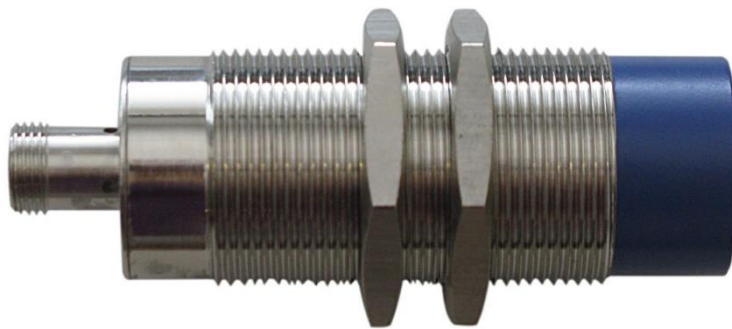


# **13.56 MHz RFID System**



## **M30 Form Factor BLUEBOX ADVANT HF**



**RS232 / RS485**

From firmware release 1.16

## 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.

## **Safety Instructions / Warning - Read before start-up!**

- The device may only be used for the intended purpose designed by the manufacturer. The operation manual should be conveniently kept available at all times for each user.
- Unauthorized changes and the use of spare parts and additional devices that have not been sold or recommended by the manufacturer may cause fire, electric shocks or injuries. Such unauthorized measures shall exclude any liability by the manufacturer.
- The liability-prescriptions of the manufacturer in the issue valid at the time of purchase are valid for the device. The manufacturer shall not be held legally responsible for inaccuracies, errors, or omissions in the manual or automatically set parameters for a device or for an incorrect application of a device.
- Repairs may be executed by the manufacturer only.
- Only qualified personnel should carry out installation, operation, and maintenance procedures.
- Use of the device and its installation must be in accordance with national legal requirements and local electrical codes.
- When working on devices the valid safety regulations must be observed.

iDTRONIC GmbH  
Ludwig-Reichling-Straße 4  
67059 Ludwigshafen  
Germany/Deutschland

Issue 1.00a  
– 21. September 2022 –

Phone: +49 621 6690094-0  
Fax: +49 621 6690094-9  
E-Mail: [info@idtronic.de](mailto:info@idtronic.de)  
Web: [idtronic.de](http://idtronic.de)

Subject to alteration without prior notice.  
© Copyright iDTRONIC GmbH 2022  
Printed in Germany

## Table of Contents

1	Introduction.....	5
2	Technical Specifications.....	6
3	Operating Features .....	9
3.1	General Parameters.....	9
4	Communication Features .....	11
4.1	Device Startup.....	12
4.2	Device Reset .....	13
4.3	General Parameters Programming .....	13
4.4	Default Parameters Programming.....	15
4.5	General Parameters Reading .....	15
4.6	FW Version Reading .....	16
4.7	Data Request .....	17
4.8	Queue Data Request.....	18
4.9	Status Reading .....	19
4.10	RF Deactivation.....	20
4.11	RF Activation .....	20
4.12	Inventory of ISO 15693 Transponders .....	20
4.13	Read Page of an ISO 15693 Transponder .....	21
4.14	Write Page of an ISO 15693 Transponder.....	22
4.15	Lock Page of an ISO 15693 Transponder.....	23
4.16	'Get System Info' Command of an ISO 15693 Transponder .....	24
4.17	'General Protocol' Command of an ISO 15693 Transponder .....	26
4.18	Inventory of ISO 14443A Transponders .....	27
4.19	Read Data Block of a MIFARE Mini/1k/4k Transponder .....	27
4.20	Write Data Block of a MIFARE Mini/1k/4k Transponder.....	29
4.21	Read Data Block of a MIFARE Ultralight Transponder .....	30
4.22	Write Data Block of a MIFARE Ultralight Transponder.....	31
4.23	Inventory of ISO 14443B Transponders .....	32
4.24	Read Data Block of an SR 176 Transponder.....	32
4.25	Write Data Block of an SR 176 Transponder .....	33
4.26	GetChipNumber Command of a JayCOS 2 Tag .....	34
4.27	GetChallenge Command of a JayCOS 2 Tag .....	35
4.28	ExternalAuthenticate Command of a JayCOS 2 Tag .....	36
4.29	ReadEEPROM Command of a JayCOS 2 Tag .....	36
4.30	WriteEEPROM Command of a JayCOS 2 Tag.....	37
4.31	'Spontaneous' Message .....	39
4.31.1	RS232 .....	39
5	Connections.....	40
6	Installation .....	43
7	Status Indications: LEDs .....	45
8	Document Revision History .....	46
A.	Supported Transponders .....	47

## 1 Introduction

The **BLUEBOX ADVANT M30 HF** hereinafter named **BLUEBOX** is a little (dimensions of the cylindrical case  $M30 \times 1.5 \times 78$  mm) read/write RFID device operating at 13.56 MHz and suitable for industrial application. The **BLUEBOX** communicates with a 'host' system (typically a PC or a PLC) through an RS232 (items 5224H and 5227H) or an RS485 (items 5225H or 5228H) serial line and acts as a joint through a set of commands between the host system and a RFID tag present near the antenna. A 'master/slave' protocol is used for the communication between the 'host' system and the **BLUEBOX**. Through the serial line, it is also possible to configure the functional parameters and to upgrade the firmware, the 'BLUEBOX Show' program of the SDK is foreseen to explicate these operations. The **BLUEBOX** is furnished with an integrated RF antenna inside the case and with a 4-pole M12 A-coded male connector (items 5224H and 5225H) or with an open end 1.5mt cable (items 5227H and 5228H).

Hereinafter the available ordering codes.

Ordering Code	Description	Interface
<b>5224H</b>	Read / write 13.56 MHz RFID device with integrated antenna. Serial RS232 communication interface. M12 connection.	RS232
<b>5225H</b>	Read / write 13.56 MHz RFID device with one external antenna. Serial RS485 communication interface. M12 connection.	RS485
<b>5227H</b>	Read / write 13.56 MHz RFID device with integrated antenna. Serial RS232 communication interface. Cable with open end connection.	RS232
<b>5228H</b>	Read / write 13.56 MHz RFID device with one external antenna. Serial RS485 communication interface. Cable with open end connection.	RS485

## 2 Technical Specifications

### 5224H:

Electrical Features	
Power Supply	24Vdc $\pm 10\%$
Power Ratings	3W
Operating Frequency	13.56 MHz $\pm 7$ kHz
Antenna	Integrated
Reading Distance	10 cm <sup>1</sup>
Supported Transponders	ISO 15693, ISO 14443A, ISO 14443B
Communication Interface	Serial RS232
Status Display	1 bicolor LED
Connections	4-pole M12 A-coded male connector

### 5225H:

Electrical Features	
Power Supply	24Vdc $\pm 10\%$
Power Ratings	3W
Operating Frequency	13.56 MHz $\pm 7$ kHz
Antenna	Integrated
Reading Distance	10 cm <sup>2</sup>
Supported Transponders	ISO 15693, ISO 14443A, ISO 14443B
Communication Interface	Serial RS485
Status Display	1 bicolor LED
Connections	4-pole M12 A-coded male connector

<sup>1</sup> Reading distance depends on transponder type, antenna and environmental conditions.

<sup>2</sup> Reading distance depends on transponder type, antenna and environmental conditions.

## 5227H:

Electrical Features	
Power Supply	24Vdc $\pm 10\%$
Power Ratings	3W
Operating Frequency	13.56 MHz $\pm 7$ kHz
Antenna	Integrated
Reading Distance	10 cm <sup>3</sup>
Supported Transponders	ISO 15693, ISO 14443A, ISO 14443B
Communication Interface	Serial RS232
Connections	2 twisted pairs overall shielded multipolar cable with free cable end. Length 1.5mt.

## 5228H:

Electrical Features	
Power Supply	24Vdc $\pm 10\%$
Power Ratings	3W
Operating Frequency	13.56 MHz $\pm 7$ kHz
Antenna	Integrated
Reading Distance	10 cm <sup>4</sup>
Supported Transponders	ISO 15693, ISO 14443A, ISO 14443B
Communication Interface	Serial RS485
Connections	2 twisted pairs overall shielded multipolar cable with free cable end. Length 1.5mt.

