mm)
- P3DY1: Distance in y-direction (unit: mm)
- V3DX1:Speed, x-direction (unit: m/s)
- V3DY1:Speed, y-direction (unit: m/s)
- OBLE1: Internal usage only

8 bit channel:

- OBID1: Object ID
- OBCO1: Internal usage only

If neither Target data nor Object data are selected, the output format will be only the part without 16 bit channel and 8 bit channel per every 250ms. (header+ending per 250 ms as “heartbeat”)

Amount of 16 bit channels		Number of 16 bit channels that provide measured data	Uint_16	2	RMS2000	Target channels:0..4 Object channels: 0..3	Target channels:0..4 Object channels: 0..3
Output channel (16 bit)	Content	Defines the content of the output channel Unit of radial distance values (DIST1) is mm	String	5	RMS2000 Target (If Target data are sent, this part contains only Target data. Object data are in another telegram)	DIST1: Radial distance AZMT1: Azimuth angle VRAD1: Radial speed AMPL1: Amplitude	44 49 53 54 31 41 5A 4D 54 31 56 52 41 44 31 41 4D 50 4C 31
					Or, RMS2000 Object (If Object data are sent, this part contains only Objct data. Object data are sent in another telegram)	P3DX1: Distance at x-direction P3DY1: Distance at y-direction V3DX1: Speed, x-direction V3DY1: Speed, y-direction OBLE1: Internal usage only	50 33 44 58 31 56 33 44 58 31 4F 42 4C 45 31
	Scale factor	Scale factor or factor of the measurement values	Real as float according to IEEE754	4	RMS2000 (Target)	DIST1: 42200000f =40d AZMT1: 3C23D70Af=0.01d VRAD1: 3C23D70Af=0.01d AMPL1: 3DCCCCCDf=0.1d	42 20 00 00 3C 23 D7 0A 3C 23 D7 0A 3D CC CC CD

					Or, RMS2000 (Object)	P3DX1: 41800000f=16d P3DY1: 41800000f=16d V3DX1: 3DCCCCCDf=0.1d V3DY1: 3DCCCCCDf=0.1d OBLE1: 3C23D70Af=0.01d	41 80 00 00 3D CC CC CD 3C 23 D7 0A
	Scale factor offset	Sets starting point of measurement	Real as float according to IEEE754	4	RMS2000	Always 0	00 00 00 00
	Amount of data	Defines the number of items on measured output	Uint_16	2	RMS2000	0000h … 0040h 0d..64d	00 00 … FF FF
	Data_1 Data_n	Data stream starting Data_1 to Data_n	Int_16	2	RMS2000	0000h … FFFFh	00 00 00 00 … 00 00 4E 20
Amount of 8 bit channels		Amount of 8 bit channels, giving out the measured data	Enum_16	2	RMS2000	Output channels: 0 or 1	Output channels: 00 or 01
Output channel (8 bit)	Content	Defines the content of the output channel	String	5	RMS2000 Target	MODE1: Internal usage only	4D 4F 44 45 31
				5	Or, RMS2000 Object	OBID1 :Object ID OBCO1: Internal usage only	4F 42 49 44 31 4F 42 43 4F 31

	Scale factor	Scale factor or of the measurement values	Real as float according to IEEE754	4	RMS2000 Target	MODE1:3F800000h=1d	3F 80 00 00
					Or, RMS2000 Object	OBID1:3F800000h=1d OBCO1: 3F800000h=1d	
	Scale factor offset	Sets starting point of measurement	Real as float according to IEEE754	4	RMS2000	00000000h	00 00 00 00
	Amount of data	Amount	Uint_16	2	RMS2000	0000h … FFFFh	00 00 … FF FF
	Data_1 Data_n	Data stream starting Data_1 to Data_n	Uint_8	1	RMS2000	00h … FFh	00 … FF
Position		Output of position data	Enum_16	2	All	No position data: 0 Position data: 1	No position data: 00 00 Position data: 00 01
Position information	X position	X-coordinate as float acco. to IEEE754	Real	4	All	0h … FFFFFFFFh	00 00 00 00 … FF FF FF FF
	Y position	Y-coordinate as float acco. to IEEE754	Real	4	All	0h … FFFFFFFFh	00 00 00 00 … FF FF FF FF
	Z position	Z-coordinate as float acco. to IEEE754	Real	4	All	0h … FFFFFFFFh	00 00 00 00 … FF FF FF FF
	X rotation	X rotation in the coordinate system	Real	4	All	0h … FFFFFFFFh	00 00 00 00 … FF FF FF FF

