TECHNICAL INFORMATION

NAV245 Telegram listing

Supplement to Telegram listing
Ranging sensors, article no. 8014631

SICK
Sensor Intelligence.

Telegram listing version

For the latest version of this telegram listing (PDF), see www.sick.com.

CONTENTS

1 INTRODUCTION 5

1.1 Function 5
1.2 Telegrams 5
1.3 Command Formats 6
1.3.1 Protocols in SOPAS ET 6
1.3.2 Binary Protocol 6
1.3.3 ASCII Protocol 6
1.3.4 Summary Example and Format Explanation 8
1.4 Data Types of Variables 9
1.5 Output Landmarks 10
1.6 Principle of the Landmark Mode 10
1.7 Synchronization NAV – AGV Timer 11
1.8 Error Messages (sFA) 12

2 LANDMARK DETECTION 13

2.1 Landmark data output configuration 13
2.1.1 Command: Config Landmark Fixed Length 13
2.1.2 Command: Set Max Landmark Length 14
2.2 Parameter for the Landmark configuration 15
2.2.1 Command: Set Reflector Size 15
2.2.2 Command: Set Reflector Type 16
2.2.3 Command: Set Sector Muting 17
2.2.4 Command: Set N Closest Reflectors 19
2.2.5 Command: Set Action Radius 20
2.2.6 Command: Set Detection Mode 22
2.3 Landmark data in 2D Measurement Mode 24
2.3.1 Command: Landmark data 24

3 GENERAL COMMANDS 26

3.1 Device Information 26
3.1.1 Command: Read Device Identification 26
3.1.2 Command: Read Serial Number 27
3.1.3 Command: Read Application Version 28
3.2 Commands with Method Call 29
3.3 Procedure of Method Call 29
3.3.1 Command: Set User Level 29
3.3.2 Command: Store Data Permanent 31
3.3.3 Command: Synchronize Timestamp 32
3.3.4 Command: Device Reset 33

4 RESULT PORT PROTOCOL 34

4.1 Introduction of the Result Port Protocol 34
4.1.1 General 34
4.1.2 Usage of result port 34
4.1.3 Result Port Payload Types 34
4.2 Framing 35
4.2.1 Checksum Calculation 35

4.2.2 Result of Landmark Detection 36
4.2.3 Scandata Result 37
4.2.4 Second Pulse Channel 37
4.3 Combination of telegrams 38
4.4 Parametrization and Visualization 39
4.4.1 Parametrization 39
4.5 Visualization in SOPAS 40
4.6 Usage Example 41
4.6.1 Separate transmission of Scandata and Landmarks 41
4.6.2 Combined Telegram with one selected result only 42
4.6.3 Combined Telegram with two selected results 43
4.6.4 Combined Telegram with two selected results and individual output intervals 44

1 Introduction

1.1 Function

This document shows how to send telegrams via terminal program to the NAV 245 and focused to extra feature of the NAV245 to detect landmarks.

Further information about the general parametrization and measurement data of the contour measurements are described in the publication **Telegram listing Ranging sensors, article number 8014631**.

The configuration for the contour measurements are are fixed to

Scan area : 270°
Scan frequency : 25 Hz
Scan resolution : 0,25 Hz

1.2 Telegrams

The focus is on ASCII (also in Hex) in SOPAS ET. Descriptions of the commands, the parameters and the expected response after sending a telegram with the compact command CoLa (**Co**mmand **La**nguage) are described. There are two types of the **Co**mmand **La**nguage: ASCII- (**A**) and Binary- (**B**) format. Parameters access as well as method calls are supported. The common framing and rules of the binary protocol are described in the Introduction as well but particular methods have to be converted from ASCII if CoLa B is used.

Attention: Some commands may change during SICK development processes. Please always use the latest version of the “Telegram Listing”.

1.3 Command Formats

1.3.1 Protocols in SOPAS ET

To connect to the telegram section in SOPAS ET choose:
Tools → Terminal → Connections → New Connection...
At this point you have the choice between using the default protocol of the connected device and creating a "user defined connection".
Attention: If you want to communicate via telegram in SOPAS ET, close all open windows of the scanner in SOPAS ET; otherwise you will get continuous status information that will make it impossible to read your requests and the corresponding responses.

1.3.2 Binary Protocol

The binary protocol of the scanner has always a fixed length and the command string can be converted from the ASCII commands described in this document. The binary protocol has a special framing so that the scanner is able to recognize the start of a binary telegram. The string has to start with 4 STX symbols (i.e. 02 02 02 02) that is followed by the length of the telegram in HEX (i.e.: 00 00 00 1B).

Example:
Binary: 02 02 02 02 00 00 00 1B 73 4D 4E 20 53 65 74 41 63 63 65 73 73 4D 6F 64 65 20 33 46 34 37 32 34 37 34 34 72

Special Characters: Header: 02 02 02 02; Length: 00 00 00 1B; Space: 20; Checksum: 72

The length can be created by counting every letter of the command in ASCII or every character pair in binary (without checksum and framing but with blanks) and convert the sum into HEX. Zeros are added in front until a string of eight characters is built.
The command itself starts after the length characters. Every single letter of the written command is converted to HEX turning into a pair of numbers, followed by a blank and then the parameters also converted in HEX. In between parameters there are no blanks. The "Checksum" is built with XOR beginning the calculation right after the length (i.e. starting with 73 in the example above).
Attention: The response has the same structure as the request. Therefore the expected parameter is also followed by a checksum which may lead to confusion.

1.3.3 ASCII Protocol

The framing of a telegram in ASCII is a <STX> at the start and an <ETX> at the end of each telegram. Commands are written as letters, followed by the parameters as defined in this document. There has to be a space in between the command and the parameters and also in between each parameter (as shown in the example below as _).
Example:
<STX>sMN_SetAccessMode_3_F4724744<ETX>

In HEX the command start with 02 and ends with 03. The spaces are marked as 20. Single numbers that are converted to ASCII always get a 3 in front.

Example:
02734D4E20536574416363657373 4D6F646520 33 20 46 34 37 32 34 37 33 34 03
(Both examples show the same command and parameters.)

1.3.3.1 Request (User to Device)

In the ASCII decimal the user may choose one of the following notations to represent the value of one parameter in a command:

- Decimal notation
  Values prefixed **with** "+" or "-" are interpreted as decimal value.
  Example: Value to be sent: 319 (= 3F hex) ➔ **ASCII:** '+‘,‘3‘,‘1‘, ‘9’

- Hexadecimal notation
  Values **without** a prefixed "+" or "-" are interpreted as a hexadecimal value
  Example: Value to be sent: 3F hex. (= 319 dec) ➔ **ASCII:** '3‘,‘F'

  The NAV245 interprets each parameter individually; therefore the different notations can be mixed within a command string.

Attention: In an ASCII hexadecimal telegram the first notation has to be used but the number has to be transformed to HEX (see example below).

Example:

If you want a value of 319 in a command written in HEX: 319 → in HEX: 2B 33 31 39

HEX: + → 2B ; - → 2D

1.3.3.2 Response (Device to User)

The scanner always responds in hexadecimal notation if no parameter (in decimal) was set before.

Attention: All values are returned from the scanner as HEX values, if they were not set before, so they have to be converted afterwards. Once a value is set in ASCII decimal it is also returned as ASCII decimal value.

1.3.4 Summary Example and Format Explanation

This paragraph shall clarify the differences between the three kinds of command formats.

At first the command basics used in the telegrams are summarized in the following table.