<sup>3</sup> Reading distance depends on transponder type, antenna and environmental conditions.

<sup>4</sup> Reading distance depends on transponder type, antenna and environmental conditions.

## 5224H, 5225H, 5227H, 5228H:

### Mechanical Features

Dimensions	M30 × 1.5 × 78 mm
Material	Nickelled brass, PC
Protection Class	IP65

## 5224H, 5225H, 5227H, 5228H:

### Environmental Conditions

Operating Temperature	-10°C ... +55°C
Storage Temperature	-40°C ... +85°C
Humidity	Up to 95%, non condensing



### 3 Operating Features

In 'continuous' mode the **BLUEBOX** is characterized by the coexistence of 2 'parallel' and asynchronous activities: the transponder identification and the communication with the 'host' system. The 'continuous' identification activity interacts with the communication activity through a buffer that contains the code of the last identified transponder or the 0 code that indicates the absence of a transponder. Due to synchronization and filtering reasons, the buffer is handled by a parameter defined as 'hold time' (to be set in the range of 0 ... 99 seconds, default value 1 second) and allows to extend 'artificially' the presence of the transponder after it leaves the antenna's influence area; this behavior is observable looking at the yellow led status that is 'on' indicating the presence of a transponder. Through the command 'data request' it is possible to get the data contained in the buffer.

The **BLUEBOX** handles also a 31 elements FIFO queue which is combined with a 'filter time' parameter (to be set in a range of 0 ... 99 seconds, default value 1 second) that prevents the queue saturation in case of a transponder 'continuous' presence. When a transponder is identified, the **BLUEBOX** compares it to the previous read transponder. If the transponder is different (it is defined as 'new'), its code will be inserted in the queue and the filter time will be started. Otherwise (the transponder is the same of the previous read one), the **BLUEBOX** verifies if the filter time is expired. In this case (the filter time is expired), the transponder is defined as 'new' and will be processed as described above, otherwise only the filter time will be rearmed. Through the command 'queue data request' and the relative 'ack', it is possible to get the data contained in the queue and unload it.

In 'continuous' mode the **BLUEBOX** can be configured to obtain the behavior of a 'spontaneous' reader that will send a message on the RS232 serial line. This feature is enabled (on) / disabled (off) a flag in the general configuration of the reader.

The **BLUEBOX** allows the execution of 'on request' functions. During the execution of these functions, the 'continuous' identification activity will be suspended temporarily; the involved commands are relative to device configuration and tag read/write specific activities.

If not required, the 'continuous' identification activity can be disabled through a flag defined in the general parameters. In this case, the **BLUEBOX** will only execute the 'on request' commands already defined above.

#### 3.1 General Parameters

Hereinafter the configurable general parameter of the **BLUEBOX**.

Parameter	Description	Range	Default
Network address	Network address of the reader.	000 ... 255	255
Baud rate	Communication baud rate on RS232/RS485 interface.	1200, 2400, 4800, 9600, 19200, 38400	19200
Data bits	Data bits on RS232/RS485 interface.	7, 8	8
Stop bits	Stop bits on RS232/RS485 interface.	1, 2	1
Parity	Parity on RS232/RS485 interface.	None, even, odd	None
Hold time	Buffer management hold time.	0 ... 99 seconds	1 sec
Filter time	Reading and tag queue management filter time.	0 ... 99 seconds 0 ... 99 minutes	1 sec
'Spontaneous' mode	Spontaneous message activation/deactivation.	Disabled, enabled	Disabled
'Continuous' mode	'Continuous' mode activation/deactivation.	Disabled, enabled	Enabled

## 4 Communication Features

The 'master/slave' protocol expects that the **BLUEBOX** (as 'slave') after the reception of a message send to him by the 'host' (as 'master'), transmits a response message after a minimum time of about 10 ms. By default, the **BLUEBOX** will apply the following parameters: address 255, baud rate 19200, 8 data bits, parity none and 1 stop bit. These parameters can be modified as specified in the 'Parameters programming' protocol command.

To simplify the explanations, the following conventions will be used:

SOH	Carattere 01h (0x01)
STX	Carattere 02h (0x02)
ETX	Carattere 03h (0x03)
EOT	Carattere 04h (0x04)
ENQ	Carattere 05h (0x05)
ACK	Carattere 06h (0x06)
NAK	Carattere 15h (0x15)
SYN	Carattere 16h (0x16)
CR	Carattere 0Dh (0x0D)
'0'...'9'	Carattere 30h ...39h (0x30 ... 0x39)
'A'...'F'	Carattere 41h ...46h (0x41 ... 0x46)
<..>	Carattere 30h ...39h (0x30 ... 0x39), 41h ...46h (0x41 ... 0x46)
<bcc>	Checksum

This is the general structure of a message:

**SOH <add h> <add l> ... <bcc> CR**

**SOH** is the opening character, **CR** is the final character, **<bcc>** is the checking character or checksum and it is calculated as 'xor' of the previous characters starting from SOH and applying the following rule: if <bcc> = SOH or <bcc> = CR or <bcc> = EOT , then <bcc> := <bcc>+1 (must be incremented of 1).

The **BLUEBOX** address is expressed with a byte (0...255 in decimal, 0x00 ... 0xFF in hexadecimal) transformed into two ASCII characters: the first ASCII character <add h> corresponds to the ASCII coding of the high nibble of the byte, while the second ASCII character <add l> corresponds to the ASCII coding of the low

nibble of the byte. Example: 255 → 0xFF → 'F' 'F'. This rule is also valid for coding a generic byte value.

For instance, the 'data request' command message for a **BLUEBOX** with address 1 will be: SOH '0' '1' ENQ ENQ CR (in hexadecimal: 0x01, 0x30, 0x31, 0x05, 0x05, 0x0D).

#### 4.1 Device Startup

During the startup phase, it is possible to configure the communication parameters of the **BLUEBOX** sending the following message (apply the following default communication settings 19200, n, 8, 1):

**STX '2' 'F' <addn h> <addn l> <bdr> <bit> <stop> <par> ETX <bcc> CR**

Dove:

<add h> <add l>	New address to be set. ASCII encoded byte.
<bdr>	RS232/RS485 communication interface baud rate. ASCII character: <ul style="list-style-type: none"> <li>'0' -&gt; 1200 bps;</li> <li>'1' -&gt; 2400 bps;</li> <li>'2' -&gt; 4800 bps;</li> <li>'3' -&gt; 9600 bps;</li> <li>'4' -&gt; 19200 bps</li> <li>'5' -&gt; 38400 bps.</li> </ul>
<bit>	RS232/RS485 communication interface data bits. ASCII character: <ul style="list-style-type: none"> <li>'7' -&gt; 7 bits;</li> <li>'8' -&gt; 8 bits.</li> </ul>
<stop>	RS232/RS485 communication interface stop bits. ASCII character: <ul style="list-style-type: none"> <li>'1' -&gt; 1 bit;</li> <li>'2' -&gt; 2 bits.</li> </ul>
<par>	RS232/RS485 communication interface parity. ASCII character: <ul style="list-style-type: none"> <li>'0' -&gt; None;</li> <li>'1' -&gt; Even;</li> <li>'2' -&gt; Odd.</li> </ul>

If the **BLUEBOX** is able to execute the command, it answers with:

## STX '2' 'F' '0' '0' <bcc> CR

### 4.2 Device Reset

This command is used to restart the **BLUEBOX** (the device has the same behavior like when it is powered up).

The 'master' sends the following command:

**SOH <adda h> <adda l> STX '3' '0' ETX <bcc> CR**

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

**SOH <add h> <add l> ACK <bcc> CR**

### 4.3 General Parameters Programming

This command is used to set the communication and operating parameters of the **BLUEBOX**.