	Y rotation	Y rotation in the coordinate system	Real	4	All	0h … FFFFFFFFh	00 00 00 00 … FF FF FF FF
	Z rotation	Z rotation in the coordinate system	Real	4	All	0h … FFFFFFFFh	00 00 00 00 … FF FF FF FF
	Rotations type	Kind of rotation	Enum_8	1	All	No rotation: 0 Pitch: 1 Roll: 2 Free: 3	No rotation: 00 Pitch: 01 Roll: 02 Free: 03
	Transmits the name of device	Device name	Uint_8	1	All	No name: 0 Name: 1	No name: 00 Name: 01
Name		Device name	Uint_16	2	All	No name: 0 Name: 1	No name: 00 00 Name: 00 01
Name information	Length	Length of name	Uint_8	1	All	0h … Fh	00 … 0F
	Name	Device name in characters	String	16	All	20h … 7Ah	20 … 7A
Comment		Comment	Uint_16	2	All	No comment: 0 Comment: 1	No comment: 00 00 Comment: 00 01
Comment information	Length	Length of comment	Uint_8	1	All	0h … Fh	00 … 0F
	Comment	Transmits a comment in characters	String	16	All	20h … 7Ah	20 … 7A
Time		Transmits a time stamp	Uint_16	2	All	No time: 0 Time: 1	No time: 00 00 Time: 00 01
Time info	Year		Uint_16	2	All	0000h … 270Fh	00 00 … 27 0F
	Month	1 to 12	Uint_8	1	All	00h … 0Ch	00 … 0C
	Day	Day of month 1 to 31	Uint_8	1	All	00h … 1Fh	00 … 1F
	Hour	0 to 23	Uint_8	1	All	00h … 17h	00 … 17
	Minute	0 to 59	Uint_8	1	All	00h … 3Bh	00 … 3B
	Second	0 to 59	Uint_8	1	All	00h … 3Bh	00 … 3B
	Microsecond	0 to 999999	Uint_32	4	All	00000000h … 000F423Fh	00 00 00 00 … 00 0F 42 3F
Event info		Display event info	Uint_16	2	All	No info: 0 Transmit info: 1	No info: 00 00 Transmit info: 00 01
Event Information	Type	Fast digital input	String	4	All	FDIN	FDIN
	Encoder position	Position of encoder when event happened	Uint_32	4	All	00000000h … FFFFFFFFh	00 00 00 00 … FF FF FF FF

	Time of event	Time (µs) of encoder when event happened	Uint_32	4	All	00000000h ... FFFFFFFFh	00 00 00 00 ... FF FF FF FF
	Angle of event	Angle of encoder when event happened	Int_32	4	All	0 ... 3600000	00 00 00 00 ... 00 36 EE 80

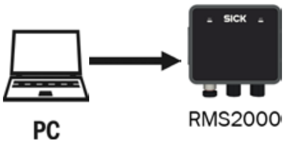
Table 39: Telegram structure: Datastream of sSN LMDradardata

NOTE

- The grey written parts are not given out by the sensor.

