Serial Radar SDK - Read Me

Devices: *iSYS-400x, iSYS-600x, iSYS-6203, IDS-1000, iSYS-5010, iSYS-5011, iSYS-5020, iSYS-5021, iSYS-5110*

Revision: *17*

Date: *07.01.2019*

Table of content

1. What is the Serial Radar SDK? 6
2. General Information 7
3. Get Radar API running in your own application 8
4. Programming Example with Serial Radar API 9
5. Used types in Serial Radar API 10
   - 5.1. Default data types 10
   - 5.2. Used structures and enumerators 10
     - 5.2.1. struct iSYSTargetList_t 10
     - 5.2.2. struct iSYSTarget_t 11
     - 5.2.3. union iSYSTargetListError_u 12
     - 5.2.4. struct iSYSRawData_t 12
     - 5.2.5. struct iSYSIQSignal_t 13
     - 5.2.6. struct iSYSComplex_t 13
     - 5.2.7. struct iSYSDetection_t 14
     - 5.2.8. struct iSYSDetectionData_t 14
     - 5.2.9. struct iSYSRangeList_t (iSYS-600X only) 15
     - 5.2.10. enum iSYSBaudrate_t 16
     - 5.2.11. enum iSYSOutput_t 16
     - 5.2.12. enum iSYSOutputnumber_t 16
     - 5.2.13. enum iSYSDirection_type_t 17
     - 5.2.14. enum iSYSVelocity_unit_t 17
     - 5.2.15. enum iSYSTemperature_unit_t 17
     - 5.2.16. enum iSYSOutput_filter_t 17
     - 5.2.17. enum iSYSFilter_signal_t 18
     - 5.2.18. enum iSYSFrequencyChannel_t 18
     - 5.2.19. enum iSYSResult_t 19
     - 5.2.20. enum iSYSTargetListError_t 20
     - 5.2.21. enum IDSDirection_type_t 20
     - 5.2.22. enum iSYSSaveLocation_t 20
     - 5.2.23. enum iSYSMeasurementMode_t 21
     - 5.2.24. enum iSYSWarningMode_t 21
     - 5.2.25. enum iSYSTargetListInterface_t 21
     - 5.2.26. enum iSYSLedConfiguration_t 21
6. Radar API Function Description 22
   - 6.1. Common Functions 22
     - 6.1.1. iSYS_initComPort (…) 22
     - 6.1.2. iSYS_initSystem (…) 22
     - 6.1.3. iSYS_exitSystem (…) 23
     - 6.1.4. iSYS_exitComPort (…) 23
     - 6.1.5. iSYS_getApiVersion (…) 23
     - 6.1.6. iSYS_ReadDeviceName (…) 23
     - 6.1.7. iSYS_StartAcquisition (…) 23
     - 6.1.8. iSYS_StopAcquisition (…) 23
     - 6.1.9. iSYS_getDeviceAddress/iSYS_setDeviceAddress (…) 24

6.1.10. iSYS_getFrequencyChannel/iSYS_setFrequencyChannel (…) 24
6.1.11. iSYS_getThresholdMin/iSYS_setThresholdMin (…) 25
6.1.12. iSYS_getThresholdSensitivityLeft/iSYS_setThresholdSensitivityLeft (…) 25
6.1.13. iSYS_getThresholdSensitivityRight/iSYS_setThresholdSensitivityRight (…) 25
6.1.14. iSYS_getLedConfiguration/ iSYS_setLedConfiguration (…) 26
6.1.15. iSYS_getOutputAngleMin/iSYS_setOutputAngleMin (…) 26
6.1.16. iSYS_getOutputAngleMax/iSYS_setOutputAngleMax (…) 26
6.1.17. iSYS_getOutputRangeMin/iSYS_setOutputRangeMin (…) 27
6.1.18. iSYS_getOutputRangeMax/iSYS_setOutputRangeMax (…) 27
6.1.19. iSYS_getOutputSignalMin/iSYS_setOutputSignalMin (…) 27
6.1.20. iSYS_getOutputSignalMax/iSYS_setOutputSignalMax (…) 28
6.1.21. iSYS_getOutputVelocityMin/iSYS_setOutputVelocityMin (…) 29
6.1.22. iSYS_getOutputVelocityMax/iSYS_setOutputVelocityMax (…) 29
6.1.23. iSYS_getOutputDirection/iSYS_setOutputDirection (…) 29
6.1.24. iSYS_getOutputFilter/iSYS_setOutputFilter (…) 30
6.1.25. iSYS_getOutputSignalFilter/iSYS_setOutputSignalFilter (…) 30
6.1.26. iSYS_getOutputAlphaFilterVelocity/iSYS_getOutputAlphaFilterVelocity (…) 31
6.1.27. iSYS_getOutputAlphaFilterRange/iSYS_getOutputAlphaFilterRange (…) 31
6.1.28. iSYS_getLineFilter1/iSYS_setLineFilter1 (…) 31
6.1.29. iSYS_getLineFilter2/iSYS_setLineFilter2 (…) 32
6.1.30. iSYS_getOutputEnable/iSYS_setOutputEnable (…) 32
6.1.31. iSYS_getOutputRisingDelay/iSYS_setOutputRisingDelay (…) 33
6.1.32. iSYS_getOutputFallingDelay/iSYS_setOutputFallingDelay (…) 33
6.1.33. iSYS_getOutputPlausibility/iSYS_setOutputPlausibility (…) 33
6.1.34. iSYS_getOutputSetting1/iSYS_setOutputSetting1 (…) 34
6.1.35. iSYS_getOutputSetting2/iSYS_setOutputSetting2 (…) 34
6.1.36. iSYS_getPotis/iSYS_setPotis (…) 35
6.1.37. iSYS_getTargetList (…) 35
6.1.38. iSYS_getTargetList16 (…) 35
6.1.39. iSYS_getTargetList32 (…) 35
6.1.40. iSYS_getDigitalOutputState (…) 36
6.1.41. iSYS_getRawData (…) 36
6.1.42. iSYS_getFftData (…) 36
6.1.43. iSYS_setFactorySetting (…) 36
6.1.44. iSYS_saveSensorSettings (…) 36
6.1.45. iSYS_saveApplicationSettings (…) 37
6.1.46. iSYS_saveSensorAndApplicationSettings (…) 37
6.1.47. iSYS_getDspHardwareVersion (…) 37
6.1.48. iSYS_getRfeHardwareVersion (…) 37
6.1.49. iSYS_getProductInfo (…) 37
6.2. iSYS-6XXX functions 38

6.2.1. iSYS_getRangeList (…) 38
6.2.2. iSYS_getMeasurementMode/iSYS_setMeasurementMode (…) 38
6.3. iSYS-6203 functions 39
6.3.1. iSYS_SetDefaultTemperatureThreshold (…) 39
6.3.2. iSYS_getTemperatureThreshold (…) 39
6.3.3. iSYS_setTemperatureThreshold (…) 39
6.3.4. iSYS_setMountingOffsetValue (…) 39
6.3.5. iSYS_getMountingOffsetValue (…) 40
6.3.6. iSYS_setRangeTemperatureWarningThresholdSwitch (…) 40
6.3.7. iSYS_getRangeTemperatureWarningThresholdSwitch (…) 40
6.3.8. iSYS_setRangeWarningThreshold (…) 40
6.3.9. iSYS_getRangeWarningThreshold (…) 41
6.3.10. iSYS_setTemperatureWarningThreshold (…) 41
6.3.11. iSYS_getTemperatureWarningThreshold (…) 41
6.3.12. iSYS_setOutputRangeMinExt (…) 41
6.3.13. iSYS_getOutputRangeMinExt (…) 42
6.3.14. iSYS_setOutputRangeMaxExt (…) 42
6.3.15. iSYS_getOutputRangeMaxExt (…) 42
6.4. IDS-1000 functions 42
6.4.1. IDS_getOutputEnable/IDS_setOutputEnable (…) 42
6.4.2. IDS_getFrequency/IDS_setFrequency (…) 43
6.4.3. IDS_getDirection/IDS_setDirection (…) 43
6.4.4. IDS_saveSettings (…) 43
6.5. iSYS-5010 functions 44
6.5.1. iSYS_getTargetClusteringEnable/iSYS_setTargetClusteringEnable (…) 44
6.5.2. iSYS_getRcsOutputEnable/iSYS_setRcsOutputEnable (…) 44
6.5.3. iSYS_getThresholdMovingTargetsNearRangeMargin/ iSYS_setThresholdMovingTargetsNearRangeMargin (…) 44
6.5.4. iSYS_getThresholdMovingTargetsMainRangeMargin/ iSYS_setThresholdMovingTargetsMainRangeMargin (…) 45
6.5.5. iSYS_getThresholdMovingTargetsLongRangeMargin/ iSYS_setThresholdMovingTargetsLongRangeMargin (…) 45
6.1. iSYS-5020 functions 46
6.1.1. iSYS_getThresholdMovingTargetsNearRangeMargin/ iSYS_setThresholdMovingTargetsNearRangeMargin (…) 46
6.1.2. iSYS_getThresholdMovingTargetsMainRangeMargin/ iSYS_setThresholdMovingTargetsMainRangeMargin (…) 46
6.1.3. iSYS_getThresholdMovingTargetsLongRangeMargin/ iSYS_setThresholdMovingTargetsLongRangeMargin (…) 47
6.1.4. iSYS_getIPConfig/iSYS_setIPConfig(…) 47
6.1.5. iSYS_getIPDestination/iSYS_setIPDestination(…) 47
6.1.6. iSYS_getTargetListInterface/ iSYS_setTargetListInterface (…) 48
6.2. iSYS-5110 functions 49
6.2.1. iSYS_getIPConfig/iSYS_setIPConfig(…) 49
6.2.2. iSYS_getIPDestination/iSYS_setIPDestination(…) 49
6.2.3. iSYS_setStopBarEnable/iSYS_getStopBarEnable(…) 50

