BLUEBOXLib

ANSI C Function Library

Up to library release 8.8.0

Preface

IDTRONIC GmbH (IDTRONIC) reserves the right to make changes to its products or services or to discontinue any product or service at any time without notice. IDTRONIC provides customer assistance in various technical areas, but does not have full access to data concerning the use and applications of customer's products. Therefore, IDTRONIC assumes no liability and is not responsible for customer applications or product or software design or performance relating to systems or applications incorporating IDTRONIC products. In addition, IDTRONIC assumes no liability and is not responsible for infringement of patents and/or any other intellectual or industrial property rights of third parties, which may result from assistance provided by IDTRONIC. IDTRONIC products are not designed, intended, authorized or warranted to be suitable for life support applications or any other life critical applications that could involve potential risk of death, personal injury or severe property or environmental damage. With the edition of this document, all previous editions become void. Indications made in this manual may be changed without previous notice. Composition of the information in this manual has been done to the best of our knowledge. IDTRONIC does not guarantee the correctness and completeness of the details given in this manual and may not be held liable for damages ensuing from incorrect or incomplete information. Since, despite all our efforts, errors may not be completely avoided, we are always grateful for your useful tips. The installation instructions given in this manual are based on advantageous boundary conditions. IDTRONIC does not give any guarantee promise for perfect function in cross environments. The companies or products mentioned in this document might be brands or brand names of the different suppliers or their subsidiaries in any country. This document may be downloaded onto a computer, stored and duplicated as necessary to support the use of the related IDTRONIC products. Any other type of duplication, circulation or storage on data carriers in any manner not authorized by IDTRONIC represents a violation of the applicable copyright laws and shall be prosecuted.

Table of Contents

1 Introduction

This manual describes the BLUEBOXLib library and its implemented functions. BLUEBOXLib is a set of ANSI C functions which allows a user program to use and configure all the Soltec BLUEBOX readers. The library is available in the following formats:

- Win32 DLL (provides the BLUEBOXLib.lib stub for Microsoft Visual C++ 6.0).
- x64 DLL (provides the BLUEBOXLib.lib stub for Microsoft Visual C++ 6.0).

2 Library Description

2.1 Typedefs

2.1.1 BLUEBOX_Handle

Name: BLUEBOX_Handle
Description: Handle type used to identify readers.
Syntax typedef int BLUEBOX_Handle;

2.2 Enumerations

2.2.1 BLUEBOX_ErrorCodes

Name: BLUEBOX_ErrorCodes
Description: Error codes enum.
Enumarator:
BLUEBOX_StatusOk: Operation completed successfully.
BLUEBOX_InitError: Initialization error.
BLUEBOX_InvalidHandle: Invalid handle.
BLUEBOX_InvalidChannel: Invalid channel.
BLUEBOX_InvalidParams: Invalid parameters.
BLUEBOX_GenericError: Generic error.
BLUEBOX_TimeoutError: Communication error.
BLUEBOX_ConnectionError: Connection error.
BLUEBOX_MemoryError: Memory al location error.
BLUEBOX_InvalidCommand: Invalid command.
BLUEBOX_TagNotFound: Tag not found.
BLUEBOX_TagError: Tag error.
BLUEBOX_AllocationError: Notify channel allocation error.
BLUEBOX_FileError: File error.
BLUEBOX_RegistrationNotFound: Registration not found.

Syntax

```
typedef enum BLUEBOX_ErrorCodes
{
        BLUEBOX_StatusOk = 0,
        BLUEBOX_InitError = -1,
        BLUEBOX_InvalidHandle = -2,
        BLUEBOX_InvalidChannel = -3,
        BLUEBOX_InvalidParams = -4,
        BLUEBOX_GenericError = -5,
```

```
        BLUEBOX_TimeoutError = -6,
        BLUEBOX_CommunicationError = -7,
        BLUEBOX_ConnectionError = -8,
        BLUEBOX_MemoryError = -9,
        BLUEBOX_InvalidCommand = -10,
        BLUEBOX_TagNotFound = -11,
        BLUEBOX_TagError = -12,
        BLUEBOX_AllocationError = -13,
        BLUEBOX_FileError = -14,
        BLUEBOX_RegistrationNotFound = -15

} BLUEBOX_ErrorCodes;
```

2.2.2 BLUEBOX_Output

Name: BLUEBOX_Output

Description: Output enum.

Enumarator: *BLUEBOX_OUTPUT_1*: Output 1.
BLUEBOX_OUTPUT_2: Output 2.

Syntax

```
typedef enum BLUEBOX_Output
{
        BLUEBOX_OUTPUT_1 = 1,
        BLUEBOX_OUTPUT_2 = 2

} BLUEBOX_Output;
```

2.2.3 BLUEBOX_Input

Name: BLUEBOX_Input

Description: Input enum.

Enumarator: *BLUEBOX_NOINPUT*: No input information.
BLUEBOX_INPUT_1: Input 1.
BLUEBOX_INPUT_2: Input 2.

Syntax

```
typedef enum BLUEBOX_Input
{
        BLUEBOX_NPINPUT= 0,
        BLUEBOX_INPUT_1 = 1,
        BLUEBOX_INPUT_2 = 2

} BLUEBOX_Input;
```

2.2.4 BLUEBOX_Antenna

Name: BLUEBOX_Antenna

Description: Antenna enum.

Enumarator:
BLUEBOX_NOANT: No antenna information.
BLUEBOX_ANT_1: Antenna nr. 1.
BLUEBOX_ANT_2: Antenna nr. 2.
BLUEBOX_ANT_3: Antenna nr. 3.
BLUEBOX_ANT_4: Antenna nr. 4.

Syntax

```
typedef enum BLUEBOX_Antenna
{
        BLUEBOX_NOANT = 0,
        BLUEBOX_ANT_1 = 1,
        BLUEBOX_ANT_2 = 2,
        BLUEBOX_ANT_3 = 3,
        BLUEBOX_ANT_4 = 4

} BLUEBOX_Antenna;
```

2.2.5 BLUEBOX_TagType

Name: BLUEBOX_TagType

Description: Tag type enum.

Enumarator:
BLUEBOX_UNDEFINED: Undefined tag.
BLUEBOX_SHORT: BLUEBOX SHORT.
BLUEBOX_MEDIUM: BLUEBOX MEDIUM.
BLUEBOX_LARGE: BLUEBOX LARGE.
BLUEBOX_EM4305: EM4305.
BLUEBOX_T5557: T5557.
BLUEBOX_Q5: Q5.
BLUEBOX_HITAG_S256: HITAG S 256.
BLUEBOX_HITAG_S2048: HITAG S 2048.
BLUEBOX_TITAN: TITAN.
BLUEBOX_ISO14443A: ISO 14443A.
BLUEBOX_MIFARE_1k: MIFARE 1k.
BLUEBOX_MIFARE_4k: MIFARE 4k.
BLUEBOX_MIFARE_UL: MIFARE Ultralight.
BLUEBOX_ISO15693: ISO 15693.
BLUEBOX_ICODE2: ICODE SLI.
BLUEBOX_ICODE_SLI_S: ICODE SLI-S.
BLUEBOX_TAG_IT_HF_I: Tag-It HF-I.
BLUEBOX_EM4035: EM4035.
BLUEBOX_LRI_64_512: LRI 64/512.
BLUEBOX_MB89R118: MB89R118.

BLUEBOX_ISO14443B: ISO 14443B.
BLUEBOX_SR176: SR176.
BLUEBOX_ISO18K6B: ISO 18000-6B.
BLUEBOX_ISO18K6C: ISO 18000-6C.
BLUEBOX_ACTIVE: ACTIVE.

Syntax

```
typedef enum BLUEBOX_TagType
{
        BLUEBOX_UNDEFINED = 0,
        BLUEBOX_SHORT = 1,
        BLUEBOX_MEDIUM = 2,
        BLUEBOX_LARGE = 3,
        BLUEBOX_Q5 = 4,
        BLUEBOX_HITAG_S256 = 5,
        BLUEBOX_HITAG_S2048 = 6,
        BLUEBOX_TITAN = 7,
        BLUEBOX_ISO14443A = 8,
        BLUEBOX_MIFARE_1k = 9,
        BLUEBOX_MIFARE_4k = 10,
        BLUEBOX_MIFARE_UL = 11,
        BLUEBOX_ISO15693 = 12,
        BLUEBOX_ICODE2 = 13,
        BLUEBOX_TAG_IT_HF_I = 14,
        BLUEBOX_EM4035 = 15,
        BLUEBOX_LRI_64_512 = 16,
        BLUEBOX_MB89R118 = 17,
        BLUEBOX_ISO14443B = 18,
        BLUEBOX_SR176 = 19,
        BLUEBOX_ISO18K6B = 20,
        BLUEBOX_ISO18K6C = 21,
        BLUEBOX_ACTIVE = 22,
        BLUEBOX_EM4305 = 23,
        BLUEBOX_T5557 = 24
        BLUEBOX_ICODE_SLI_S = 25

} BLUEBOX_TagType;
```

2.2.6 BLUEBOX_ICODE_SLI_S_PasswordIdentifier

Name: BLUEBOX_ICODE_SLI_S_PasswordIdentifier

Description: ICODE SLI-S password identifier enum.

Enumarator:
BLUEBOX_ICODE_SLI_S_PWD_READ: Read.
BLUEBOX_ICODE_SLI_S_PWD_WRITE: Write.
BLUEBOX_ICODE_SLI_S_PWD_PRIVACY: Privacy.
BLUEBOX_ICODE_SLI_S_DESTROY_SLI_S: Destroy SLI-S.

BLUEBOX_ICODE_SLI_S_EAS: EAS.

Syntax

```
typedef enum BLUEBOX_ICODE_SLI_S_PasswordIdentifier
{
	BLUEBOX_ICODE_SLI_S_PWD_READ = 0x01,
	BLUEBOX_ICODE_SLI_S_PWD_WRITE = 0x02,
	BLUEBOX_ICODE_SLI_S_PWD_PRIVACY = 0x04,
	BLUEBOX_ICODE_SLI_S_PWD_DESTROY_SLI_S = 0x08,
	BLUEBOX_ICODE_SLI_S_PWD_EAS = 0x10

} BLUEBOX_ICODE_SLI_S_PasswordIdentifier;
```

2.2.7 BLUEBOX_ICODE_SLI_S_ProtectionStatus

Name: BLUEBOX_ICODE_SLI_S_ProtectionStatus

Description: ICODE SLI-S protection status enum.

Enumarator:

32-bit Password Protection:

BLUEBOX_ICODE_SLI_S_PROTECT_PUBLIC: Public.

BLUEBOX_ICODE_SLI_S_PROTECT_READ_AND_WRITE_BY_READ_PWD: Protect Read and Write by Read password.

BLUEBOX_ICODE_SLI_S_PROTECT_WRITE_BY_WRITE_PWD: Protect Write by Write password.

BLUEBOX_ICODE_SLI_S_PROTECT_READ_BY_READ_PWD_AND_WRITE_BY_WRITE_PWD: Protect Read by Read password and Write by Write password.

64-bit Password Protection:

BLUEBOX_ICODE_SLI_S_PROTECT_PUBLIC: Public.

BLUEBOX_ICODE_SLI_S_PROTECT_READ_AND_WRITE_BY_READ_AND_WRITE_PWD: Protect Read and Write by Read plus Write password.

BLUEBOX_ICODE_SLI_S_PROTECT_WRITE_BY_READ_AND_WRITE_PWD: Protect Write by Read plus Write password.

Syntax

```
typedef enum BLUEBOX_ICODE_SLI_S_ProtectionStatus
{
	BLUEBOX_ICODE_SLI_S_PROTECT_PUBLIC = 0x00,
	BLUEBOX_ICODE_SLI_S_PROTECT_READ_AND_WRITE_BY_READ_PWD = 0x01,
	BLUEBOX_ICODE_SLI_S_PROTECT_WRITE_BY_WRITE_PWD = 0x10,
	BLUEBOX_ICODE_SLI_S_PROTECT_READ_BY_READ_PWD_AND_WRITE_BY_WRITE_PWD = 0x11,
	BLUEBOX_ICODE_SLI_S_PROTECT_READ_AND_WRITE_BY_READ_AND_WRITE_PWD = 0x01,
	BLUEBOX_ICODE_SLI_S_PROTECT_WRITE_BY_READ_AND_WRITE_PWD = 0x10,

} BLUEBOX_ICODE_SLI_S_ProtectionStatus;
```

2.2.8 BLUEBOX_MIFARE_Key

Name: BLUEBOX_MIFARE_Key

Description: MIFARE key enum.

Enumarator: *BLUEBOX_MIFARE_KEY_A*: Key A.
BLUEBOX_MIFARE_KEY_B: Key B.

Syntax

```
typedef enum BLUEBOX_MIFARE_Key
{
      BLUEBOX_MIFARE_KEY_A = 0,
      BLUEBOX_MIFARE_KEY_B = 1

} BLUEBOX_MIFARE_Key;
```

2.2.9 BLUEBOX_ISO18K6C_Bank

Name: BLUEBOX_ISO18K6C_Bank

Description: ISO18000-6B tag's memory bank enum.

Enumarator: *BLUEBOX_ISO18K6C_BANK_RESERVED*: Reserved.
BLUEBOX_ISO18K6C_BANK_EPC: EPC.
BLUEBOX_ISO18K6C_BANK_TID: TID.
BLUEBOX_ISO18K6C_BANK_USER: User.

Syntax

```
typedef enum BLUEBOX_ISO18K6C_Bank
{
      BLUEBOX_ISO18K6C_BANK_RESERVED = 0,
      BLUEBOX_ISO18K6C_BANK_EPC = 1,
      BLUEBOX_ISO18K6C_BANK_TID = 2,
      BLUEBOX_ISO18K6C_BANK_USER = 3

} BLUEBOX_ISO18K6C_Bank;
```

2.2.10 BLUEBOX_ISO18K6C_PasswordPermission

Name: BLUEBOX_ISO18K6C_PasswordPermission

Description: The ISO 18000-6C tag password permission values enum.