The 'master' sends the following command:

**SOH <adda h> <adda l> STX '2' 'F' <addn h> <addn l> <bdr> <bit> <stop> <par> <man h> <man l> '0' '0' <filt h> <filt l> <flag h> <flag l> ETX <bcc> CR**

Dove:

<adda h> <adda l>	Reader address. ASCII encoded byte.
<addn h> <addn l>	New address to be set. ASCII encoded byte.
<bdr>	RS232/RS485 communication interface baud rate. ASCII character: <ul style="list-style-type: none"> <li>'0' -&gt; 1200 bps;</li> <li>'1' -&gt; 2400 bps;</li> <li>'2' -&gt; 4800 bps;</li> <li>'3' -&gt; 9600 bps;</li> <li>'4' -&gt; 19200 bps</li> <li>'5' -&gt; 38400 bps.</li> </ul>

<bit>	RS232/RS485 communication interface data bits. ASCII character: <ul style="list-style-type: none"> <li>• '7' -&gt; 7 bits;</li> <li>• '8' -&gt; 8 bits.</li> </ul>
<stop>	RS232/RS485 communication interface stop bits. ASCII character: <ul style="list-style-type: none"> <li>• '1' -&gt; 1 bit;</li> <li>• '2' -&gt; 2 bits.</li> </ul>
<par>	RS232/RS485 communication interface parity. ASCII character: <ul style="list-style-type: none"> <li>• '0' -&gt; None;</li> <li>• '1' -&gt; Even;</li> <li>• '2' -&gt; Odd.</li> </ul>
<man h> <man l>	Hold time. ASCII encoded byte: <ul style="list-style-type: none"> <li>• Decimal 0 ... 99 for time in seconds (0 ... 99 seconds);</li> </ul>
<filt h> <filt l>	Filter time. ASCII encoded byte: <ul style="list-style-type: none"> <li>• Decimal 0 ... 99 for time in seconds (0 ... 99 seconds);</li> <li>• Decimal 100 ... 199 for time in minutes (0 ... 99 minutes).</li> </ul>
<flag h> <flag l>	Flags. ASCII encoded byte whose bits are dedicated to disable (0 value) or enable (1 value) functions: <ul style="list-style-type: none"> <li>• Bit 7 ... bit4: Not used;</li> <li>• Bit 3: 'Spontaneous' mode, (1=ON);</li> <li>• Bit 2 ... bit 1: Not used;</li> <li>• Bit 0: 'Continuous' mode, (1=OFF).</li> </ul>

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

**SOH <add h> <add l> ACK <bcc> CR**



After the command execution, the **BLUEBOX** resets itself to apply the new parameters.

#### 4.4 Default Parameters Programming

This command is used to set the default values of the communication and parameters of the **BLUEBOX**.

The 'master' sends the following command:

**SOH <add h> <add l> STX '3' '1' ETX <bcc> CR**

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

**SOH <add h> <add l> ACK <bcc> CR**



After the command execution, the **BLUEBOX** resets itself to apply the new parameters.

#### 4.5 General Parameters Reading

This command is used to get the values of the communication and operating general parameters of the **BLUEBOX**.

The 'master' sends the following command:

**SOH <add h> <add l> STX '2' 'A' ETX <bcc> CR**

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** able to execute the command), it answers with:

**SOH <add h> <add l> STX '2' 'A' <add h> <add l> <bdr> <bit> <stop> <par> <man h> <man l> '0' '0' <filt h> <filt l> <flag h> <flag l> ETX <bcc> CR**

Where:

<adda h> <adda l>

Reader address. ASCII encoded byte.

<bdr>	<p>RS232/RS485 communication interface baud rate. ASCII character:</p> <ul style="list-style-type: none"> <li>• '0' -&gt; 1200 bps;</li> <li>• '1' -&gt; 2400 bps;</li> <li>• '2' -&gt; 4800 bps;</li> <li>• '3' -&gt; 9600 bps;</li> <li>• '4' -&gt; 19200 bps;</li> <li>• '5' -&gt; 38400 bps.</li> </ul>
<bit>	<p>RS232/RS485 communication interface data bits. ASCII character:</p> <ul style="list-style-type: none"> <li>• '7' -&gt; 7 bits;</li> <li>• '8' -&gt; 8 bits.</li> </ul>
<stop>	<p>RS232/RS485 communication interface stop bits. ASCII character:</p> <ul style="list-style-type: none"> <li>• '1' -&gt; 1 bit;</li> <li>• '2' -&gt; 2 bits.</li> </ul>
<par>	<p>RS232/RS485 communication interface parity. ASCII character:</p> <ul style="list-style-type: none"> <li>• '0' -&gt; None;</li> <li>• '1' -&gt; Even;</li> <li>• '2' -&gt; Odd.</li> </ul>
<man h> <man l>	<p>Hold time. ASCII encoded byte:</p> <ul style="list-style-type: none"> <li>• Decimal 0 ... 99 for time in seconds (0 ... 99 seconds);</li> </ul>
<filt h> <filt l>	<p>Filter time. ASCII encoded byte:</p> <ul style="list-style-type: none"> <li>• Decimal 0 ... 99 for time in seconds (0 ... 99 seconds);</li> <li>• Decimal 100 ... 199 for time in minutes (0 ... 99 minutes).</li> </ul>
<flag h> <flag l>	<p>Flags. ASCII encoded byte whose bits are dedicated to disable (0 value) or enable (1 value) functions:</p> <ul style="list-style-type: none"> <li>• Bit 7 ... bit4: Not used;</li> <li>• Bit 3: 'Spontaneous' mode, (1=ON);</li> <li>• Bit 2 ... bit 1: Not used;</li> <li>• Bit 0: 'Continuous' mode, (1=OFF).</li> </ul>

#### 4.6 FW Version Reading

The 'master' sends the following command:



**SOH <add h> <add l> STX '3' '4' ETX <bcc> CR**

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** able to execute the command), it answers with:

**SOH <add h> <add l> STX '3' '4' <vf 01 h> <vf 01 l> <vf 02 h> <vf 02 l> ... <vf 15 h> <vf 15 l> <vf 16 h> <vf 16 l> ETX <bcc> CR**

Where:

<vf 01 h> <vf 01 l>	ASCII coding of the byte 1 of the string.
...	...
<vf 16 h> <vf 16 l>	ASCII coding of the byte 16 of the string.

In this case the 16 bytes are represented by a string of 16 ASCII characters that define the version. Example 'TINYOEM\_HF\_1.00' indicates that this is a **TINYOEM** (BLUEBOX OEM) in **HF** configuration (**H**igh **F**requency 13.56 MHz) with firmware version **1.00**.

## 4.7 Data Request

This command sends back the code of the eventual transponder that is present in the buffer. When 'continuous' mode is enabled, the reply is immediate because the **BLUEBOX** sends back the data hold in the buffer that is managed by the 'continuous' identification activity; otherwise, the **BLUEBOX** performs readily the identification task under time out protection and sends back the result of the operation.

The 'master' sends the following command:

**SOH <add h> <add l> ENQ <bcc> CR**

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

**SOH <add h> <add l> STX <type h> <type l> <UID 1 h> <UID 1 l>... <UID i h> <UID i l>... <UID n h> <UID n l> ETX <bcc> CR**

Where:

<type h> <type l>	Transponder type. See Annex A for tag type and UID length table.
i	1 ... n (the UID length). See Annex A for tag type and UID length table.
<UID i h> <UID i l>	i-th byte of the UID of the identified tag. ASCII encoded byte..

If the **BLUEBOX** doesn't have any valid UID (no tag present), it will answer with:

**SOH <add h> <add l> STX '0' '0' '0' '0' '0' '0' '0' '0' '0' '0' ETX <bcc> CR**

demonstrating to the 'master' its presence in the network.

#### 4.8 Queue Data Request

In 'continuous' mode, when the **BLUEBOX** finds a 'new' transponder, it inserts the code in the FIFO queue. This command sends back the first present code in the queue.