4.3 Status

4.3.1 Read device ident



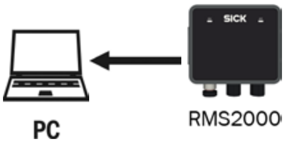
Telegram structure: sRN DeviceIdent						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Read	String	3	All	sRN	73 52 4E
Command	Read ident	String	11	All	DeviceIdent	44 65 76 69 63 65 49 64 65 6E 74

Table 40: Telegram structure: sRN DeviceIdent

Example: sRN DeviceIdent

CoLa A	ASCII	<STX>sRN{SPC}DeviceIdent<ETX>
	Hex	02 73 52 4E 20 44 65 76 69 63 65 49 64 65 6E 74 03
CoLa B	Binary	02 02 02 02 00 00 00 0F 73 52 4E 20 44 65 76 69 63 65 49 64 65 6E 74 25

Table 41: Example: sRN DeviceIdent



Telegram structure: sRA DeviceIdent						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Answer	String	3	All	sRA	73 52 41
Command	Start the device	String	11	All	DeviceIdent	44 65 76 69 63 65 49 64 65 6E 74
Value	Length of ident	Enum_16	1	All	0 ... 22h	0 ... 22h
Value	Ident information	String		All	(See example)	(See example)
Value	Length of version	Enum_16	1	All	0 ... 22h	0 ... 22h
Value	Version information	String		All	(See example)	(See example)

Table 42: Telegram structure: sRA DeviceIdent

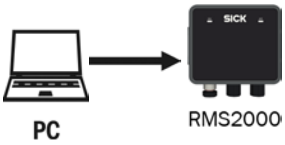
Example: sRA DeviceIdent

CoLa A	ASCII	<STX>sRA{SPC}DeviceIdent{SPC}7{SPC}RMSxxxx{SPC}A{SPC}1.6.0.159R<ETX>
	Hex	Always ASCII answer
CoLa B	Binary	02 02 02 02 00 00 00 25 73 52 41 20 44 65 76 69 63 65 49 64 65 6E 74 20 00 07 52 4D 53 78 78 78 78 00 0A 31 2E 36 2E 30 2E 31 35 39 52 3D

Table 43: Example: sRA DeviceIdent

4.3.2 Read device type

This telegram asks for the device type.



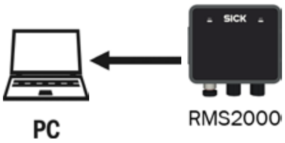
Telegram structure: sRN DItype						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Read	String	3	All	sRN	73 52 4E
Command	Ask state	String	6	All	DItype	44 49 74 79 70 65

Table 44: Telegram structure: sRN DItype

Example: sRN DItype

CoLa A	ASCII	<STX>sRN{SPC}DItype<ETX>
	Hex	02 73 52 4E 20 44 49 74 79 70 65 03
CoLa B	Binary	02 02 02 02 00 00 00 0A 73 52 4E 20 44 49 74 79 70 65 5A

Table 45: Example: sRN DItype



Telegram structure: sRA DItype						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Answer	String	3	All	sRA	73 52 41
Command	Ask state	String	6	All	DItype	44 49 74 79 70 65
Length of type key	Number of digits of the following type code length	Uint_16	1	All	0d … 18d (0h … 12h)	00 … FF
Device type	Type code of the device	String	(var.)	All	(Device type)	(Device type)

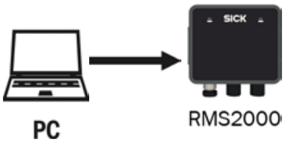
Table 46: Telegram structure: sRA DItype

Example for RMS2000

CoLa A	ASCII	<STX>sRA{SPC}DItype{SPC}F{SPC}RMS2731C-636111<ETX>
	Hex	02 73 52 41 20 44 49 74 79 70 65 20 46 20 52 4D 53 32 37 33 31 43 2D 36 33 36 31 31 31 03
CoLa B	Binary	02 02 02 02 00 00 00 1C 73 52 41 20 44 49 74 79 70 65 20 00 0F 52 4D 53 32 37 33 31 43 2D 36 33 36 31 31 31 5D

Table 47: Example for sRA DItype

4.3.3 Read serial number



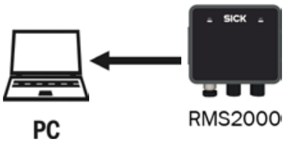
Telegram structure: sRN SerialNumber						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Read	String	3	All	sRN	73 52 4E
Command	Read Serial Number	String	12	All	SerialNumber	53 65 72 69 61 6C 4E 75 6D 62 65 72