Enumarator: *BLUEBOX_ISO18K6C_TAG_PWD_PERM_ACCESSIBLE*: Accessible from either opened and secured states.
BLUEBOX_ISO18K6C_TAG_PWD_PERM_ALWAYS_ACCESSIBLE: Permanently accessible from either opened and secured states. It couldn't be locked.
BLUEBOX_ISO18K6C_TAG_PWD_PERM_SECURED_ACCESSIBLE: Accessible only from secured state.
BLUEBOX_ISO18K6C_TAG_PWD_PERM_ALWAYS_NOT_ACCESSIBLE: Not accessible from either opened or secured

states.
BLUEBOX_ISO18K6C_TAG_PWD_PERM_NO_CHANGE: No change in accessible options.

Syntax

```
typedef enum BLUEBOX_ISO18K6C_PasswordPermission
{
   BLUEBOX_ISO18K6C_TAG_PWD_PERM_ACCESSIBLE = 0,
   BLUEBOX_ISO18K6C_TAG_PWD_PERM_ALWAYS_ACCESSIBLE = 1,
   BLUEBOX_ISO18K6C_TAG_PWD_PERM_SECURED_ACCESSIBLE = 2,
   BLUEBOX_ISO18K6C_TAG_PWD_PERM_ALWAYS_NOT_ACCESSIBLE = 3,
   BLUEBOX_ISO18K6C_TAG_PWD_PERM_NO_CHANGE = 4

} BLUEBOX_ISO18K6C_PasswordPermission;
```

2.2.11 BLUEBOX_ISO18K6C_MemoryPermission

Name: BLUEBOX_ISO18K6C_MemoryPermission

Description: The ISO 18000-6C tag memory permission values enum.

Enumarator:
BLUEBOX_ISO18K6C_TAG_MEM_PERM_WRITABLE: Writable from either opened and secured states.
BLUEBOX_ISO18K6C_TAG_MEM_PERM_ALWAYS_WRITABLE: Permanently writable from either opened and secured states. It couldn't be locked.
BLUEBOX_ISO18K6C_TAG_MEM_PERM_SECURED_WRITABLE: Writable only from secured state.
BLUEBOX_ISO18K6C_TAG_MEM_PERM_ALWAYS_NOT_WRITABLE: Not writable from either opened or secured states.
BLUEBOX_ISO18K6C_TAG_MEM_PERM_NO_CHANGE: No change in writable options.

Syntax

```
typedef enum BLUEBOX_ISO18K6C_MemoryPermission
{
   BLUEBOX_ISO18K6C_TAG_MEM_PERM_WRITABLE= 0,
   BLUEBOX_ISO18K6C_TAG_MEM_PERM_ALWAYS_WRITABLE= 1,
   BLUEBOX_ISO18K6C_TAG_MEM_PERM_SECURED_WRITABLE= 2,
   BLUEBOX_ISO18K6C_TAG_MEM_PERM_ALWAYS_NOT_WRITABLE= 3,
   BLUEBOX_ISO18K6C_TAG_MEM_PERM_NO_CHANGE= 4

} BLUEBOX_ISO18K6C_MemoryPermission;
```

2.2.12 BLUEBOX_Reader

Name: BLUEBOX_Reader

Description: Reader (primary, auxiliary, …) ID.

Enumarator:
BLUEBOX_PRIMARY_READER: Primary reader.
BLUEBOX_AUXILIARY_1_READER: First auxiliary reader.
BLUEBOX_AUXILIARY_2_READER: 2nd auxiliary reader.

Syntax

```
typedef enum BLUEBOX_Reader
{
```

```
        BLUEBOX_PRIMARY_READER = 0,
        BLUEBOX_AUXILIARY_1_READER = 1,
        BLUEBOX_AUXILIARY_2_READER = 1,

} BLUEBOX_Reader;
```

2.3 Definitions

2.3.1 BLUEBOX_EM4305_ID_SIZE

Name: BLUEBOX_EM4305_ID_SIZE
Description: EM4305 tag's ID size in bytes.
Syntax #define BLUEBOX_EM4305_ID_SIZE (4)

2.3.2 BLUEBOX_T5557_ID_SIZE

Name: BLUEBOX_T5557_ID_SIZE
Description: T5557 tag's ID size in bytes.
Syntax #define BLUEBOX_T5557_ID_SIZE (8)

2.3.3 BLUEBOX_Q5_ID_SIZE

Name: BLUEBOX_Q5_ID_SIZE
Description: Q5 tag's ID size in bytes.
Syntax #define BLUEBOX_Q5_ID_SIZE (5)

2.3.4 BLUEBOX_HITAGS_ID_SIZE

Name: BLUEBOX_HITAGS_ID_SIZE
Description: HITAG S tag's ID size in bytes.
Syntax #define BLUEBOX_HITAGS_ID_SIZE (4)

2.3.5 BLUEBOX_HITAGS_PAGE_SIZE

Name: BLUEBOX_HITAGS_PAGE_SIZE
Description: HITAG S tag's memory page size in bytes.
Syntax #define BLUEBOX_HITAGS_PAGE_SIZE (4)

2.3.6 BLUEBOX_TITAN_ID_SIZE

Name: BLUEBOX_TITAN_ID_SIZE
Description: TITAN tag's ID size in bytes.
Syntax #define BLUEBOX_TITAN_ID_SIZE (8)

2.3.7 BLUEBOX_TITAN_PASSWORD_SIZE

Name: BLUEBOX_TITAN_PASSWORD_SIZE
Description: TITAN tag's password size in bytes.
Syntax #define BLUEBOX_TITAN_PASSWORD_SIZE (4)

2.3.8 BLUEBOX_TITAN_PAGE_SIZE

Name: BLUEBOX_TITAN_PAGE_SIZE
Description: TITAN tag's memory page size in bytes.
Syntax #define BLUEBOX_TITAN_PAGE_SIZE (4)

2.3.9 BLUEBOX_ISO15693_UID_SIZE

Name: BLUEBOX_ISO15693_UID_SIZE
Description: ISO 15693 tag's UID size in bytes.
Syntax #define BLUEBOX_ISO15693_UID_SIZE (8)

2.3.10 BLUEBOX_ICODE_SLI_S_RND_SIZE

Name: BLUEBOX_ICODE_SLI_S_RND_SIZE
Description: ICODE SLI-S tag's random number size in bytes.
Syntax #define BLUEBOX_ICODE_SLI_S_RND_SIZE (2)

2.3.11 BLUEBOX_ICODE_SLI_S_PWD_SIZE

Name: BLUEBOX_ICODE_SLI_S_PWD_SIZE
Description: ICODE SLI-S tag's password size in bytes.
Syntax #define BLUEBOX_ICODE_SLI_S_PWD_SIZE (4)

2.3.12 BLUEBOX_MIFARE_1k_UID_SIZE

Name: BLUEBOX_MIFARE_1k_UID_SIZE
Description: MIFARE 1k tag's UID size in bytes.
Syntax #define BLUEBOX_MIFARE_1k_UID_SIZE (4)

2.3.13 BLUEBOX_MIFARE_1k_BLOCK_SIZE

Name: BLUEBOX_MIFARE_1k_BLOCK_SIZE
Description: MIFARE 1k tag's memory block size in bytes.
Syntax #define BLUEBOX_MIFARE_1k_BLOCK_SIZE (16)

2.3.14 BLUEBOX_MIFARE_4k_UID_SIZE

Name: BLUEBOX_MIFARE_4k_UID_SIZE
Description: MIFARE 4k tag's UID size in bytes.
Syntax #define BLUEBOX_MIFARE_4k_UID_SIZE (4)

2.3.15 BLUEBOX_MIFARE_4k_BLOCK_SIZE

Name: BLUEBOX_MIFARE_4k_BLOCK_SIZE
Description: MIFARE 4k tag's memory block size in bytes.
Syntax #define BLUEBOX_MIFARE_4k_BLOCK_SIZE (16)

2.3.16 BLUEBOX_MIFARE_UL_UID_SIZE

Name: BLUEBOX_MIFARE_UL_UID_SIZE
Description: MIFARE Ultralight tag's UID size in bytes.
Syntax #define BLUEBOX_MIFARE_UL_UID_SIZE (7)

2.3.17 BLUEBOX_MIFARE_UL_BLOCK_SIZE

Name: BLUEBOX_MIFARE_UL_BLOCK_SIZE
Description: MIFARE Ultralight tag's memory block size in bytes.
Syntax #define BLUEBOX_MIFARE_UL_BLOCK_SIZE (4)

2.3.18 BLUEBOX_MIFARE_KEY_SIZE

Name: BLUEBOX_MIFARE_KEY_SIZE
Description: MIFARE tag's key size in bytes.
Syntax #define BLUEBOX_MIFARE_KEY_SIZE (6)

2.3.19 BLUEBOX_SR176_UID_SIZE

Name: BLUEBOX_SR176_UID_SIZE
Description: SR176 tag's UID size in bytes.

Syntax #define BLUEBOX_SR176_UID_SIZE (8)

2.3.20 BLUEBOX_SR176_BLOCK_SIZE

Name: BLUEBOX_SR176_BLOCK_SIZE
Description: SR176 tag's memory block size in bytes.
Syntax #define BLUEBOX_SR176_BLOCK_SIZE (2)

2.3.21 BLUEBOX_ISO18K6B_UID_SIZE

Name: BLUEBOX_ISO18K6B_UID_SIZE
Description: ISO 18000-6B tag's UID size in bytes.
Syntax #define BLUEBOX_ISO18K6B_UID_SIZE (8)

2.3.22 BLUEBOX_ISO18K6B_BLOCK_SIZE

Name: BLUEBOX_ISO18K6B_BLOCK_SIZE
Description: ISO 18000-6B tag's memory block size in bytes.
Syntax #define BLUEBOX_ISO18K6B_BLOCK_SIZE (8)

2.3.23 BLUEBOX_ISO18K6C_UID_SIZE

Name: BLUEBOX_ISO18K6C_UID_SIZE
Description: ISO 18000-6C tag's UID maximum size in bytes.
Syntax #define BLUEBOX_ISO18K6C_UID_SIZE (20)

2.3.24 BLUEBOX_ISO18K6C_BLOCK_SIZE

Name: BLUEBOX_ISO18K6C_BLOCK_SIZE
Description: ISO 18000-6C tag's memory block size in bytes.
Syntax #define BLUEBOX_ISO18K6C_BLOCK_SIZE (2)

2.3.25 BLUEBOX_ISO18K6C_ACC_PWD_SIZE

Name: BLUEBOX_ISO18K6C_ACC_PWD_SIZE
Description: ISO 18000-6C tag's access password size in bytes.
Syntax #define BLUEBOX_ISO18K6C_ACC_PWD_SIZE (4)

2.3.26 BLUEBOX_ISO18K6C_KILL_PWD_SIZE

Name: BLUEBOX_ISO18K6C_KILL_PWD_SIZE
Description: ISO 18000-6C tag's kill password size in bytes.

Syntax #define BLUEBOX_ISO18K6C_KILL_PWD_SIZE (4)

2.3.27 BLUEBOX_ACTIVE_UID_SIZE

Name: BLUEBOX_ACTIVE_UID_SIZE
Description: ACTIVE tag's UID size in bytes.
Syntax #define BLUEBOX_ACTIVE_UID_SIZE (8)

2.4 Data Structures

2.4.1 BLUEBOX_Tag

Name: BLUEBOX_Tag
Description: Tag identification struct.
Data fields:
TagType: Tag type.
Id: Pointer to tag ID.
Length: The length of the tag ID in bytes.
Antenna: The antenna that identifies the tag.
Input: The input (direction) of tag identification.

Syntax

```
typedef struct BLUEBOX_Tag
{
        BLUEBOX_TagType TagType;
        unsigned char *Id;
        int Length;
        BLUEBOX_Antenna Antenna;
        BLUEBOX_Input Input;

} BLUEBOX_Tag;
```

2.4.2 BLUEBOX_Notify

Name: BLUEBOX_Notify
Description: Tag notification struct.
Data fields:
Address: The address of the reader that identifies the tag.
TagType: Tag type.
Id: Pointer to the tag ID.
Length: The length of the tag ID in bytes.
Antenna: The antenna which have identified the tag.
Input: The input (direction) of tag identification.

Syntax

```
typedef struct BLUEBOX_Tag
{
        unsigned char Address;
        BLUEBOX_TagType TagType;
        unsigned char *Id;
        int Length;
        BLUEBOX_Antenna Antenna;
        BLUEBOX_Input Input;

} BLUEBOX_Tag;
```

2.4.3 BLUEBOX_ICODE_SLI_S_BlockProtectionStatus

Name: BLUEBOX_ICODE_SLI_S_BlockProtectionStatus

Description: ICODE SLI-S block protection status struct.

Data fields: *Lock*: Lock bit (Write access condition).
ReadPasswordProtected: Read password protected.
WritePasswordProtected: Write password protected.
PageProtectionLock: Page protection lock.

Syntax

```
typedef struct BLUEBOX_ICODE_SLI_S_BlockProtectionStatus
{
        int LockBit;
        int ReadPasswordProtected;
        int WritePasswordProtected;
        int PageProtectionLock;

} BLUEBOX_ICODE_SLI_S_BlockProtectionStatus;
```

2.4.4 BLUEBOX_Registration

Name: BLUEBOX_Registration

Description: Registration struct.

Data fields: *TagType*: Tag type.
Id: Pointer to the tag ID.
Length: The length of the tag ID in bytes.
Antenna: The antenna which have identified the tag.
Input: The input which have activated the identification procedure.

