

HF RFID System



BLUEBOX GEN 2 HF MR COMMUNICATION PROTOCOL

BLUEBOX
RFid System

RS232 / RS485 / Ethernet

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

Reference table of the devices object of this manual:

Code	Communication Interface	Antenna
5231HM	Serial	1 External Channel
5232HM	Ethernet	1 External Channel

Table of Contents

1	Introduction	5
2	Operating Features.....	6
3	Communication Features.....	8
3.1	Introduction	8
3.2	Device Reset	9
3.3	Device Serial Number Reading	9
3.4	MAC Address Reading	9
3.5	FW Version Reading	10
3.6	Temperature Reading	11
3.7	Date/Time Reading	11
3.8	Date/Time Programming	12
3.9	Parameters Programming.....	12
3.10	Ethernet Parameters Programming	14
3.11	RF Parameters Programming	14
3.12	Parameters Reading	15
3.13	Ethernet Parameters Reading	16
3.14	RF Parameters Reading	17
3.15	Default Parameters Programming	17
3.16	Digital Output Activation	17
3.17	Status Reading	18
3.18	RF Deactivation.....	19
3.19	RF Activation	19
3.20	Data Request.....	19
3.21	Queue Data Request.....	20
3.22	ISO 15693 Transponders 'Inventory' Command	20
3.23	Reading a Data Block of an ISO 15693 Transponder	21
3.24	Writing a Data Block of an ISO 15693 Transponder	22
3.25	Locking a Data Block of an ISO 15693 Transponder	22
3.26	ISO 15693 Transponder 'Get System Info' Command	23
3.27	ISO 15693 Transponder 'General Protocol' Command	24
3.28	'Spontaneous' Message	25
A.	Supported Transponders	26

1 Introduction

The **BLUEBOX GEN2 INDUSTRIAL HF MR** hereinafter named **BLUEBOX** is a read/write RFID device for industrial application that communicates with a 'host' system (typically a PC or a PLC) through a RS232/RS485 serial line (item 5231HM) or through an 10-100M Ethernet connection (item 5232HM). The **BLUEBOX** acts as a joint through a set of commands between the host system and the rfid tag/s (or transponder/s) present near the antenna/s. The same 'master/slave' protocol is used for the communication between the host system ('master') and the **BLUEBOX** ('slave'), independently of the kind of connection (point to point, multidrop net, Ethernet). An USB connection, working as virtual com, is also available and is principally used as service port to configure the functional parameters and to update the firmware of the device, the 'BLUEBOX show' program of the SDK is foreseen to explicate these operations. Furthermore the **BLUEBOX** is able to handle 2 channels of digital I/O; each channel can be used as output to drive a low side load or as input either driven by a 'pnp' output or by a 'clean' contact. Warning, when the I/O is used as input, do not use it also as output to avoid conflicts! The **BLUEBOX** is available with 1 external RF antenna (item 5232HM).

Compared to the **BLUEBOX GEN2 INDUSTRIAL HF SR** device (item 5231H), the **BLUEBOX GEN2 INDUSTRIAL HF MR** device (item 5231HM) differs for the following reasons:

- increased RF power which leads to higher performance in terms of read/write distance;
- only the ISO 15693 transponders are supported;
- specific 50 ohm antennas with an IP65 industrial TNC connector have to be used;
- ability to set the RF power level and mode of communication transponder/reader (ASK with 1 subcarrier / FSK with 2 subcarriers).

2 Operating Features

Supported transponders (or tags) and relative associated functions are described in annexe A.

In 'continuous' mode the **BLUEBOX** is characterized by the coexistence of 2 'parallel' and asynchronous activities: the tag identification consisting of a repetitive inventory of tags through an anticollision loop 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 tags or that is empty indicating the absence of tags. Due to synchronization and filtering reasons, the buffer is handled for each tag by a parameter defined as 'hold time' same as 'filter time' defined below (to be set in the range of 0 ... 99 seconds or 0 ... 99 minutes, default value 1 second) and allows to extend 'artificially' the presence of the tag after it leaves the antenna's influence area; this behavior is observable looking at the antenna led status that is 'on' indicating the presence of tag/s and also through the activation of the digital output channel 1 (if its 'automatic' management is enabled by the flag defined in the general parameters). Through the command 'data request' it is possible to get the data contained in the buffer (tag/s ID).