The 'master' sends the following command:

**SOH <add h> <add l> SYN <bcc> CR**

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

**SOH <add h> <add l> STX <type h> <type l> <UID 1 h> <UID 1 l>... <UID i h> <UID i l>... <UID n h> <UID n l> ETX <bcc> CR**

Where:

<type h> <type l>	Transponder type. See Annex A for tag type and UID length table.
i	1 ... n (the UID length). See Annex A for tag type and UID length table.
<UID i h> <UID i l>	i-th byte of the UID of the identified tag. ASCII encoded byte..

If the queue is empty, the **BLUEBOX** will answer with:

**SOH <add h> <add l> STX '0' '0' '0' '0' '0' '0' '0' '0' '0' '0' ETX <bcc> CR**

demonstrating to the 'master' its presence in the network.

To delete the received code from the queue, the 'master' reply to the **BLUEBOX** with:

**SOH <add h> <add l> ACK <bcc> CR**

#### 4.9 Status Reading

The **BLUEBOX** will answer to this command with a series of information about the current status and particularly about the digital inputs status.

The 'master' sends the following command:

**SOH <add1> <add0> STX '3' '6' ETX <bcc> CR**

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

**SOH <add h> <add l> STX '3' '6' <sta hh> <sta hl> <sta lh> <sta ll> ETX <bcc> CR**

Where:

<sta hh> <sta hl> <sta  
lh> <sta ll>

**BLUEBOX** status. ASCII encoded word whose bits has the following meaning:

- Bit 15...14: Not used;
- Bit 13: RF status (0=off, 1=on);
- Bit 12: 'Continuous' mode status (0=disabled, 1=enabled);
- Bit 11...18: Not used;
- Bit 7: Solder Jumper 4 status (1=off);
- Bit 6: Solder Jumper 3 status (1=off);
- Bit 5: Solder Jumper 2 status (1=off);
- Bit 4: Solder Jumper 1 status (1=off);
- Bit 3...0: Not used.

#### 4.10 RF Deactivation

In 'continuous' mode, this command is used to suspend the activity of the RF antennas connected to the **BLUEBOX**; see also 'RF activation' command.

The 'master' sends the following command:

**SOH <add h> <add l> STX '3' '8' ETX <bcc> CR**

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

**SOH <add h> <add l> ACK <bcc> CR**

#### 4.11 RF Activation

In 'continuous' mode, this command is used to resume the activity of the RF antennas connected to the **BLUEBOX**; see also 'RF deactivation' command.

The 'master' sends the following command:

**SOH <add h> <add l> STX '3' '9' ETX <bcc> CR**

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

**SOH <add h> <add l> ACK <bcc> CR**

#### 4.12 Inventory of ISO 15693 Transponders

This command is used to get the buffer that contains the UID code of the identified ISO 15693 transponders that are present near the antenna.

The 'master' sends the following command:

**SOH <add h> <add l> STX '1' '0' ETX <bcc> CR**

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

### SOH <add h> <add l> NAK <bcc> CR

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

if at least one tag is present

**SOH <add h> <add l> STX '1' '0' '0' '0' <UID 1 1 h> <UID 1 1 l>... <UID 1 i h> <UID 1 i l>... <UID 1 8 h> <UID 1 8 l> ... <UID j 1 h> <UID j 1 l>... <UID j i h> <UID j i l>... <UID j 8 h> <UID j 8 l> ... <UID n 1 h> <UID n 1 l>... <UID n i h> <UID n i l>... <UID n 8 h> <UID n 8 l> ETX <bcc> CR**

Where:

i	1 ... 8.
J	1 ... n.
n	Number of identified tags.
<UID j i h> <UID j i l>	i-th byte of the UID of the j-th identified tag. ASCII encoded byte.

b) if some error is occurred during the transaction

**SOH <add h> <add l> STX '1' '0' '0' '2' ETX <bcc> CR**

c) if no tag is present

**SOH <add h> <add l> STX '1' '0' '0' '1' ETX <bcc> CR**

## 4.13 Read Page of an ISO 15693 Transponder

This command is used to get a data block of a known (UID) ISO 15693 transponder. Note that the number of bytes of a block depends on the transponder type; for example, the **NXP ICODE2 (or ICODE SLI)** transponder is organized in blocks of 4 bytes, the **Fujitsu MB89R118** transponder is organized in blocks of 8 bytes, for more details see the specific transponder data sheet.

The 'master' sends the following command:

**SOH <add h> <add l> STX '1' '1' <UID 1 h> <UID 1 l>... <UID i h> <UID i l>... <UID 8 h> <UID 8 l> <pag h> <pag l> ETX <bcc> CR**

Where:

i	1 ... 8.
<UID i h> <UID i l>	i-th byte of the UID of the tag. ASCII encoded byte.
<pag h> <pag l>	Address of the page to read. ASCII encoded byte.

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the reader is able to execute the command), it answers with:

a) if the addressed tag is present and the data bytes have been successfully read

**SOH <add h> <add l> STX '1' '1' '0' '0' <data 1 h> <data 1 l>... <data i h> <data i l>... <data n h> <data n l> ETX <bcc> CR**

Where:

i	1 ... n.
n	Number of bytes on the page read (4, 8).
<data i h> <data i l>	i-th byte of the page read from tag. ASCII encoded byte.

b) if the addressed tag do not support the requested blocks or if some error is occurred during the transaction

**SOH <add h> <add l> STX '1' '1' '0' '2' ETX <bcc> CR**

c) if the addressed tag is not present

**SOH <add h> <add l> STX '1' '1' '0' '1' ETX <bcc> CR**

#### 4.14 Write Page of an ISO 15693 Transponder

This command is used to write a data block of a known (UID) ISO 15693 transponder. Note that the number of bytes of a block depends on the transponder type; for example, the **NXP ICODE2 (or ICODE SLI)** transponder is organized in blocks of 4 bytes, the **Fujitsu MB89R118** transponder is organized in blocks of 8 bytes, for more details see the specific transponder data sheet.

The 'master' sends the following command:

**SOH <add h> <add l> STX '1' '2' <UID 1 h> <UID 1 l>... <UID i h> <UID i l>... <UID 8 h> <UID 8 l> <pag h> <pag l> <data 1 h> <data 1 l>... <data j h> <data j l>... <data n h> <data n l> ETX <bcc> CR**

Where:

i	1 ... 8.
<UID i h> <UID i l>	i-th byte of the UID of the tag. ASCII encoded byte.
<pag h> <pag l>	Address of the page to write. ASCII encoded byte.
J	1 ... n.
n	Number of bytes of the page to write (4, 8).
<data j h> <data j l>	j-th byte of the page to write. ASCII encoded byte.

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

a) if the addressed tag is present and the data bytes have been successfully written

**SOH <add h> <add l> STX '1' '2' '0' '0' ETX <bcc> CR**

b) if the addressed tag is present but errors occurred

**SOH <add h> <add l> STX '1' '2' '0' '2' ETX <bcc> CR**

c) if the addressed tag is not present

**SOH <add h> <add l> STX '1' '2' '0' '1' ETX <bcc> CR**

#### 4.15 Lock Page of an ISO 15693 Transponder

This command is used to lock a data block of a known (UID) ISO 15693 transponder.

The 'master' sends the following command:

**SOH <add h> <add l> STX '1' '3' <UID 1 h> <UID 1 l> ... <UID i h> <UID i l> ... <UID 8 h> <UID 8 l> <pag h> <pag l> ETX <bcc> CR**

Where:

i	1 ... 8.
<UID i h> <UID i l>	i-th byte of the UID of the tag. ASCII encoded byte.
<pag h> <pag l>	Address of the page to lock. ASCII encoder byte.

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

a) if the addressed tag is present and it has been successfully locked

**SOH <add h> <add l> STX '1' '3' '0' '0' ETX <bcc> CR**

b) if the addressed tag is present but errors occurred

**SOH <add h> <add l> STX '1' '3' '0' '2' ETX <bcc> CR**

c) if the addressed tag is not present

**SOH <add h> <add l> STX '1' '3' '0' '1' ETX <bcc> CR**

#### 4.16 'Get System Info' Command of an ISO 15693 Transponder

This command is used to get the system info data block of a known (UID) ISO 15693 transponder. For more details see the specific transponder data sheet.