Table 48: Telegram structure: sRN SerialNumber

Example: sRN SerialNumber

CoLa A	ASCII	<STX>sRN{SPC}SerialNumber<ETX>
	Hex	02 73 52 4E 20 53 65 72 69 61 6C 4E 75 6D 62 65 72 03
CoLa B	Binary	02 02 02 02 00 00 00 10 73 52 4E 20 53 65 72 69 61 6C 4E 75 6D 62 65 72 4C

Table 49: Example: sRN SerialNumber



Telegram structure: sRA SerialNumber						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Answer	String	3	All	sRA	73 52 41
Command	Read device name	String	12	All	SerialNumber	53 65 72 69 61 6C 4E 75 6D 62 65 72
Value	Number of digits of the following serial number length	Uint_16	1	All	0d ... 255d (0h ... FF)	00 ... FF
Value	SerialNumber	String	max16	All	(Serial Number)	(SerialNumber)

Table 50: Telegram structure: sRA SerialNumber

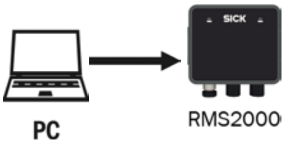
Example: sRA SerialNumber

CoLa A	ASCII	<STX>sRA{SPC} SerialNumber {SPC}8 {SPC}12345678<ETX>
	Hex	02 73 52 41 20 53 65 72 69 61 6C 4E 75 6D 62 65 72 20 38 20 31 32 33 34 35 36 37 38 03
CoLa B	Binary	02 02 02 02 00 00 00 1B 73 52 41 20 53 65 72 69 61 6C 4E 75 6D 62 65 72 20 00 08 31 32 33 34 35 36 37 38 63

Table 51: Example: sRA SerialNumber

4.3.4 Read order number

This telegram reads the device order number.



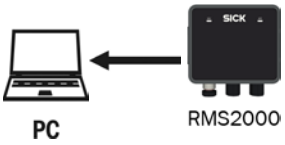
Telegram structure: sRN OrdNum						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Read	String	3	All	sRN	73 52 4E
Command	Read state	String	6	All	OrdNum	4F 72 64 4E 75 6D

Table 52: Telegram structure: sRN OrdNum

Example: sRN OrdNum

CoLa A	ASCII	<STX>sRN{SPC}OrdNum<ETX>
	Hex	02 73 52 4E 20 4F 72 64 4E 75 6D 03
CoLa B	Binary	02 02 02 02 00 00 00 0A 73 52 4E 20 4F 72 64 4E 75 6D 40

Table 53: Example: sRN OrdNum



Telegram structure: sRA OrdNum						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Answer	String	3	All	sRA	73 52 41
Command	Read state	String	6	All	OrdNum	4F 72 64 4E 75 6D
Value	Number of digits of the following order number length	Uint_16	1	All	0d … 32d (0h … 20h)	00 … FF
Order number	Order number	String	max16	All	0000000 … 9999999	00 00 00 00 00 00 00 … FF FF FF FF FF FF FF

Table 54: Telegram structure: sRA OrdNum

Example: sRA OrdNum

CoLa A	ASCII	<STX>sRA{SPC}OrdNum{SPC}7{SPC}1128853<ETX>
	Hex	02 73 52 41 20 4F 72 64 4E 75 6D 20 37 20 31 31 32 38 38 35 33 03
CoLa B	Binary	02 02 02 02 00 00 00 14 73 52 41 20 4F 72 64 4E 75 6D 20 00 07 31 31 32 38 38 35 33 5C

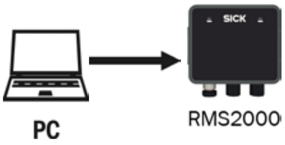
Table 55: Example for sRA OrdNum

4.4 Interfaces

4.4.1 Set IP address

IMPORTANT

- Save permanently to set values. Changes will be active after rebooting the device.
- Settings must correspond with network in which scanner is used. Else device cannot be found any more.