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 by name	sRN	73 52 4E	
Write by name	sWN	73 57 4E	
Method	sMN	73 4D 4E	
Space	{SPC}	20	20

Tab.1-1: Command basics

The following table shows the layout that is used to describe the commands in this document but in order to demonstrate the difference between the two protocols the “Values Binary” column is added.

Telegram structure: sMN SetAccessMode				
Command part	**Description**	**Type**	**Values ASCII**	**Values Binary**
Command Type	SOPAS by name	String	**sMN**	**73 4D 4E**
Command	User level	String	**SetAccessMode**	**53 65 74 41 63 63 65 73 73 4D 6F 64 65**
User level	Select user level	Int_8	**2 = maintenance** **3 = authorized client**	**02 = maintenance** **03 = authorized client** **04 = service**
Password	“Hash” - value for the user level “Maintenance” “Authorized Client” “Service”	UInt_32	**B21ACE26** **F4724744** **81BE23AA**	**B2 1A CE 26** **F4 72 47 44** **81BE23AA**

Tab.1-2: Telegram structure: sMN SetAccessMode

The last table shows a telegram sting in the three different command formats (as they can be seen if used in the SOPAS telegram).

Example String:

sMN SetAccessMode 03 F4724744	
ASCII	<STX>sMN{SPC}SetAccessMode{SPC}3{SPC}F4724744<ETX>
HEX	02 73 4D 4E 20 53 65 74 41 63 63 65 73 73 4D 6F 64 65 20 33 20 46 34 37 32 34 37 34 34 03
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

Tab.1-3: Example string

1.4 Data Types of Variables

The following data types are specified for the variable values in the telegram:

Type	Size (bits)	Range of Value (decimal)	Sign
Bool_1	8	0 or 1	unsigned
UInt_8	8	0 ... 255	unsigned
Int_8	8	–128 ... +127	signed
UInt_16	16	0 ... 65535	unsigned
Int_16	16	–32768 ... +32767	signed
UInt_32	32	0 ... 4294967295	unsigned
Int_32	32	–2147483648 ... +2147483647	signed
Float_32	32	$\pm\sim10^{-44,85}$ ... $+10^{38,53}$	signed
Enum_8	8	0...255	unsigned
Enum_16	16	0..65535	unsigned
String	Depends on content	Note: Strings not terminated by 0	

Tab.1-4: Data types of variables

Note

- The figures in the table column "Size (bits)" refer to the binary transmission of numerical parameters. The ASCII representation differs from these numbers depending on their notation as decimal or hexadecimal values.

- The figures in the table column "Range of value (decimal)" refer to the mathematically possible Range/Value of values for the variables. The current range of values may differ and is described in the “Command” chapter for each individual parameter of a command.

1.5 Output Landmarks

This sequence will start the output of the landmark data.

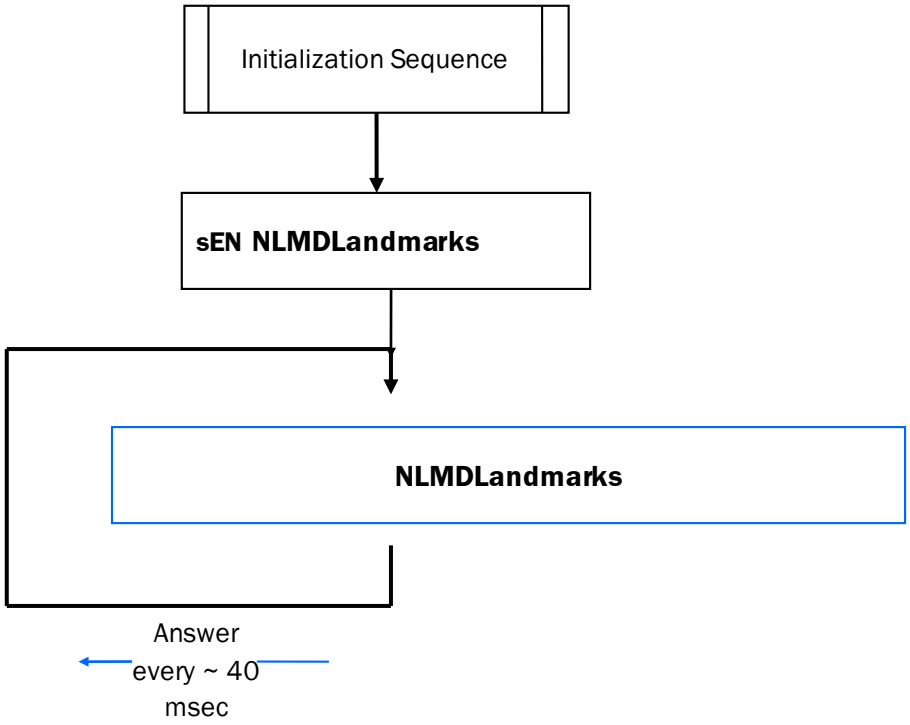
	Step	Command	Parameter	Values
1	Activate the output	**sEN NLMDLandmarks**	Measurement	**1**

Tab.1-5: Example for an initialization sequence

1.6 Principle of the Landmark Mode

This chapter shows the sequences from the initialization of the output of the landmark in order to calculate in an external computer the actual postion.
Once the output is activated the NAV245 will send approx.every 40 msec new landmark data.

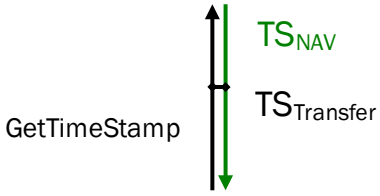
Landmark event Loop :



1.7 Synchronization NAV – AGV Timer

The formula shows the resulting timing offset after requesting the scanner time between the timer of the AGV and the NAV245 and considers also the influence of the transmission time.

GetSyncTimestamp $TS_{offset} = (TS_{AGV} - TS_{Transfer}) - TS_{NAV}$



TS_{offset}	Resulting timing offset
TS_{AGV}	Timestamp of AGV
TS_{NAV}	Timestamp of NAV245
$TS_{Transfer}$	Transfer time via RS232 or Ethernet interface

1.8 Error Messages (sFA)

All commands are initially interpreted by the CoLa-A or CoLa-B command interpreter, before the actual execution of the command follows. If the command interpreter determines an error, it responds with an error message in the form of "FA <Error number>". This is valid for all commands with direct variable access (“RA”/“WA"), events, methods and other commands.Common error numbers are described in the following table.

Error Number (ASCII)	Error Description
1	Access authorization for execution of the method is missing (low user level)
2	Unknown method name / index
3	Unknown variable / index
4	Variable or parameter is out of Range/Value
5	Invalid data
6	Unknown error
7	Memory Overflow
8	Parameter is missing, memory underflow
9	Unknown error type
A	Access authorization to write the variable is missing (low user level)
B	Unknown command for the name server
C	Unknown CoLa command
D	Synchronous method is still running (only one currently possible)
E	Flex Array is too large
F	Unknown Event name / index
10	CoLa-A Type overflow (value is higher than allowed for the type)
11	Illegal CoLa-A character (eg - or letter)
12	No message from the operating system SOPAS
13	No response from the operating system SOPAS
14	Internal error

Tab.1-8: Error messages of the command interpreter

2 Landmark detection

2.1 Landmark data output configuration

2.1.1 Command: Config Landmark Fixed Length

The command NLMDFormatFixedLength is used to control the output format of the landmark data that uses the event *NLMDLandmarks*). Default setting transmits the data in flexible length, depending on the number of measured landmarks.