The 'master' sends the following command:

**SOH <add h> <add l> STX '1' '4' <UID 1 h> <UID 1 l> ... <UID i h> <UID i l> ... <UID 8 h> <UID 8 l> ETX <bcc> CR**

Where:

i	1 ... 8.
<UID i h> <UID i l>	i-th byte of the UID of the tag. ASCII encoded byte.

If the addressed **BLUEBOX** is not able to execute the command, it answers:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise it answers,

a) if the addressed tag is present and the data bytes have been successfully read:



**SOH <add h> <add l> STX '1' '4' '0' '0' <flg h> <flg l> <UID 1 h> <UID 1 l> ... <UID i h> <UID i l> ... <UID 8 h> <UID 8 l> <dsf h> <dsf l> <afi h> <afi l> <msbs h> <msbs l> <msnb h> <msnb l> <icr h> <icr l> ETX <bcc> CR**

Where:

<flg h> <flg l>	Info Flags of the tag. ASCII encoded byte. Single bits are dedicated to specify the presence of the following fields (0 → absent, 1 → present): <ul style="list-style-type: none"> <li>• Bit 7...4: Not used;</li> <li>• Bit 3: IC Reference (1 byte);</li> <li>• Bit 2: Memory Size (2 bytes);</li> <li>• Bit 1: AFI (1 byte);</li> <li>• Bit 0: DSFID (1 byte).</li> </ul>
<UID i h> <UID i l>	i-th byte of the UID of the tag. ASCII encoded byte.
i	1 ... 8
<dsf h> <dsf l>	DSFID of the tag. ASCII encoded byte. Field present only if bit 0 of Info Flags is set.
<afi h> <afi l>	AFI of the tag. ASCII encoded byte. Field present only if bit 1 of Info Flags is set.
<msbs h> <msbs l>	Memory Size – Block Size in bytes. ASCII encoded byte. 0x00 (1 byte) ... 0x1F (32 bytes). Field present only if bit 2 of Info Flags is set.
<msnb h> <msnb l>	Memory Size – Number of Blocks. ASCII encoded byte. 0x00 (1 block) ... 0xFF (256 blocks). Field present only if bit 2 of Info Flags is set.
<icr h> <icr l>	IC Reference. ASCII encoded byte. Field present only if bit 3 of Info Flags is set.

b) if the addressed tag is present but errors occurred:

**SOH <add h> <add l> STX '1' '4' '0' '2' ETX <bcc> CR**

c) if the addressed tag is not present:

**SOH <add h> <add l> STX '1' '4' '0' '1' ETX <bcc> CR**

#### 4.17 'General Protocol' Command of an ISO 15693 Transponder

This command allows to send any ISO 15693 general format protocol command (flags field, command code field, parameters fields, application data fields) to a ISO 15693 transponder and to receive, in case of successful operation, the response of the transponder (flag field, parameters fields, data fields). For more details see the specific transponder data sheet and ISO 15693 protocol. If the 'continuous' mode is enabled, it will be suspended by the execution of this command and will be suspended as long as this command is used; it will be resumed automatically when another type of command will be executed.

The 'master' sends the following command:

**SOH <add h> <add l> STX '1' '5' <data 1 h> <data 1 l>... <data i h> <data i l>... <data n h> <data n l> ETX <bcc> CR**

Where:

i	1 ... n.
n	Number of bytes to send to the tag.
<data i h> <data i l>	i-th byte to send to the tag. ASCII encoded byte.

If the addressed **BLUEBOX** is not able to execute the command, it answers:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise it answers,

a) if the addressed tag is present and the data bytes have been successfully sent:

**SOH <add h> <add l> STX '1' '5' '0' '0' <data 1 h> <data 1 l>... <data i h> <data i l>... <data n h> <data n l> ETX <bcc> CR**

Where:

i	1 ... n.
n	Number of bytes received from the tag.
<data i h> <data i l>	i-th byte received from the tag. ASCII encoded byte.

b) if the addressed tag is present but errors occurred:

**SOH <add h> <add l> STX '1' '5' '0' '2' ETX <bcc> CR**

c) if the addressed tag is not present:

**SOH <add h> <add l> STX '1' '5' '0' '1' ETX <bcc> CR**

#### 4.18 Inventory of ISO 14443A Transponders

This command is used to get the UID code of a ISO 14443A transponder (MIFARE Ultralight, MIFARE 1k, MIFARE 4k, MIFARE Mini, MIFARE Plus 2k, MIFARE Plus 4k) that is present near the antenna.

The 'master' sends the following command:

**SOH <add h> <add l> STX '1' '8' ETX <bcc> CR**

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

a) if at least one tag is present

**SOH <add h> <add l> STX '1' '8' '0' '0' <UID 1 h> <UID 1 l> ... <UID i h> <UID i l> ... <UID m h> <UID m l> ETX <bcc> CR**

Where:

i	1 ... n (the UID length). See Annex A for tag type and UID length table.
<UID i h> <UID i l>	i-th byte of the UID of the identified tag. ASCII encoded byte.

b) if some error is occurred during the transaction

**SOH <add h> <add l> STX '1' '8' '0' '2' ETX <bcc> CR**

c) if no tag is present

**SOH <add h> <add l> STX '1' '8' '0' '1' ETX <bcc> CR**

#### 4.19 Read Data Block of a MIFARE Mini/1k/4k Transponder

This command is used to get a data block (16 bytes) of a known (UID) MIFARE Mini/1k/4k transponder. For more details see the specific transponder data sheet.

The 'master' sends the following command:

**SOH <add h> <add l> STX '1' '9' <UID 1 h> <UID 1 l>... <UID i h> <UID i l>... <UID 4 h> <UID 4 l> <kt h> <kt l> <key 1 h> <key 1 l> ... <key j h> <key j l> ... <key 6 h> <key 6 l> <block h> <block l> ETX <bcc> CR**

Where:

i	1 ... 4.
<UID i h> <UID i l>	i-th byte of the UID of the tag. ASCII encoded byte.
<kt h> <kt l>	Key type to access the tag's memory. ASCII encoded byte: <ul style="list-style-type: none"> <li>• 0x00: Key A;</li> <li>• 0x01: Key B.</li> </ul>
j	1 ... 6.
<key j h> <key j l>	j-th byte of the key. ASCII encoded byte.
<block h> <block l>	Memory block to read. ASCII encoded byte (0x00 ... 0x13 for MIFARE Mini, 0x00 ... 0x3F for MIFARE 1k, 0x00 ... 0xFF for MIFARE 4k)

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the reader is able to execute the command), it answers with:

a) if the addressed tag is present and the data bytes have been successfully read

**SOH <add h> <add l> STX '1' '9' '0' '0' <data 1 h> <data 1 l> ... <data i h> <data i l> ... <data 16 h> <data 16 l> ETX <bcc> CR**

Where:

i	1 ... 16.
<data i h> <data i l>	i-th byte read from the tag. ASCII encoded byte.

b) if the addressed tag do not support the requested blocks or if some error is occurred during the transaction

**SOH <add h> <add l> STX '1' '9' '0' '2' ETX <bcc> CR**

c) if the addressed tag is not present

**SOH <add h> <add l> STX '1' '9' '0' '1' ETX <bcc> CR**

## 4.20 Write Data Block of a MIFARE Mini/1k/4k Transponder

This command is used to write a data block (16 bytes) of a known (UID) MIFARE Mini/1k/4k transponder. For more details see the specific transponder data sheet.

The 'master' sends the following command:

**SOH <add h> <add l> STX '1' 'A' <UID 1 h> <UID 1 l> ... <UID i h> <UID i l> ... <UID 4 h> <UID 4 l> <kt h> <kt l> <key 1 h> <key 1 l> ... <key j h> <key j l> ... <key 6 h> <key 6 l> <block h> <block l> <data 1 h> <data 1 l> ... <data k h> <data k l> ... <data 16 h> <data 16 l> ETX <bcc> CR**

Where:

i	1 ... 4.
<UID i h> <UID i l>	i-th byte of the UID of the tag. ASCII encoded byte.
<kt h> <kt l>	Key type to access the tag's memory. ASCII encoded byte: <ul style="list-style-type: none"> <li>• 0x00: Key A;</li> <li>• 0x01: Key B.</li> </ul>
j	1 ... 6.
<key j h> <key j l>	j-th byte of the key. ASCII encoded byte.
<block h> <block l>	Memory block to write. ASCII encoded byte (0x00 ... 0x13 for MIFARE Mini, 0x00 ... 0x3F for MIFARE 1k, 0x00 ... 0xFF for MIFARE 4k)
k	1 ... 16
<data i h> <data i l>	i-th byte to write in the tag. ASCII encoded byte.