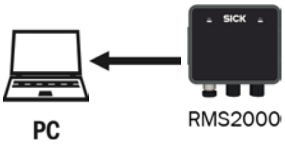
Telegram structure: sWN EIIpAddr (Authorized client)						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Write	String	3	All	sWN	73 57 4E
Command	Set IP address	String	8	All	EIIpAddr	45 49 49 50 41 64 64 72
IP address	Set values in hex	Uint_32	4	All	00 00 00 00h ... FF FF FF FFh (decimal values unwieldy)	00 00 00 00 ... FF FF FF FF

Table 56: Telegram structure: sWN EIIpAddr

Example: sWN EIIpAddr 192.168.0.2

CoLa A	ASCII	<STX>sWN{SPC}EIIpAddr{SPC}C0{SPC}A8{SPC}0{SPC}2<ETX>
	Hex	02 73 57 4E 20 45 49 49 70 41 64 64 72 20 43 30 20 41 38 20 30 20 32 03
CoLa B	Binary	02 02 02 02 00 00 00 11 73 57 4E 20 45 49 49 70 41 64 64 72 20 C0 A8 00 02 06

Table 57: Example: sWN EIIpAddr 192.168.0.2



Telegram structure: sWA EIIpAddr						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Answer	String	3	All	sWA	73 57 41
Command	Set IP address	String	8	All	EIIpAddr	45 49 49 50 41 64 64 72

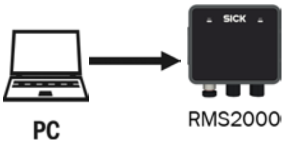
Table 58: Telegram structure: sWA EIIpAddr

Example: sWA EIIpAddr

CoLa A	ASCII	<STX>sWA{SPC}EIIpAddr<ETX>
	Hex	02 73 57 41 20 45 49 49 70 41 64 64 72 03
CoLa B	Binary	02 02 02 02 00 00 00 0D 73 57 41 20 45 49 49 70 41 64 64 72 20 63

Table 59: Example: sWA EIIpAddr

4.4.2 Read IP address



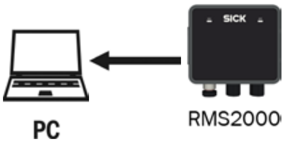
Telegram structure: sRN EIIpAddr						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Read	String	3	All	sRN	73 52 4E
Command	Read IP address	String	8	All	EIIpAddr	45 49 49 50 41 64 64 72

Table 60: Telegram structure: sRN EIIpAddr

Example: srN EIIpAddr

CoLa A	ASCII	<STX>sRN{SPC}EIIpAddr<ETX>
	Hex	02 73 57 4E 20 45 49 49 70 41 64 64 72 03
CoLa B	Binary	02 02 02 02 00 00 00 0C 73 52 4E 20 45 49 49 70 41 64 64 72 49

Table 61: Example: srN EIIpAddr



Telegram structure: sRA EIIpAddr						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Answer	String	3	All	sRA	73 52 41
Command	Set IP address	String	8	All	EIIpAddr	45 49 49 50 41 64 64 72
IP address	Set values in hex	Uint_32	4	All	00 00 00 00h … FF FF FF FFh (decimal values unwieldy)	00 00 00 00 … FF FF FF FF

Table 62: Telegram structure: sRA EIIpAddr

Example: sRA EIIpAddr 192.168.0.2

CoLa A	ASCII	<STX>sRA{SPC}EIIpAddr{SPC}C0{SPC}A8{SPC}00{SPC}02<ETX>
	Hex	02 73 57 41 20 45 49 49 70 41 64 64 72 03
CoLa B	Binary	02 02 02 02 00 00 00 11 73 52 41 20 45 49 49 70 41 64 64 72 20 C0 A8 00 02 0C

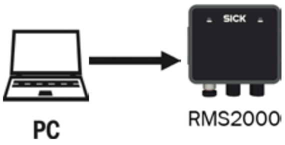
Table 63: Example: sRA EIIpAddr 192.168.0.2

4.4.3 Set Ethernet gateway

Change Ethernet gateway IP address (TCP/IP)

IMPORTANT

- Save permanently to set values. Changes will be active after rebooting the device.
- Settings must correspond with network in which scanner is used. Else device cannot be found any more.