Syntax

```
typedef struct BLUEBOX_Registration
{
        BLUEBOX_TagType TagType;
        unsigned char *Id;
        int Length;
        BLUEBOX_Antenna Antenna;
        BLUEBOX_Input Input;
```

} BLUEBOX_Registration;

2.5 Functions

2.5.1 BLUEBOX_GetSwRelease

Name:	BLUEBOX_GetSwRelease
Reader:	All readers.
Description:	This function gets the software release of the library.
Parameters:	[out] SwRel: Software release of the library.
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk.
Syntax:	BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_GetSwRelease (char *SwRel);

2.5.2 BLUEBOX_Init

Name:	BLUEBOX_Init
Reader:	All readers.
Description:	This function creates an opaque handle to identify a module attached to PC.
Parameters:	[out] Handle: The handle that identifies the reader.
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InitError.
Syntax:	BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Init (BLUEBOX_Handle *Handle);

2.5.3 BLUEBOX_End

Name:	BLUEBOX_End
Reader:	All readers.
Description:	This function notifies the library the end of operation and frees the allocated memory. Implicity calls the

	BLUEBOX_Close function if the connection with the reader is open.
Parameters:	[in] Handle: The handle that identifies the reader.
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle.
Syntax:	BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_End (BLUEBOX_Handle *Handle);

2.5.4 BLUEBOX_Open

Name:	BLUEBOX_Open
Reader:	All readers.
Description:	This function opens the connection with the reader. If already connected it tries to close the connection and then opens it.
Parameters:	[in] Handle: The handle that identifies the reader.
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle. BLUEBOX_ConnectionError. BLUEBOX_GenericError.
Syntax:	BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Open (BLUEBOX_Handle *Handle);

2.5.5 BLUEBOX_Close

Name:	BLUEBOX_Close
Reader:	All readers.
Description:	This function closes the connection with the reader.
Parameters:	[in] Handle: The handle that identifies the reader.
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle. BLUEBOX_ConnectionError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Close (BLUEBOX_Handle *Handle);

2.5.6 BLUEBOX_SetAddress

Name: BLUEBOX_SetAddress

Reader: All readers.

Description: This function sets the reader address.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Address: Address to use to communicate with the reader.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_SetAddress (BLUEBOX_Handle *Handle, unsigned char Address);

2.5.7 BLUEBOX_SetDevice

Name: BLUEBOX_SetDevice

Reader: All readers.

Description: This function sets the reader type, frequency, range, and other parameters needed to communicate correctly with the reader

Parameters: [in] Handle: The handle that identifies the reader.
[in] Type: The reader type string. Use one of the strings listed below:
“DESKTOP”: Desktop reader.
“INDUSTRIAL”: Industrial reader.
“TINYOEM”: OEM reader like OEM HF or OEM LF.
“EASYOEM”: OEM reader like OEM HF E.
“PORTAL”: Portal reader.
“BB2 DESKTOP”: Gen2 desktop reader.
“BB2 INDUSTRIAL”: Gen2 industrial reader.
“BB2 BASIC”: Gen2 basic reader.
[in] Frequency: The reader frequency string. Use one of the strings listed below:
“LF”: Low Frequency (125 kHz).
“HF”: High Frequency (13.56 MHz).
“UHF”: Ultra High Frequency (860 – 960 MHz).

	“MICROWAVE”: Microwave 2.4 GHz. [in] Range: The reader range string. Use one of the strings listed below: “SHORT”: Short range. “MID”: Mid range. “LONG”: Long range. [in] Antennas: The reader antennas string. Use one of the values listed below: “SINGLE”: Single antenna. “DUAL”: Dual antennas. “QUAD”: Quad antennas. [in] Major: The firmware version major number. [in] Minor: The firmware version minor number. [in] Variant: The firmware variant. One char which identifies the firmware variant.
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle. BLUEBOX_InvalidParams.
Syntax:	BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_SetDevice (BLUEBOX_Handle *Handle, char *Type, char *Frequency, char *Range, char *Antennas, int Major, int Minor, char Variant);
Remarks:	The type, frequency, range and antennas permitted combinations are the following (N stands for NULL string): “DESKTOP”, “LF”, N, N: DESKTOP LF. “DESKTOP”, “HF”, N, N: DESKTOP HF. “INDUSTRIAL”, “LF”, “SHORT”, “SINGLE”: INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL. “INDUSTRIAL”, “LF”, “SHORT”, “DUAL”: INDUSTRIAL LF SHORT RANGE DUAL CHANNEL. “INDUSTRIAL”, “LF”, “LONG”, “SINGLE”: INDUSTRIAL LF LONG RANGE SINGLE CHANNEL. “INDUSTRIAL”, “HF”, “SHORT”, “SINGLE”: INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL. “INDUSTRIAL”, “HF”, “SHORT”, “DUAL”: INDUSTRIAL HF SHORT RANGE DUAL CHANNEL. “INDUSTRIAL”, “HF”, “MID”, “SINGLE”: INDUSTRIAL HF MID RANGE SINGLE CHANNEL. “INDUSTRIAL”, “HF”, “LONG”, “SINGLE”: INDUSTRIAL HF LONG RANGE SINGLE CHANNEL. “INDUSTRIAL”, “HF”, “LONG”, “QUAD”: INDUSTRIAL HF

LONG RANGE QUAD CHANNEL.
“INDUSTRIAL”, “UHF”, “SHORT”, “SINGLE”: INDUSTRIAL UHF SHORT RANGE SINGLE CHANNEL.
“INDUSTRIAL”, “UHF”, “MID”, “SINGLE”: INDUSTRIAL UHF MID RANGE SINGLE CHANNEL.
“INDUSTRIAL”, “UHF”, “LONG”, “QUAD”: INDUSTRIAL UHF LONG RANGE QUAD CHANNEL.
“INDUSTRIAL”, “MICROWAVE”, N, N: INDUSTRIAL ACTIVE.
“TINYOEM”, “LF”, N, N: OEM LF.
“TINYOEM”, “HF”, N, N: OEM HF.
“EASYOEM”, “HF”, N, N: OEM HF E.
“PORTAL”, “UHF”: PORTAL UHF.

2.5.8 BLUEBOX_GetDevice

Name: BLUEBOX_GetDevice

Reader: All readers.

Description: This function gets the reader type, frequency, range, and other parameters needed to communicate correctly with the reader

Parameters: [in] Handle: The handle that identifies the reader.
[out] Type: The reader type string. One of the strings listed below:
“DESKTOP”: Desktop reader.
“INDUSTRIAL”: Industrial reader.
“TINYOEM”: OEM reader like OEM HF or OEM LF.
“EASYOEM”: OEM reader like OEM HF E.
“PORTAL”: PORTAL reader.
“BB2 DESKTOP”: Gen2 desktop reader.
“BB2 INDUSTRIAL”: Gen2 industrial reader.
“BB2 BASIC”: Gen2 basic reader.
[out] Frequency: The reader frequency string. One of the strings listed below:
“LF”: Low Frequency (125 kHz).
“HF”: High Frequency (13.56 MHz).
“UHF”: Ultra High Frequency (860 – 960 MHz).
“MICROWAVE”: Microwave 2.4 GHz.
[out] Range: The reader range string. One of the strings listed below:
“SHORT”: Short range.
“MID”: Mid range.
“LONG”: Long range.
[out] Antennas: The reader antennas string. One of the values listed below:

	"SINGLE": Single antenna. "DUAL": Dual antennas. "QUAD": Quad antennas. [out] Major: The firmware version major number. [out] Minor: The firmware version minor number. [out] Variant: The firmware variant. One char which identifies the firmware variant.
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle.
Syntax:	BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_GetDevice (BLUEBOX_Handle *Handle, char *Type, char *Frequency, char *Range, char *Antennas, int *Major, int *Minor, char *Variant);

2.5.9 BLUEBOX_SetChannel

Name:	BLUEBOX_SetChannel
Reader:	All readers.
Description:	This function sets the notification channel to use with the reader.
Parameters:	[in] Handle: The handle that identifies the reader. [in] Channel: Channel to use string. Use one of the strings listed below: "RS232": RS232. "RS485": RS485. "TCP": TCP. [in] Settings: Channel settings string. Use one of the form listed below and depending on channel to use: RS232: "<port name>,<baud rate>,<data bits>,<parity>,<stop bits>,<retx>,<timeout>" (e.g. "COM1,19200,8,n,1,5,60000"). RS485: "<port name>,<baud rate>,<data bits>,<parity>,<stop bits>,<retx>,<timeout>" (e.g. "COM1,19200,8,n,1,5,60000"). TCP: "<ip>:<port>,<timeout>" (e.g. "192.168.4.200:3000,60000").
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle.

BLUEBOX_InvalidParams.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_SetChannel (BLUEBOX_Handle *Handle, char *Channel, char *Settings);

Remarks The <timeout> field is expressed in ms. RS232 is also used with USB Virtual Com interfaces.

2.5.10 BLUEBOX_GetFwRelease

Name: BLUEBOX_GetFwRelease

Reader: All readers.

Description: This function gets the firmware release of the reader.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Reader: The reader to read the version firmware. Use one of the values listed below and defined in BLUEBOX_Reader in BLUEBOXLib.h.
BLUEBOX_PRIMARY_READER: The primary reader.
BLUEBOX_AUXILIARY_1_READER: The 1st auxiliary reader.
BLUEBOX_AUXILIARY_2_READER: The 2nd auxiliary reader.
[out] FwRel: The firmware release of the reader.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_GetFwRelease (BLUEBOX_Handle *Handle, BLUEBOX_Reader Reader, char *FwRel);

Remarks This function must be called after opening the connection with the reader.

2.5.11 BLUEBOX_Reset

Name: BLUEBOX_Reset

Reader: All readers.

Description: This function resets the reader.

Parameters: [in] Handle: The handle that identifies the reader.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_GetReset (BLUEBOX_Handle *Handle);

2.5.12 BLUEBOX_ReadParameters

Name: BLUEBOX_ReadParameters

Reader: All readers.

Description: This function reads the general parameters of the reader.

Parameters: [in] Handle: The handle that identifies the reader.
[out] Parameters: General parameters set in the reader.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ReadParameters (BLUEBOX_Handle *Handle, unsigned char *Parameters);

Remarks This function must be called after opening the connection with the reader and after reading the firmware version.
See the reader technical manual for the Parameters format.

2.5.13 BLUEBOX_WriteParameters

Name: BLUEBOX_WriteParameters

Reader: All readers.

Description: This function writes the general parameters of the reader.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Parameters: General parameters to be set in the reader.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_WriteParameters (BLUEBOX_Handle *Handle, unsigned char *Parameters);

Remarks See the reader technical manual for the Parameters format.

2.5.14 BLUEBOX_DefaultParameters

Name: BLUEBOX_DefaultParameters

Reader: All readers.

Description: This function resets the parameters to the factory default values.

Parameters: [in] Handle: The handle that identifies the reader.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_DefaultParameters (BLUEBOX_Handle *Handle);

2.5.15 BLUEBOX_ReadConfiguration

Name: BLUEBOX_ReadConfiguration

Reader: All readers.

Description: This function reads a configuration page of the reader.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Page: The configuration page number to read.

[out] Config: Configuration set in the reader.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ReadConfiguration (BLUEBOX_Handle *Handle, int Page, unsigned char * Config);

Remarks The configuration buffer size in bytes depends on the page and it is 7 bytes length for pages between 00h and 7Fh, and 14 bytes length for pages between 80h and FFh.

2.5.16 BLUEBOX_WriteConfiguration

Name: BLUEBOX_WriteConfiguration

Reader: All readers.

Description: This function writes a configuration page of the reader.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Page: The configuration page number to write.
[in] Config: The configuration to be set in the reader.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_WriteConfiguration (BLUEBOX_Handle *Handle, int Page, unsigned char *Config);

Remarks The configuration buffer size in bytes depends on the page and it is 7 bytes length for pages between 00h and 7Fh, and 14 bytes length for pages between 80h and FFh.

2.5.17 BLUEBOX_DefaultConfiguration

Name: BLUEBOX_DefaultConfiguration

Reader: All readers.

Description: This function resets the configuration to the factory default values.

Parameters: [in] Handle: The handle that identifies the reader.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.

Syntax:
```
BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_DefaultConfiguration (BLUEBOX_Handle *Handle);
```

2.5.18 BLUEBOX_DataRequest

Name: BLUEBOX_DataRequest

Reader: All readers except of BLUEBOX PORTAL UHF.

Description: This function reads data from buffer. The tags array contain all the tag ID and other information related to every tag such as antenna and ID length.

Parameters: [in] Handle: The handle that identifies the reader.
[out] Tags: The array containing the tags read.
[out] TagsNo: The number of tags in the array.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_MemoryError.
BLUEBOX_TagNotFound.

Syntax:
```
BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_DataRequest (BLUEBOX_Handle *Handle,
```

BLUEBOX_Tag **Tags, int *TagsNo);

2.5.19 BLUEBOX_QueueRequest

Name: BLUEBOX_QueueRequest

Reader: All readers except of BLUEBOX PORTAL UHF.

Description: This function reads data from queue. The tags array contain all the tag ID and other information related to every tag such as antenna and ID length.

Parameters: [in] Handle: The handle that identifies the reader.
[out] Tags: The array containing the tags read.
[out] TagsNo: The number of tags in the array.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_MemoryError.
BLUEBOX_TagNotFound.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_QueueRequest (BLUEBOX_Handle *Handle, BLUEBOX_Tag **Tags, int *TagsNo);

2.5.20 BLUEBOX_FreeTagsMemory

Name: BLUEBOX_FreeTagsMemory

Reader: All readers except of BLUEBOX PORTAL UHF.

Description: This function frees the memory allocated to store tags by BLUEBOX_DataRequest or BLUEBOX_QueueRequest or inventoy commands.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Tags: The array containing the tags read.
[in] TagsNo: The number of tags in the array.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall

BLUEBOX_FreeTagsMemory (BLUEBOX_Handle *Handle, BLUEBOX_Tag **Tags, int *TagsNo);

2.5.21 BLUEBOX_AllocateNotifyChannel

Name: BLUEBOX_AllocateNotifyChannel

Reader: All readers except of BLUEBOX PORTAL UHF.

Description: This function allocates a notify channel in order to start a tag notification

Parameters: [in] Handle: The handle that identifies the reader.
[in] Address: The address of the reader.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidParams.
BLUEBOX_AllocationError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_AllocateNotifyChannel (BLUEBOX_Handle *Handle, unsigned char Address);

2.5.22 BLUEBOX_DeallocateNotifyChannel

Name: BLUEBOX_DeallocateNotifyChannel

Reader: All readers except of BLUEBOX PORTAL UHF.

Description: This function deallocates a notify channel in order to stop a tag notification

Parameters: [in] Handle: The handle that identifies the reader.
[in] Address: The address of the reader.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidChannel.
BLUEBOX_AllocationError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_DeallocateNotifyChannel (BLUEBOX_Handle *Handle, unsigned char Address);

2.5.23 BLUEBOX_GetNotification

Name: BLUEBOX_GetNotification

Reader: All readers except of BLUEBOX PORTAL UHF.