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

a) if the addressed tag is present and the data bytes have been successfully written

**SOH <add h> <add l> STX '1' 'A' '0' '0' ETX <bcc> CR**

b) if the addressed tag is present but errors occurred

**SOH <add h> <add l> STX '1' 'A' '0' '2' ETX <bcc> CR**

c) if the addressed tag is not present

**SOH <add h> <add l> STX '1' 'A' '0' '1' ETX <bcc> CR**

#### 4.21 Read Data Block of a MIFARE Ultralight Transponder

This command is used to get a data block (4 bytes) of a known (UID) MIFARE Ultralight transponder. For more details see the specific transponder data sheet.

The 'master' sends the following command:

**SOH <add h> <add l> STX '1' 'B' <UID 1 h> <UID 1 l>... <UID i h> <UID i l>... <UID 7 h> <UID 7 l> <block h> <block l> ETX <bcc> CR**

Where:

i	1 ... 7.
<UID i h> <UID i l>	i-th byte of the UID of the tag. ASCII encoded byte.
<block h> <block l>	Data block to read. ASCII encoded byte (0x00 ... 0x0F)

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the reader is able to execute the command), it answers with:

a) if the addressed tag is present and the data bytes have been successfully read

**SOH <add h> <add l> STX '1' 'B' '0' '0' <data 1 h> <data 1 l> ... <data i h> <data i l> ... <data 4 h> <data 4 l> ETX <bcc> CR**

Where:

i	1 ... 4.
<data i h> <data i l>	i-th byte read from the tag. ASCII encoded byte.

b) if the addressed tag do not support the requested blocks or if some error is occurred during the transaction

**SOH <add h> <add l> STX '1' 'B' '0' '2' ETX <bcc> CR**

c) if the addressed tag is not present

**SOH <add h> <add l> STX '1' 'B' '0' '1' ETX <bcc> CR**

#### 4.22 Write Data Block of a MIFARE Ultralight Transponder

This command is used to write a data block (4 bytes) of a known (UID) MIFARE Ultralight transponder. For more details see the specific transponder data sheet.

The 'master' sends the following command:

**SOH <add h> <add l> STX '1' 'C' <UID 1 h> <UID 1 l> ... <UID i h> <UID i l> ... <UID 7 h> <UID 7 l> <block h> <block l> <data 1 h> <data 1 l> ... <data j h> <data j l> ... <data 4 h> <data 4 l> ETX <bcc> CR**

Where:

i	1 ... 7.
<UID i h> <UID i l>	i-th byte of the UID of the tag. ASCII encoded byte.
<block h> <block l>	Data block to write. ASCII encoded byte (0x00 ... 0x0F)
j	1 ... 4.
<data i h> <data i l>	i-th byte to write in the tag. ASCII encoded byte.

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

a) if the addressed tag is present and the data bytes have been successfully written

**SOH <add h> <add l> STX '1' 'C' '0' '0' ETX <bcc> CR**

b) if the addressed tag is present but errors occurred

**SOH <add h> <add l> STX '1' 'C' '0' '2' ETX <bcc> CR**

c) if the addressed tag is not present

**SOH <add h> <add l> STX '1' 'C' '0' '1' ETX <bcc> CR**

#### 4.23 Inventory of ISO 14443B Transponders

This command is used to get the UID code of a ISO 14443B transponder (SR 176 and SRI 512) that is present near the antenna.

The 'master' sends the following command:

**SOH <add h> <add l> STX '2' '0' ETX <bcc> CR**

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the reader is able to execute the command), it answers with:

a) if at least one tag is present

**SOH <add h> <add l> STX '2' '0' '0' '0' <UID 1 h> <UID 1 l> ... <UID i h> <UID i l> ... <UID 7 h> <UID 7 l> ETX <bcc> CR**

Where:

i	1 ... 7.
<UID i h> <UID i l>	i-th byte of the UID of the identified tag. ASCII encoded byte.

b) if some error is occurred during the transaction

**SOH <add h> <add l> STX '2' '0' '0' '2' ETX <bcc> CR**

c) if no tag is present

**SOH <add h> <add l> STX '2' '0' '0' '1' ETX <bcc> CR**

#### 4.24 Read Data Block of an SR 176 Transponder

This command is used to get a data block (2 bytes) of a known (UID) SR 176 transponder. For more details see the specific transponder data sheet.

The 'master' sends the following command:

**SOH <add h> <add l> STX '2' '1' <UID 1 h> <UID 1 l>... <UID i h> <UID i l>... <UID 8 h> <UID 8 l> <block h> <block l> ETX <bcc> CR**

Where:



i	1 ... 8.
<UID i h> <UID i l>	i-th byte of the UID of the tag. ASCII encoded byte.
<block h> <block l>	Data block to read. ASCII encoded byte (0x00 ... 0x0F)

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the reader is able to execute the command), it answers with:

a) if the addressed tag is present and the data bytes have been successfully read

**SOH <add h> <add l> STX '2' '1' '0' '0' <data 1 h> <data 1 l> ... <data i h> <data i l> ... <data 2 h> <data 2 l> ETX <bcc> CR**

Where:

i	1 ... 2.
<data i h> <data i l>	i-th byte read from the tag. ASCII encoded byte.

b) if the addressed tag do not support the requested blocks or if some error is occurred during the transaction

**SOH <add h> <add l> STX '2' '1' '0' '2' ETX <bcc> CR**

c) if the addressed tag is not present

**SOH <add h> <add l> STX '2' '1' '0' '1' ETX <bcc> CR**

#### 4.25 Write Data Block of an SR 176 Transponder

This command is used to write a data block (2 bytes) of a known (UID) SR 176 transponder. For more details see the specific transponder data sheet.

The 'master' sends the following command:

**SOH <add h> <add l> STX '2' '2' <UID 1 h> <UID 1 l> ... <UID i h> <UID i l> ... <UID 8 h> <UID 8 l> <block h> <block l> <data 1 h> <data 1 l> <data 2 h> <data 2 l> ETX <bcc> CR**

Where:

i	1 ... 8.
---	----------

<UID i h> <UID i l>	i-th byte of the UID of the tag. ASCII encoded byte.
<block h> <block l>	Data block to write. ASCII encoded byte (0x00 ... 0x0F)
k	1 ... 2.
<data i h> <data i l>	i-th byte to write in the tag. ASCII encoded byte.

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

a) if the addressed tag is present and the data bytes have been successfully written

**SOH <add h> <add l> STX '2' '2' '0' '0' ETX <bcc> CR**

b) if the addressed tag is present but errors occurred

**SOH <add h> <add l> STX '2' '2' '0' '2' ETX <bcc> CR**

c) if the addressed tag is not present

**SOH <add h> <add l> STX '2' '2' '0' '1' ETX <bcc> CR**

#### 4.26 GetChipNumber Command of a JayCOS 2 Tag

This command allows to read the ChipNumber of a JayCOS 2 tag.

The 'master' sends the following command:

**SOH <add h> <add l> STX '2' '3' ETX <bcc> CR**

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

a) if the addressed tag is present and the command have been successfully executed

**SOH <add h> <add l> STX '2' '3' '0' '0' <chip 0 h> <chip 0 l> ... <chip i h> <chip i l> ... <chip 7 h> <chip 7 l> ETX <bcc> CR**

Where:

i	1 ... 8.
<chip i h> <chip i l>	i-th byte of the ChipNumber of the tag. ASCII encoded byte.

b) if the addressed tag is present but errors occurred

**SOH <add h> <add l> STX '2' '3' '0' '2' ETX <bcc> CR**

c) if the addressed tag is not present

**SOH <add h> <add l> STX '2' '3' '0' '1' ETX <bcc> CR**

#### 4.27 GetChallenge Command of a JayCOS 2 Tag

This command allows to received a random number of 8 bytes generated by a JayCOS 2 tag.