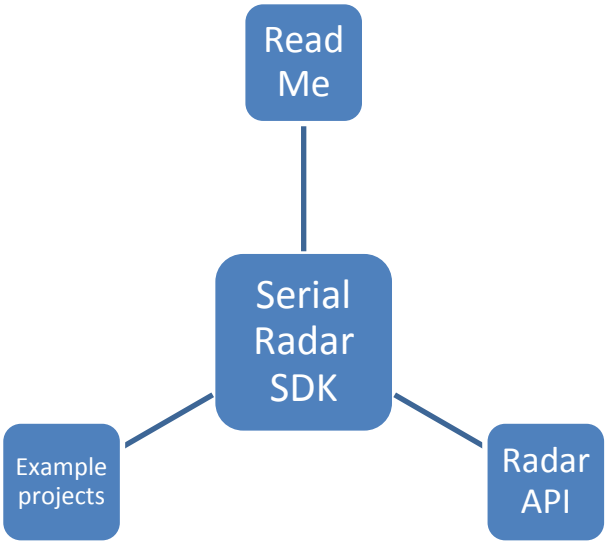
6.2.4. iSYS_setStopBarRange/iSYS_getStopBarRange(…)....50
6.2.5. iSYS_setStopBarAngle/iSYS_getStopBarAngle(…) 51

History

Document revision	Date	Change log
1	2016-01-27	Initial Release
2	2016-02-18	enum iSYSFrequencyChannel updated
3	2016-02-23	Added more information's about units of measurement and value limits.
4	2016-03-29	Added functions for IDS-1000 Added enum IDSDirection_type
5	2016-04-11	Added iSYS-6203
6	2016-04-12	Added new function and Enum Fixed some wrong definitons
7	2016-11-28	Added functions for iSYS-5010
8	2017-03-16	Added functions for iSYS-5110
9	2017-03-28	Adapted programming example chapter
10	2017-05-09	Added Baud Rate 230400 to enum iSYSBaudrate_t Added example for iSYS-6005 to programming example chapter
11	2017-10-28	Added functions for iSYS-6203
12	2017-11-30	Added functions for iSYS-5010
13	2017-12-20	Added iSYS-5020
14	2018-01-09	Added functions for iSYS-5110
15	2018-05-08	Added functions for iSYS-5020
16	2018-11-22	Added functions for iSYS-5011 and iSYS-5021
17	2019-01-07	Added function for switching on/off iSYS-400x LED

1. What is the Serial Radar SDK?

The Serial Radar SDK includes the following components:



- Read me
  Description of the following components "How to use the Serial Radar SDK?"
- Serial Radar API
  DLL and header file for easy access to system functionality
- *Example projects for an easy start*

2. General Information

- The Serial Radar API was built on MS Windows 7 32-bit system with *Microsoft Visual C++ Compiler 10.0 (x86) and MINGW 4.8*
- The Serial Radar API was tested with MS Windows 7 32-bit
- The Serial Radar API is incompatible with MS Windows XP, Vista and older MS Windows versions
- To guaranty 8 Byte alignment of defined structures the union type is used
- The Serial Radar API is designed for synchronal use only (blocking application)
- Do not modify delivered Radar API files
- Use the Radar API specific data types defined in serial_radarSDK_basicTypes.h (see 5)

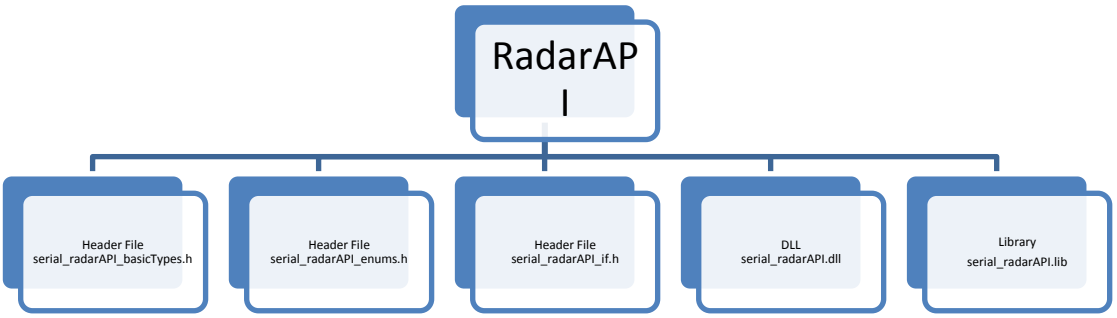
The Serial Radar SDK is designed for usage with all iSYS systems with serial interface (iSYS-4XXX, iSYS-5XXX, iSYS-6XXX, and IDS1000). But some commands works only with specific devices.

The documentation of radarAPI functions is structured in:

- common functions
- iSYS-6XXX functions
- iSYS-6203 functions
- IDS-1000 functions
- iSYS-5010 functions
- iSYS-5011 functions
- iSYS-5020 functions
- iSYS-5021 functions
- iSYS-5110 functions

3. Get Radar API running in your own application

The Serial Radar API consists of the following components:



- serial_radarAPI_basicTypes.h
  Contains all basic type definitions used by the Serial Radar API

  (copy this into your project folder)
- serial_radarAPI_enums.h
  Contains all enum definitions used by the Serial Radar API

  (copy this into your project folder)
- serial_radarAPI_if.h
  Contains all necessary definitions, type definitions, structures and commands featured by the Serial Radar API

  (copy this into your project folder)
- serial_radarAPI.dll
  Library file which contains all functions referred in the header file

  (copy this into the folder where your executable is located)
- serial_radarAPI.lib

  *Links against *.dll functions*

  (copy this into your project folder)

Each callable function (from Serial Radar API) has a return value that inherits information about the status of the function. This return value is of type *iSYSResult_t.* If *iSYSResult_t* is ERR_OK the command was successfully completed. Otherwise an error occurred during operation.
See also delivered example projects how to use radar API.

4. Programming Example with Serial Radar API

In the delivered software package are following examples included.

- *serial_radarAPI_example_DOTNET_x86*

  Example project how to use the serial radar API in .Net.
  It supports only iSYS-4xxx and iSYS-6xxx.

- *serial_radarAPI_example_Qt_x86*

  Example project how to use the serial radar API in Qt (C++).
  It supports only iSYS-4xxx and iSYS-6xxx.

- serial_radarAPI_iSYS-4001_example_VC10_x86

  Example project how to use the serial radar API in Visual Studio 2010 (C++). The project includes functions supported by iSYS-4001.

- *serial_radarAPI_iSYS-5010_example_VC10_x86*

  Example project how to use the serial radar API in Visual Studio 2010 (C++). The project includes all functions supported by iSYS-5010.

- *serial_radarAPI_iSYS-5020_example_VC10_x86*

  Example project how to use the serial radar API in Visual Studio 2010 (C++). The project includes all functions supported by iSYS-5020.

- *serial_radarAPI_iSYS-5110_example_VC10_x86*

  Example project how to use the serial radar API in Visual Studio 2010 (C++). The project includes all functions supported by iSYS-5110.

- serial_radarAPI_iSYS-6003_example_VC10_x86

  Example project how to use the serial radar API in Visual Studio 2010 (C++). The project includes only function for iSYS-6003.

- serial_radarAPI_iSYS-6004_example_VC10_x86

  Example project how to use the serial radar API in Visual Studio 2010 (C++). The project includes only function for iSYS-6004.

- serial_radarAPI_iSYS-6005_example_VC10_x86

  Example project how to use the serial radar API in Visual Studio 2010 (C++). The project includes only function for iSYS-6005.

- serial_radarAPI_iSYS-6006_example_VC10_x86

  Example project how to use the serial radar API in Visual Studio 2010 (C++). The project includes only function for iSYS-6006.

5. Used types in Serial Radar API

In the Serial Radar API specific data types are used. You will find them in the *serial_radarAPI_basicTypes.h* and *serial_radarAPI_if.h* file.

5.1. Default data types

In *radar_API_basicTypes.h* are data types defined for the Serial Radar API. For full compatibility use these data types as well.

Data Type	Definition
`bool_t`	1 bit boolean
`float32_t`	32 bit floating point
`sint8_t / uint8_t`	8 bit signed / unsigned integer
`sint16_t / uint16_t`	16 bit signed / unsigned integer
`sint32_t / uint32_t`	32 bit signed / unsigned integer

5.2. Used structures and enumerators

There are structures and enumerators used in Radar API. These are defined in radarAPI_if.h and make function call easier for the user. Please use these types as well.

Note:	To avoid alignment errors some enumerators are also defined as union and the union type is used.