The **BLUEBOX** handles also a 31 elements FIFO queue which is combined with the 'filter time' general parameter (to be set in a range of 0 ... 99 seconds or 0 ... 99 minutes, default value 1 second) that prevents the queue saturation in case of a tag 'continuous' presence. At the end of each inventory session, for each identified tag, the **BLUEBOX** compares it to the list of previous read transponder/s. If the considered transponder do not belong to the list, it is defined as 'new', its code (tag ID) will be inserted in the queue and the filter time will be started. Otherwise (the transponder belong to the list), 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 (tag ID) and unload it.

Every time that a new transponder is identified, the buzzer will be activated for 250ms if its 'automatic' management is enabled by the flag defined in the operating parameters.

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 and on the Ethernet channel every time that a 'new' transponder is identified.

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.

List of configurable parameters:

Parameters	Range / Choice	Default
Protocol Network address	000 ... 255	255
Serial communication Baud rate	1200, 2400, 4800, 9600, 19200, 38400	19200
Serial communication Data bits	7, 8	8
Serial communication Stop bits	1, 2	1
Serial communication Parity	None, even, odd	None
Hold time	0 ... 99 seconds	1 sec
Filter time	0 ... 99 seconds 0 ... 99 minutes	1 sec
Buzzer	Disabled, enabled	Enabled
Output channel 1	Disabled, enabled	Disabled
'Continuous mode'	Disabled, enabled	Enabled
'Spontaneous reader'	Disabled, enabled	Disabled
Ethernet IP Address	-	192.168.004.200
Ethernet TCP Port	0 ... 65535	3000
Ethernet Subnet Mask	-	255.255.255.0
Ethernet Gateway Address	0 ... 65535	0.0.0.0

List of configurable RF parameters:

Parameters	Range / Choice	Default
Transponder / reader communication mode	ASK, FSK	ASK
RF power level	0 (min) ... 7 (max)	6

3 Communication Features

3.1 Introduction

The 'master/slave' serial protocol expects that the **BLUEBOX** (as 'slave') after the reception of a message sent to it by the 'host' (as 'master'), sends back an answer message after a minimum time of about 10 ms. For the communication through the serial line interface (items 5231HM), by default, the **BLUEBOX** will apply the following parameters: 'master/slave' protocol 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. For the communication through the ethernet interface (item 5232HM), by default, the **BLUEBOX** will apply the following parameters: 'master/slave' protocol address 255, IP address 192.168.004.200, port 3000, subnet mask 255.255.255.0 and gateway address 0.0.0.0. These parameters can be modified as specified in the 'Parameters programming' protocol command for the 'master/slave' protocol address and in the 'Ethernet Parameters programming' protocol command.

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

SOH	Character 01h (0x01)
STX	Character 02h (0x02)
ETX	Character 03h (0x03)
EOT	Character 04h (0x04)
ENQ	Character 05h (0x05)
ACK	Character 06h (0x06)
NAK	Character 15h (0x15)
SYN	Character 16h (0x16)
CR	Character 0Dh (0x0D)
'0'...'9'	Character 30h ...39h (0x30 ... 0x39)
'A'...'F'	Character 41h ...46h (0x41 ... 0x46)
<...>	Character 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 $\langle bcc \rangle = \text{SOH}$ or $\langle bcc \rangle = \text{CR}$ or $\langle bcc \rangle = \text{EOT}$, then $\langle bcc \rangle := \langle bcc \rangle + 1$ (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 $\langle \text{add h} \rangle$ corresponds to the ASCII coding of the high nibble of the byte, while the second ASCII character $\langle \text{add l} \rangle$ corresponds to the ASCII coding of the low nibble of the byte. Example: $255 \rightarrow 0xFF \rightarrow \text{'F' 'F'}$. 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).