The 'master' sends the following command:

**SOH <add h> <add l> STX '2' '4' ETX <bcc> CR**

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

a) if the addressed tag is present and the command have been successfully executed

**SOH <add h> <add l> STX '2' '4' '0' '0' <rnd 0 h> < rnd 0 l> ... < rnd i h> < rnd i l> ... < rnd 7 h> < rnd 7 l> ETX <bcc> CR**

Where:

i	1 ... 8.
<rnd i h> <rnd i l>	i-th byte of the random number generated by the tag. ASCII encoded byte.

b) if the addressed tag is present but errors occurred

**SOH <add h> <add l> STX '2' '4' '0' '2' ETX <bcc> CR**

c) if the addressed tag is not present

**SOH <add h> <add l> STX '2' '4' '0' '1' ETX <bcc> CR**

#### 4.28 ExternalAuthenticate Command of a JayCOS 2 Tag

This command allows to send the authentication key of 8 byte to a JayCOS 2 tag.

The 'master' sends the following command:

**SOH <add h> <add l> STX '2' '5' <ref h> <ref l> <key 0 h> <key 0 l> ... <key i h> <key i l> ... <key 7 h> <key 7 l> ETX <bcc> CR**

Where:

<ref h> <ref l>	Index of the key to use. ASCII encoded byte.
i	1 ... 8.
<key i h> <key i l>	i-th byte of the key to use. ASCII encoded byte.

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

a) if the addressed tag is present and the command have been successfully executed

**SOH <add h> <add l> STX '2' '5' '0' '0' ETX <bcc> CR**

b) if the addressed tag is present but errors occurred

**SOH <add h> <add l> STX '2' '5' '0' '2' ETX <bcc> CR**

c) if the addressed tag is not present

**SOH <add h> <add l> STX '2' '5' '0' '1' ETX <bcc> CR**

#### 4.29 ReadEEPROM Command of a JayCOS 2 Tag

This command allows to read the EEPROM of a JayCOS 2 tag.

The 'master' sends the following command:

**SOH <add h> <add l> STX '2' '6' <addr hh> <addr hl> <addr lh> <addr ll> <len h> <len l> ETX <bcc> CR**

Where:

<adr hh> <adr hl> <adr lh> <adr ll>	Address of the memory location to read. ASCII encoded word.
<len h> <len l>	Number of bytes to read (max 32 byte). ASCII encoded byte.

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

a) if the addressed tag is present and the command have been successfully executed

**SOH <add h> <add l> STX '2' '6' '0' '0' <data 0 h> <data 0 l> ... <data i h> <data i l> ... <data n h> <data n l> ETX <bcc> CR**

i	1 ... n.
n	Number of bytes read.
<data i h> <data i l>	i-th byte read from the tag's memory. ASCII encoded byte.

b) if the addressed tag is present but errors occurred

**SOH <add h> <add l> STX '2' '6' '0' '2' ETX <bcc> CR**

c) if the addressed tag is not present

**SOH <add h> <add l> STX '2' '6' '0' '1' ETX <bcc> CR**

#### 4.30 WriteEEPROM Command of a JayCOS 2 Tag

This command allows to write in the EEPROM of a JayCOS 2 tag.

The 'master' sends the following command:

**SOH <add h> <add l> STX '2' '7' <addr hh> <addr hl> <addr lh> <addr ll> <len h> <len l> <data 0 h> <data 0 l> ... <data i h> <data i l> ... <data n h> <data n l> ETX <bcc> CR**

Where:

<addr hh> <addr hl> <addr lh> <addr ll>	Address of the memory location to write. ASCII encoded word.
<len h> <len l>	Number of bytes to write (max 32 byte). ASCII encoded byte.
i	1 ... n.
n	Number of bytes to write.
<data i h> <data i l>	i-th byte to read in the tag's memory. ASCII encoded byte.

If the addressed **BLUEBOX** is not able to execute the command, it answers with:

**SOH <add h> <add l> NAK <bcc> CR**

Otherwise (the addressed **BLUEBOX** is able to execute the command), it answers with:

a) if the addressed tag is present and the command have been successfully executed

**SOH <add h> <add l> STX '2' '7' '0' '0' ETX <bcc> CR**

b) if the addressed tag is present but errors occurred

**SOH <add h> <add l> STX '2' '7' '0' '2' ETX <bcc> CR**

c) if the addressed tag is not present

**SOH <add h> <add l> STX '2' '7' '0' '1' ETX <bcc> CR**

## 4.31 'Spontaneous' Message

### 4.31.1 RS232

In 'continuous' mode, if the 'spontaneous' feature is set on, the **BLUEBOX** will send the following message on the RS232 serial line every time that it will find a 'new' tag:

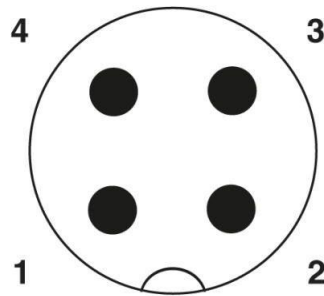
**STX <type h> <type l> <UID 1 h> <UID 1 l>... <UID i h> <UID i l>... <UID n h> <UID n l> ETX <bcc> CR**

where:

<type h> <type l>	Transponder type. See Annex A for tag type and UID length table.
i	1 ... n (the UID length). See Annex A for tag type and UID length table.
<UID i h> <UID i l>	i-th byte of the UID of the identified tag. ASCII encoded byte.
<bcc>	Block check character or checksum calculated as 'xor' of the previous characters starting from STX applying the following rule: if <bcc> = STX or <bcc> = CR, then <bcc> := <bcc>+1 (increment of 1)

## 5 Connections

### 5224H:



M12 A-coded male

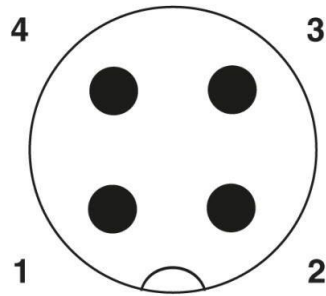
Pin	No	Min	Typical	Max	Description
+PWR	1	18Vdc	24Vdc	36Vdc	Positive DC power supply (+24V ± 10%)
RS232-Tx	2				RS232 Transmit (to host)
-PWR (GND)	3				DC power supply return (GND)
RS232-Rx	4				RS232 Receive (from host)



The shield of the cable is internally connected to GND.



## 5225H:



M12 A-coded male

Pin	No	Min	Typical	Max	Description
+PWR	1	18Vdc	24Vdc	36Vdc	Positive DC power supply (+24V ± 10%)
RS485-RT+	2				RS485 connection (positive)
-PWR (GND)	3				DC power supply return (GND)
RS485-RT-	4				RS485 connection (negative)



The shield of the cable is internally connected to GND.

## 5227H:

Pin	Pair	Wire	Min	Typical	Max	Description
+PWR	Red / black	Red	18Vdc	24Vdc	36Vdc	Positive DC power supply (+24V ± 10%)
-PWR (GND)	Red / black	Black				DC power supply return (GND)
RS232-Tx	White / black	White				RS232 Transmit (to host)
RS232-Rx	White / black	Black				RS232 Receive (from host)



The shield of the cable is internally connected to GND.