Request Write

Command Syntax: **sWN NLMDFormatFixedLenght**

Command part	Description	Type	Range/Value
Command type	Request (SOPAS write by name)	String	**sWN**
Command	Set Config Landmark data	String	**NLMDFormatFixedLength**
Fixedlength	If true, NLMDLandmarks's flexarray is filled to NLMDFormatMaxLength with dummy landmarks	Bool1	**0 = inactive (Default)** **1 = active**

Response

Command Syntax: **sWA NLMDFormatFixedLength**

Command part	Description	Type	Range/Value
Command type	Response (SOPAS write by name)	String	**sWA**
Command	Settings of Landmark data	String	**NLMDFormatFixedLength**

Request Read

Command Syntax: **sRN NLMDFormatFixedLength**

Command part	Description	Type	Range/Value
Command type	Request (SOPAS read by name)	String	**sRN**
Command	Read scan data format	String	**NLMDFormatFixedLength**

Response

Command Syntax: **sRA NLMDFormatFixedLength**

Command part	Description	Type	Range/Value
Command type	Response (SOPAS read by name)	String	**sRA**
Command	Settings of Landmark data	String	**NLMDFormatFixedLength**
Fixedlength	If true, NLMDLandmarks's flexarray is filled to NLMDFormatMaxLength with dummy landmarks	Bool1	**0 = inactive (Default)** **1 = active**

2.1.2 Command: Set Max Landmark Length

The command *Set Max Landmark Length* is used to control the maximum number of reflectors to be given out by the event *NLMDLandmarks*). Default setting is a max number of 40 landmarks.

Request Write

Command Syntax: **sWN NLMDFormatMaxLenght**

Command part	Description	Type	Range/Value
Command type	Request (SOPAS write by name)	String	**sWN**
Command	Set maximum of Landmark data	String	**NLMDFormatMaxLength**
MaxLength	Max quantity of transferred landmarks	UInt_16	**Range 0 – 60** **Default: 40**

Response

Command Syntax: **sWA NLMDFormatMaxLength**

Command part	Description	Type	Range/Value
Command type	Response (SOPAS write by name)	String	**sWA**
Command		String	**NLMDFormatMaxLength**

Request Read

Command Syntax: **sRN NAVFormatMaxLenght**

Command part	Description	Type	Range/Value
Command type	Request (SOPAS read by name)	String	**sRN**
Command	Read maximum of landmarkdata	String	**NLMDFormatMaxLength**

Response

Command Syntax: **sRA NAVFormatMaxLength**

Command part	Description	Type	Range/Value
Command type	Response (SOPAS read by name)	String	**sRA**
Command	Set maximum of Landmark data	String	**NLMDFormatMaxLength**
Fixedlength	Max quantity of transferred landmarks	UInt_16	**Range 0 – 60** **Default: 40**

2.2 Parameter for the Landmark configuration

2.2.1 Command: Set Reflector Size

This command parameterizes the reflector size.

Request Write

Command syntax: **sWN NLMDReflSize** *size*

Command part	Description	Type	Range/Value
Command type	Request (SOPAS write by name)	String	**sWN**
Command	Set reflector size	String	**NLMDReflSize**
Size	Size of reflector (diameter in case of cylindrical reflectors)	UInt_16	**10 ... 250 mm (Default: 76 mm)**

Response

Command syntax: **sWA NLMDReflSize**

Command part	Description	Type	Range/Value
Command type	Response (SOPAS write by name)	String	**sWA**
Command	Set reflector size	String	**NLMDReflSize**

Request Read

Command syntax: **sRN NLMDReflSize**

Command part	Description	Type	Range/Value
Command type	Request (SOPAS read by name)	String	**sRN**
Command	Read reflector size	String	**NLMDReflSize**

Response

Command syntax: **sRA NLMDReflSize** *size*

Command part	Description	Type	Range/Value
Command type	Response (SOPAS read by name)	String	**sRA**
Command	Read reflector size	String	**NLMDReflSize**
Size	Size of reflector (diameter in case of cylindrical reflectors)	UInt_16	**10... 250 mm** **Default: 76 mm**

2.2.2 Command: Set Reflector Type

This command parameterizes the reflector type (flat or cylindrical).

Request Write

Command syntax: **sWN NLMDReflType** *type*

Command part	Description	Type	Range/Value
Command type	Request (SOPAS write by name)	String	**sWN**
Command	Set reflector type	String	**NLMDReflType**
Type	Type of reflector	Enum_8	**1 = flat** **2 = cylindric (Default)**

Response

Command syntax: **sWA NLMDReflType**

Command part	Description	Type	Range/Value
Command type	Response (SOPAS write by name)	String	**sWA**
Command	Set reflector type	String	**NLMDReflType**

Request Read

Command syntax: **sRN NLMDReflType**

Command part	Description	Type	Range/Value
Command type	Request (SOPAS read by name)	String	**sRN**
Command	Get reflector type	String	**NLMDReflType**

Response

Command syntax: **sRA NLMDReflType** *type*

Command part	Description	Type	Range/Value
Command type	Response (SOPAS read by name)	String	**sRA**
Command	Get reflector type	String	**NLMDReflType**
Type	Type of reflector	Enum_8	**1 = flat** **2 = cylindric**

2.2.3 Command: Set Sector Muting

For situations where the scan area of the NAV245 may be covered, it is possible to mute different angle areas for the landmark detection. Up to four sectors could be activated.

Request Write

Command syntax: **sWN NLMDMutedSectors** *sector0 sector1 sector2 sector3*

Command part		**Description**	**Type**	**Range/Value**
Command type		Request (SOPAS read by name)	String	**sWN**
Command		Configuration of sector muting	String	**NLMDMutedSectors**
4 x sector	AngleFrom	Start angle of the sector	UInt_32	**-90000 ... 270000 mdeg (Default: 0)**
	AngleTo	End angle of the sector	UInt_32	**-90000 ...270000 mdeg (Default: 0)**
	Active	Activation of the sector	Bool_1	**0 = inactive (Default) 1 = active**

Response

Command syntax: **sWA NLMDMutedSectors**

Command part	**Description**	**Type**	**Range/Value**
Command type	Response (SOPAS read by name)	String	**sWA**
Command	Configuration of sector muting	String	**NLMDMutedSectors**

Request Read

Command syntax: **sRN NLMDMutedSectors**

Command part	Description	Type	Range/Value
Command type	Request (SOPAS read by name)	String	**sRN**
Command	Configuration of sector muting	String	**NLMDMutedSectors**

Response

Command syntax: **sRA NLMDMutedSectors** *sector0 sector1 sector2 sector3*

Command part		Description	Type	Range/Value
Command type		Response (SOPAS read by name)	String	**sRA**
Command		Configuration of sector muting	String	**NLMDMutedSectors**
4 x sector	AngleFrom	Start angle of the sector	UInt_32	**-90000 ... 270000 mdeg**
	AngleTo	End angle of the sector	UInt_32	**-90000 ... 270000 mdeg**
	Active	Activation of the sector	Bool_1	**0 = inactive** **1 = active**

2.2.4 Command: Set N Closest Reflectors

This command sets the maximum quantity of closest reflectors for the output. If N = 0, all valid reflectors are chosen. The quantity of “N closest reflectors” may be changed while landmark detection.