3.2 Device Reset

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

The 'master' sends the following command:

SOH $\langle \text{add h} \rangle \langle \text{add l} \rangle \text{STX '3' '0' ETX } \langle bcc \rangle \text{ CR}$

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

SOH $\langle \text{add h} \rangle \langle \text{add l} \rangle \text{NAK } \langle bcc \rangle \text{ CR}$

Otherwise it answers:

SOH $\langle \text{add h} \rangle \langle \text{add l} \rangle \text{ACK } \langle bcc \rangle \text{ CR}$

3.3 Device Serial Number Reading

This command is used to get the SN code of the **BLUEBOX** (unique for each device and assigned during the production process), the SN is constituted by 6 bytes.

The 'master' sends the following command:

SOH $\langle \text{add h} \rangle \langle \text{add l} \rangle \text{STX '2' 'A' '0' '1' ETX } \langle bcc \rangle \text{ CR}$

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

SOH $\langle \text{add h} \rangle \langle \text{add l} \rangle \text{NAK } \langle bcc \rangle \text{ CR}$

Otherwise it answers:

SOH $\langle \text{add h} \rangle \langle \text{add l} \rangle \text{STX '2' 'A' '0' '1' } \langle \text{sn 1 h} \rangle \langle \text{sn 1 l} \rangle \dots \langle \text{sn i h} \rangle \langle \text{sn i l} \rangle \dots \langle \text{sn 6 h} \rangle \langle \text{sn 6 l} \rangle \text{ETX } \langle bcc \rangle \text{ CR}$

Where:

i	1 ... 6.
$\langle \text{sn i h} \rangle \langle \text{sn i l} \rangle$	i-th byte of the sn of the device. ASCII encoded byte.

Note: the SN is a numeric code constituted by 12 digits, the bytes of the SN are BCD-coded and so every byte encodes 2 digits.

3.4 MAC Address Reading

This command is used to get the MAC address of the **BLUEBOX** (unique for each device), the MAC address is constituted by 6 bytes.

The 'master' sends the following command:

SOH $\langle \text{add h} \rangle \langle \text{add l} \rangle \text{STX '2' 'A' '0' '2' ETX } \langle bcc \rangle \text{ CR}$

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

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

Otherwise it answers:

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

Where:

i	1 ... 6.
<mac i h> <mac i l>	i-th byte of the mac address of the device. ASCII encoded byte.

3.5 FW Version Reading

This command is used to get the firmware version of the **BLUEBOX**.

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:

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

Otherwise it answers:

**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 a string of 16 ASCII characters that defines the version. The general form is: **'BB_TWO_HM y.yy '**; y.yy gives the firmware version (example **1.16**).

It is also possible to get the firmware version of the HFM reader module mounted in the **BLUEBOX**.

The 'master' sends the following command to get the firmware version of module:

SOH <add h> <add l> STX '3' '4' '0' '1' ETX <bcc> CR

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

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

Otherwise it answers:

**SOH <add h> <add l> STX '3' '4' '0' '1' <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 a string of 16 ASCII characters that defines the version. The general form is: **'MIDRANGE x.x ';** x.xx gives the firmware version (example **1.7**).

3.6 Temperature Reading

This command sends back the internal temperature of the **BLUEBOX** measured by the on board temperature sensor.

The 'master' sends the following command:

SOH <add h> <add l> '3' 'A' <bcc> CR

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

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

Otherwise it answers:

SOH <add h> <add l> STX '3' 'A' <temp 1 h> <temp 1 l> <temp 2 h> <temp 2 l> ETX <bcc> CR

Where:

<temp 1 h> <temp 1 l>	Integer value of the temperature in °C. ASCII encoded byte.
<temp 2 h> <temp 2 l>	Fractional value of the temperature. ASCII encoded byte. Bits 7, 6, 5 encode the fractional value in steps of 0.125 °C: 00000000b → .000 °C 00100000b → .125 °C ... 11100000b → .875°C.

Example: 28h, E0h → 40.875 °C.