5.2.1. `struct iSYSTargetList_t`

The *iSYSTargetList_t* inherits the information about all detections made from the running system.

```
typedef struct iSYSTargetList {
    union iSYSTargetListError_u error;
    uint8_t outputNumber;
    uint16_t nrOfTargets;
    uint32_t clippingFlag;
    iSYSTarget_t targets[MAX_TARGETS];
} iSYSTargetList_t;
```

- *iSYSTargetListError_u error*

Inherits information about the list status. (see 5.2.3)

- *uint8_t outputNumber*

Information about the selected output number

- *uint16_t nrOfTargets*

Information about the number of detected targets for the selected output number.

- *uint32_t clippingFlag*

Information if the radar signal is overdriven and no vaild targetlist is available.
0 = signal ok, 1 = signal overdriven

- *iSYSTarget_t targets[MAX_TARGETS]*

Inherits the Targetlist (see 5.2.2). Maximum Targets are 35.

5.2.2. struct iSYSTarget_t

The iSYSTarget_t struct contains the values for a target.

```
typedef struct iSYSTarget {
    float32_t velocity;
    float32_t range;
    float32_t signal;
    float32_t angle;
} iSYSTarget_t;
```

- *float32_t velocity*

Radial velocity in m/s.

- *float32_t range*

Range in m.

- *float32_t signal*

Signal indicator in dB.

- *float32_t angle*

Angle of the target in °.

5.2.3. union iSYSTargetListError_u

The iSYSTargetListError_u inherits the error status for the targetlist.

```
union iSYSTargetListError_u
{
    iSYSTargetListError_t iSYSTargetListError;
    uint32_t dummy;
};
```

- *iSYSTargetListError_t iSYSTargetListError*

Error status code for the targetlist (see 5.2.20)

- *uint32_t dummy*

Dummy value. Not used.

5.2.4. struct iSYSRawData_t

The *iSYSRawData_t* inherits information about the raw signal list.

```
typedef struct iSYSRawData{
  iSYSIQSignal_t signal;
  uint32_t signalNrOfSamples;
  uint32_t signalNrOfIQPairs;
}iSYSRawData_t;
```

- *iSYSIQSignal_t signal;*

Inherits the raw signal list (see 5.2.5).

- *uint32_t signalNrOfSamples;*

Information about the number of samples for each I and Q signal.

- *uint32_t signalNrOfIOPairs;*

Information about the number of IQ pairs.

5.2.5. struct iSYSIQSignal_t

The *iSYSIQSignal_t* contains the sampled IQ Signals from the iSYS-System ADC.

```
typedef struct iSYSIQSignal{
  iSYSComplex_t IQ1A[MAX_DATA_SIZE];
  iSYSComplex_t IQ1B[MAX_DATA_SIZE];
  iSYSComplex_t IQ1C[MAX_DATA_SIZE];
  iSYSComplex_t IQ1D[MAX_DATA_SIZE];
}iSYSIQSignal_t;
```

- *iSYSComplex_t IQ1A;*

Inherits the IQ1A data (see 5.2.6). Maximum data size is 2048 samples.

- *iSYSComplex_t IQ1B;*

Inherits the IQ1B data (see 5.2.6). Maximum data size is 2048 samples.

- *iSYSComplex_t IQ1C;*

Inherits the IQ1C data (see 5.2.6). Maximum data size is 2048 samples.

- *iSYSComplex_t IQ1D;*

Inherits the IQ1D data (see 5.2.6). Maximum data size is 2048 samples.

5.2.6. struct iSYSComplex_t

The *iSYSIQSignal_t* contains the real (I) and imaginary (Q) value.

```
typedef struct iSYSComplex{
    float32_t real;
    float32_t imag;
}iSYSComplex_t;
```

- *float32_t real;*

Contains the real (I) value, sampled from the ISYS-System ADC.

- *float32_t imag;*

Contains the imaginary (Q) value, sampled from the ISYS-System ADC.

5.2.7. struct iSYSDetection_t

The iSYSDetection_t inherits the information about the fft data and number of detections.

```
typedef struct iSYSDetection{
  iSYSDetectionData_t detection;
  uint32_t signalNrOfSamples;
  uint32_t nrOfDetections;
}iSYSDetection_t;
```

- *iSYSDetectionData_t detection;*

Inherits the information about the fft magnitude list and threshold list. (see 5.2.8)

- *uint32_t signalNrOfSamples;*

Contains the number of samples.

- *uint32_t nrOfDetections;*

Contains the number of detections.

5.2.8. struct iSYSDetectionData_t

The iSYSDetectionData_t contains the fft data and threshold data.

```
typedef struct iSYSDetectionData{
  uint32_t clippingFlag;
  sint16_t fftMagnitude[MAX_DATA_SIZE];
  sint16_t threshold[MAX_DATA_SIZE];
}iSYSDetectionData_t;
```

- *uint32_t clippingFlag;*

Information if the radar signal is overdriven and no vaild detections are available.
0 = signal ok, 1 = signal overdriven

- *sint16_t fftMagnitude[MAX_DATA_SIZE];*

Contains the fft magnitude data. Maximum data size is 2048 samples.

- *sint16_t threshold[MAX_DATA_SIZE];*

Contains the threshold. Maximum data size is 2048 samples.

5.2.9. struct iSYSRangeList_t (iSYS-600X only)

The iSYSRangeList_t contains the range and serveral other results from iSYS-600X systems.

```
typedef struct iSYSRangeList{
    sint32_t range;
    sint16_t amplitude;
    sint16_t sensor_temperature;
    sint32_t standard_deviation;
    sint32_t variance;
    sint32_t statistical_min;
    sint32_t statistical_max;
} iSYSRangeList_t;
```

- *sint32_t range*

Contains the range in µm

- *sint16_t amplitude*

Contains the signal strength in dB.

- *sint16_t sensor_temperature*

Contains the sensor temperature in °C.

- *sint32_t standard_deviation*

Statistical value. Currently not implemented.

- *sint32_t statistical_min*

Statistical value. Currently not implemented.

- *sint32_t statistical_max*

Statistical value. Currently not implemented.

5.2.10. enum iSYSBaudrate_t

Enum for all available baudrates (default 115200).

iSYSBaudrate_t	Description
ISYS_BAUDRATE_110	Baudrate 110
ISYS_BAUDRATE_300	Baudrate 300
ISYS_BAUDRATE_600	Baudrate 600
ISYS_BAUDRATE_1200	Baudrate 1.200
ISYS_BAUDRATE_2400	Baudrate 2.400
ISYS_BAUDRATE_4800	Baudrate 4.800
ISYS_BAUDRATE_9600	Baudrate 9.600
ISYS_BAUDRATE_19200	Baudrate 19.200
ISYS_BAUDRATE_38400	Baudrate 38.400
ISYS_BAUDRATE_57600	Baudrate 57.600
ISYS_BAUDRATE_115200	Baudrate 115.200
ISYS_BAUDRATE_256000	Baudrate 256.000
ISYS_BAUDRATE_512000	Baudrate 512.000
ISYS_BAUDRATE_230400	Baudrate 230.400

5.2.11. enum iSYSOutput_t

Enum for iSYS digital output configuration

iSYSOutput_t	Description
ISYS_OUTPUT_OFF	Digital output deactivated
ISYS_OUTPUT_DIGITAL	Digital output activated digital IO
ISYS_OUTPUT_PWM	Digital output activated with pwm signal

5.2.12. enum iSYSOutputnumber_t

Enum for iSYS digital output selection

iSYSOutputnumber_t	Description
ISYS_OUTPUT_1	Digital output 1
ISYS_OUTPUT_2	Digital output 2
ISYS_OUTPUT_3	Digital output 3

5.2.13. enum iSYSDirection_type_t

Enum for iSYS direction types

iSYSDirection_type_t	Description
ISYS_TARGET_DIRECTION_APPROACHING	Target is approaching
ISYS_TARGET_DIRECTION_RECEDING	Target is receding
ISYS_TARGET_DIRECTION_BOTH	Target is approaching and receding
ISYS_TARGET_DIRECTION_END	Defines the end of enum

5.2.14. enum iSYSVelocity_unit_t

Enum for iSYS velocity units

iSYSVelocity_unit_t	Description
ISYS_VELOCITY_UNIT_MS	Meter per second (default)
ISYS_VELOCITY_UNIT_KMH	Kilometer per hour
ISYS_VELOCITY_UNIT_MPH	Miles per hour

5.2.15. enum iSYSTemperature_unit_t

Enum for iSYS temperatures units

iSYSTemperature_unit_t	Description
ISYS_TEMPERATURE_UNIT_KELVIN	Kelvin
ISYS_TEMPERATURE_UNIT_CELSIUS	Celsius (default)
ISYS_TEMPERATURE_UNIT_FAHRENHEIT	Fahrenheit

5.2.16. enum iSYSOutput_filter_t

Enum for iSYS single target output filter

iSYSOutput_filter_t	Description
ISYS_OUTPUT_FILTER_HIGHEST_SIGNAL	Highest signal
ISYS_OUTPUT_FILTER_MEAN	Mean
ISYS_OUTPUT_FILTER_MEDIAN	Median
ISYS_OUTPUT_FILTER_MIN	Min
ISYS_OUTPUT_FILTER_MAX	Max
iSYS_OUTPUT_FILTER_END	Defines the end of enum

5.2.17. enum iSYSFilter_signal_t

Enum for iSYS single target output filter. Defines which signal is used for the Output Filter.