## 5228H:

Pin	Pair	Wire	Min	Typical	Max	Description
+PWR	Red / black	Red	18Vdc	24Vdc	36Vdc	Positive DC power supply (+24V ± 10%)
-PWR (GND)	Red / black	Black				DC power supply return (GND)
RS485-RT+	White / black	White				RS485 connection (positive)
RS485-RT-	White / black	Black				RS485 connection (negative)



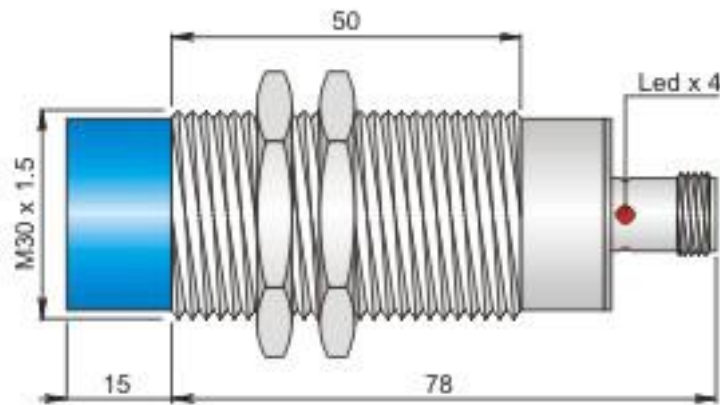
The shield of the cable is internally connected to GND.

## 6 Installation

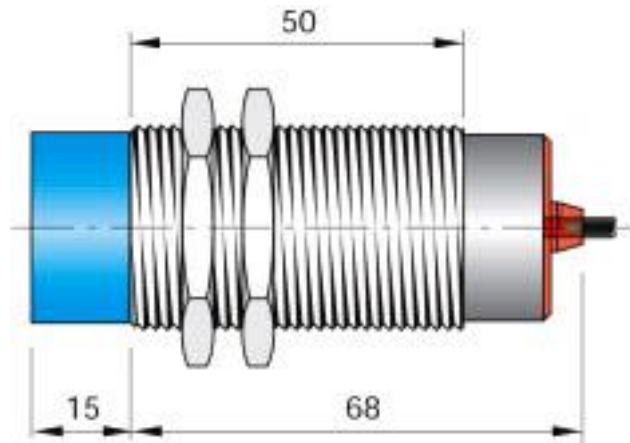
The **BLUEBOX** (items 5224H, 5225H, 5227H, 5228H) is furnished with an integrated RF antenna inside the case.

The read range of an RFID system always depends on various factors like antenna size, transponder size, transponder IC type, orientation between transponder and reader antenna, position of the transponder versus the reader antenna, noise environment, metallic environment, etc. Therefore all data about read ranges can only be typical values measured under laboratory conditions. In real live applications the read range may differ from the data mentioned in the datasheet.

### 5225H, 5226H:



### 5227H, 5228H:








Dimensions in mm.

Fix the **BLUEBOX** using the provided ring nuts and mounting them on non metallic supports (plastic, plexiglas, ...).

Connect the cable wires.

## 7 Status Indications: LEDs

The following table specifies the meaning of LEDs status.

LED	Color	State	Meaning
SYSTEM/TAG	 (red)	On	<ul style="list-style-type: none"> <li>System error.</li> <li>System initialization.</li> <li>System upgrade.</li> </ul>
	 (green)	Blinking	<ul style="list-style-type: none"> <li>Antenna active, no tag detected in 'continuous' mode.</li> </ul>
	 (green)	Slow Blink	<ul style="list-style-type: none"> <li>Antenna not active in 'continuous' mode.</li> <li>'Continuous' mode disabled.</li> </ul>
	 (green)	On	<ul style="list-style-type: none"> <li>Antenna active, tag detected in 'continuous mode.</li> <li>System initialization.</li> </ul>
	 (off)	Off	<ul style="list-style-type: none"> <li>Power supply for the device is missing.</li> <li>Hardware defect.</li> </ul>

### LED state definition

State	Definition
On	The indicator is constantly on
Off	The indicator is constantly off
Blinking	The indicator turns on and off with a frequency of 2 Hz: on for 250 ms, followed by off for 250 ms
Slow Blink	The indicator turns on and off with a frequency of 1 Hz: on for 500 ms, followed by off for 500 ms
Fast Blink	The indicator turns on for 50ms and then off.

## 8 Document Revision History

Date	Revision	Description
24/07/15	1.00	First release.

## A. Supported Transponders

Supported transponders by the **BLUEBOX** are:

Manuf.	Part	Standard	Chip	Notes / Functions
NXP	Mifare Ultralight (UID 7 bytes)	ISO 14443A	MF0 IC U1	Inventory Read Page (4 bytes) Write Page (4 bytes)
NXP	Mifare 1k (UID 4 bytes)	ISO 14443A	MF1 IC S50	Inventory Read Page (16 bytes) Write Page (16 bytes)
NXP	Mifare 4k (UID 4 bytes)	ISO 14443A	MF1 IC S70	Inventory Read Page (16 bytes) Write Page (16 bytes)
NXP	Mifare 1k (UID 7 bytes)	ISO 14443A	MF1 IC S50	Inventory Read Page (16 bytes) Write Page (16 bytes)
NXP	Mifare 4k (UID 7 bytes)	ISO 14443A	MF1 IC S70	Inventory Read Page (16 bytes) Write Page (16 bytes)
NXP	Mifare Desfire 2k (UID 7 bytes)	ISO 14443A	MF3 IC D21	Inventory
NXP	Mifare Desfire 4k (UID 7 bytes)	ISO 14443A	MF3 IC D41	Inventory
NXP	Mifare Desfire 8k (UID 7 bytes)	ISO 14443A	MF3 IC D81	Inventory
NXP	Mifare Plus S 2k (UID 7 bytes)	ISO 14443A	MF1 SPLUS 60	Inventory
NXP	Mifare Plus S 4k (UID 7 bytes)	ISO 14443A	MF1 SPLUS 80	Inventory
NXP	Mifare Plus X 2k (UID 7 bytes)	ISO 14443A	MF1 PLUS 60	Inventory
NXP	Mifare Plus X 4k (UID 7 bytes)	ISO 14443A	MF1 PLUS 80	Inventory
NXP	I CODE SLI	ISO 15693	SL2 IC S20	Inventory Read block (4 bytes) Write block (4 bytes) Lock block Get system info
TI	TAG-IT HF-I	ISO 15693		Inventory Read block (4 bytes) Write b block (4 bytes) Lock block Get system info

Manuf.	Part	Standard	Chip	Notes / Functions
MEM	EM 4035	ISO 15693		Inventory Read block (8 bytes) Write block (8 bytes) Lock block Get system info
STM	LRI 64	ISO 15693		Inventory Read block (1 byte) Write block (1 byte) Get system info
STM	LRI 512	ISO 15693		Inventory Read block (4 bytes) Write block (4 bytes) Lock block Get system info
FUJITSU	MB89R118	ISO 15693		Inventory Read block (8 bytes) Write block (8 bytes) Get system info
INFINEON	MY-D 02P	ISO 15693	SRF 55V02P	Inventory Read block (4 bytes) Write block (4 bytes) Lock block
INFINEON	MY-D 10P	ISO 15693	SRF 55V10P	Inventory Read block (4 bytes) Write block (4 bytes)
STM	SR 176 (UID 8 bytes)	ISO 14443B	SRF 55V02P	Read UID Read block (2 bytes) Write block (2 bytes) Lock block
STM	SRI 512 (UID 8 bytes)	ISO 14443B	SRF 55V10P	Read UID



Type coding for supported transponders:

Type Code	Part	UID Length
0x11	Mifare 1k (UID 4)	4 bytes
0x12	Mifare 4k (UID 4)	4 bytes
0x13	Mifare Ultralight	7 bytes
0x14	Mifare Desfire 2k/4k/8k	7 bytes
0x15	Mifare 1k (UID 7)	7 bytes
0x16	Mifare 4k (UID 7)	7 bytes
0x17	Mifare Plus S/X 2k	7 bytes
0x18	Mifare Plus S/X 4k	7 bytes
0x20	Generic ISO 15693	8 bytes
0x21	I CODE SLI	8 bytes
0x22	TAG-IT HF-I	8 bytes
0x23	EM 4035	8 bytes
0x24	LRI 64/512	8 bytes
0x25	MB89R118	8 bytes
0x26	MY-D 02P/10P	8 bytes
0x31	SR 176	8 bytes
0x32	SRI 512	8 bytes