3.7 Date/Time Reading

This command sends back the date/time of the **BLUEBOX** available on the internal real time clock device.

The 'master' sends the following command:

SOH <add h> <add l> '2' '8' <bcc> CR

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

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

Otherwise it answers:

SOH <add h> <add l> STX '2' '8' <yea 1 h> <yea 1 l> <yea 2 h> <yea 2 l> <mon h> <mon l> <day h> <day l> <hou h> <hou l> <min h> <min l> <sec h> <sec l> ETX <bcc> CR

Where:

<yea 1 h> <yea 1 l>	Year value thousands and hundereds. ASCII encoded byte. BCD encoded byte.
<yea 2 h> <yea 2 l>	Year value tens and units. ASCII encoded byte. BCD encoded byte.
<mon h> <mon l>	Month value tens and units. ASCII encoded byte. BCD encoded byte.

<day h> <day l>	Day value tens and units. ASCII encoded byte. BCD encoded byte.
<hou h> <hou l>	Hour value tens and units. ASCII encoded byte. BCD encoded byte.
<min h> <min l>	Minute value tens and units. ASCII encoded byte. BCD encoded byte.
<sec h> <sec l>	Second value tens and units. ASCII encoded byte. BCD encoded byte.

3.8 Date/Time Programming

This command is used to set the date/time of the **BLUEBOX** in the internal real time clock device.

The 'master' sends the following command:

SOH <add h> <add l> STX '2' '9' <yea 1 h> <yea 1 l> <yea 2 h> <yea 2 l> <mon h> <mon l> <day h> <day l> <hou h> <hou l> <min h> <min l> <sec h> <sec l> ETX <bcc> CR

Where:

<yea 1 h> <yea 1 l>	Year value thousands and hundreds. ASCII encoded byte. BCD encoded byte.
<yea 2 h> <yea 2 l>	Year value tens and units. ASCII encoded byte. BCD encoded byte.
<mon h> <mon l>	Month value tens and units. ASCII encoded byte. BCD encoded byte.
<day h> <day l>	Day value tens and units. ASCII encoded byte. BCD encoded byte.
<hou h> <hou l>	Hour value tens and units. ASCII encoded byte. BCD encoded byte.
<min h> <min l>	Minute value tens and units. ASCII encoded byte. BCD encoded byte.
<sec h> <sec l>	Second value tens and units. ASCII encoded byte. BCD 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:

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

3.9 Parameters Programming

This command is used to set the protocol address, the serial communication (don't care for ethernet communication) and the functional 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> '0' '1' '0' '3' <filt h> <filt l> <flag h>
 <flag l> ETX <bcc> CR**

Where:

<adda h> <adda l>	Protocol address. ASCII encoded byte.
<addn h> <addn l>	New protocol address to be set. ASCII encoded byte.
<bdr>	RS232/RS485 communication interface baud rate. ASCII character: <ul style="list-style-type: none"> • '0' -> 1200 bps • '1' -> 2400 bps • '2' -> 4800 bps • '3' -> 9600 bps • '4' -> 19200 bps • '5' -> 38400 bps.
<bit>	RS232/RS485 communication interface data bits. ASCII character: <ul style="list-style-type: none"> • '7' -> 7 bits • '8' -> 8 bits.
<stop>	RS232/RS485 communication interface stop bits. ASCII character: <ul style="list-style-type: none"> • '1' -> 1 bit • '2' -> 2 bits.
<par>	RS232/RS485 communication interface parity. ASCII character: <ul style="list-style-type: none"> • '0' -> None • '1' -> Even • '2' -> Odd.
<filt h> <filt l>	Filter time. ASCII encoded byte: <ul style="list-style-type: none"> • Decimal 0 ... 99 for time in seconds (0 ... 99 seconds) • Decimal 100 ... 199 for time in minutes (0 ... 99 minutes).
<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"> • Bit 7: automatic buzzer management • Bit 6: automatic output 1 management • Bit 5 ... bit 4: not used • Bit 3: 'spontaneous' message • Bit 2 ... bit 1: not used • Bit 0: to disable 'continuous' mode.