Description: This function reads data from notification buffer. The tags array contain all the tag ID and other information related to every tag such as antenna and ID length.

Parameters: [in] Handle: The handle that identifies the reader.
[out] Tags: The array containing the tags read.
[out] TagsNo: The number of tags in the array.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_MemoryError.
BLUEBOX_TagNotFound.
BLUEBOX_AllocationError.

Syntax:
```
BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_GetNotification (BLUEBOX_Handle *Handle, BLUEBOX_Notify **Tags, int *TagsNo);
```

2.5.24 BLUEBOX_FreeNotifyMemory

Name: BLUEBOX_FreeNotifyMemory

Reader: All readers except of BLUEBOX PORTAL UHF.

Description: This function frees the memory allocated to store tags by BLUEBOX_GetNotification commands.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Tags: The array containing the tags read.
[in] TagsNo: The number of tags in the array.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.

Syntax:
```
BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_FreeNotifyMemory (BLUEBOX_Handle *Handle, BLUEBOX_Notify **Tags, int *TagsNo);
```

2.5.25 BLUEBOX_SetOutput

Name:	BLUEBOX_SetOutput
Reader:	BLUEBOX OEM, BLUEBOX INDUSTRIAL, BLUEBOX GEN2 INDUSTRIAL and BASIC readers.
Description:	This function sets an output behavior.
Parameters:	[in] Handle: The handle that identifies the reader. [in] Output: The value that identifies the output. Use one of the values defined in BLUEBOX_Output in BLUEBOXLib.h. [in] Period: Activation period from 1 (0x01) to 99 (0x63) seconds. Use 0x80 to continuously deactivate the output and 0x81 to continuously activate the output.
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle. BLUEBOX_ConnectionError. BLUEBOX_InvalidCommand. BLUEBOX_TimeoutError. BLUEBOX_CommunicationError. BLUEBOX_InvalidParams.
Syntax:	BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_SetOutput (BLUEBOX_Handle *Handle, BLUEBOX_Output Output, unsigned char Period);

2.5.26 BLUEBOX_GetReaderStatus

Name:	BLUEBOX_GetReaderStatus
Reader:	All readers.
Description:	This function reads the status of the reader.
Parameters:	[in] Handle: The handle that identifies the reader. [out] Status: The status of the reader.
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle.

BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_GetReaderStatus (BLUEBOX_Handle *Handle, BLUEBOX_ReaderStatus *Status);

Remarks See the reader technical manual for the Status format.

2.5.27 BLUEBOX_RfOnOff

Name: BLUEBOX_RfOnOff

Reader: All readers except of BLUEBOX PORTAL UHF.

Description: This function sets RF ON/OFF.

Parameters: [in] Handle: The handle that identifies the reader.
[out] OnOff: Flag to activate/deactivate the RF. Use one of the values listed below:
= 0: To deactivate the RF.
!= 0: To activate the RF.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_InvalidParams.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_RfOnOff (BLUEBOX_Handle *Handle, short OnOff);

2.5.28 BLUEBOX_ReadID_EM4305

Name: BLUEBOX_ReadID_EM4305

Reader: BLUEBOX DESKTOP LF, BLUEBOX OEM LF, BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP LF, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to read the ID of an EM4305 tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to read the tag's ID. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[out] TagType: One of the values listed below and defined in BLUEBOX_TagType enum in BLUEBOXLib.h:
BLUEBOX_EM4305: EM4305.
[out] Data: The data read from the tag's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ReadID_EM4305 (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, BLUEBOX_TagType *TagType, void *Data);

2.5.29 BLUEBOX_Write_EM4305

Name: BLUEBOX_Write_EM4305

Reader: BLUEBOX DESKTOP LF, BLUEBOX OEM LF, BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP LF, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to writes a EM4305 tag with one of the codes defined below:
BLUEBOX SHORT: 5 bytes, UNIQUE equivalent.
BLUEBOX MEDIUM: 10 bytes.
BLUEBOX LONG: 20 bytes.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to write the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[in] Data: The data to write in the tag's memory.
[in] Length: The number of bytes to write. Allowed values are 5, 10 and 20.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Write_EM4305 (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Data, int Length);

2.5.30 BLUEBOX_ReadID_T5557

Name: BLUEBOX_ReadID_T5557

Reader: BLUEBOX DESKTOP LF, BLUEBOX OEM LF, BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP LF, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to read the ID of a T5557 tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to read the tag's ID. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[out] TagType: One of the values listed below and defined in BLUEBOX_TagType enum in BLUEBOXLib.h:
BLUEBOX_T5557: T5557.

[out] Data: The data read from the tag's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ReadID_T5557 (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, BLUEBOX_TagType *TagType, void *Data);

2.5.31 BLUEBOX_Write_T5557

Name: BLUEBOX_Write_T5557

Reader: BLUEBOX DESKTOP LF, BLUEBOX OEM LF, BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP LF, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to writes a T5557 tag with one of the codes defined below:
BLUEBOX SHORT: 5 bytes, UNIQUE equivalent.
BLUEBOX MEDIUM: 10 bytes.
BLUEBOX LONG: 20 bytes.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to write the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[in] Data: The data to write in the tag's memory.
[in] Length: The number of bytes to write. Allowed values are 5, 10 and 20.

Return: An error code about the execution of the function. One of

	the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle. BLUEBOX_ConnectionError. BLUEBOX_InvalidCommand. BLUEBOX_TimeoutError. BLUEBOX_CommunicationError. BLUEBOX_InvalidParams.
Syntax:	BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Write_T5557 (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Data, int Length);

2.5.32 BLUEBOX_ReadID_Q5

Name:	BLUEBOX_ReadID_Q5
Reader:	BLUEBOX DESKTOP LF, BLUEBOX OEM LF, BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP LF, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL.
Description:	This function allows to read the ID of a Q5 tag.
Parameters:	[in] Handle: The handle that identifies the reader. [in] Antenna: The antenna to use to read the tag's ID. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h. BLUEBOX_ANT_1: Antenna 1. BLUEBOX_ANT_2: Antenna 2. [out] TagType: One of the values listed below and defined in BLUEBOX_TagType enum in BLUEBOXLib.h: BLUEBOX_Q5: Q5. [out] Data: The data read from the tag's memory.
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle. BLUEBOX_ConnectionError. BLUEBOX_InvalidCommand. BLUEBOX_TimeoutError. BLUEBOX_CommunicationError.

BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ReadID_Q5 (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, BLUEBOX_TagType *TagType, void *Data);

2.5.33 BLUEBOX_Write_Q5

Name: BLUEBOX_Write_Q5

Reader: BLUEBOX DESKTOP LF, BLUEBOX OEM LF, BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP LF, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to writes a Q5 tag with one of the codes defined below:
BLUEBOX SHORT: 5 bytes, UNIQUE equivalent.
BLUEBOX MEDIUM: 10 bytes.
BLUEBOX LONG: 20 bytes.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to write the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[in] Data: The data to write in the tag's memory.
[in] Length: The number of bytes to write. Allowed values are 5, 10 and 20.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall

BLUEBOX_Write_Q5 (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Data, int Length);

2.5.34 BLUEBOX_ReadID_HITAGS

Name: BLUEBOX_ReadID_HITAGS

Reader: BLUEBOX DESKTOP LF, BLUEBOX OEM LF, BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP LF, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to read the ID of a HITAG S tag (UID).

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to read the tag's ID. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[out] TagType: One of the values listed below and defined in BLUEBOX_TagType enum in BLUEBOXLib.h:
BLUEBOX_HITAG_S256: HITAG S 256.
BLUEBOX_HITAG_S2048: HITAG S 2048.
[out] Data: The data read from the tag's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ReadID_HITAGS (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, BLUEBOX_TagType *TagType, void *Data);

2.5.35 BLUEBOX_Write_HITAGS

Name: BLUEBOX_Write_HITAGS

Reader: BLUEBOX DESKTOP LF, BLUEBOX OEM LF, BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP LF, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to writes a HITAG S tag with one of the codes defined below:
BLUEBOX SHORT: 5 bytes, UNIQUE equivalent.
BLUEBOX MEDIUM: 10 bytes.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to write the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[in] Data: The data to write in the tag's memory.
[in] Length: The number of bytes to write. Allowed values are 5 and 10.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Write_HITAGS (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Data, int Length);

2.5.36 BLUEBOX_ReadPage_HITAGS

Name: BLUEBOX_ReadPage_HITAGS

Reader: BLUEBOX DESKTOP LF, BLUEBOX OEM LF, BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL LF SHORT RANGE DUAL CHANNEL,

	BLUEBOX GEN2 DESKTOP LF, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL.
Description:	This function allows to read a page of a HITAG S tag.
Parameters:	[in] Handle: The handle that identifies the reader. [in] Antenna: The antenna to use to read the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h. BLUEBOX_ANT_1: Antenna 1. BLUEBOX_ANT_2: Antenna 2. [in] Id: The ID of the tag to read. [in] Page: The page of the tag's memory to read (0 – 63). [in] Data: The data read from the tag's memory.
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle. BLUEBOX_ConnectionError. BLUEBOX_InvalidCommand. BLUEBOX_TimeoutError. BLUEBOX_CommunicationError. BLUEBOX_InvalidParams. BLUEBOX_TagNotFound. BLUEBOX_TagError.
Syntax:	BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ReadPage_HITAGS (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, int Page, void *Data);

2.5.37 BLUEBOX_WritePage_HITAGS

Name:	BLUEBOX_WritePage_HITAGS
Reader:	BLUEBOX DESKTOP LF, BLUEBOX OEM LF, BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP LF, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to write a page of a HITAG S tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to write the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[in] Id: The ID of the tag to write.
[in] Page: The page of the tag's memory to write (0 – 63).
[in] Data: The data written to the tag's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_WritePage_HITAGS (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, int Page, void *Data);

2.5.38 BLUEBOX_ReadID_TITAN

Name: BLUEBOX_ReadID_TITAN

Reader: BLUEBOX DESKTOP LF, BLUEBOX OEM LF, BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP LF, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to read the ID of a TITAN tag (ID + SN).

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to read the tag's ID. Use one of the values listed below and defined in

BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[out] TagType: One of the values listed below and defined in BLUEBOX_TagType enum in BLUEBOXLib.h:
BLUEBOX_TITAN: TITAN.
[out] Data: The data read from the tag's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ReadID_TITAN (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, BLUEBOX_TagType *TagType, void *Data);

2.5.39 BLUEBOX_Reset_TITAN

Name: BLUEBOX_Reset_TITAN

Reader: BLUEBOX DESKTOP LF, BLUEBOX OEM LF, BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP LF, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to reset a TITAN tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to reset the tag. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.

Return: An error code about the execution of the function. One of the values listed below and defined in

BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Reset_TITAN (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna);

2.5.40 BLUEBOX_Login_TITAN

Name: BLUEBOX_Login_TITAN

Reader: BLUEBOX DESKTOP LF, BLUEBOX OEM LF, BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP LF, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to login a TITAN tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to login the tag. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[in] Password: The password to use to login the tag.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.

BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Login_TITAN (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Password);

2.5.41 BLUEBOX_WritePassword_TITAN

Name: BLUEBOX_WritePassword_TITAN

Reader: BLUEBOX DESKTOP LF, BLUEBOX OEM LF, BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP LF, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to write the password of a TITAN tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to write the tag's password. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[in] OldPwd: The old password of the tag.
[in] NewPwd: The new password of the tag.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_WritePassword_TITAN (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *OldPwd, void *NewPwd);

2.5.42 BLUEBOX_SelectiveRead_TITAN

Name: BLUEBOX_SelectiveRead_TITAN

Reader: BLUEBOX DESKTOP LF, BLUEBOX OEM LF, BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP LF, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to selective read of a TITAN tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to read the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[in] Address: The address of the memory to read.
[in] Length: The number of words (4 bytes length) to read from the tag's memory.
[out] Data: The data read from the tag's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_SelectiveRead_TITAN (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, int Address, int Length, void *Data);

2.5.43 BLUEBOX_SelectiveWrite_TITAN

Name: BLUEBOX_SelectiveWrite_TITAN

Reader: BLUEBOX DESKTOP LF, BLUEBOX OEM LF, BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP LF, BLUEBOX GEN2

INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to selective write of a TITAN tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to write the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[in] Address: The address of the memory to write.
[in] Length: The number of words (4 bytes length) to write to the tag's memory.
[in] Data: The data to write to the tag's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax:
```
BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_SelectiveWrite_TITAN (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, int Address, int Length, void *Data);
```

2.5.44 BLUEBOX_Inventory_ISO15693

Name: BLUEBOX_Inventory_ISO15693

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX INDUSTRIAL HF MID RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF LONG RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF LONG RANGE QUAD CHANNEL, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF MID RANGE

SINGLE CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.