Request Write

Command syntax: **sWN NLMDnClosest** *nClosest*

Command part	Description	Type	Range/Value
Command type	Request (SOPAS read by name)	String	**sWN**
Command	Set N closest reflectors	String	**NLMDnClosest**
NClosest	Quantity of N closest reflectors	UInt_8	**0 ... 60** **(Default: 0 = all)**

Response

Command syntax: **sWA NLMDnClosest**

Command part	Description	Type	Range/Value
Command type	Response (SOPAS read by name)	String	**sWA**
Command	Set N closest reflectors	String	**NLMDnClosest**

Request Read

Command syntax: **sRN NLMDnClosest**

Command part	Description	Type	Range/Value
Command type	Request (SOPAS read by name)	String	**sRN**
Command	Readout quantity N closest reflectors	String	**NLMDnClosest**

Response

Command syntax: **sRA NLMDnClosest** *nClosest*

Command part	Description	Type	Range/Value
Command type	Response (SOPAS read by name)	String	**sRA**
Command	Return quantity of N closest reflectors	String	**NLMDnClosest**
NClosest	Quantity of N closest reflectors	UInt_8	**0 ... 60**

2.2.5 Command: Set Action Radius

The action radius defines an area in the surrounding of NAV245. The reflectors outside of the limited area will be ignored.

The parameters *rFr* and *rTo* define this area as a circular ring. With this command the action radius area may be change while measuring and landmark detection.

Default: *rFr* = 450 mm, *rTo* = 50000 mm.

Request Write

Command syntax: **sWN NLMDActionRadius** *rFr rTo*

Command part	Description	Type	Range/Value
Command type	Request (SOPAS read by name)	String	**sWN**
Command	Set Action Radius	String	**NLMDActionRadius**
RFr	Minimum Actions Radius	UInt_32	**500 ... 50000 mm** **(Default: 500)**
RTo	Maximum Action Radius	UInt_32	**500 ... 50000 mm** **(Default: 50000)**

Response

Command syntax: **sWA NLMDActionRadius**

Command part	Description	Type	Range/Value
Command type	Response (SOPAS read by name)	String	**sWA**
Command	Set action radius	String	**NLMDActionRadius**

Request Read

Command syntax: **sRN NLMDActionRadius**

Command part	Description	Type	Range/Value
Command type	Request (SOPAS read by name)	String	**sRN**
Command	Request action radius	String	**NLMDActionRadius**

Response

Command syntax: **sRA NLMDActionRadius** *rFr rTo*

Command part	Description	Type	Range/Value
Command type	Response (SOPAS read by name)	String	**sRA**
Command	Request action radius	String	**NLMDActionRadius**
RFr	Minimum Actions Radius	UInt_32	**500 ... 50000 mm**
RTo	Maximum Action Radius	UInt_32	**500.... 50000 mm**

2.2.6 Command: Set Detection Mode

The Detection Mode defines the filtering of landmarks. The setting may be set during the movement of the AGV according to the local situation.

Optimal (Default)

Only landmarks with ensured high accuracy given out

Standard

More landmarks available, but with possible higher measurement error.

- Filter *PartlyCovered* is active, but with tolerances
- Filter *ReflectiveAlignment* is disabled.

Availability

Highest availability of landmarks, but

- Filter *Partly covered* is disabled.
- Filter *ReflectiveAlignment* is disabled.

Situation of reflective alignment:

Request Write

Command syntax: **sWN NLMDDetectionMode** *Mode*

Command part	Description	Type	Range/Value
Command type	Request (SOPAS read by name)	String	**sWN**
Command	Set Detection Mode	String	**NLMDDetectionMode**
Mode	Detection Mode	Enum_8	**10: Optimal (Default)** **20: Standard** **30: Availability**

Response

Command syntax: **sWA NLMDActionRadius**

Command part	Description	Type	Range/Value
Command type	Response (SOPAS read by name)	String	**sWA**
Command	Detection Mode	String	**NLMDDetectionMode**

Request Read

Command syntax: **sRN NLMDDetectionMode**

Command part	Description	Type	Range/Value
Command type	Request (SOPAS read by name)	String	**sRN**
Command	Request Detection Mode	String	**NLMDDetectionMode**

Response

Command syntax: **sRA NLMDDetectionMode**

Command part	Description	Type	Range/Value
Command type	Response (SOPAS read by name)	String	**sRA**
Command	Request Detection Mode	String	**NLMDDetectionMode**
	Detection Mode	Enum_8	**10: Optimal (Default)** **20: Standard** **30: Availability**

2.3 Landmark data in 2D Measurement Mode

2.3.1 Command: Landmark data

This command activates the data output of the currently recognized landmarks of the NAV245.

The format of the transferred landmark data could be set by the commands *NLMDFormatFixedLength and NLMDFormatMaxLenght. The scanner will answer witin 40 msec*

The contour may requested by the event LMDscandata.

Request

Command Syntax: **sEN NLMDLandmarks**

Command part	Description	Type	Range/Value
Command type	Response (SOPAS method acknowledge by name)	String	**sEN**
Command	Get detected landmarks	String	**NLMDLandmarks**
Measurement	Activate output		**1: output** **0: No output**

Response (Event based answer)

Command Syntax: **sSN NLMDLandmarks**

Command part	Description	Type	Range/Value
Command type	SOPAS answer for events	String	sEA
Command	Event based answer of new landmarks	String	NLMDLandmarks
Version	Version of telegram format	UInt_16	
Errorcode		UInt_8	00: OK 01: UNKNOWNERROR
Timestamp	Timestamp of related polar scan data	UInt_32	0...4.294.967.295µs (0xffffffff)
ScanCount	Number of Scan	UInt_16	
TelegramCount	Telegram number	UInt_16	
Content	Flexarray fixed or depending on the number of detected reflectors	UInt_32	Bit 0x01: Flexarrays with fixed length
Landmarks	Flexarray length	UInt_16	Max Length=Max_FEATURE_NUM (60)
N x Landmark			Max 60x
Timestamp	Timestamp, when Landmark was measured	UInt_32	0...4.294.967.295µs (0xffffffff)
X	Cartesian x	Int_32	0..[mm]
Y	Cartesian y	Int_32	0.. [mm]
Distance	Polar Distance	UInt_32	0..50000[mm]
Angle	Measured angle	Int_32	-90000 ... 270000mdeg
Type	Type of Landmark	UInt_16	Type of Landmark 01: REFLECTOR_FLAT 02:REFLECTOR_CYLINDRICAL
Reserved	Not Used / Reserved	UInt_32	
Reserved	Not Used / Reserved	UInt_32	
Size	Size of Landmark	UInt_16	10...250 mm (Default: 76 mm [mm]
Hits	Counts of hits	UInt_16	
RSSI	Measn RSSI values	UInt_16	0....4096 [digit]
Reserved	Reserved	UInt_32	
Start	Index of related Scandata, where Landmarks begins	UInt_16	
Stop	Index of related Scandata, where Landmarks ends	UInt_16	

3 General Commands

3.1 Device Information

3.1.1 Command: Read Device Identification

Request Read

Command Syntax: **sRN DeviceIdent**

Command part	Description	Type	Range/Value
Command type	Request (SOPAS read by name)	String	**sRN**
Command	Read Device identification	String	**DeviceIdent**

Response

Command Syntax: **sRA DeviceIdent** *sizeName name sizeVersion version*

Command part	Description	Type	Range/Value
Command type	Response (SOPAS read by name)	String	**sRA**
Command	Read Device identification	String	**DeviceIdent**
SizeName	Length of name string	UInt_8	**0 … 255**
Name	Device name	String	**…**
SizeVersion	Length of version string	UInt_8	**0 … 255**
Version	SOPAS-Device version	String	**…**