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

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

Otherwise it answers:

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

Note: after the command execution, the **BLUEBOX** resets itself to apply the new parameters.

3.10 Ethernet Parameters Programming

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

The 'master' sends the following command:

SOH <add h> <add l> STX '3' 'D' '8' '0' <IP 1 h> <IP 1 l> <IP 2 h> <IP 2 l> <IP 3 h> <IP 3 l> <IP 4 h> <IP 4 l> <port hh> <port hl> <port lh> <port ll> <subnet 1 h> <subnet 1 l> <subnet 2 h> <subnet 2 l> <subnet 3 h> <subnet 3 l> <subnet 4 h> <subnet 4 l> <gateway 1 h> <gateway 1 l> <gateway 2 h> <gateway 2 l> <gateway 3 h> <gateway 3 l> <gateway 4 h> <gateway 4 l> ETX <bcc> CR

Where:

<IP 1 h> <IP 1 l> ... <IP 4 h> <IP 4 l>	IP address. ASCII encoded 4-byte array.
<port hh> <port hl> <port lh> <port ll>	Communication port. ASCII encoded word.
<subnet 1 h> <subnet 1 l> ... <subnet 4 h> <subnet 4 l>	Subnet mask. ASCII encoded 4-byte array.
<gateway 1 h> <gateway 1 l> ... <gateway 4 h> <gateway 4 l>	Gateway address. ASCII encoded 4-byte array.

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

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

Otherwise it answers:

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

Note: the new settings are valid only after a **BLUEBOX** reset.

3.11 RF Parameters Programming

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

The 'master' sends the following command:

SOH <add h> <add l> STX '3' 'D' '0' '0' <type h> <type l> <pwr h> <pwr l> '0' '0' '0' '0' '0' '0' '0' '0' '0' '0' ETX <bcc> CR

Where:

<type h> <type l>	Transponder / reader communication mode (ASK / FSK). ASCII encoded byte: <ul style="list-style-type: none"> • 0x20: ASK • 0x30: FSK.
<pwr h> <pwr l>	RF power level. ASCII encoded byte: <ul style="list-style-type: none"> • 0 (min) ... 7 (max).

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

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

Otherwise it answers:

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

3.12 Parameters Reading

This command is used to get the values of the protocol address, the serial communication and the functional operating 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:

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

Otherwise it answers:

**SOH <add h> <add l> STX '2' 'A' <add h> <add l> <bdr> <bit>
<stop> <par> <man h> <man l> '0' '1' <filt h> <filt l> <flag h>
<flag l> ETX <bcc> CR**

Where:

<add h> <add l>	Protocol address. ASCII encoded byte.
<bdr>	RS232/RS485 communication interface baud rate. ASCII character: <ul style="list-style-type: none"> • '0' -> 1200 bps • '1' -> 2400 bps • '2' -> 4800 bps • '3' -> 9600 bps • '4' -> 19200 bps • '5' -> 38400 bps.
<bit>	RS232/RS485 communication interface data bits. ASCII character: <ul style="list-style-type: none"> • '7' -> 7 bits • '8' -> 8 bits.
<stop>	RS232/RS485 communication interface stop bits. ASCII character: <ul style="list-style-type: none"> • '1' -> 1 bit • '2' -> 2 bits.

<par>	RS232/RS485 communication interface parity. ASCII character: <ul style="list-style-type: none"> • '0' -> None • '1' -> Even • '2' -> Odd.
<man h> <man l>	Hold time. ASCII encoded byte: <ul style="list-style-type: none"> • Decimal 0 ... 99 for time in seconds (0 ... 99 seconds).
<filt h> <filt l>	Filter time. ASCII encoded byte: <ul style="list-style-type: none"> • Decimal 0 ... 99 for time in seconds (0 ... 99 seconds) • Decimal 100 ... 199 for time in minutes (0 ... 99 minutes).
<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"> • Bit 7: automatic buzzer management • Bit 6: automatic output 1 management • Bit 5 ... bit 4: not used • Bit 3: 'spontaneous' message • Bit 2 ... bit 1: not used • Bit 0: to disable 'continuous' mode.