iSYSFilter_signal_t	Description
ISYS_SIGNAL_FILTER_OFF	Single target output filter deactivated
ISYS_SIGNAL_FILTER_VELOCITY_RADIAL	Velocity radial
ISYS_SIGNAL_FILTER_RANGE_RADIAL	Range radial
ISYS_SIGNAL_FILTER_ANGLE	Currently not implemented
ISYS_SIGNAL_FILTER_VELOCITY_X	Currently not implemented
ISYS_SIGNAL_FILTER_VELOCITY_Y	Currently not implemented
ISYS_SIGNAL_FILTER_RANGE_X	Currently not implemented
ISYS_SIGNAL_FILTER_RANGE_Y	Currently not implemented
ISYS_SIGNAL_FILTER_END	Defines the end of enum

5.2.18. enum iSYSFrequencyChannel_t

Enum for iSYS Frequency channels.

iSYSFrequencyChannel_t	Description
ISYS_CHANNEL_1	Channel 1
ISYS_CHANNEL_2	Channel 2
ISYS_CHANNEL_3	Channel 3
ISYS_CHANNEL_4	Channel 4
ISYS_CHANNEL_5	Channel 5
ISYS_CHANNEL_6	Channel 6
ISYS_CHANNEL_7	Channel 7
ISYS_CHANNEL_8	Channel 8

5.2.19. enum iSYSResult_t

iSYSResult_t	Description
ERR_OK	No error. Function was excuted successful
ERR_FUNCTION_DEPRECATED	Function deprecated, do not use this function anymore
ERR_DLL_NOT_FINISHED	Not used.
ERR_HANDLE_NOT_INITIALISED	Handle not initialised
ERR_COMPORT_DOESNT_EXIST	Selected serial port doesn´t exist
ERR_COMPORT_CANT_INITIALIZE	Could not initialize serial port
ERR_COMPORT_ACCESS_DENIED	Selected serial port access denied
ERR_COMPORT_BAUDRATE_NOT_VALID	Serial port doesn´t support the selected baudrate
ERR_COMPORT_CANT_OPEN	Could not open serial port
ERR_COMPORT_CANT_SET_FLOW_CONTROL	Could not set flow control
ERR_COMPORT_CANT_SET_PARITY	Could not set parity
ERR_COMPORT_CANT_SET_STOP_BITS	Could not set stop bits
ERR_COMPORT_CANT_SET_DATA_BITS	Could not set data bits
ERR_COMPORT_CANT_SET_BAUDRATE	Could not set baudrate
ERR_COMPORT_ALREADY_INITIALIZED	Serial port already initialized
ERR_COMPORT_EQUALS_NULL	Serial port not initialized
ERR_COMPORT_NOT_OPEN	Could not open serial port
ERR_COMPORT_NOT_READABLE	Serial port not readable
ERR_COMPORT_NOT_WRITEABLE	Serial port not writeable
ERR_COMPORT_CANT_WRITE	Could not write to serial port
ERR_COMPORT_CANT_READ	Could not read from serial port
ERR_COMMAND_NOT_WRITTEN	Command was not written to serial port
ERR_COMMAND_NOT_READ	Command was not read from serial port
ERR_COMMAND_NO_DATA_RECEIVED	No data received
ERR_COMMAND_NO_VALID_FRAME_FOUND	No valid data frame found
ERR_COMMAND_RX_FRAME_DAMAGED	Data frame damaged
ERR_COMMAND_FAILURE	Command was not executed by the iSYS system
ERR_UNDEFINED_READ	Could not read data frame
ERR_COMPORT_LESS_DATA_READ	Not enough data received to have a valid data frame
ERR_COMPORT_SYSTEM_INIT_FAILED	System initialization failed
ERR_COMPORT_SYSTEM_ALREADY_INITIALIZED	System already initialised
ERR_COMMAND_RX_FRAME_LENGTH	Wrong rx data frame length
ERR_COMMAND_MAX_DATA_OVERFLOW	Data overflow
ERR_COMMAND_MAX_IQPAIRS_OVERFLOW	IO pairs overflow
ERR_COMMAND_NOT_ACCEPTED	Command not accepted by sensor
ERR_NULL_POINTER	Pointer does not refer to a valid object.

ERR_CALC_CORRECTION_PARAMS	only for internal use
ERR_PARAMETER_OUT_OF_RANGE	Parameter is out of range

5.2.20. enum iSYSTargetListError_t

Enum for iSYS Target list errors.

iSYSTargetListError_t	Description
TARGET_LIST_OK	No error. Targetlist is valid
TARGET_LIST_FULL	Targetlist full
TARGET_LIST_REFRESHED	Targetlist refreshed
TARGET_LIST_ALREADY_REQUESTED	Targetlist already requested
TARGET_LIST_ACQUISITION_NOT_STARTED	Acquisition not started, could not read targetlist

5.2.21. enum IDSDirection_type_t

Enum for IDS direction settings.

iSYSTargetListError_t	Description
IDS_DIRECTION_POSITIVE	Set direction to positive
IDS_DIRECTION_NEGATIVE	Set direction to negative

5.2.22. enum iSYSSaveLocation_t

Enum that defines if configuration setting is stored in RAM or EEPROM.

iSYSSaveLocation_t	Description
ISYS_LOCATION_RAM	Value is stored volatile.
ISYS_LOCATION_EEPROM	Value is stored non volatile.

5.2.23. enum iSYSMeasurementMode_t

*Enum to set the measurement mode (see caption **Measurement Mode** in **iSYS_serial_interface_protocol_revX.pdf**).*

iSYSFrequencyChannel_t	Description
ISYS_MEASUREMENT_MODE_FAST	Fast measurement
ISYS_MEASUREMENT_MODE_ACCURATE	Accurate measurement

5.2.24. enum iSYSWarningMode_t

Enum to set the warning mode for range or temperature threshold (iSYS6203 only)

iSYSWarningMode _t	Description
ISYS_TEMPERATURE_WARNING	Temperature warning mode
ISYS_RANGE_WARNING	Range warning mode

5.2.25. enum iSYSTargetListInterface_t

Enum to set the target list output interface

iSYSTargetListInterface_t	Description
ISYS_TARGETLIST_IF_DISABLE	Disable target list output
ISYS_TARGETLIST_IF_SPI	Output target list via SPI interface
ISYS_TARGETLIST_IF_ETH	Output target list via Ethernet interface

5.2.26. enum iSYSLedConfiguration_t

Enum to set the target list output interface

iSYSTargetListInterface_t	Description
ISYS_LED_CONFIG_OFF	Disable LED functionality
ISYS_LED_CONFIG_ON	Enable LED functionality
ISYS_LED_CONFIG_DISABLE_BLINKING	Disable LED yellow blinking only

6. Radar API Function Description

This chapter describes all available functions including a calling example. It is structured in 2 sections. There are functions that could be used for all systems and functions that are only valid for special systems.

6.1. Common Functions

6.1.1. iSYS_initComPort (…)

To open the serial connection to a connected iSYS-400X, ISYS-600X use `iSYS_initComPort` (…). Note that a COM port can only be occupied by one connection. Make sure that your COM is not blocked from another application.

Input:

```
iSYSHandle_t *pHandle, uint8_t comportNr, iSYSBaudrate_t baudrate
```

For each connection to a system, the user has to create an empty `iSYSHandle_t pHandle` that is a unique identifier to communicate with this unique system. This function will initiate the connection and initialize `pHandle`.

6.1.2. iSYS_initSystem (…)

To initialize a connection to a system use `iSYS_initSystem (…).`

Input:

```
iSYSHandle_t pHandle, uint8_t destAddress, uint32_t timeout
```

The user has to initialize each system with this function. Every system has a device address which has to be initialized. With this address, it is possible to connect more than one system over RS485 to only one serial port.

Address	Description
0	Broadcast address
1	Master device address
2 to 255	Individual sensor slave addresses (default: iSYS-400X = 128 , iSYS-5XXX = 100 , iSYS-600X = 100)

6.1.3. iSYS_exitSystem (…)

To close (disconnect) an existing connection use iSYS_exitSystem(…).

Input:

```
iSYSHandle_t pHandle, uint8_t destAddress
```

6.1.4. iSYS_exitComPort (…)

To close (disconnect) an existing serial port connection use iSYS_exitSystem(…).

Input:

```
iSYSHandle_t pHandle
```

6.1.5. iSYS_getApiVersion (…)

To get the Serial Radar API version use iSYS_getApiVersion(…).

Input:

```
float32_t *version
```

6.1.6. iSYS_ReadDeviceName (…)

To get the devicename and serialnumber in ASCII format use iSYS_ReadDeviceName(…).

To read the full device name you need 21 characters.

Input:

```
iSYSHandle_t pHandle, char *devicename_array, uint16_t array_length,
uint8_t destAddress, uint32_t timeout
```

6.1.7. iSYS_StartAcquisition (…)

To start the acquisition use iSYS_StartAcquisition (…).

Input:

```
iSYSHandle_t pHandle, uint8_t destAddress, uint32_t timeout
```

6.1.8. iSYS_StopAcquisition (…)

To stop the acquisition use iSYS_StopAcquisition (…).