3.1.2 Command: Read Serial Number

Request Read

Command Syntax: **sRN SerialNumber**

Command part	Description	Type	Range/Value
Command type	Request (SOPAS read by name)	String	**sRN**
Command	Read serial number	String	**SerialNumber**

Response

Command Syntax: **sRA SerialNumber** *sizeNumber serialNumber*

Command part	Description	Type	Range/Value
Command type	Response (SOPAS read by name)	String	**sRA**
Command	Read Serial number	String	**SerialNumber**
SizeNumber	Length serial number string	UInt_8	**0 ... 16**
SerialNumber	Serial number	String	**...**

3.1.3 Command: Read Application Version

Request Read

Command Syntax: **sRN FirmwareVersion**

Command part	Description	Type	Range/Value
Command type	Request (SOPAS read by name)	String	**sRN**
Command	Read Application software version	String	**FirmwareVersion**

Response

Command Syntax: **sRA FirmwareVersion** *sizeVersion version*

Command part	Description	Type	Range/Value
Command type	Response (SOPAS read by name)	String	**sRA**
Command	Application software version	String	**FirmwareVersion**
SizeVersion	Length of Application version string	UInt_8	**0 … 255**
Version	Application software version	String	**…**

3.2 Commands with Method Call

Calling a method by command, instructs the NAV245 to process data or measurements. Methods can be called including data to be processed. The method responds with a structure of one or more return values. Whether a method can be called depends on the user level.

The NAV245 processes synchronous methods:

Synchronous commands **sMN** block the NAV245 until the results of the method are returned by **sAN**.

3.3 Procedure of Method Call

3.3.1 Command: Set User Level

Selects a user level and transfers its password to the NAV245. The password is encoded (Hash Value)

User level	Password	Hash-Value
Operator	main	**B21ACE26**
Authorized client	client	**F4724744**
Service	servicelevel	**81BE23AA**

Request

Command Syntax: **sMN SetAccessMode** *newMode password*

Command part	Description	Type	Range/Value
Command type	Request (SOPAS method by name)	String	**sMN**
Command	Set user level	String	**SetAccessMode**
NewModel	User level code	Int_8	**2 = operator** **3 = authorized client** **4 = service**
Password	Password Hash Value	UInt_32	**00000000 ... FFFFFFF**

Response

Command Syntax: **sAN SetAccessMode** *success*

Command part	**Description**	**Type**	**Range/Value**
Command type	Request (SOPAS answer)	String	**sAN**
Command	Set user level	String	**SetAccessMode**
Success	Mode successfully set	Bool	**0 = no** **1 = yes**

3.3.2 Command: Store Data Permanent

This command stores all settings of the NAV245 permanent, to be recalled after power on.

Request

Command Syntax: **sMN mEEwriteall**

Command part	Description	Type	Range/Value
Command type	Request (SOPAS method by name)	String	**sMN**
Command	Store data permanent	String	**mEEwriteall**

Response

Command Syntax: **sAN mEEwriteall** *success*

Command part	Description	Type	Range/Value
Command type	Response (SOPAS answer)	String	**sAN**
Command	Store data permanent	String	**mEEwriteall**
Success	Shows if the command was successful.	Bool	**0 = no** **1 = yes**

3.3.3 Command: Synchronize Timestamp

This command returns the internal timestamp of the NAV245 to synchronize it with the vehicle controller's clock. The NAV245 clock is a free running *UInt_32* counter with 1µs increment. While reading the timestamp, a pulse is generated on the digital output to compensate the transmission time, if necessary.

Request

Command Syntax: **sMN GetSyncTimestamp**

Command part	Description	Type	Range/Value
Command type	Request (SOPAS method by name)	String	**sMN**
Command	Sync Timestamp	String	**GetSyncTimestamp**

Response

Command Syntax: **sAN GetSyncTimestamp** *errorCode timestamp*

Command part	Description	Type	Range/Value
Command type	Response (SOPAS answer)	String	**sAN**
Command	Sync Timestamp	String	**GetSyncTimestamp**
ErrorCode	Error number	Enum_8	**0 = no error** **7 = general error**
Timestamp	Internal NAV245 timestamp	UInt_32	**0...4.294.967.295 µs (0xffffffff)**

3.3.4 Command: Device Reset

Method shuts the device down but saves the parameter before shutdown ist executed

Request

Command Syntax: **sMN mSCsoftreset**

Command part	Description	Type	Range/Value
Command type	Request (SOPAS method by name)	String	**sMN**
Command	Device reset	String	**mSCsoftreset**

Response

Command Syntax: **sAN mSCsoftreset** *errorCode*

Command part	Description	Type	Range/Value
Command type	Response (SOPAS method by name)	String	**sAN**
Command	Device reset	String	**mSCsoftreset**

4 Result Port Protocol

4.1 Introduction of the Result Port Protocol

4.1.1 General

The result port telegram has been introduced in the NAV245 series starting from the firmware version V1.12.

The telegrams are optimized for a short transmission time and the data structure is are arranged for a memcopy that means to copy the data from the receiving buffer to the main memory for the further processing.

4.1.2 Usage of result port

For parametrization and request of data either the IP ports 2111 or 2112 could be used.
IP Port 2201 is used for unidirectional data output streaming.

Port 2111 SOPAS CoLaA-Protocol
Port 2112 SOPAS CoLaA/B-Protocol
Port 2113 future use SOPAS
Port 2201 ResultPort

The result port telegrams consist of a header structure followed by specific data. The payload type field indicates the transmitted result. Available are results of raw scan data, reflector detection and localization.

Each result could be enabled separately.

4.1.3 Result Port Payload Types

Result Payload Type	ID	Description
Scan Data	0x0101	Raw scan data (distance, angle, remission)
Reflector Detection	0x0601	Result of reflector detection (Available in Mode LMDetection)
Scandata (Little Endian)	0x0181	Raw scan data in little endian format
Reflector Detection (Little Endian)	0x0681	Result of reflector detection in little endian format (Available in Mode LMDetection)

These combinations are possible (either in Big Endian or Little Endian Format) :

LANDMARK Detection Mode

- Scan Data only
- Reflector Detection only
- Scan Data + Reflector Detection

4.2 Framing

The Result Protocol consists of an unique header structure and telegram specific content. The telegram content is determined by the field *PayloadType*.

Section	Fields		Description	Type	Size [Byte]	Total Size
HEADER	MagicWord	"SICK"	Magic word (0x53 0x49 0x43 0x4B)	4x UInt8	4	
	Length		Length of telegram. Header, payload and Trailer.	UInt32	4	
	PayloadType		Payload datatype (see table 2.2)	UInt16	2	
	PayloadVersion		Version of PayloadType structure.	UInt16	2	
	OrderNumber		Order number of device	UInt32	4	
	SerialNumber		Serial of device	UInt32	4	
	FW-Version		Softwareversion of device	20x UInt8	20	
	TelegramCounter		Telegram counter since power on.	UInt32	4	
	SystemTime		System time of related scan data.	NTP	8	52

Section	Fields		Description	Type	Size [Byte]	Total Size
Trailer	Checksum		CRC16 over Length without 2 bytes of trailer	UInt16	2	2

4.2.1 Checksum Calculation

Current implementation is using the standard EDP1 / EDP2 CRC16 calculation.