3.13 Ethernet Parameters Reading

This command is used to get the values of the ethernet communication parameters of the **BLUEBOX**.

The 'master' sends the following command:

SOH <add h> <add l> STX '3' 'C' '8' '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 it answers:

SOH <add h> <add l> STX '3' 'C' <IP 1 h> <IP 1 l> <IP 2 h> <IP 2 l> <IP 3 h> <IP 3 l> <IP 4 h> <IP 4 l> <port hh> <port hl> <port lh> <port ll> <subnet 1 h> <subnet 1 l> <subnet 2 h> <subnet 2 l> <subnet 3 h> <subnet 3 l> <subnet 4 h> <subnet 4 l> <gateway 1 h> <gateway 1 l> <gateway 2 h> <gateway 2 l> <gateway 3 h> <gateway 3 l> <gateway 4 h> <gateway 4 l> ETX <bcc> CR

Where:

<IP 1 h> <IP 1 l> ... <IP 4 h> <IP 4 l>	IP address. ASCII encoded 4-byte array.
<port hh> <port hl> <port lh> <port ll>	Communication port. ASCII encoded word.

<subnet 1 h> <subnet 1 l>

...

<subnet 4 h> <subnet 4 l>

Subnet mask. ASCII encoded 4-byte array.

<gateway 1 h> <gateway 1 l>

...

< gateway 4 h> < gateway 4 l>

Gateway address. ASCII encoded 4-byte array.

3.14 RF Parameters Reading

This command is used to get the values of the protocol address, the serial communication and the functional operating parameters of the **BLUEBOX**.

The 'master' sends the following command:

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

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

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

Otherwise it answers:

SOH <add h> <add l> STX '3' 'C' <type h> <type l> <pwr h> <pwr l> '0' '0' '0' '0' '0' '0' '0' '0' '0' '0' ETX <bcc> CR

Where:

<type h> <type l>

Transponder / reader communication mode (ASK / FSK).
ASCII encoded byte:

- 0x20: ASK
- 0x30: FSK.

<pwr h> <pwr l>

RF power level. ASCII encoded byte:

- 0 (min) ... 7 (max).

3.15 Default Parameters Programming

This command is used to set the default values of the communication, general and RF 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:

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

Otherwise it answers:

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

Note: after the command execution, the **BLUEBOX** resets itself to apply the new parameters.

3.16 Digital Output Activation

This command is used to excite each individual output and to set also the duration in case of impulsive use.

The 'master' sends the following command:

SOH <add h> <add l> STX '3' '7' <can h> <can l> <dur h> <dur l> ETX <bcc> CR

Where:

<can h> <can l>	Relay to activate. ASCII encoded byte: <ul style="list-style-type: none"> • 0x01 -> Output 1 • 0x02 -> Output 2.
<dur h> <dur l>	Activation time. ASCII encoded byte: <ul style="list-style-type: none"> • 0x01 ... 0x63 (1 ... 99 seconds -> in case of 'impulsive' activation • 0x81 -> 'Continuous' activation • 0x80 -> Deactivation.

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

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

Otherwise it answers:

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

3.17 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:

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

Otherwise it answers:

SOH <add h> <add l> STX '3' '6' <sta 1 h> <sta 1 l> <sta 2 h> <sta 2 l> ETX <bcc> CR

Where:

<sta 1 h> <sta 1 l>	BLUEBOX status byte 1. ASCII encoded byte whose bits have the following meaning: <ul style="list-style-type: none"> • Bit 7, 6: Not used • Bit 5: RF status (0=off, 1=on) • Bit 4: 'continuous' mode status (0=disab., 1=enab.) • Bit 3, 2: Not used • Bit 1: Output 2 status (1=activated) • Bit 0: Output 1 status (1=activated).
<sta 2 h> <sta 2 l>	BLUEBOX status byte 2. ASCII encoded byte whose bits have the following meaning: <ul style="list-style-type: none"> • Bit 7...2: Not used • Bit 1: Input 2 status (1=activated) • Bit 0: Input 1 status (1=activated).