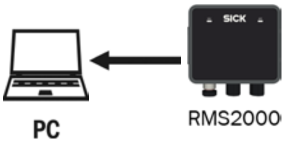
Telegram structure: sWN Elgate						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Write	String	3	All	sWN	73 57 4E
Command	Set gateway adress	String	6	All	Elgate	45 49 67 61 74 65
Gateway address	Set values in hex	Uint_32	4	All	00 00 00 00h ... FF FF FF FFh (decimal values unwieldy)	00 00 00 00 ... FF FF FF FF

Table 64: Telegram structure: sWN Elgate

Example: sWN Elgate 192.168.0.1

CoLa A	ASCII	<STX>sWN{SPC}Elgate{SPC}C0{SPC}A8{SPC}00{SPC}01<ETX>
	Hex	02 73 57 4E 20 45 49 67 61 74 65 20 43 30 20 41 38 20 30 30 20 30 31 03
CoLa B	Binary	02 02 02 02 00 00 00 0F 73 57 4E 20 45 49 67 61 74 65 20 C0 A8 00 01 18

Table 65: Example: sWN Elgate 192.168.0.1



Telegram structure: sWA Elgate						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Answer	String	3	All	sWA	73 57 41
Command	Set gateway address	String	6	All	Elgate	45 49 67 61 74 65

Table 66: Telegram structure: sWA Elgate

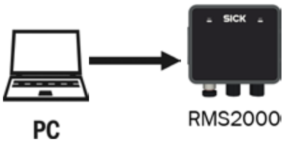
Example: sWA Elgate

CoLa A	ASCII	<STX>sWA{SPC}Elgate<ETX>
	Hex	02 73 57 41 20 45 49 67 61 74 65 03
CoLa B	Binary	02 02 02 02 00 00 00 0A 73 57 41 20 45 49 67 61 74 65 20 7E

Table 67: Example: sWA Elgate

4.4.4 Read Ethernet gateway

Read for the Ethernet gateway (TCP/IP)



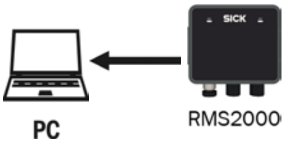
Telegram structure: sRN Elgate						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Read	String	3	All	sRN	73 52 4E
Command	Read gateway address	String	6	All	Elgate	45 49 67 61 74 65

Table 68: Telegram structure: sRN Elgate

Example: sRN Elgate

CoLa A	ASCII	<STX>sRN{SPC}Elgate<ETX>
	Hex	02 73 52 4E 20 45 49 67 61 74 65 03
CoLa B	Binary	02 02 02 02 00 00 00 0A 73 52 4E 20 45 49 67 61 74 65 54

Table 69: Example: sRN Elgate



Telegram structure: sRA Elgate						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Answer	String	3	All	sRA	73 52 41
Command	Read gateway address	String	6	All	Elgate	45 49 67 61 74 65
Gateway address	Values in hex	Uint_32	4	All	00 00 00 00h ... FF FF FF FFh (decimal values unwieldy)	00 00 00 00 ... FF FF FF FF

Table 70: Telegram structure: sRA Elgate

Example: sRA EIgate 192.168.0.1

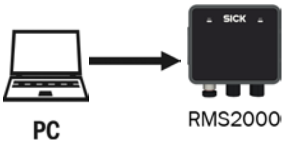
CoLa A	ASCII	<STX>sRA{SPC}EIgate{SPC}C0{SPC}A8{SPC}00{SPC}01<ETX>
	Hex	02 73 52 41 20 45 49 67 61 74 65 20 C0 A8 00 01 03
CoLa B	Binary	02 02 02 02 00 00 00 0F 73 52 41 20 45 49 67 61 74 65 20 C0 A8 00 01 12

Table 71: Example: sRA EIgate 192.168.0.1

4.4.5 Set IP mask

IMPORTANT

- Save permanently to set values. Changes will be active after rebooting the device.
- Settings must correspond with network in which scanner is used. Else device cannot be found any more.