Input:

```
iSYSHandle_t pHandle, uint8_t destAddress, uint32_t timeout
```

6.1.9. iSYS_getDeviceAddress/iSYS_setDeviceAddress (…)

To get/set the device address use iSYS_getDeviceAddress/iSYS_setDeviceAddress(…).

Input:

```
iSYS_getDeviceAddress:
iSYSHandle_t pHandle,uint8_t *deviceaddress, uint8_t destAddress,
uint32_t timeout
iSYS_setDeviceAddress:
iSYSHandle_t pHandle, uint8_t deviceaddress, uint8_t destAddress ,
uint32_t timeout
```

6.1.10. iSYS_getFrequencyChannel/iSYS_setFrequencyChannel (…)

To get/set the frequency channel use iSYS_getFrequencyChannel/iSYS_setFrequencyChannel (…).

Sensor must be out of measurement mode (iSYS_StopAcquisition).

Input:

```
iSYS_getFrequencyChannel:
iSYSHandle_t pHandle, iSYSFrequencyChannel_t *channel, uint8_t
destAddress, uint32_t timeout
iSYS_setFrequencyChannel:
iSYSHandle_t pHandle, iSYSFrequencyChannel_t channel, uint8_t
destAddress , uint32_t timeout
```

The different iSYS sensors support only support only a limited number of frequency channels. For a list of the available frequency channels refer to the device data sheet or the table below

Device	Available # of frequency channels
• *iSYS-6003*	Channel 1
• *iSYS-4001, iSYS-4002, iSYS-4003, iSYS-4009, iSYS-4010*	Channel 1, Channel 2, Channel 3, Channel 4
• *iSYS-4004, iSYS-4011, iSYS-4012*	Channel 1, Channel 2, Channel 3
• *iSYS-5010*	Channel 1, Channel 2
• *iSYS-5110*	Channel 1, Channel 2

6.1.11. iSYS_getThresholdMin/iSYS_setThresholdMin (…)

To get/set the threshold min use `iSYS_getThresholdMin/iSYS_setThresholdMin (…).`

The values are transmitted as sint16_t data type in tenth of dB. Only writing of values between -30.0 dB and +30.0 dB is valid. Requesting invalid values is replied with failure.

Input:

```
iSYS_getThresholdMin:
iSYSHandle_t pHandle, sint16_t *sensitivity, uint8_t destAddress,
uint32_t timeout
iSYS_setThresholdMin:
iSYSHandle_t pHandle, sint16_t sensitivity, uint8_t destAddress,
uint32_t timeout
```

6.1.12. iSYS_getThresholdSensitivityLeft/iSYS_setThresholdSensitivityLeft (…)

To get/set the threshold sensitivity left use `iSYS_getThresholdSensitivityLeft/iSYS_setThresholdSensitivityLeft (…).`

Threshold sensitivity for left FFT-side in tenth of dB, e.g. iSYS-4001 positive velocities. Only writing of values between -30.0 dB and +30.0 dB is valid.

Input:

```
iSYS_getThresholdSensitivityLeft:
iSYSHandle_t pHandle, uint16_t *threshold, uint8_t destAddress,
uint32_t timeout
iSYS_setThresholdSensitivityLeft:
iSYSHandle_t pHandle, uint16_t threshold, uint8_t destAddress,
uint32_t timeout
```

6.1.13. iSYS_getThresholdSensitivityRight/iSYS_setThresholdSensitivityRight (…)

To get/set the threshold sensitivity right use `iSYS_getThresholdSensitivityRight/iSYS_setThresholdSensitivityRight (…).`

Threshold sensitivity for right FFT-side in tenth of dB, e.g. iSYS-4001 negative velocities. Only writing of values between -30.0 dB and +30.0 dB is valid.

Input:

```
iSYS_getThresholdSensitivityRight:
iSYSHandle_t pHandle, uint16_t *threshold, uint8_t destAddress,
uint32_t timeout
iSYS_setThresholdSensitivityRight:
iSYSHandle_t pHandle, uint16_t threshold, uint8_t destAddress,
uint32_t timeout
```

6.1.14. iSYS_getLedConfiguration/ iSYS_setLedConfiguration (…)

To get/set the LED functionality (iSYS-4001, -4002, 4003 only)
iSYS_getLedConfiguration/ iSYS_setLedConfiguration (…).

Input:

```
iSYS_getLedConfiguration:
iSYSHandle_t pHandle, iSYSLedConfiguration_t *config, uint8_t destAddress,
uint32_t timeout
iSYS_setLedConfiguration:
iSYSHandle_t pHandle, iSYSLedConfiguration_t config, uint8_t destAddress,
uint32_t timeout
```

6.1.15. iSYS_getOutputAngleMin/iSYS_setOutputAngleMin (…)

To get/set the output angle min use
iSYS_getOutputAngleMin/iSYS_setOutputAngleMin (…).

Min possible angle (value in tenth of degree)

Input:

```
iSYS_getOutputAngleMin:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*angle, uint8_t destAddress, uint32_t timeout
iSYS_setOutputAngleMin:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*angle, uint8_t destAddress, uint32_t timeout
```

6.1.16. iSYS_getOutputAngleMax/iSYS_setOutputAngleMax (…)

To get/set the output angle max use iSYS_getOutputAngleMax/iSYS_setOutputAngleMax (…).

Min possible angle (value in tenth of degree).

Input:

```
iSYS_getOutputAngleMax:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*angle, uint8_t destAddress, uint32_t timeout
iSYS_setOutputAngleMax:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t angle,
uint8_t destAddress, uint32_t timeout
```

6.1.17. iSYS_getOutputRangeMin/iSYS_setOutputRangeMin (…)

To get/set the output range min use iSYS_getOutputRangeMin/iSYS_setOutputRangeMin (…).

Min possible range (value in tenth of m).

Input:

```
iSYS_getOutputRangeMin:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*range, uint8_t destAddress, uint32_t timeout
iSYS_setOutputRangeMin:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t range,
uint8_t destAddress, uint32_t timeout
```

6.1.18. iSYS_getOutputRangeMax/iSYS_setOutputRangeMax (…)

To get/set the output range max use iSYS_getOutputRangeMax/iSYS_setOutputRangeMax (…).

Max possible range (value in tenth of m).

Input:

```
iSYS_getOutputRangeMax:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*range, uint8_t destAddress, uint32_t timeout
iSYS_setOutputRangeMax:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t range,
uint8_t destAddress, uint32_t timeout
```

6.1.19. iSYS_getOutputSignalMin/iSYS_setOutputSignalMin (…)

To get/set the output signal min use iSYS_getOutputSignalMin/iSYS_setOutputSignalMin (…).

Min possible signal strength (value in tenth of dB).

Input:

```
iSYS_getOutputSignalMin:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*signal, uint8_t destAddress, uint32_t timeout
iSYS_setOutputSignalMin:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
signal, uint8_t destAddress, uint32_t timeout
```

6.1.20. iSYS_getOutputSignalMax/iSYS_setOutputSignalMax (…)

To get/set the output signal max use
iSYS_getOutputSignalMax/iSYS_setOutputSignalMax (…).

Max possible signal strength (value in tenth of dB).

Input:

```
iSYS_getOutputSignalMax:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*signal, uint8_t destAddress, uint32_t timeout
iSYS_setOutputSignalMax:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
signal, uint8_t destAddress, uint32_t timeout
```

6.1.21. iSYS_getOutputVelocityMin/iSYS_setOutputVelocityMin (…)

To get/set the output velocity min use
iSYS_getOutputVelocityMin/iSYS_setOutputVelocityMin (…).

Min possible velocity (value in tenth of $\frac{m}{s}$)

Input:

```
iSYS_getOutputVelocityMin:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*velocity, uint8_t destAddress, uint32_t timeout
iSYS_setOutputVelocityMin:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
velocity, uint8_t destAddress, uint32_t timeout
```

6.1.22. iSYS_getOutputVelocityMax/iSYS_setOutputVelocityMax (…)

To get/set the output velocity max use
iSYS_getOutputVelocityMax/iSYS_setOutputVelocityMax (…).

Max possible velocity (value in tenth of $\frac{m}{s}$)

Input:

```
iSYS_getOutputVelocityMax:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*velocity, uint8_t destAddress, uint32_t timeout
iSYS_setOutputVelocityMax:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
velocity, uint8_t destAddress, uint32_t timeout
```

6.1.23. iSYS_getOutputDirection/iSYS_setOutputDirection (…)

To get/set the output direction use
iSYS_getOutputDirection/iSYS_setOutputDirection (…).

Moving direction of detected targets

Input:

```
iSYS_getOutputDirection:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber,
iSYSDirection_type_t *direction, uint8_t destAddress, uint32_t timeout
iSYS_setOutputDirection:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber,
iSYSDirection_type_t direction, uint8_t destAddress, uint32_t timeout
```

6.1.24. iSYS_getOutputFilter/iSYS_setOutputFilter (…)

To get/set the output filter use iSYS_getOutputFilter/iSYS_setOutputFilter (…).

Type of single target filter

- 0=highest amplitude
- 1=mean
- 2=median
- 3=min
- 4=max

Input:

```
iSYS_getOutputFilter:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber,
iSYSOutput_filter_t *filter, uint8_t destAddress, uint32_t timeout
iSYS_setOutputFilter:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber,
iSYSOutput_filter_t filter, uint8_t destAddress, uint32_t timeout
```

6.1.25. iSYS_getOutputSignalFilter/iSYS_setOutputSignalFilter (…)

To get/set the output filter signal use
iSYS_getOutputSignalFilter/iSYS_setOutputSignalFilter (…).