CRC-CCITT

Polynom 0x1021 (without leading 1: x16 + x12 + x5 + 1)

Start value 0xffff

4.2.2 Result of Landmark Detection

By parameter *ER1FctLMDetectFixedLength* the number of transmitted landmarks is either variable according to the number of seen landmarks or fixed.

Parameter *ER1FctLMDetectMaxLength* defines the maximum number of transmitted landmarks.

If enabling *ER1FctLMDetectFixedLength,* empty landmark positions will be filled with a zero value up to *ER1FctLMDetectMaxLength.*

Section	Fields		Description	Type	Size [Byte]	Total Size
LANDMARKS	ErrorCode		0: OK 1: UNKNOWNERROR	UInt16	2	
	ScanCounter		Counter of related scan data	UInt32	4	
	Content		Flag pattern: 0x01=Output with fixed length	UInt32	4	
	LandmarkNum		Number of landmark data	UInt16	2	
	0 – 60 1. Maximum length is configurable. Default is 40. 2. Telegram is filled with zero, if number of seen landmarks is smaller than maximum length.	Timestamp	Timestamp of Reflector	UInt32	4	
		X	Cartesian coordinates	Int32	4	
		Y		Int32	4	
		Distance	Polar coordinates	UInt32	4	
		Angle		Int32	4	
		Type	Type of Landmark	UInt16	2	
		ID	Global ID of Landmark in Navigation Mode	UInt32	4	
		Reserved		UInt32	4	
		Size	Size	UInt16	2	
		Hit count	Number of hits	UInt16	2	
		RSSI	Mean echo amplitude	UInt16	2	
		Reserved		UInt32	4	
		Index begin	Start index of the landmark in related scandata	UInt16	2	
		Index end	End index	UInt16	2	12+N*44 = **2656**

4.2.3 Scandata Result

Section	Fields		Description	Type	Size [Byte]	Total Size
Header	ErrorCode		0: OK 1: UNKNOWNERROR	UInt16	2	
	ScanCounter		Number of scans	UInt32	4	
	Timestamp		Timestamp of Scandata	UInt32	4	
	DeviceState		Device status	UInt16	2	
	ScanFreq		Scan frequency	UInt32	4	16
	ChannelNum	32bit channnels	Amount of 32Bit channels, always 0.	Uint16	2	2
	ChannelNum	16bit channnels	Amount of 16Bit channels	Uint16	2	2
16 bit Channel	Channel Header	Content	Content: Dist1; RSSI1	6x UInt8	6	
		ScaleFactor	Scale factor	Float32	4	
		ScaleOffset	Scale factor offset	Float32	4	
		StartAngle	Start angle of scan (1/10.000°)	UInt32	4	
		Steps	Angle res. (1/10.000°)	UInt16	2	
		ScanPoints	Number of scan points	UInt16	2	22
	1082x	ScanData	Data : Distance or RSSI with 16bit	Int16	2	2164
						4392

4.2.4 Second Pulse Channel

The NAV24x device is able to evaluate a second pulse on each laser beam.
By enabling the Filter “N To 1”, the application uses the second measurement instead of the first. Also this filtered scan data channel is provided by the result port.

4.3 Combination of telegrams

A combined telegram has the following structure:

Telegram	Section	Size [Byte]	Total Size
1	HEADER	52	
	SCAN	4392	
	TRAILER	2	
2	HEADER	52	
	LANDMARKS	16+N*44 = 2656	
	TRAILER	2	
			7156

4.4 Parametrization and Visualization

4.4.1 Parametrization

Parametrization is done in SOPAS CoLa protocol or SOPAS ET GUI. The Result Port itself is mentioned as unidirectional data output stream.

Available Parameters:

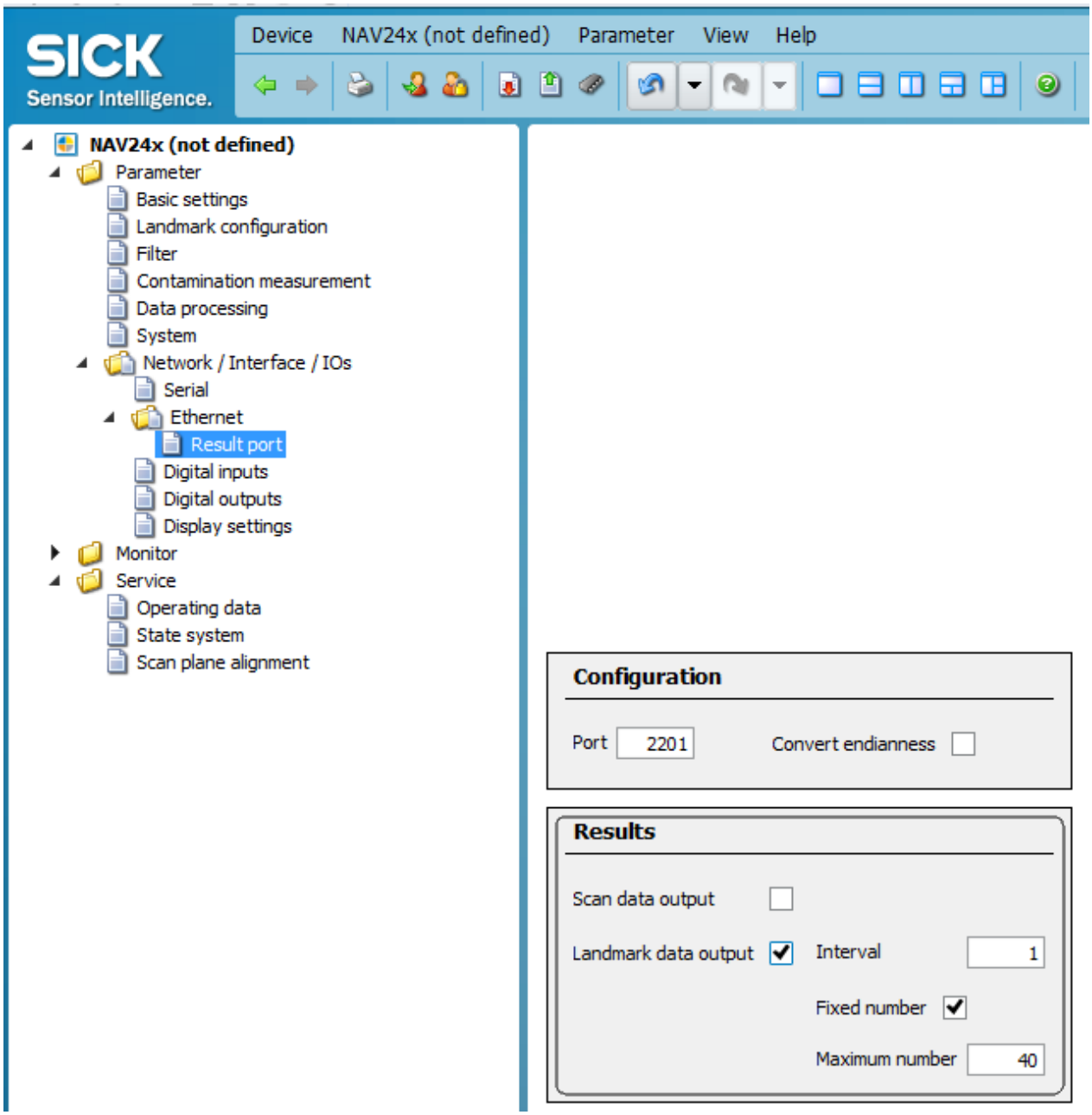
Name	Type	Description
RS1Port	UInt16 (default 2201)	Port for TCP/IP communication of EtherResult port 1
ER1Request	UInt16 (default 0xffff)	Requests specific number for all activated results on EtherResult port 1. System decrements parameter for each result processing cycle independently from the specific Interval parameter. It is based on internal scan data processing. Default value of 0xffff means unlimited number of results.
ER1RequestConvertEndianness	Bool (default false)	Switches all enabled results to their converted equivalent. (Not the header)
ER1CombineTelegrams	Bool (default false)	Causes a combination of all active results in one telegram
ER1FctScanEn	Bool (default false)	Activation of Scan Data EtherResult Port 1
ER1FctScanInterval	UInt16 (default 1)	Interval for output of result messages (1 = each scan is sent out, 2 = each second scan is sent out, ...)
ER1FctLMDetectEn	Bool (default false)	Activation of Landmark Detection EtherResult Port 1
ER1FctLMDetectInterval	UInt16 (default 1)	Interval for output of result messages (1 = each scan is sent out, 2 = each second scan is sent out, ...)
ER1FctLMDetectFixedLength	Bool (default true)	A fixed number of landmarks are transmitted. If fewer landmarks detected, zero values are transmitted.
ER1FctLMDetectMaxLength	UInt16 (default 40, max 60)	Limitation of transmitted landmark number.