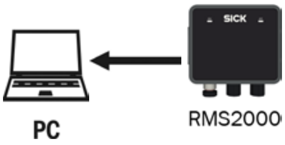
Telegram structure: sWN EImask						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Write	String	3	All	sWN	73 57 4E
Command	Set IP mask	String	6	All	EImask	45 49 6D 61 73 6B
IP mask	Set values in hex	Uint_32	4	All	00 00 00 00h ... FF FF FF FFh (decimal values unwieldy)	00 00 00 00 ... FF FF FF FF

Table 72: Telegram structure: sWN EImask

Example: sWN EImask 255.255.254.0

CoLa A	ASCII	<STX>sWN{SPC}EImask{SPC}FF{SPC}FF{SPC}FE{SPC}00<ETX>
	Hex	02 73 57 4E 20 45 49 6D 61 73 6B 20 46 46 20 46 46 20 46 45 20 30 30 03
CoLa B	Binary	02 02 02 02 00 00 00 0F 73 57 4E 20 45 49 6D 61 73 6B 20 FF FF FE 00 8C

Table 73: Example: sWN EImask 255.255.254.0



Telegram structure: sWA EImask						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Answer	String	3	All	sWA	73 57 41
Command	Set IP mask	String	6	All	EImask	45 49 6D 61 73 6B

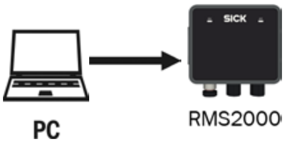
Table 74: Telegram structure: sWA EImask

Example: sWA EImask

CoLa A	ASCII	<STX>sWA{SPC}EImask<ETX>
	Hex	02 73 57 41 20 45 49 6D 61 73 6B 03
CoLa B	Binary	02 02 02 02 00 00 00 0A 73 57 41 20 45 49 6D 61 73 6B 20 7D

Table 75: Example: sWA EImask

4.4.6 Read IP mask



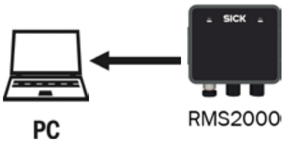
Telegram structure: sRN EImask						
Telegram part	**Description**	**Variable**	**Length**	**Sensor**	**Values CoLa A (ASCII)**	**Values CoLa B (Binary)**
Command type	Read	String	3	All	sRN	73 52 4E
Command	Read IP mask	String	6	All	EImask	45 49 6D 61 73 6B

Table 76: Telegram structure: sRN EImask

Example: sRN EImask

CoLa A	ASCII	<STX>sRN{SPC}EImask<ETX>
	Hex	02 73 52 4E 20 45 49 6D 61 73 6B 03
CoLa B	Binary	02 02 02 02 00 00 00 0A 73 52 4E 20 45 49 6D 61 73 6B 57

Table 77: Example: sRN EImask



Telegram structure: sRA EImask						
Telegram part	Description	Variable	Length	Sensor	Values CoLa A (ASCII)	Values CoLa B (Binary)
Command type	Answer	String	3	All	sRA	73 52 41
Command	Read IP mask	String	6	All	EImask	45 49 6D 61 73 6B
IP mask	Values in hex	Uint_32	4	All	00 00 00 00h ... FF FF FF FFh (decimal values unwieldy)	00 00 00 00 ... FF FF FF FF

Table 78: *Telegram structure: sRA EImask*

Example: sRA EImask 255.255.254.0

CoLa A	ASCII	<STX>sRA{SPC}EImask{SPC}FF{SPC}FF{SPC}FE{SPC}00<ETX> <STX>sRN{SPC}EImask<ETX>
	Hex	02 73 52 41 20 45 49 6D 61 73 6B 20 45 49 6D 61 73 6B 03
CoLa B	Binary	02 02 02 02 00 00 00 0F 73 52 41 20 45 49 6D 61 73 6B 20 FF FF FE 00 86