Signal for single target filter

- 0=off
- 1=velocity radial
- 2=range radial

Input:

```
iSYS_getOutputSignalFilter:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber,
iSYSFilter_signal_t *signal, uint8_t destAddress, uint32_t timeout
iSYS_setOutputSignalFilter:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber,
iSYSFilter_signal_t signal, uint8_t destAddress, uint32_t timeout
```

6.1.26. iSYS_getOutputAlphaFilterVelocity/iSYS_getOutputAlphaFilterVelocity (…)

To get/set the output velocity alpha filter use `iSYS_getOutputAlphaFilterVelocity/iSYS_setOutputAlphaFilterVelocity (…).`

Alpha-filter parameter for velocity set between 0 and 100 in percent.

Input:

```
iSYS_getOutputAlphaFilterVelocity:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*filter, uint8_t destAddress, uint32_t timeout
iSYS_setOutputAlphaFilterVelocity:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
filter, uint8_t destAddress, uint32_t timeout
```

6.1.27. iSYS_getOutputAlphaFilterRange/iSYS_getOutputAlphaFilterRange (…)

To get/set the output range alpha filter use `iSYS_getOutputAlphaFilterRange/iSYS_setOutputAlphaFilterRange (…).`

Alpha-filter parameter for velocity set between 0 and 100 in percent.

Input:

```
iSYS_getOutputAlphaFilterRange:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*filter, uint8_t destAddress, uint32_t timeout
iSYS_setOutputAlphaFilterRange:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
filter, uint8_t destAddress, uint32_t timeout
```

6.1.28. iSYS_getLineFilter1/iSYS_setLineFilter1 (…)

To get/set the Linefilter1 use `iSYS_getLineFilter1/iSYS_setLineFilter1 (…).`

Line filter 1 setting

1. uint16_t: Enable (0= off, 1 = on)
2. uint16_t: frequency in Hz
3. uint16_t: offset in tenth of dB

Input:

```
iSYS_getLineFilter1:
iSYSHandle_t pHandle, uint16_t *enable, uint16_t *frequency, uint16_t
*offset, uint8_t destAddress, uint32_t timeout
iSYS_setLineFilter1:
iSYSHandle_t pHandle, uint16_t enable, uint16_t frequency, uint16_t
offset, uint8_t destAddress, uint32_t timeout
```

6.1.29. iSYS_getLineFilter2/iSYS_setLineFilter2 (…)

To get/set the Linefilter2 use iSYS_getLineFilter2/iSYS_setLineFilter2 (…).

Line filter 2 setting

1. uint16_t: Enable (0= off, 1 = on)
2. uint16_t: frequency in Hz
3. uint16_t: offset in tenth of dB

Input:

```
iSYS_getLineFilter2:
iSYSHandle_t pHandle, uint16_t *enable, uint16_t *frequency, uint16_t
*offset, uint8_t destAddress, uint32_t timeout
iSYS_setLineFilter2:
iSYSHandle_t pHandle, uint16_t enable, uint16_t frequency, uint16_t
offset, uint8_t destAddress, uint32_t timeout
```

6.1.30. iSYS_getOutputEnable/iSYS_setOutputEnable (…)

To get/set the output enable use iSYS_getOutputEnable/iSYS_setOutputEnable (…).

Digital output enable

- 0 = off
- 1 = digital
- 2 = pwm

Input:

```
iSYS_getOutputEnable:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, iSYSOutput_t
*enable, uint8_t destAddress, uint32_t timeout
iSYS_setOutputEnable:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, iSYSOutput_t
enable, uint8_t destAddress, uint32_t timeout
```

6.1.31. iSYS_getOutputRisingDelay/iSYS_setOutputRisingDelay (…)

To get/set the output rising delay use iSYS_getOutputRisingDelay/iSYS_setOutputRisingDelay (…).

Rising delay in multiples of iSYS cycle time.

Input:

```
iSYS_getOutputRisingDelay:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*delay, uint8_t destAddress, uint32_t timeout
iSYS_setOutputRisingDelay:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t delay,
uint8_t destAddress, uint32_t timeout
```

6.1.32. iSYS_getOutputFallingDelay/iSYS_setOutputFallingDelay (…)

To get/set the output rising delay use iSYS_getOutputFallingDelay/iSYS_setOutputFallingDelay (…).

Falling delay in multiples of iSYS cycle time.

Input:

```
iSYS_getOutputFallingDelay:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*delay, uint8_t destAddress, uint32_t timeout
iSYS_setOutputFallingDelay:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t delay,
uint8_t destAddress, uint32_t timeout
```

6.1.33. iSYS_getOutputPlausibility/iSYS_setOutputPlausibility (…)

To get/set the output plausibility use iSYS_getOutputPlausibility/iSYS_setOutputPlausibility (…).

Input:

```
iSYS_getOutputPlausibility:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*plausibility, uint8_t destAddress, uint32_t timeout
iSYS_setOutputPlausibility:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*plausibility, uint8_t destAddress, uint32_t timeout
```

6.1.34. iSYS_getOutputSetting1/iSYS_setOutputSetting1 (…)

To get/set the output setting1 use iSYS_getOutputSetting1/iSYS_setOutputSetting1 (…).

Setting 1 for the digital outputs

- 0 = active low side
- 1 = active high side
- 2 = totem pole

Input:

```
iSYS_getOutputSetting1:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*setting, uint8_t destAddress, uint32_t timeout
iSYS_setOutputSetting1:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
setting, uint8_t destAddress, uint32_t timeout
```

6.1.35. iSYS_getOutputSetting2/iSYS_setOutputSetting2 (…)

To get/set the output setting2 use iSYS_getOutputSetting2/iSYS_setOutputSetting2 (…).

Setting 2 for the digital outputs

- 0 = normally open
- 1 = normally closed

Input:

```
iSYS_getOutputSetting2:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*setting, uint8_t destAddress, uint32_t timeout
iSYS_setOutputSetting2:
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
setting, uint8_t destAddress, uint32_t timeout
```

6.1.36. iSYS_getPotis/iSYS_setPotis (…)

To get/set the potentiometer value use iSYS_getPotis/iSYS_setPotis (…).

iSYS-4001,4002,4003 only.

Input:

```
iSYS_getPotis:
iSYSHandle_t pHandle, uint16_t *value, uint8_t destAddress , uint32_t
timeout
iSYS_setPotis:
iSYSHandle_t pHandle, uint16_t value, uint8_t destAddress , uint32_t
timeout
```

6.1.37. iSYS_getTargetList (…)

To get targetlist use iSYS_getTargetList (…). The serial radar API will automatically decide if the targetlist is 16Bit or 32Bit.

Sensor must be in measurement mode (iSYS_StartAcquisition).

Input:

```
iSYSHandle_t pHandle, iSYSTargetList_t *pTargetList,
iSYSOutputNumber_t outputnumber, uint8_t destAddress, uint32_t timeout
```

6.1.38. iSYS_getTargetList16 (…)

To get a 16Bit targetlist use iSYS_getTargetList16 (…).

Sensor must be in measurement mode (iSYS_StartAcquisition).

Input:

```
iSYSHandle_t pHandle, iSYSTargetList_t *pTargetList,
iSYSOutputNumber_t outputnumber, uint8_t destAddress, uint32_t timeout
```

6.1.39. iSYS_getTargetList32 (…)

To get a 32Bit targetlist use iSYS_getTargetList32 (…).

Sensor must be in measurement mode (iSYS_StartAcquisition).

Input:

```
iSYSHandle_t pHandle, iSYSTargetList_t *pTargetList,
iSYSOutputNumber_t outputnumber, uint8_t destAddress, uint32_t timeout
```

6.1.40. iSYS_getDigitalOutputState (…)

To get the digital outputstate use iSYS_getDigitalOutputState (…).

Sensor must be in measurement mode (iSYS_StartAcquisition).

Input:

```
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, sint16_t
*value, uint8_t destAddress, uint32_t timeout
```

6.1.41. iSYS_getRawData (…)

To get the raw signal data use iSYS_getRawData (…).

Sensor must be in measurement mode (iSYS_StartAcquisition).

Input:

```
iSYSHandle_t pHandle, iSYSRawData_t *data, uint8_t destAddress,
uint32_t timeout
```

6.1.42. iSYS_getFftData (…)

To get the fft data und number of detections use iSYS_getFftData (…).

Sensor must be in measurement mode (iSYS_StartAcquisition).

Input:

```
iSYSHandle_t pHandle, iSYSDetection_t *data, uint8_t destAddress,
uint32_t timeout
```

6.1.43. iSYS_setFactorySetting (…)

To set the factory default settings use iSYS_setFactorySetting (…).

Input:

```
iSYSHandle_t pHandle, uint8_t destAddress, uint32_t timeout
```

6.1.44. iSYS_saveSensorSettings (…)

To save the sensor settings to EEPROM use iSYS_saveSensorSettings (…).

Input:

```
iSYSHandle_t pHandle, uint8_t destAddress, uint32_t timeout
```

6.1.45. iSYS_saveApplicationSettings (…)

To save the application settings to EEPROM use iSYS_saveApplicationSettings (…).