4.5 Visualization in SOPAS

The configuration could be done by Cola Command or by SOPAS.

In the menu Ethernet -> Result port are the options to select the format to Big Endian (MSB first / default) or Little Endian (LSB first) by “Convert endianness” and to select the output of Scan data and / or Landmark data.

If the output of the Landmark configuration data is selected the Landmark Dataformat maybe defined the maximum number of landmarks that will be given out.



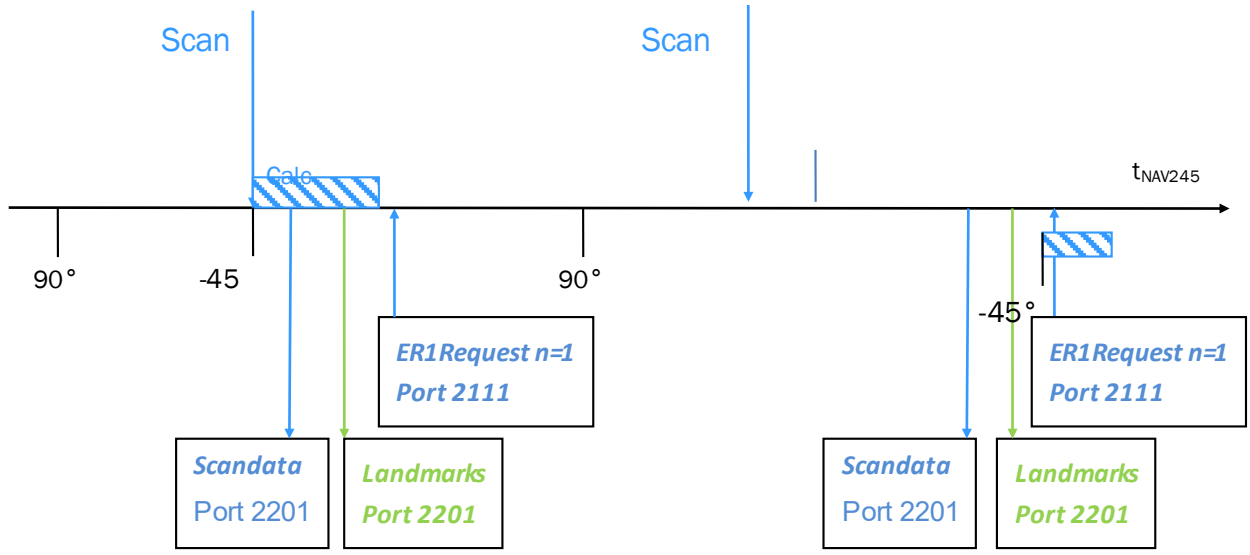
4.6 Usage Example

4.6.1 Separate transmission of Scandata and Landmarks

This example shows how to use the Result Port.

Activation of the results for raw scan data and landmark detection by setting *ER1FctScanEn* and *ER1FctLMDetectEn* to true.

By sending *ER1Request n* via IP port 2111, the scanner will transmit n data results after processing via port 2201.
The scanner will transmit landmark data and scan data approximately every 40 ms.

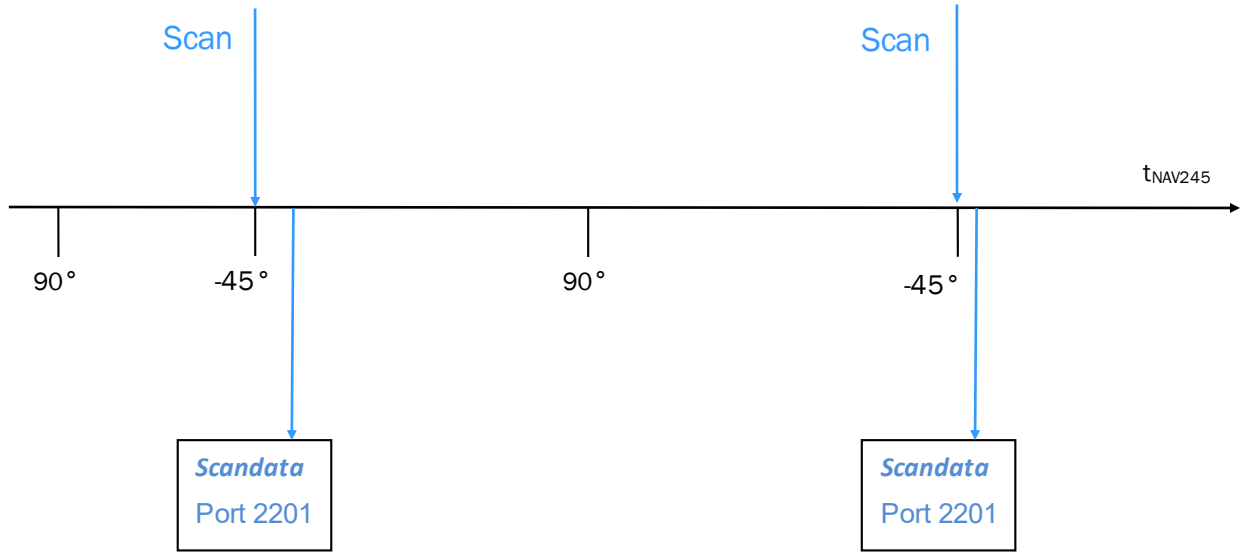


4.6.2 Combined Telegram with one selected result only

After activation of *ER1CombineTelegrams* results will be combined in a single telegram as follows:

If *ER1CombineTelegrams* is active but only one result is active, the parameter has no effect.