Table 79: *Example: sRA EImask 255.255.254.0*

5 List of tables

a

Australia
Phone +61 (3) 9457 0600
1800 33 48 02 – tollfree
E-Mail sales@sick.com.au

Austria
Phone +43 (0) 2236 62288-0
E-Mail office@sick.at

Belgium/Luxembourg
Phone +32 (0) 2 466 55 66
E-Mail info@sick.be

Brazil
Phone +55 11 3215-4900
E-Mail comercial@sick.com.br

Canada
Phone +1 905.771.1444
E-Mail cs.canada@sick.com

Czech Republic
Phone +420 234 719 500
E-Mail sick@sick.cz

Chile
Phone +56 (2) 2274 7430
E-Mail chile@sick.com

China
Phone +86 20 2882 3600
E-Mail info.china@sick.net.cn

Denmark
Phone +45 45 82 64 00
E-Mail sick@sick.dk

Finland
Phone +358-9-25 15 800
E-Mail sick@sick.fi

France
Phone +33 1 64 62 35 00
E-Mail info@sick.fr

Germany
Phone +49 (0) 2 11 53 010
E-Mail info@sick.de

Greece
Phone +30 210 6825100
E-Mail office@sick.com.gr

Hong Kong
Phone +852 2153 6300
E-Mail ghk@sick.com.hk

Hungary
Phone +36 1 371 2680
E-Mail ertekesites@sick.hu

India
Phone +91-22-6119 8900
E-Mail info@sick-india.com

Israel
Phone +972 97110 11
E-Mail info@sick-sensors.com

Italy
Phone +39 02 27 43 41
E-Mail info@sick.it

Japan
Phone +81 3 5309 2112
E-Mail support@sick.jp

Malaysia
Phone +603-8080 7425
E-Mail enquiry.my@sick.com

Mexico
Phone +52 (472) 748 9451
E-Mail mexico@sick.com

Netherlands
Phone +31 (0) 30 229 25 44
E-Mail info@sick.nl

New Zealand
Phone +64 9 415 0459
0800 222 278 – tollfree
E-Mail sales@sick.co.nz

Norway
Phone +47 67 81 50 00
E-Mail sick@sick.no

Poland
Phone +48 22 539 41 00
E-Mail info@sick.pl

Romania
Phone +40 356-17 11 20
E-Mail office@sick.ro

Russia
Phone +7 495 283 09 90
E-Mail info@sick.ru

Singapore
Phone +65 6744 3732
E-Mail sales.gsg@sick.com

Slovakia
Phone +421 482 901 201
E-Mail mail@sick-sk.sk

Slovenia
Phone +386 591 78849
E-Mail office@sick.si

South Africa
Phone +27 10 060 0550
E-Mail info@sickautomation.co.za

South Korea
Phone +82 2 786 6321/4
E-Mail infokorea@sick.com

Spain
Phone +34 93 480 31 00
E-Mail info@sick.es

Sweden
Phone +46 10 110 10 00
E-Mail info@sick.se

Switzerland
Phone +41 41 619 29 39
E-Mail contact@sick.ch

Taiwan
Phone +886-2-2375-6288
E-Mail sales@sick.com.tw

Thailand
Phone +66 2 645 0009
E-Mail marcom.th@sick.com

Turkey
Phone +90 (216) 528 50 00
E-Mail info@sick.com.tr

United Arab Emirates
Phone +971 (0) 4 88 65 878
E-Mail contact@sick.ae

United Kingdom
Phone +44 (0)17278 31121
E-Mail info@sick.co.uk

USA
Phone +1 800.325.7425
E-Mail info@sick.com

Vietnam
Phone +65 6744 3732
E-Mail sales.gsg@sick.com

Detailed addresses and further locations at **www.sick.com**

SICK AG | Waldkirch | Germany | www.sick.com

SICK
Sensor Intelligence.