Input:

```
iSYSHandle_t pHandle, uint8_t destAddress, uint32_t timeout
```

6.1.46. iSYS_saveSensorAndApplicationSettings (…)

To save the sensor and application settings to EEPROM use iSYS_saveSensorAndApplicationSettings (…).

Input:

```
iSYSHandle_t pHandle, uint8_t destAddress, uint32_t timeout
```

6.1.47. iSYS_getDspHardwareVersion (…)

To get the DSP hardware version of sensor.

Input:

```
iSYSHandle_t pHandle, uint16_t *major, uint16_t *fix, uint16_t *minor,
uint8_t destAddress, uint32_t timeout
```

6.1.48. iSYS_getRfeHardwareVersion (…)

To get the RFE hardware version of sensor.

Input:

```
iSYSHandle_t pHandle, uint16_t *major, uint16_t *fix, uint16_t *minor,
uint8_t destAddress, uint32_t timeout
```

6.1.49. iSYS_getProductInfo (…)

To get the sensor configuration (e.g. 4001 or 4002).

Input:

```
iSYSHandle_t pHandle, uint16_t *productInfo, uint8_t destAddress,
uint32_t timeout
```

6.2. iSYS-6XXX functions

6.2.1. iSYS_getRangeList (…)

To get the rangelist use iSYS_getRangeList (…).

Input:

```
iSYSHandle_t pHandle, iSYSRangeList_t *pRangeList, uint8_t
destAddress, uint32_t timeout
```

6.2.2. iSYS_getMeasurementMode/iSYS_setMeasurementMode (…)

To get/set the measurement mode use (…).
You can select the fast mode with 50ms cycletime, or the accurate mode with 200ms cycletime.

Input:

```
iSYS_getMeasurementMode:
iSYSHandle_t pHandle, iSYSMeasurementMode_t *mode, uint8_t
destAddress, uint32_t timeout
iSYS_setMeasurementMode:
iSYSHandle_t pHandle, iSYSMeasurementMode_t mode, uint8_t destAddress,
uint32_t timeout
```

Sensor must be out of measurement mode (iSYS_StopAcquisition).

This setting cannot be stored into nonvolatile memory up to firmware version iSYS-6003_v0.36.

6.3. iSYS-6203 functions

6.3.1. iSYS_SetDefaultTemperatureThreshold (…)

To set default values of temperature threshold and get default values back

Input:

```
iSYSHandle_t pHandle, float32_t *tempThldOut1, float32_t
*tempThldOut2, float32_t *tempThldOut3, uint8_t destAddress, uint32_t
timeout
```

6.3.2. iSYS_getTemperatureThreshold (…)

To get the values of temperature threshold. The place represent the save location. 0 = RAM; 1 = EEPROM

Input:

```
iSYSHandle_t pHandle, uint8_t place, float32_t *tempThldOut1,
float32_t *tempThldOut2, float32_t *tempThldOut3, uint8_t destAddress,
uint32_t timeout
```

6.3.3. iSYS_setTemperatureThreshold (…)

To set the values of temperature threshold. The place represent the save location.

Input:

```
iSYSHandle_t pHandle, iSYSSaveLocation_t place, float32_t
tempThldOut1, float32_t tempThldOut2, float32_t tempThldOut3, uint8_t
destAddress, uint32_t timeout
```

6.3.4. iSYS_setMountingOffsetValue (…)

To set the value of mounting offset in 1[mm].

Input:

```
iSYSHandle_t pHandle, uint32_t mountingOffset, uint8_t destAddress,
uint32_t timeout
```

6.3.5. iSYS_getMountingOffsetValue (…)

To get the value of mounting offset in 1[mm].

Input:

```
iSYSHandle_t pHandle, uint32_t *mountingOffset, uint8_t destAddress,
uint32_t timeout
```

6.3.6. iSYS_setRangeTemperatureWarningThresholdSwitch (…)

To set the switch for choosing between range or temperature threshold for outputs 1 and 2. The location represents the save location: 0 = RAM; 1 = EEPROM
The selectedOutput represents the regarding output: 1 = Output1; 2 = Output2

Input:

```
iSYSHandle_t pHandle, iSYSSaveLocation_t location, iSYSOutputNumber_t
selectedOutput, iSYSWarningMode_t warningMode, uint8_t destAddress,
uint32_t timeout
```

6.3.7. iSYS_getRangeTemperatureWarningThresholdSwitch (…)

To get the switch for choosing between range or temperature threshold for outputs 1 and 2. The location represents the save location: 0 = RAM; 1 = EEPROM
The selectedOutput represents the regarding output: 1 = Output1; 2 = Output2

Input:

```
iSYSHandle_t pHandle, iSYSSaveLocation_t location, iSYSOutputNumber_t
selectedOutput, iSYSWarningMode_t *warningMode, uint8_t destAddress,
uint32_t timeout
```

6.3.8. iSYS_setRangeWarningThreshold (…)

To set the threshold value for range warning in 1000.1[mm]. The location represents the save location: 0 = RAM; 1 = EEPROM
The selectedOutput represents the regarding output: 1 = Output1; 2 = Output2

Input:

```
iSYSHandle_t pHandle, iSYSSaveLocation_t location, iSYSOutputNumber_t
selectedOutput, uint32_t rangeWarningThreshold, uint8_t destAddress,
uint32_t timeout
```

6.3.9. iSYS_getRangeWarningThreshold (…)

To get the threshold value for range warning in 1000.1[mm]. The location represents the save location: 0 = RAM; 1 = EEPROM
The selectedOutput represents the regarding output: 1 = Output1; 2 = Output2

Input:

```
iSYSHandle_t pHandle, iSYSSaveLocation_t location, iSYSOutputNumber_t selectedOutput, uint32_t *rangeWarningThreshold, uint8_t destAddress, uint32_t timeout
```

6.3.10. iSYS_setTemperatureWarningThreshold (…)

To set the threshold value for range warning in 1.00[°C]. The location represents the save location: 0 = RAM; 1 = EEPROM
The selectedOutput represents the regarding output: 1 = Output1; 2 = Output2

Input:

```
iSYSHandle_t pHandle, iSYSSaveLocation_t location, iSYSOutputNumber_t selectedOutput, float32_t temperatureWarningThreshold, uint8_t destAddress, uint32_t timeout
```

6.3.11. iSYS_getTemperatureWarningThreshold (…)

To get the threshold value for range warning in 1.00[°C]. The location represents the save location: 0 = RAM; 1 = EEPROM
The selectedOutput represents the regarding output: 1 = Output1; 2 = Output2

Input:

```
iSYSHandle_t pHandle, iSYSSaveLocation_t location, iSYSOutputNumber_t selectedOutput, float32_t *temperatureWarningThreshold, uint8_t destAddress, uint32_t timeout
```

6.3.12. iSYS_setOutputRangeMinExt (…)

To set the range minimum in 0.01[m].
The outputnumber represents the regarding output: 1 = Output1; 2 = Output2

Input:

```
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t range, uint8_t destAddress, uint32_t timeout
```

6.3.13. iSYS_getOutputRangeMinExt (…)

To get the range minimum in 0.01[m].
The outputnumber represents the regarding output: 1 = Output1; 2 = Output2

Input:

```
iSYSHandle_t pHandle, iSYSOutputNumber_t *outputnumber, uint16_t
range, uint8_t destAddress, uint32_t timeout
```

6.3.14. iSYS_setOutputRangeMaxExt (…)

To set the range maximum in 0.01[m].
The outputnumber represents the regarding output: 1 = Output1; 2 = Output2

Input:

```
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t range,
uint8_t destAddress, uint32_t timeout
```

6.3.15. iSYS_getOutputRangeMaxExt (…)

To set the range maximum in 0.01[m].
The outputnumber represents the regarding output: 1 = Output1; 2 = Output2

Input:

```
iSYSHandle_t pHandle, iSYSOutputNumber_t outputnumber, uint16_t
*range, uint8_t destAddress, uint32_t timeout
```

6.4. IDS-1000 functions

6.4.1. IDS_getOutputEnable/IDS_setOutputEnable (…)

To get/set the output state of IDS-1000 use
`IDS_getOutputEnable/IDS_setOutputEnable(…)`.

- 0 = Output OFF
- 1 = Output ON (default after repower)

Input:

```
IDS_getOutputEnable:
iSYSHandle_t pHandle, bool_t *enable, uint8_t destAddress
IDS_setOutputEnable:
iSYSHandle_t pHandle, bool_t enable, uint8_t destAddress
```

6.4.2. IDS_getFrequency/IDS_setFrequency (…)

To get/set the frequency [in Hz] of IDS-1000 use `IDS_getFrequency/IDS_setFrequency(…)`. Values between 15 and 1000000 are valid.

Input:

```
IDS_getFrequency:
iSYSHandle_t pHandle, uint32_t *frequency, uint8_t destAddress
IDS_setFrequency:
iSYSHandle_t pHandle, uint32_t frequency, uint8_t destAddress
```

6.4.3. IDS_getDirection/IDS_setDirection (…)

To get/set the direction of IDS-1000 use `IDS_getDirection/IDS_setDirection(…)`.