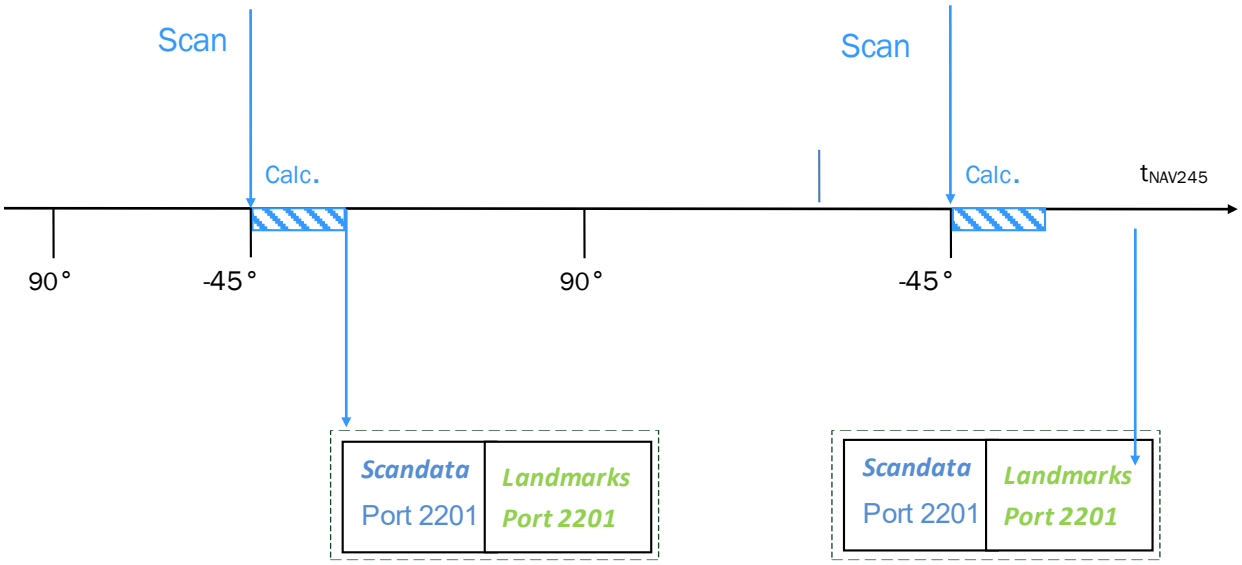
Parameter	Value
ER1Request	0xffff
ER1RequestConvertEndianness	0 (false)
ER1CombineTelegrams	1 (true)
ER1FctScanEn	1 (true)
ER1FctScanInterval	1



4.6.3 Combined Telegram with two selected results

If *ER1CombineTelegrams* and more than one result are active, all results will be sent in a single telegram once they are available (in this case scan data and landmarks are available after calculation time).

Parameter	Value
ER1Request	0xffff
ER1RequestConvertEndianness	0 (false)
ER1CombineTelegrams	1 (true)
ER1FctScanEn	1 (true)
ER1FctScanInterval	1
ER1FctLMDetectEn	1 (true)
ER1FctLMDetectInterval	1

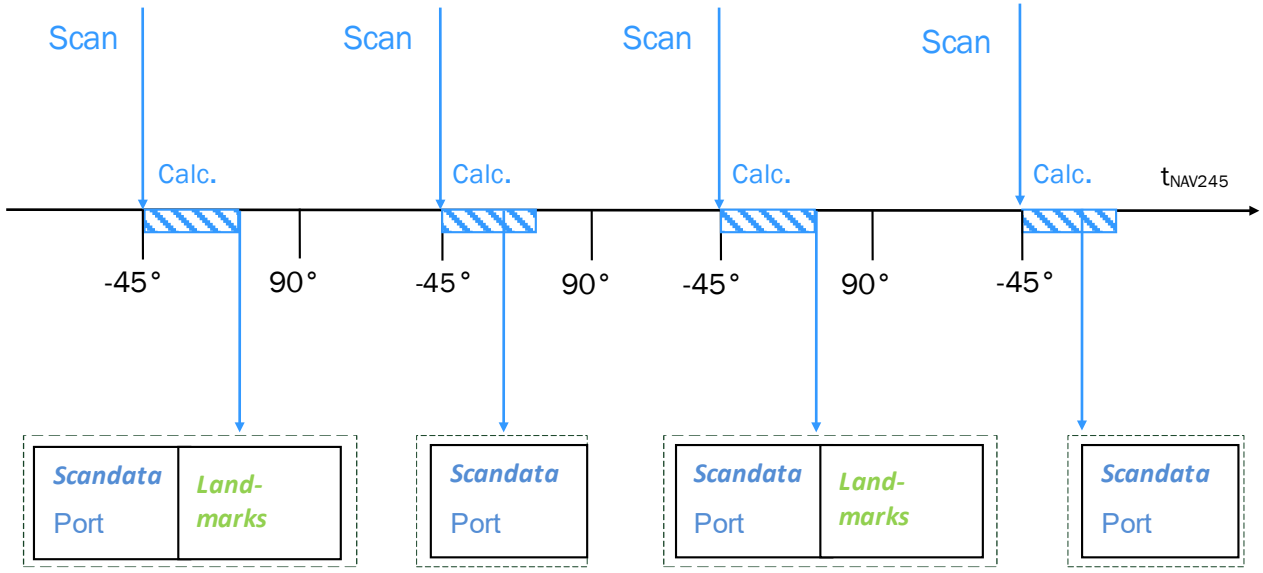


4.6.4 Combined Telegram with two selected results and individual output intervals

If *ER1CombineTelegrams* and more than one result are active and the interval parametrization of the results differ from each other, the behavior is as follows.

In this example the interval of landmarks is set to 2 and the interval of scans is set to 1 instead.

Parameter	Value
ER1Request	0xffff
ER1RequestConvertEndianness	0 (false)
ER1CombineTelegrams	1 (true)
ER1FctScanEn	1 (true)
ER1FctScanInterval	1
ER1FctLMDetectEn	1 (true)
ER1FctLMDetectInterval	2



8018823/15W6/en/2019-07-17

Australia
Phone +61 3 9457 0600
1800 334 802 – toll free E-Mail
sales@sick.com.au

Austria
Phone +43 22 36 62 28 8-0
E-Mail office@sick.at

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

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

Canada
Phone +1 905 771 1444
E-Mail information@sick.com

Czech Republic
Phone +420 2 57 91 18 50
E-Mail sick@sick.cz

Chile
Phone +56 2 2274 7430
E-Mail info@schadler.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 +358-9-2515 800
E-Mail sick@sick.fi

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

Germany
Phone +49 211 5301-301
E-Mail info@sick.de

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

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

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

Israel
Phone +972 4 6881000
E-Mail info@sick-sensors.com

Italy
Phone +39 (0)2 27 43 41
E-Mail info@sick.it

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

Malaysia
Phone +6 03 8080 7425
E-Mail enquiry.my@sick.com

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

Netherlands
Phone +31 30 2044 000
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 +40 356 171 120
E-Mail office@sick.ro

Russia
Phone +7 495 775 05 30
E-Mail info@sick.ru

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

Slovakia
Phone +421 482 901201
E-Mail mail@sick-sk.sk

Slovenia
Phone +386 591 788 49
E-Mail office@sick.si

South Africa
Phone +27 11 472 3733
E-Mail info@sickautomation.co.za

South Korea
Phone +82 2 786 6321
E-Mail info@sickkorea.net

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 2645 0009
E-Mail Ronnie.Lim@sick.com

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

United Arab Emirates
Phone +971 (0) 4 8865 878
E-Mail info@sick.ae

United Kingdom
Phone +44 1727 831121
E-Mail info@sick.co.uk

USA
Phone +1 800 325 7425
E-Mail info@sick.com

Vietnam
Phone +84 945452999
E-Mail Ngo.Duy.Linh@sick.com

Further locations at **www.sick.com**

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

SICK
Sensor Intelligence.