Description: This function sends an inventory command with anticollision to read all the ISO 15693 tags. The tags array contain all the tag ID and other information related to every tag such as antenna and ID length.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to inventory the tag's. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_NOANT: No antenna info. Use only with quad channel readers.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[in] UseAFI: Flag to use the AFI in inventory procedure. Use one of the values listed below:
== 0: To not use AFI.
!= 0: To use AFI.
[in] AFI: The AFI field to use in inventory procedure. It's an optional parameter needed when UseAFI != 0.
[out] Tags: The array containing the tags read.
[out] TagsNo: The number of tags in the array.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Inventory_ISO15693 (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, short UseAFI, unsigned char AFI, BLUEBOX_Tag **Tags, int *TagsNo);

2.5.45 BLUEBOX_ReadPage_ISO15693

Name: BLUEBOX_ReadPage_ISO15693

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF SHORT RANGE DUAL

CHANNEL, BLUEBOX INDUSTRIAL HF MID RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF LONG RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF LONG RANGE QUAD CHANNEL, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF MID RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to read a page of a ISO 15693 tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to read the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_NOANT: No antenna info. Use only with quad channel readers.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[in] Id: The ID of the tag to read.
[in] Page: The page of the tag's memory to read (0 – 255).
[in] Size: The size of the page to read in bytes (4, 8).
[out] Data: The data read from the tag's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ReadPage_ISO15693 (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, int Page, int Size, void *Data);

2.5.46 BLUEBOX_WritePage_ISO15693

Name: BLUEBOX_WritePage_ISO15693

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL,

	BLUEBOX INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX INDUSTRIAL HF MID RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF LONG RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF LONG RANGE QUAD CHANNEL, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF MID RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.
Description:	This function allows to write a page of a ISO 15693 tag.
Parameters:	[in] Handle: The handle that identifies the reader. [in] Antenna: The antenna to use to write the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h. BLUEBOX_NOANT: No antenna info. Use only with quad channel readers. BLUEBOX_ANT_1: Antenna 1. BLUEBOX_ANT_2: Antenna 2. [in] Id: The ID of the tag to write. [in] Page: The page of the tag's memory to write (0 – 255). [in] Size: The size of the page to write in bytes (4, 8). [in] Data: The data to write to the tag's memory.
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle. BLUEBOX_ConnectionError. BLUEBOX_InvalidCommand. BLUEBOX_TimeoutError. BLUEBOX_CommunicationError. BLUEBOX_InvalidParams. BLUEBOX_TagNotFound. BLUEBOX_TagError.
Syntax:	BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_WritePage_ISO15693 (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, int Page, int Size, void *Data);

2.5.47 BLUEBOX_LockPage_ISO15693

Name:	BLUEBOX_LockPage_ISO15693
Reader:	BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX

	INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX INDUSTRIAL HF MID RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF LONG RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF LONG RANGE QUAD CHANNEL, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF MID RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.
Description:	This function allows to lock a page of a ISO 15693 tag.
Parameters:	[in] Handle: The handle that identifies the reader. [in] Antenna: The antenna to use to lock the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h. BLUEBOX_NOANT: No antenna info. Use only with quad channel readers. BLUEBOX_ANT_1: Antenna 1. BLUEBOX_ANT_2: Antenna 2. [in] Id: The ID of the tag to lock. [in] Page: The page of the tag's memory to lock (0 – 255).
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle. BLUEBOX_ConnectionError. BLUEBOX_InvalidCommand. BLUEBOX_TimeoutError. BLUEBOX_CommunicationError. BLUEBOX_InvalidParams. BLUEBOX_TagNotFound. BLUEBOX_TagError.
Syntax:	BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_LockPage_ISO15693 (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, int Page);

2.5.48 BLUEBOX_Write_AFI_ISO15693

Name:	BLUEBOX_Write_AFI_ISO15693
Reader:	BLUEBOX INDUSTRIAL HF MID RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF MID RANGE SINGLE

CHANNEL.

Description: This function allows to write the AFI of a ISO 15693 tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to write the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
[in] Id: The ID of the tag to write.
[in] Afi: The tag's AFI to be written.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Write_AFI_ISO15693 (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, unsigned char Afi);

2.5.49 BLUEBOX_Lock_AFI_ISO15693

Name: BLUEBOX_Lock_AFI_ISO15693

Reader: BLUEBOX INDUSTRIAL HF MID RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF MID RANGE SINGLE CHANNEL.

Description: This function allows to lock the AFI of a ISO 15693 tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to lock the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
[in] Id: The ID of the tag to lock.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.

BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Lock_AFI_ISO15693 (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id);

2.5.50 BLUEBOX_GetRandomNumber_ICODE_SLI_S

Name: BLUEBOX_GetRandomNumber_ICODE_SLI_S

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to get a random number from an ICODE SLI-S tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
[in] Id: The ID of the tag.
[out] Random: The random number received from the tag.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_GetRandomNumber_ICODE_SLI_S

(BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, void *Random);

2.5.51 BLUEBOX_SetPassword_ICODE_SLI_S

Name: BLUEBOX_SetPassword_ICODE_SLI_S

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to transmit a password to an ICODE SLI-S tag to get access to the different protected functionalities of the tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
[in] Id: The ID of the tag.
[in] PwdId: The password identifier which identifies the type of the password to transmit.
[in] Password: The password to transmit to the tag.
[in] Random: The random number previously received from the tag.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_SetPassword_ICODE_SLI_S (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, BLUEBOX_ICODE_SLI_S_PasswordIdentifier PwdId, void *Password, void *Random);

2.5.52 BLUEBOX_WritePassword_ICODE_SLI_S

Name: BLUEBOX_WritePassword_ICODE_SLI_S

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to write a new password to an ICODE SLI-S tag if the related old password has already been transmitted with a Set Password command before and the addressed password is not locked.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
[in] Id: The ID of the tag.
[in] PwdId: The password identifier which identifies the type of the password to write.
[in] Password: The new password to write to the tag.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_WritePassword_ICODE_SLI_S (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, BLUEBOX_ICODE_SLI_S_PasswordIdentifier PwdId, void *Password);

2.5.53 BLUEBOX_LockPassword_ICODE_SLI_S

Name: BLUEBOX_LockPassword_ICODE_SLI_S

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX

GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to lock a password of an ICODE SLI-S tag if the related old password has already been transmitted with a Set Password command before and the addressed password is not locked.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
[in] Id: The ID of the tag.
[in] PwdId: The password identifier which identifies the type of the password to lock.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_LockPassword_ICODE_SLI_S (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, BLUEBOX_ICODE_SLI_S_PasswordIdentifier PwdId);

2.5.54 BLUEBOX_64BitPasswordProtection_ICODE_SLI_S

Name: BLUEBOX_64BitPasswordProtection_ICODE_SLI_S

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to activate the 64-bit password protection. This mode can be enabled if both the Read and Write password have already been transmitted with a Set Password command before.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
[in] Id: The ID of the tag.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_64BitPasswordProtection_ICODE_SLI_S (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id);

2.5.55 BLUEBOX_ProtectPage_ICODE_SLI_S

Name: BLUEBOX_ProtectPage_ICODE_SLI_S

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to change the protection condition of a page of an ICODE SLI-S if the related passwords have already been transmitted with a Set Password command before and the addressed page is not locked.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.

BLUEBOX_ANT_1: Antenna 1.
[in] Id: The ID of the tag.
[in] PageNo: The number of the page.
[in] Status: The protection status. Use one of the values defined in BLUEBOX_ICODE_SLI_S_ProtectionStatus enum.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ProtectPage_ICODE_SLI_S (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, int PageNo, BLUEBOX_ICODE_SLI_S_ProtectionStatus Status);

2.5.56 BLUEBOX_LockPageProtectionCondition_ICODE_SLI_S

Name: BLUEBOX_LockPageProtectionCondition_ICODE_SLI_S

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to lock the page protection condition of a page of an ICOE SLI-S if the related passwords have already been transmitted with a Set Password command before.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
[in] Id: The ID of the tag.
[in] PageNo: The number of the page.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_LockPageProtectionCondition_ICODE_SLI_S (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, int PageNo);

2.5.57 BLUEBOX_GetMultipleBlockProtectionStatus_ICODE_SLI_S

Name: BLUEBOX_GetMultipleBlockProtectionStatus_ICODE_SLI_S

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to get the block protection status of the requested blocks of an ICODE SLI-S.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
[in] Id: The ID of the tag.
[in] BlockNo: The number of the first block.
[in] Length: The number of blocks.
[out] Status: The block protection status of the requested blocks array.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.

BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_GetMultipleBlockProtectionStatus_ICODE_SLI_S (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, int BlockNo, int Length, BLUEBOX_ICODE_SLI_S_BlockProtectionStatus *Status);

2.5.58 BLUEBOX_Destroy_SLI_S_ICODE_SLI_S

Name: BLUEBOX_Destroy_SLI_S_ICODE_SLI_S

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to destroy an ICODE SLI-S tag. It can be destroyed if the Destroy SLI-S password has been transmitted before. This command is irreversible.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
[in] Id: The ID of the tag.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Destroy_SLI_S_ICODE_SLI_S (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna,

void *Id);

2.5.59 BLUEBOX_EnablePrivacy_ICODE_SLI_S

Name: BLUEBOX_EnablePrivacy_ICODE_SLI_S

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to set an ICODE SLI-S into Privacy mode if the Privacy password has already been transmitted before.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
[in] Id: The ID of the tag.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_EnablePrivacy_ICODE_SLI_S (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id);

2.5.60 BLUEBOX_Inventory_ISO14443A

Name: BLUEBOX_Inventory_ISO14443A

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX OEM HF E, BLUEBOX INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP HF,

	BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.
Description:	This function sends an inventory command with anticollision to read all ISO 14443A tags. The tags array contain all the tag ID and other information related to every tag such as antenna and ID length.
Parameters:	[in] Handle: The handle that identifies the reader. [in] Antenna: The antenna to use to inventory tags. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h. BLUEBOX_ANT_1: Antenna 1. BLUEBOX_ANT_2: Antenna 2. [out] Tags: The array containing the tags read. [out] TagsNo: The number of tags in the array.
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle. BLUEBOX_ConnectionError. BLUEBOX_InvalidCommand. BLUEBOX_TimeoutError. BLUEBOX_CommunicationError. BLUEBOX_InvalidParams. BLUEBOX_TagNotFound.
Syntax:	BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Inventory_ISO14443A (BLUEBOX_Handle *Handle, BLUEBOX_Tag **Tags, int *TagsNo);
Remarks	Read only one ISO 14443A tag.

2.5.61 BLUEBOX_ReadBlock_MIFARE_1k

Name:	BLUEBOX_ReadBlock_MIFARE_1k
Reader:	BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX OEM HF E, BLUEBOX INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.
Description:	This function allows to read a block of memory of a

MIFARE 1k tag.

Parameters:

[in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to read the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[in] Id: The ID of the tag to read.
[in] KeyType: The key type to use. Use one of the values listed below and defined in BLUEBOX_MifareKey enum in BLUEBOXLib.h.
BLUEBOX_MIFARE_KEY_A: Key A.
BLUEBOX_MIFARE_KEY_B: Key B.
[in] Key: The key to use to read the tag's memory.
[in] Block: The page of the tag's memory to read (0 – 63).
[out] Data: The data read from the tag's memory.

Return:

An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax:

```
BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ReadBlock_MIFARE_1k (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, BLUEBOX_MifareKey KeyType, void *Key, int Block, void *Data);
```

2.5.62 BLUEBOX_WriteBlock_MIFARE_1k

Name:

BLUEBOX_WriteBlock_MIFARE_1k

Reader:

BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX OEM HF E, BLUEBOX INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF

SHORT RANGE SINGLE CHANNEL.

Description: This function allows to write a block of memory of a MIFARE 1k tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to write the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[in] Id: The ID of the tag to write.
[in] KeyType: The key type to use. Use one of the values listed below and defined in BLUEBOX_MifareKey enum in BLUEBOXLib.h.
BLUEBOX_MIFARE_KEY_A: Key A.
BLUEBOX_MIFARE_KEY_B: Key B.
[in] Key: The key to use to write the tag's memory.
[in] Block: The page of the tag's memory to write (0 – 63).
[in] Data: The data to write to the tag's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_WriteBlock_MIFARE_1k (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, BLUEBOX_MifareKey KeyType, void *Key, int Block, void *Data);

2.5.63 BLUEBOX_ReadBlock_MIFARE_4k

Name: BLUEBOX_ReadBlock_MIFARE_4k

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX OEM HF E, BLUEBOX INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP HF,

	BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.
Description:	This function allows to read a block of memory of a MIFARE 4k tag.
Parameters:	[in] Handle: The handle that identifies the reader. [in] Antenna: The antenna to use to read the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h. BLUEBOX_ANT_1: Antenna 1. BLUEBOX_ANT_2: Antenna 2. [in] Id: The ID of the tag to read. [in] KeyType: The key type to use. Use one of the values listed below and defined in BLUEBOX_MifareKey enum in BLUEBOXLib.h. BLUEBOX_MIFARE_KEY_A: Key A. BLUEBOX_MIFARE_KEY_B: Key B. [in] Key: The key to use to read the tag's memory. [in] Block: The page of the tag's memory to read (0 – 255). [out] Data: The data read from the tag's memory.
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle. BLUEBOX_ConnectionError. BLUEBOX_InvalidCommand. BLUEBOX_TimeoutError. BLUEBOX_CommunicationError. BLUEBOX_InvalidParams. BLUEBOX_TagNotFound. BLUEBOX_TagError.
Syntax:	BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ReadBlock_MIFARE_4k (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, BLUEBOX_MifareKey KeyType, void *Key, int Block, void *Data);

2.5.64 BLUEBOX_WriteBlock_MIFARE_4k

Name:	BLUEBOX_WriteBlock_MIFARE_4k
Reader:	BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX

OEM HF E, BLUEBOX INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to write a block of memory of a MIFARE 4k tag.

Parameters:
[in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to write the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[in] Id: The ID of the tag to write.
[in] KeyType: The key type to use. Use one of the values listed below and defined in BLUEBOX_MifareKey enum in BLUEBOXLib.h.
BLUEBOX_MIFARE_KEY_A: Key A.
BLUEBOX_MIFARE_KEY_B: Key B.
[in] Key: The key to use to write the tag's memory.
[in] Block: The page of the tag's memory to write (0 – 255).
[in] Data: The data to write to the tag's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_WriteBlock_MIFARE_4k (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, BLUEBOX_MifareKey KeyType, void *Key, int Block, void *Data);

2.5.65 BLUEBOX_ReadBlock_MIFARE_Ultralight

Name: BLUEBOX_ReadBlock_MIFARE_Ultralight

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX OEM HF E, BLUEBOX INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to read a block of memory of a MIFARE Ultralight tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to read the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[in] Id: The ID of the tag to read.
[in] Block: The page of the tag's memory to read (0 – 15).
[out] Data: The data read from the tag's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ReadBlock_MIFARE_Ultralight (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, int Block, void *Data);

2.5.66 BLUEBOX_WriteBlock_MIFARE_Ultralight

Name: BLUEBOX_WriteBlock_MIFARE_Ultralight

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX OEM HF E, BLUEBOX INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF SHORT

RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to write a block of memory of a MIFARE Ultralight tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to write the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[in] Id: The ID of the tag to write.
[in] Block: The page of the tag's memory to write (0 – 15).
[in] Data: The data to write to the tag's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_WriteBlock_MIFARE_Ultralight (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, int Block, void *Data);

2.5.67 BLUEBOX_Inventory_ISO14443B

Name: BLUEBOX_Inventory_ISO14443B

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE

SINGLE CHANNEL.