Input:

```
IDS_getFrequency:
iSYSHandle_t pHandle, IDSDirection_type_t *direction, uint8_t
destAddress
IDS_setFrequency:
iSYSHandle_t pHandle, IDSDirection_type_t direction, uint8_t
destAddress
```

6.4.4. IDS_saveSettings (…)

To save the current **frequency** and **direction** settings in EEProm use `IDS_saveSettings(…)`.

Input:

```
iSYSHandle_t pHandle, uint8_t destAddress
```

6.5. iSYS-5010 functions

6.5.1. iSYS_getTargetClusteringEnable/iSYS_setTargetClusteringEnable (…)

To set and get target clustering configuration of iSYS-5010.

Input:

```
iSYS_setTargetClusteringEnable:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, uint8_t enable,
uint8_t destAddress, uint32_t timeout
iSYS_getTargetClusteringEnable:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, uint8_t *enable,
uint8_t destAddress, uint32_t timeout
```

6.5.2. iSYS_getRcsOutputEnable/iSYS_setRcsOutputEnable (…)

To set and get RCS output configuration of iSYS-5010.

Input:

```
iSYS_setRcsOutputEnable:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, uint8_t enable,
uint8_t destAddress, uint32_t timeout
iSYS_getRcsOutputEnable:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, uint8_t *enable,
uint8_t destAddress, uint32_t timeout
```

6.5.3. iSYS_getThresholdMovingTargetsNearRangeMargin/ iSYS_setThresholdMovingTargetsNearRangeMargin (…)

To set and get the moving target near range margin (near range detection sensitivity).

location: 0 = RAM; 1 = EEPROM

margin: margin/sensitivity value in tenth of dB

Input:

```
iSYS_setThresholdMovingTargetsNearRangeMargin:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, sint16_t margin,
uint8_t destAddress, uint32_t timeout
iSYS_getThresholdMovingTargetsNearRangeMargin:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, sint16_t *margin,
uint8_t destAddress, uint32_t timeout
```

6.5.4. iSYS_getThresholdMovingTargetsMainRangeMargin/ iSYS_setThresholdMovingTargetsMainRangeMargin (…)

To set and get the moving target main range margin (main range detection sensitivity).

location: 0 = RAM; 1 = EEPROM

margin: margin/sensitivity value in tenth of dB

Input:

```
iSYS_setThresholdMovingTargetsMainRangeMargin:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, sint16_t margin,
uint8_t destAddress, uint32_t timeout
iSYS_getThresholdMovingTargetsMainRangeMargin:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, sint16_t *margin,
uint8_t destAddress, uint32_t timeout
```

6.5.5. iSYS_getThresholdMovingTargetsLongRangeMargin/ iSYS_setThresholdMovingTargetsLongRangeMargin (…)

To set and get the moving target main range margin (long range detection sensitivity).

location: 0 = RAM; 1 = EEPROM

margin: margin/sensitivity value in tenth of dB

Input:

```
iSYS_setThresholdMovingTargetsLongRangeMargin:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, sint16_t margin,
uint8_t destAddress, uint32_t timeout
iSYS_getThresholdMovingTargetsLongRangeMargin:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, sint16_t *margin,
uint8_t destAddress, uint32_t timeout
```

6.1. iSYS-5020 functions

6.1.1. iSYS_getThresholdMovingTargetsNearRangeMargin/ iSYS_setThresholdMovingTargetsNearRangeMargin (…)

To set and get the moving target near range margin (near range detection sensitivity).

location: 0 = RAM; 1 = EEPROM

margin: margin/sensitivity value in tenth of dB

Input:

```
iSYS_setThresholdMovingTargetsNearRangeMargin:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, sint16_t margin,
uint8_t destAddress, uint32_t timeout
iSYS_getThresholdMovingTargetsNearRangeMargin:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, sint16_t *margin,
uint8_t destAddress, uint32_t timeout
```

6.1.2. iSYS_getThresholdMovingTargetsMainRangeMargin/ iSYS_setThresholdMovingTargetsMainRangeMargin (…)

To set and get the moving target main range margin (main range detection sensitivity).

location: 0 = RAM; 1 = EEPROM

margin: margin/sensitivity value in tenth of dB

Input:

```
iSYS_setThresholdMovingTargetsMainRangeMargin:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, sint16_t margin,
uint8_t destAddress, uint32_t timeout
iSYS_getThresholdMovingTargetsMainRangeMargin:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, sint16_t *margin,
uint8_t destAddress, uint32_t timeout
```

6.1.3. iSYS_getThresholdMovingTargetsLongRangeMargin/ iSYS_setThresholdMovingTargetsLongRangeMargin (…)

To set and get the moving target main range margin (long range detection sensitivity).

location: 0 = RAM; 1 = EEPROM

margin: margin/sensitivity value in tenth of dB

Input:

```
iSYS_setThresholdMovingTargetsLongRangeMargin:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, sint16_t margin,
uint8_t destAddress, uint32_t timeout
iSYS_getThresholdMovingTargetsLongRangeMargin:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, sint16_t *margin,
uint8_t destAddress, uint32_t timeout
```

6.1.4. iSYS_getIPConfig/iSYS_setIPConfig(…)

To set and get IP configuration of iSYS-5020.

Input:

```
iSYS_setIPConfig:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, iSYSIPConfig_t
ipConfig, uint8_t destAddress, uint32_t timeout
iSYS_getIPConfig:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, iSYSIPConfig_t
*ipConfig, uint8_t destAddress, uint32_t timeout
```

6.1.5. iSYS_getIPDestination/iSYS_setIPDestination(…)

To set and get IP destination configuration of iSYS-5020.

Input:

```
iSYS_setIPDestination:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, iSYSIPDestination_t
ipDestination, uint8_t destAddress, uint32_t timeout
iSYS_getIPDestination:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, iSYSIPDestination_t
*ipDestination, uint8_t destAddress, uint32_t timeout
```

6.1.6. iSYS_getTargetListInterface/ iSYS_setTargetListInterface (…)

To set and get target list output interface of iSYS-5110.

Input:

```
iSYS_setTargetListInterface:
iSYSHandle_t pHandle, iSYSSaveLocation_t location,
iSYSTargetListInterface_t interf, uint8_t destAddress, uint32_t
timeout
iSYS_getTargetListInterface:
iSYSHandle_t pHandle, iSYSSaveLocation_t location,
iSYSTargetListInterface_t *pInterface, uint8_t destAddress, uint32_t
timeout
```

6.2. iSYS-5110 functions

6.2.1. iSYS_getIPConfig/iSYS_setIPConfig(…)

To set and get IP configuration of iSYS-5110.

Input:

```
iSYS_setIPConfig:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, iSYSIPConfig_t
ipConfig, uint8_t destAddress, uint32_t timeout
iSYS_getIPConfig:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, iSYSIPConfig_t
*ipConfig, uint8_t destAddress, uint32_t timeout
```

6.2.2. iSYS_getIPDestination/iSYS_setIPDestination(…)

To set and get IP destination configuration of iSYS-5110.

Input:

```
iSYS_setIPDestination:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, iSYSIPDestination_t
ipDestination, uint8_t destAddress, uint32_t timeout
iSYS_getIPDestination:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, iSYSIPDestination_t
*ipDestination, uint8_t destAddress, uint32_t timeout
```

6.2.3. iSYS_setStopBarEnable/iSYS_getStopBarEnable(…)

Enable stop bar zone functionality of iSYS-5110.

location: 0 = RAM; 1 = EEPROM
zoneNumber: 0 = Zone 1; 1 = Zone 2
enable: 0 = disable, 1 = enable

Input:

```
iSYS_setStopBarEnable:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, uint8_t zoneNumber, uint8_t enable, uint8_t destAddress, uint32_t timeout
iSYS_getStopBarEnable:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, uint8_t zoneNumber, uint8_t *enable, uint8_t destAddress, uint32_t timeout
```

6.2.4. iSYS_setStopBarRange/iSYS_getStopBarRange(…)

Set minRange and maxRange in [m] for stop bar zone.

location: 0 = RAM; 1 = EEPROM
zoneNumber: 0 = Zone 1; 1 = Zone 2
minRange/maxRange: value in m

Input:

```
iSYS_setStopBarRange:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, uint8_t zoneNumber, uint16_t minRange, uint16_t maxRange, uint8_t destAddress, uint32_t timeout
iSYS_getStopBarRange:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, uint8_t zoneNumber, uint16_t *minRange, uint16_t *maxRange, uint8_t destAddress, uint32_t timeout
```

6.2.5. iSYS_setStopBarAngle/iSYS_getStopBarAngle(…)

Set minAngle and maxAngle in [°] for stop bar zone.

location: 0 = RAM; 1 = EEPROM
zoneNumber: 0 = Zone 1; 1 = Zone 2
minAngle/maxAngle: value in tenth of °

Input:

```
iSYS_setStopBarAngle:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, uint8_t zoneNumber,
float32_t minAngle, float32_t maxAngle, uint8_t destAddress, uint32_t
timeout
iSYS_getStopBarAngle:
iSYSHandle_t pHandle, iSYSSaveLocation_t location, uint8_t zoneNumber,
float32_t *minAngle, float32_t *maxAngle, uint8_t destAddress,
uint32_t timeout
```

InnoSenT GmbH
Am Roedertor 30
97499 Donnersdorf
GERMANY
Tel.: +49 95289518-0
E-Mail: info@innosent.de
www.innosent.de