Description: This function sends an inventory command with anticollision to read all ISO 14443B tags. The tags array contain all the tag ID and other information related to every tag such as antenna and ID length.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to inventory tags. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[out] Tags: The array containing the tags read.
[out] TagsNo: The number of tags in the array.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Inventory_ISO14443B (BLUEBOX_Handle *Handle, BLUEBOX_Tag **Tags, int *TagsNo);

Remarks Read only one ISO 14443B tag.

2.5.68 BLUEBOX_ReadBlock_SR176

Name: BLUEBOX_ReadBlock_SR176

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to read a block of memory of a SR176 tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to read the tag's

memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.
[in] Id: The ID of the tag to read.
[in] Block: The page of the tag's memory to read (0 – 63).
[out] Data: The data read from the tag's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ReadBlock_SR176 (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, int Block, void *Data);

2.5.69 BLUEBOX_WriteBlock_SR176

Name: BLUEBOX_WriteBlock_SR176

Reader: BLUEBOX DESKTOP HF, BLUEBOX OEM HF, BLUEBOX INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL.

Description: This function allows to write a block of memory of a SR176 tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Antenna: The antenna to use to write the tag's memory. Use one of the values listed below and defined in BLUEBOX_Antenna enum in BLUEBOXLib.h.
BLUEBOX_ANT_1: Antenna 1.
BLUEBOX_ANT_2: Antenna 2.

[in] Id: The ID of the tag to write.
[in] Block: The page of the tag's memory to write (0 – 63).
[in] Data: The data to write to the tag's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_WriteBlock_SR176 (BLUEBOX_Handle *Handle, BLUEBOX_Antenna Antenna, void *Id, int Block, void *Data);

2.5.70 BLUEBOX_ReadRfParameters

Name: BLUEBOX_ReadRfParameters

Reader: BLUEBOX INDUSTRIAL HF MID/LONG RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF LONG RANGE QUAND CHANNEL, BLUEBOX INDUSTRIAL UHF MID RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL UHF LONG RANGE QUAD CHANNEL, BLUEBOX INDUSTRIAL ACTIVE, BLUEBOX PORTAL UHF, BLUEBOX GEN2 INDUSTRIAL HF MID RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL UHF MID RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC UHF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 UHF MID RANGE SINGLE CHANNEL.

Description: This function reads the RF parameters of the reader.

Parameters: [in] Handle: The handle that identifies the reader.
[out] Parameters: RF parameters set in the reader.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.

BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ReadRfParameters (BLUEBOX_Handle *Handle, unsigned char *Parameters);

Remarks See the reader technical manual for the Parameters format.
This functions could be replaced with BLUEBOX_ReadConfiguration (2.5.15).

2.5.71 BLUEBOX_WriteRfParameters

Name: BLUEBOX_WriteRfParameters

Reader: BLUEBOX INDUSTRIAL HF MID/LONG RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL HF LONG RANGE QUAND CHANNEL, BLUEBOX INDUSTRIAL UHF MID RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL UHF LONG RANGE QUAD CHANNEL, BLUEBOX INDUSTRIAL ACTIVE, BLUEBOX PORTAL UHF, BLUEBOX GEN2 INDUSTRIAL HF MID RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL UHF MID RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC UHF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 UHF MID RANGE SINGLE CHANNEL.

Description: This function writes the RF parameters of the reader.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Parameters: RF parameters to be set in the reader.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_WriteRfParameters (BLUEBOX_Handle *Handle, unsigned char *Parameters);

Remarks See the reader technical manual for the Parameters format.
This functions could be replaced with

BLUEBOX_ReadConfiguration (2.5.16).

2.5.72 BLUEBOX_Inventory_ISO18K6B

Name: BLUEBOX_Inventory_ISO18K6B

Reader: BLUEBOX INDUSTRIAL UHF LONG RANGE QUAD CHANNEL.

Description: This function sends an inventory command with anticollision to read all the ISO 18000-6B tags.

Parameters: [in] Handle: The handle that identifies the reader.
[out] Tags: The array containing the tags read.
[out] TagsNo: The number of tags in the array.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Inventory_ISO18K6B (BLUEBOX_Handle *Handle, BLUEBOX_Tag **Tags, int *TagsNo);

2.5.73 BLUEBOX_Read_ISO18K6B

Name: BLUEBOX_Read_ISO18K6B

Reader: BLUEBOX INDUSTRIAL UHF LONG RANGE QUAD CHANNEL.

Description: This function allows to read the memory of a ISO 18000-6B tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Uid: The UID of the tag to be read.
[in] Address: The starting address of the tag's memory to be read.
[in] Nblocks: The number of 8-bytes blocks to be read (1 ... 8).
[out] Data: The data read from the tag's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in

BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Read_ISO18K6B (BLUEBOX_Handle *Handle, void *Uid, void *Pwd, BLUEBOX_ISO18K6C_Bank Bank, int Address, int Length, void *Data);

2.5.74 BLUEBOX_Write_ISO18K6B

Name: BLUEBOX_Write_ISO18K6B

Reader: BLUEBOX INDUSTRIAL UHF LONG RANGE QUAD CHANNEL.

Description: This function allows to write the memory of a ISO 18000-6B tag.

Parameters:
[in] Handle: The handle that identifies the reader.
[in] Uid: The UID of the tag to be written.
[in] Address: The starting address of the tag's memory to be written.
[in] Length: The number of bytes to be written (1 … 32).
[in] Data: The data to be written into the tag's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Write_ISO18K6B (BLUEBOX_Handle *Handle, void *Uid, int Address, int Length, void *Data);

2.5.75 BLUEBOX_Inventory_ISO18K6C

Name: BLUEBOX_Inventory_ISO18K6C

Reader: BLUEBOX INDUSTRIAL UHF MID RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL UHF LONG RANGE QUAD CHANNEL., BLUEBOX GEN2 DESKTOP UHF, BLUEBOX GEN2 INDUSTRIAL UHF MID RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC UHF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC UHF MID RANGE SINGLE CHANNEL.

Description: This function sends an inventory command with anticollision to read all the ISO 18000-6C tags.

Parameters: [in] Handle: The handle that identifies the reader.
[out] Tags: The array ☐ontaining the tags read.
[out] TagsNo: The number of tags in the array.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Inventory_ISO18K6C (BLUEBOX_Handle *Handle, BLUEBOX_Tag **Tags, int *TagsNo);

2.5.76 BLUEBOX_Read_ISO18K6C

Name: BLUEBOX_Read_ISO18K6C

Reader: BLUEBOX INDUSTRIAL UHF MID RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL UHF LONG RANGE QUAD CHANNEL., BLUEBOX GEN2 DESKTOP UHF, BLUEBOX GEN2 INDUSTRIAL UHF MID RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC UHF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC UHF MID RANGE SINGLE CHANNEL.

Description: This function allows to read the memory of a ISO 18000-6C tag.

Parameters: [in] Handle: The handle that identifies the reader.

[in] Uid: The UID of the tag to be read.
[in] Pwd: The access password to read the tag. Set to 0 if no password is required.
[in] Bank: The memory bank to be read. One of the values listed below and defined in BLUEBOX_ISO18K6C_Bank in BLUEBOXLib.h:
BLUEBOX_ISO18K6C_BANK_RESERVED: Reserved.
BLUEBOX_ISO18K6C_BANK_EPC: EPC.
BLUEBOX_ISO18K6C_BANK_TID: TID.
BLUEBOX_ISO18K6C_BANK_USER: User.
[in] Address: The starting address of the tag's memory to be read.
[in] Length: The number of 16-bits words to be read (1 … 4).
[out] Data: The data read from the tag's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Read_ISO18K6C (BLUEBOX_Handle *Handle, void *Uid, void *Pwd, BLUEBOX_ISO18K6C_Bank Bank, int Address, int Length, void *Data);

2.5.77 BLUEBOX_Write_ISO18K6C

Name: BLUEBOX_Write_ISO18K6C

Reader: BLUEBOX INDUSTRIAL UHF MID RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL UHF LONG RANGE QUAD CHANNEL., BLUEBOX GEN2 DESKTOP UHF, BLUEBOX GEN2 INDUSTRIAL UHF MID RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC UHF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC UHF MID RANGE SINGLE CHANNEL.

Description: This function allows to write the memory of a ISO 18000-6C tag.

Parameters:
[in] Handle: The handle that identifies the reader.
[in] Uid: The UID of the tag to be written.
[in] Pwd: The access password to write the tag. Set to 0 if no password is required.
[in] Bank: The memory bank to be written. One of the values listed below and defined in BLUEBOX_ISO18K6C_Bank in BLUEBOXLib.h:
BLUEBOX_ISO18K6C_BANK_RESERVED: Reserved.
BLUEBOX_ISO18K6C_BANK_EPC: EPC.
BLUEBOX_ISO18K6C_BANK_TID: TID.
BLUEBOX_ISO18K6C_BANK_USER: User.
[in] Address: The starting address of the tag's memory to be written.
[in] Length: The number of 2-bits words to be written (1 … 4).
[in] Data: The data to be written into the tag's memory.

Return:
An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax:
BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Write_ISO18K6C (BLUEBOX_Handle *Handle, void *Uid, void *Pwd, BLUEBOX_ISO18K6C_Bank Bank, int Address, int Length, void *Data);

2.5.78 BLUEBOX_Lock_ISO18K6C

Name:
BLUEBOX_Lock_ISO18K6C

Reader:
BLUEBOX INDUSTRIAL UHF MID RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL UHF LONG RANGE QUAD CHANNEL., BLUEBOX GEN2 DESKTOP UHF, BLUEBOX GEN2 INDUSTRIAL UHF MID RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC UHF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC UHF MID RANGE SINGLE CHANNEL.

Description:
This function allows to lock the password and memory of a ISO 18000-6C tag.

Parameters:

[in] Handle: The handle that identifies the reader.
[in] Uid: The UID of the tag to be written.
[in] Pwd: The access password to write the tag. Must be not 0.
[in] KillPwd: To lock the kill password. One of the values listed below and defined in BLUEBOX_ISO18K6C_PasswordPermission in BLUEBOXLib.h:
BLUEBOX_ISO18K6C_TAG_PWD_PERM_ACCESSIBLE: Accessible from both opened and secured states.
BLUEBOX_ISO18K6C_TAG_PWD_PERM_ALWAYS_ACCESSIBLE: Permanently accessible from both opened and secured states and may never be locked.
BLUEBOX_ISO18K6C_TAG_PWD_PERM_SECURED_ACCESSIBLE: Accessible only from secured state.
BLUEBOX_ISO18K6C_TAG_PWD_PERM_ALWAYS_NOT_ACCESSIBLE: Not accessible from either opened or secured states.
BLUEBOX_ISO18K6C_TAG_PWD_PERM_NO_CHANGE: No change.
[in] AccessPwd: To lock the access password. One of the values listed below and defined in BLUEBOX_ISO18K6C_PasswordPermission in BLUEBOXLib.h:
BLUEBOX_ISO18K6C_TAG_PWD_PERM_ACCESSIBLE: Accessible from both opened and secured states.
BLUEBOX_ISO18K6C_TAG_PWD_PERM_ALWAYS_ACCESSIBLE: Permanently accessible from both opened and secured states and may never be locked.
BLUEBOX_ISO18K6C_TAG_PWD_PERM_SECURED_ACCESSIBLE: Accessible only from secured state.
BLUEBOX_ISO18K6C_TAG_PWD_PERM_ALWAYS_NOT_ACCESSIBLE: Not accessible from either opened or secured states.
BLUEBOX_ISO18K6C_TAG_PWD_PERM_NO_CHANGE: No change.
[in] EPCMemory: To lock the EPC memory. One of the values listed below and defined in BLUEBXO_ISO18K6C_MemoryPermission in BLUEBOXLib.h:
BLUEBOX_ISO18K6C_TAG_MEM_PERM_WRITABLE: Writable from both opened and secured states.
BLUEBOX_ISO18K6C_TAG_MEM_PERM_ALWAYS_WRITABLE: Permanently writable from both opened and secured states and may never be locked.
BLUEBOX_ISO18K6C_TAG_MEM_PERM_SECURED_WRITA

BLE: Writable only from secured state.
BLUEBOX_ISO18K6C_TAG_MEM_PERM_ALWAYS_NOT_WRITABLE: Not writable from either opened or secured states.
BLUEBOX_ISO18K6C_TAG_MEM_PERM_NO_CHANGE: No change.
[in] TIDMemory: To lock the TID memory. One of the values listed below and defined in BLUEBXO_ISO18K6C_MemoryPermission in BLUEBOXLib.h:
BLUEBOX_ISO18K6C_TAG_MEM_PERM_WRITABLE: Writable from both opened and secured states.
BLUEBOX_ISO18K6C_TAG_MEM_PERM_ALWAYS_WRITABLE: Permanently writable from both opened and secured states and may never be locked.
BLUEBOX_ISO18K6C_TAG_MEM_PERM_SECURED_WRITABLE: Writable only from secured state.
BLUEBOX_ISO18K6C_TAG_MEM_PERM_ALWAYS_NOT_WRITABLE: Not writable from either opened or secured states.
BLUEBOX_ISO18K6C_TAG_MEM_PERM_NO_CHANGE: No change.
[in] UserMemory: To lock the user memory. One of the values listed below and defined in BLUEBXO_ISO18K6C_MemoryPermission in BLUEBOXLib.h:
BLUEBOX_ISO18K6C_TAG_MEM_PERM_WRITABLE: Writable from both opened and secured states.
BLUEBOX_ISO18K6C_TAG_MEM_PERM_ALWAYS_WRITABLE: Permanently writable from both opened and secured states and may never be locked.
BLUEBOX_ISO18K6C_TAG_MEM_PERM_SECURED_WRITABLE: Writable only from secured state.
BLUEBOX_ISO18K6C_TAG_MEM_PERM_ALWAYS_NOT_WRITABLE: Not writable from either opened or secured states.
BLUEBOX_ISO18K6C_TAG_MEM_PERM_NO_CHANGE: No change.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.

BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Lock_ISO18K6C (BLUEBOX_Handle *Handle, void *Uid, void *Pwd, BLUEBOX_ISO18K6C_PasswordPermission KillPwd, BLUEBOX_ISO18K6C_PasswordPermission AccessPwd, BLUEBOX_ISO18K6C_MemoryPermission EPCMemory, BLUEBOX_ISO18K6C_MemoryPermission TIDMemory, BLUEBOX_ISO18K6C_MemoryPermission UserMemory);

2.5.79 BLUEBOX_Kill_ISO18K6C

Name: BLUEBOX_Kill_ISO18K6C

Reader: BLUEBOX INDUSTRIAL UHF MID RANGE SINGLE CHANNEL, BLUEBOX INDUSTRIAL UHF LONG RANGE QUAD CHANNEL., BLUEBOX GEN2 DESKTOP UHF, BLUEBOX GEN2 INDUSTRIAL UHF MID RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC UHF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC UHF MID RANGE SINGLE CHANNEL.

Description: This function allows to kill a ISO 18000-6C tag.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Uid: The UID of the tag to be killed.
[in] Pwd: The kill password to kill the tag

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_InvalidParams.
BLUEBOX_TagNotFound.
BLUEBOX_TagError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_Kill_ISO18K6C (BLUEBOX_Handle *Handle, void *Uid, void *Pwd);

2.5.80 BLUEBOX_FwUpgrade

Name: BLUEBOX_FwUpgrade

Reader: All readers.

Description: This function allows to upgrade the BLUEBOX readers firmware.

Parameters: [in] Handle: The handle that identifies the reader.
[in] FileName: The binary file name with the firmware to send to the reader.
[in] Reader: The reader to upgrade. Use one of the values listed below and defined in BLUEBOX_Reader in BLUEBOXLib.h.
BLUEBOX_PRIMARY_READER: To upgrade the primary reader.
BLUEBOX_AUXILIARY_1_READER: To upgrade the 1st auxiliary reader.
BLUEBOX_AUXILIARY_2_READER: To upgrade the 2nd auxiliary reader.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.
BLUEBOX_GenericError.
BLUEBOX_InvalidParams.
BLUEBOX_FileError.

Syntax:
```
BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall
BLUEBOX_FwUpgrade (BLUEBOX_Handle *Handle,
BLUEBOX_UpgReader Reader, char *FileName);
```

2.5.81 BLUEBOX_ReadNumberOfRegistrations

Name: BLUEBOX_ReadNumberOfRegistrations

Reader: BLUEBOX PORTAL UHF.

Description: This function reads the number of registrations saved in the reader's memory.

Parameters: [in] Handle: The handle that identifies the reader.
[out] Registrations: The number of registrations saved in the reader's memory.

Return: An error code about the execution of the function. One of

	the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle. BLUEBOX_ConnectionError. BLUEBOX_InvalidCommand. BLUEBOX_TimeoutError. BLUEBOX_CommunicationError.
Syntax:	BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ReadNumberOfRegistrations (BLUEBOX_Handle *Handle, int *Registrations);

2.5.82 BLUEBOX_ReadOlderRegistration

Name:	BLUEBOX_ReadOlderRegistration
Reader:	BLUEBOX PORTAL UHF.
Description:	This function reads the older registration saved in the reader's memory.
Parameters:	[in] Handle: The handle that identifies the reader. [out] Index: The index of the registration read from reader's memory. [out] Registration: The registration read from the reader's memory.
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle. BLUEBOX_ConnectionError. BLUEBOX_InvalidCommand. BLUEBOX_TimeoutError. BLUEBOX_CommunicationError.
Syntax:	BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ReadOlderRegistration (BLUEBOX_Handle *Handle, int *Index, BLUEBOX_Registration *Registration);

2.5.83 BLUEBOX_CancelOlderRegistration

Name:	BLUEBOX_CancelOlderRegistration
Reader:	BLUEBOX PORTAL UHF.
Description:	This function cancels the older registration saved in the reader's memory.

Parameters: [in] Handle: The handle that identifies the reader.
[in] Index: The index of the registration to cancel from reader's memory.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_CancelOlderRegistration (BLUEBOX_Handle *Handle, int Index);

2.5.84 BLUEBOX_CancelAllRegistrations

Name: BLUEBOX_CancelAllRegistrations

Reader: BLUEBOX PORTAL UHF.

Description: This function cancels all the registrations saved in the reader's memory.

Parameters: [in] Handle: The handle that identifies the reader.

Return: An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h:
BLUEBOX_StatusOk.
BLUEBOX_InvalidHandle.
BLUEBOX_ConnectionError.
BLUEBOX_InvalidCommand.
BLUEBOX_TimeoutError.
BLUEBOX_CommunicationError.

Syntax: BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_CancelAllRegistrations (BLUEBOX_Handle *Handle);

2.5.85 BLUEBOX_ReadPreviousRegistration

Name: BLUEBOX_ReadPreviousRegistration

Reader: BLUEBOX PORTAL UHF.

Description: This function reads a previous registration saved in the reader's memory.

Parameters: [in] Handle: The handle that identifies the reader.

	[in] Index: The index of the registration to be read from reader's memory. [out] Registration: The registrations read from the reader's memory.
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle. BLUEBOX_ConnectionError. BLUEBOX_InvalidCommand. BLUEBOX_TimeoutError. BLUEBOX_CommunicationError.
Syntax:	BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_ReadPreviousRegistration (BLUEBOX_Handle *Handle, int Index, BLUEBOX_Registration *Registration);

2.5.86 BLUEBOX_GenericCommand

Name:	BLUEBOX_GenericCommand
Reader:	All readers.
Description:	This function sends a generic command using the BLUEBOX protocol.
Parameters:	[in] Handle: The handle that identifies the reader. [in] Command: The command to send to the reader. [out] Reply: The reply got from the reader.
Return:	An error code about the execution of the function. One of the values listed below and defined in BLUEBOX_ErrorCodes in BLUEBOXLib.h: BLUEBOX_StatusOk. BLUEBOX_InvalidHandle. BLUEBOX_ConnectionError. BLUEBOX_InvalidCommand. BLUEBOX_TimeoutError. BLUEBOX_CommunicationError.
Syntax:	BLUEBOXLib_API BLUEBOX_ErrorCodes __stdcall BLUEBOX_GenericCommand (BLUEBOX_Handle *Handle, char *Command, char *Reply);

3 BlueBox Gen1 Functions Table

	BLUEBOX OEM LF	BLUEBOX OEM HF	BLUEBOX OEM HF E	BLUEBOX DESKTOP LF	BLUEBOX DESKTOP HF	BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL	BLUEBOX INDUSTRIAL LF SHORT RANGE DUAL CHANNEL	BLUEBOX INDUSTRIAL LF LONG RANGE SINGLE CHANNEL	BLUEBOX INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL	BLUEBOX INDUSTRIAL HF SHORT RANGE DUAL CHANNEL	BLUEBOX INDUSTRIAL HF MID RANGE SINGLE CHANNEL	BLUEBOX INDUSTRIAL HF LONG RANGE SINGLE CHANNEL	BLUEBOX INDUSTRIAL HF LONG RANGE QUAD CHANNEL	BLUEBOX INDUSTRIAL UHF MID RANGE SINGLE CHANNEL	BLUEBOX INDUSTRIAL UHF LONG RANGE QUAD CHANNEL	BLUEBOX INDUSTRIAL ACTIVE	BLUEBOX PORTAL UHF
BLUEBOX_GetSwRelease	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_Init	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_End	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_Open	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_Close	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_SetAddress	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_SetDevice	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_GetDevice	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_SetChannel	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_GetFwRelease	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_Reset	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_ReadParameters	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_WriteParameters	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_DefaultParameters	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_ReadConfiguration											✓	✓	✓	✓	✓	✓	✓
BLUEBOX_WriteConfiguration											✓	✓	✓	✓	✓	✓	✓
BLUEBOX_DefaultConfiguration											✓	✓	✓	✓	✓	✓	✓
BLUEBOX_DataRequest	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_QueueRequest	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_FreeTagsMemory	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_AllocateNotifyChannel	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_DeallocateNotifyChannel	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_GetNotification	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_FreeNotifyMemory	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_SetOutput	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_GetReaderStatus	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_RfOnOff	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_ReadID_EM4305	✓			✓		✓	✓										
BLUEBOX_Write_EM4305	✓			✓		✓	✓										
BLUEBOX_ReadID_T5557	✓			✓		✓	✓										
BLUEBOX_Write_T5557	✓			✓		✓	✓										
BLUEBOX_ReadID_Q5	✓			✓		✓	✓										
BLUEBOX_WriteQ5	✓			✓		✓	✓										
BLUEBOX_ReadID_HITAGS	✓			✓		✓	✓										
BLUEBOX_Write_HITAGS	✓			✓		✓	✓										
BLUEBOX_ReadPage_HITAGS	✓			✓		✓	✓										
BLUEBOX_WritePage_HITAGS	✓			✓		✓	✓										
BLUEBOX_ReadID_TITAN	✓			✓		✓	✓										
BLUEBOX_Reset_TITAN	✓			✓		✓	✓										
BLUEBOX_Login_TITAN	✓			✓		✓	✓										
BLUEBOX_WritePassword_TITAN	✓			✓		✓	✓										
BLUEBOX_SelectiveRead_TITAN	✓			✓		✓	✓										
BLUEBOX_SelectiveWrite_TITAN	✓			✓		✓	✓										
BLUEBOX_Inventory_ISO15693		✓			✓				✓	✓	✓	✓	✓				
BLUEBOX_ReadPage_ISO15693		✓			✓				✓	✓	✓	✓	✓				
BLUEBOX_WritePage_ISO15693		✓			✓				✓	✓	✓	✓	✓				
BLUEBOX_LockPage_ISO15693		✓			✓				✓	✓	✓	✓	✓				
BLUEBOX_Write_AFI_ISO15693											✓	✓	✓				
BLUEBOX_Lock_AFI_ISO15693											✓	✓	✓				
BLUEBOX_GetRandomNumber_ICODE_SLI_S		✓			✓												
BLUEBOX_SetPassword_ICODE_SLI_S		✓			✓												
BLUEBOX_WritePassword_ICODE_SLI_S		✓			✓												
BLUEBOX_LockPassword_ICODE_SLI_S		✓			✓												
BLUEBOX_64BitPasswordProtection_ICODE_SLI_S		✓			✓												

BLUEBOX_ProtectPage_ICODE_SLI_S		✓			✓											
BLUEBOX_LockPageProtectionCondition_ICODE_SLI_S		✓			✓											
BLUEBOX_GetMultipleBlockProtectionStatus_ICODE_SLI_S		✓			✓											
BLUEBOX_Destroy_SLI_S_ICODE_SLI_S		✓			✓											
BLUEBOX_EnablePrivacy_ICODE_SLI_S		✓			✓											
BLUEBOX_Inevntory_ISO14443A		✓	✓		✓				✓	✓						
BLUEBOX_ReadBlock_MIFARE_1k		✓	✓		✓				✓	✓						
BLUEBOX_WriteBlock_MIFARE_1k		✓	✓		✓				✓	✓						
BLUEBOX_ReadBlock_MIFARE_4k		✓	✓		✓				✓	✓						
BLUEBOX_WriteBlock_MIFARE_4k		✓	✓		✓				✓	✓						
BLUEBOX_ReadBlock_MIFARE_Ultralight		✓	✓		✓				✓	✓						
BLUEBOX_WriteBlock_MIFARE_Ultralight		✓	✓		✓				✓	✓						
BLUEBOX_Inventory_ISO14443B		✓			✓				✓	✓						
BLUEBOX_ReadBlock_SR176		✓			✓				✓	✓						
BLUEBOX_WriteBlock_SR176		✓			✓				✓	✓						
BLUEBOX_ReadRfParameters											✓	✓	✓	✓	✓	✓	✓
BLUEBOX_WriteRfParameters											✓	✓	✓	✓	✓	✓	✓
BLUEBOX_Inventory_ISO18K6B															✓		
BLUEBOX_Read_ISO18K6B															✓		
BLUEBOX_Write_ISO18K6B															✓		
BLUEBOX_Inventory_ISO18K6C														✓	✓		
BLUEBOX_Read_ISO18K6C														✓	✓		
BLUEBOX_Write_ISO18K6C														✓	✓		
BLUEBOX_Lock_ISO18K6C														✓	✓		
BLUEBOX_Kill_ISO18K6C														✓	✓		
BLUEBOX_FwUpgrade	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_ReadNumberOfRegistrations																	✓
BLUEBOX_ReadOlderRegistration																	✓
BLUEBOX_CancelOlderRegistration																	✓
BLUEBOX_CancelAllRegistrations																	✓
BLUEBOX_ReadPreviousRegistration																	✓
BLUEBOX_GenericCommand	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

4 BlueBox Gen2 Functions Table

	BLUEBOX GEN2 DESKTOP LF	BLUEBOX GEN2 DESKTOP HF	BLUEBOX GEN2 DESKTOP UHF	BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL	BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL	BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CHANNEL	BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL	BLUEBOX GEN2 INDUSTRIAL HF MID RANGE SINGLE CHANNEL	BLUEBOX GEN2 INDUSTRIAL UHF MID RANGE SINGLE CHANNEL	BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL	BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL	BLUEBOX GEN2 BASIC UHF SHORT RANGE SINGLE CHANNEL	BLUEBOX GEN2 BASIC UHF MID RANGE SINGLE CHANNEL
BLUEBOX_GetSwRelease	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_Init	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_End	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_Open	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_Close	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_SetAddress	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_SetDevice	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_GetDevice	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_SetChannel	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_GetFwRelease	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_Reset	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_ReadParameters	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_WriteParameters	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_DefaultParameters	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_ReadConfiguration			✓	✓	✓	✓	✓	✓	✓				
BLUEBOX_WriteConfiguration			✓	✓	✓	✓	✓	✓	✓				
BLUEBOX_DefaultConfiguration			✓	✓	✓	✓	✓	✓	✓				
BLUEBOX_DataRequest	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_QueueRequest	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_FreeTagsMemory	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_AllocateNotifyChannel	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_DeallocateNotifyChannel	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_GetNotification	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_FreeNotifyMemory	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_SetOutput				✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_GetReaderStatus	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_RfOnOff	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_ReadID_EM4305	✓			✓	✓					✓			
BLUEBOX_Write_EM4305	✓			✓	✓					✓			
BLUEBOX_ReadID_T5557	✓			✓	✓					✓			
BLUEBOX_Write_T5557	✓			✓	✓					✓			
BLUEBOX_ReadID_Q5	✓			✓	✓					✓			
BLUEBOX_WriteQ5	✓			✓	✓					✓			
BLUEBOX_ReadID_HITAGS	✓			✓	✓					✓			
BLUEBOX_Write_HITAGS	✓			✓	✓					✓			
BLUEBOX_ReadPage_HITAGS	✓			✓	✓					✓			
BLUEBOX_WritePage_HITAGS	✓			✓	✓					✓			
BLUEBOX_ReadID_TITAN	✓			✓	✓					✓			
BLUEBOX_Reset_TITAN	✓			✓	✓					✓			
BLUEBOX_Login_TITAN	✓			✓	✓					✓			
BLUEBOX_WritePassword_TITAN	✓			✓	✓					✓			
BLUEBOX_SelectiveRead_TITAN	✓			✓	✓					✓			
BLUEBOX_SelectiveWrite_TITAN	✓			✓	✓					✓			
BLUEBOX_Inventory_ISO15693		✓				✓	✓	✓			✓		
BLUEBOX_ReadPage_ISO15693		✓				✓	✓	✓			✓		
BLUEBOX_WritePage_ISO15693		✓				✓	✓	✓			✓		
BLUEBOX_LockPage_ISO15693		✓				✓	✓	✓			✓		
BLUEBOX_Write_AFI_ISO15693								✓					
BLUEBOX_Lock_AFI_ISO15693								✓					
BLUEBOX_GetRandomNumber_ICODE_SLI_S		✓				✓	✓				✓		

BLUEBOX_SetPassword_ICODE_SLI_S		✓				✓	✓				✓		
BLUEBOX_WritePassword_ICODE_SLI_S		✓				✓	✓				✓		
BLUEBOX_LockPassword_ICODE_SLI_S		✓				✓	✓				✓		
BLUEBOX_64BitPasswordProtection_ICODE_SLI_S		✓				✓	✓				✓		
BLUEBOX_ProtectPage_ICODE_SLI_S		✓				✓	✓				✓		
BLUEBOX_LockPageProtectionCondition_ICODE_SLI_S		✓				✓	✓				✓		
BLUEBOX_GetMultipleBlockProtectionStatus_ICODE_SLI_S		✓				✓	✓				✓		
BLUEBOX_Destroy_SLI_S_ICODE_SLI_S		✓				✓	✓				✓		
BLUEBOX_EnablePrivacy_ICODE_SLI_S		✓				✓	✓				✓		
BLUEBOX_Inevntory_ISO14443A		✓				✓	✓				✓		
BLUEBOX_ReadBlock_MIFARE_1k		✓				✓	✓				✓		
BLUEBOX_WriteBlock_MIFARE_1k		✓				✓	✓				✓		
BLUEBOX_ReadBlock_MIFARE_4k		✓				✓	✓				✓		
BLUEBOX_WriteBlock_MIFARE_4k		✓				✓	✓				✓		
BLUEBOX_ReadBlock_MIFARE_Ultralight		✓				✓	✓				✓		
BLUEBOX_WriteBlock_MIFARE_Ultralight		✓				✓	✓				✓		
BLUEBOX_Inventory_ISO14443B		✓				✓	✓				✓		
BLUEBOX_ReadBlock_SR176		✓				✓	✓				✓		
BLUEBOX_WriteBlock_SR176		✓				✓	✓				✓		
BLUEBOX_ReadRfParameters			✓					✓	✓			✓	✓
BLUEBOX_WriteRfParameters			✓					✓	✓			✓	✓
BLUEBOX_Inventory_ISO18K6B													
BLUEBOX_Read_ISO18K6B													
BLUEBOX_Write_ISO18K6B													
BLUEBOX_Inventory_ISO18K6C			✓						✓			✓	✓
BLUEBOX_Read_ISO18K6C			✓						✓			✓	✓
BLUEBOX_Write_ISO18K6C			✓						✓			✓	✓
BLUEBOX_Lock_ISO18K6C			✓						✓			✓	✓
BLUEBOX_Kill_ISO18K6C			✓						✓			✓	✓
BLUEBOX_FwUpgrade	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLUEBOX_ReadNumberOfRegistrations													
BLUEBOX_ReadOlderRegistration													
BLUEBOX_CancelOlderRegistration													
BLUEBOX_CancelAllRegistrations													
BLUEBOX_ReadPreviousRegistration													
BLUEBOX_GenericCommand	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

5 Document Revision History

Revision	Date	Description
1.00	30/04/10	First release.
1.01	05/05/10	Changes in supported readers list to add the new readers managed from the library release 2.0.0. Added the document revision history section. Added definitions, functions, enums and structs to manage the new readers. Changes in BLUEBOX_SetChannel function parameters. Changes in nibble coding and gain enums definitions (ref. LF readers). Changes in BLUEBOX_GeneralParameters struct definition. Changes in BLUEBOX_TagType enum definition. Changes in MIFARE key enum definitions (ref. HF readers). Changes in BLUEBOX_ReaderStatus struct definition (deleted the Line flag in BLUEBOX INDUSTRIAL readers definition).
1.02	18/06/10	Changes in supported readers list to add the new readers managed from the library release 3.0.0. Added the 'spontaneous' message notifications by adding the functions: • BLUEBOX_AllocateNotifyChannel • BLUEBOX_DeallocateNotifyChannel • BLUEBOX_GetNotification and the error codes: • BLUEBOX_AllocationError and the structs: • BLUEBOX_Notify Added functions to manage the device type: • BLUEBOX_SetDevice • BLUEBOX_GetDevice Added the ISO 15693 AFI (Application Family Identifier) management in BLUEBOX_Inventory_ISO15693 function. Added functions to manage ISO 15693 AFI (Application Family Identifier): • BLUEBOX_Write_AFI_ISO15693

		• BLUEBOX_Lock_AFI_ISO15693 Improved BLUEBOX_Close and BLUEBOX_End functions. The BLUEBOX_End function implicity calls the BLUEBOX_Close function. Changes in BLUEBOX_SetChannel function parameters strings, added the retransmission numbers at the end of Settings string with RS232/RS485 interface. Added the BLUEBOX_Tag struct definition. Changes in BLUEBOX_RfParameters struct definition. Added the upgrade firmware by adding the functions: • BLUEBOX_FwUpgrade • BLUEBOX_GetUpgradeStatus and the error codes: • BLUEBOX_FileError and the enums: • BLUEBOX_UpgReader Changes in definitions to uniform the code writing rules. Added the library release information in document revision history table.
1.03	08/07/10	Corrections in the document revision history in description of the 1.02 document release. Changes in supported readers list to add the new readers managed from the library release 4.0.0. Changes in BLUEBOX_SetChannel function parameters strings, added the communication timeout of Settings string with all interfaces. Changes in BLUEBOX_GetFwRelease to allow the auxiliary reader's fw version reading. Changes in configuration functions: • BLUEBOX_ReadParameters; • BLUEBOX_WriteParameters; • BLUEBOX_ReadRfParameters; • BLUEBOX_WriteRfParameters; parameters to make them more flexible. Also deleted all the enumarations and structures related to them. Changes in BLUEBOX_GetReaderStatus function parameters to make it more flexible. Also deleted all the enumarations and structures related to it. Changed the BLUEBOX_UpgReader □numeration to BLUEBOX_Reader and also changed its items names. Added the BLUEBOX_TagError error code in the following functions: • BLUEBOX_Inventory_ISO15693;

		• BLUEBOX_Inventory_ISO14443A; • BLUEBOX_Inventory_ISO14443B.
1.04	02/08/10	Changed the BLUEBOX_GetDevice and BLUEBOX_SetDevice function parameters by adding the firmware version major and minor numbers to manage different features in different firmware versions.
1.05	05/08/10	Added the EM4305 and T5557 tags management in BLUEBOX INDUSTRIAL LF SHORT RANGE SINGLE/DUAL CHANNEL, BLUEBOX OEM LF and BLUEBOX DESKTOP LF by adding, in BLUEBOX_TagType the enumerators: • BLUEBOX_EM4305; • BLUEBOX_T5557; and definitions: • BLUEBOX_EM4305_ID_SIZE; • BLUEBOX_T5557_ID_SIZE; and functions: • BLUEBOX_ReadID_EM4305; • BLUEBOX_Write_EM4305; • BLUEBOX_ReadID_T5557; • BLUEBOX_Write_T5557.
1.06	05/10/10	Corrections in the document revision history in description of the 1.04 document release. Deleted the Antenna parameter from BLUEBOX_RfOnOff function. Deleted the BLUEBOX_GetUpgradeStatus function. Added the BLUEBOX PORTAL UHF reader management. The affected functions are: • BLUEBOX_SetDevice; • BLUEBOX_GetDevice; • BLUEBOX_GetFwRelease; • BLUEBOX_ReadParameters; • BLUEBOX_WriteParameters; • BLUEBOX_DefaultParameters; • BLUEBOX_GetReaderStatus; • BLUEBOX_ReadRfParameters; • BLUEBOX_WriteRfParameters; the functions added are: • BLUEBOX_ReadNumberOfRegistrations; • BLUEBOX_ReadOlderRegistration; • BLUEBOX_CancelOlderRegistration;

		• BLUEBOX_CancelAllRegistrations; • BLUEBOX_ReadPreviousRegistration; the affected definitions / enumarations / structures are: • BLUEBOX_ErrorCodes; the definitions / enumarations / structures added are: • BLUEBOX_Input; • BLUEBOX_Registration.
1.07	17/01/11	Added the ISO 18000-6C (EPC C1G2) with variable UID size tags management. The affected definitions are: • BLUEBOX_ISO18K6C_UID_SIZE. Added the section remarks in functions: • BLUEBOX_GetFwRelease; • BLUEBOX_ReadParameters.
1.08	09/09/11	Added the firmware release related to this technical manual in the first page. Added the x64 architecture support in section 1. Deleted the BLUEBOX INDUSTRIAL UHF SHORT RANGE SINGLE CHANNEL reader management (replaced with the MID RANGE one). Added the customization management through the variant management. The affected functions are: • BLUEBOX_SetDevice; • BLUEBOX_GetDevice. Changed the BLUEBOX_AllocateNotifyChannel parameters and function prototype. Changes in section 'Document Revision History' (this section).
1.09	24/10/11	Extended the BLUEBOX_Input enumeration to support the 'no input' case. Increased the maximum tag's ID length supported (BLUEBOX_MAX_ID_LENGTH definition). Changes to BLUEBOX_Tag and BLUEBOX_Notify structures.
1.10	19/01/12	Deleted the maximum tag's ID length definitions (BLUEBOX_MAX_ID_LENGTH). Changed the tag's ID management from static array to dynamic array. The affected structures are: • BLUEBOX_Tags; • BLUEBOX_Notify; • BLUEBOX_Registration. The affected functions are:

		• BLUEBOX_FreeTagsMemory; • BLUEBOX_FreeNotifyMemory.
1.11	22/02/12	Added the ICODE SLI-S tag management in BLUEBOX DESKTOP HF and BLUEBOX OEM HF by adding, in BLUEBOX_TagType the enumerators: • BLUEBOX_ICODE_SLI_S; and enumerations: • BLUEBOX_ICODE_SLI_S_PasswordIdentifier; • BLUEBOX_ICODE_SLI_S_ProtectionStatus; and definitions: • BLUEBOX_ICODE_SLI_S_RND_SIZE; • BLUEBOX_ICODE_SLI_S_PWD_SIZE; and structures: • BLUEBOX_ICODE_SLI_S_BlockProtectionStatus; and functions: • BLUEBOX_GetRandomNumber_ICODE_SLI_S; • BLUEBOX_SetPassword_ICODE_SLI_S; • BLUEBOX_WritePassword_ICODE_SLI_S; • BLUEBOX_LockPassword_ICODE_SLI_S; • BLUEBOX_64BitPasswordProtection_ICODE_SLI_S • BLUEBOX_ProtectPage_ICODE_SLI_S; • BLUEBOX_LockPageProtectionCondition_ICODE_SLI_S; • BLUEBOX_GetMultipleBlockProtectionStatus_ICODE_SLI_S; • BLUEBOX_Destroy_SLI_S_ICODE_SLI_S; • BLUEBOX_EnablePrivacy_ICODE_SLI_S;
1.12	10/10/12	Added the BLUEBOX Gen2 readers support to the library (BLUEBOX GEN2 DESKTOP LF, BLUEBOX GEN2 DESKTOP HF, BLUEBOX GEN2 DESKTOP UHF, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL LF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE SINGLE CAHNNEL, BLUEBOX GEN2 INDUSTRIAL HF SHORT RANGE DUAL CHANNEL, BLUEBOX GEN2 INDUSTRIAL HF MID RANGE SINGLE CHANNEL, BLUEBOX GEN2 INDUSTRIAL UHF MID RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC LF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC HF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC UHF SHORT RANGE SINGLE CHANNEL, BLUEBOX GEN2 BASIC MID RANGE SINGLE CHANNEL. Deleted the list of the supported readers in section 1 and

		added sections 4 and 5 with the readers supported functions tables. Added the management of a second auxiliary reader. The affected enumartions are: • BLUEBOX_Reader. The affected functions are: • BLUEBOX_GetFwRelease; • BLUEBOX_FwUpgrade. Added the BLUEBOX_Reset function. Added the management of the configuration pages of the readers. The added functions are: • BLUEBOX_ReadConfiguration; • BLUEBOX_WriteConfiguration; • BLUEBOX_DefaultConfiguration. Removed remarks in BLUEBOX_SetDevice and BLUEBOX_GetDevice functions. Added the BLUEBOX_GenericCommand function.