3.18 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:

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

Otherwise it answers:

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

3.19 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:

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

Otherwise it answers:

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

3.20 Data Request

This command sends back the code (tag ID) of the eventual transponder/s 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 (inventory) 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:

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

Otherwise, it answers,

a) if transponder/s have been identified by the antenna:

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

Where:

i	0 ... m
m	UID length, 8 byte for ISO 15693 transponders.
j	0 ... n

n	Number of identified tags.
<type j h> <type j l>	Transponder type. See Annexe A for type assignment table.
<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 no transponder is identified by the antenna:

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

3.21 Queue Data Request

In 'continuous' mode, when the **BLUEBOX** finds a 'new' transponder, it inserts its 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:

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

Otherwise, it answers:

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

<type h> <type l>	<ul style="list-style-type: none"> Transponder type. See Annexe A for type assignment table.
i	0 ... n
n	UID length, 8 byte for ISO 15693 transponders.
<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' '0' ETX <bcc> CR

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

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

3.22 ISO 15693 Transponders 'Inventory' Command

This command is used to get 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:

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

Otherwise it answers,

a) 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

3.23 Reading a Data Block 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 and the number of blocks depends on the transponder type; for example, the **ICODE2** transponder is organized in blocks of 4 bytes, the **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> <blk h> <blk 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.
<blk h> <blk l>	Address of the block to read. 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' '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 block read.

<data i h> <data i l>

i-th byte of the block read from tag. ASCII encoded byte.

b) if the addressed tag do not support the requested block 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

3.24 Writing a Data Block 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 **ICODE2** transponder is organized in blocks of 4 bytes, the **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> <blk h> <blk 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.
<blk h> <blk l>	Address of the block to write. ASCII encoded byte.
j	1 ... n.
n	Number of bytes of the block to write.
<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:

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 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

3.25 Locking a Data Block of an ISO 15693 Transponder

This command is used to lock a 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' '3' (for antenna 2: '9' '3') <UID 1 h>

**<UID 1 I> ... <UID i h> <UID i I> ... <UID 8 h> <UID 8 I> <blk h>
<blk I> ETX <bcc> CR**

Where:

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

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

SOH <add h> <add I> NAK <bcc> CR

Otherwise it answers,

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

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

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

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

c) if the addressed tag is not present:

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

3.26 ISO 15693 Transponder 'Get System Info' Command

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 I> STX '1' '4' <UID 1 h> <UID 1 I> ... <UID i h>
<UID i I> ... <UID 8 h> <UID 8 I> ETX <bcc> CR**

Where:

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

where:

<flg h> <flg I>	<p>Info Flags of the tag. ASCII encoded byte.</p> <p>Single bits are dedicated to specify the presence of the following fields (0 → absent, 1 → present):</p> <ul style="list-style-type: none"> • Bit 7 ... bit 4: not used • Bit 3: IC Reference (1 byte) • Bit 2: Memory Size (2 bytes) • Bit 1: AFI (1 byte)
-----------------	--

	<ul style="list-style-type: none"> Bit 0: DSFID (1 byte).
<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

3.27 ISO 15693 Transponder 'General Protocol' Command

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

3.28 'Spontaneous' Message

In 'continuous' mode, if the 'spontaneous' feature is set on (see parameters), the **BLUEBOX** will send the following message on the serial line (item 5231HM) and on the Ethernet channel (item 5232HM) 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 Annexe A for type assignment table.
i	1 ... 8 (the UID length).
<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)

A. Supported Transponders

Supported transponders by the **BLUEBOX** are:

Manuf.	Part	Standard	Chip	Notes / Functions
NXP	ICODE2	15693	SL2 IC S20	Inventory Read block (4 bytes) Write block (4 bytes) Lock block Get system info
TI	TAG-IT HF-I	15693		Inventory Read block (4 bytes) Write b block (4 bytes) Lock block Get system info
MEM	EM 4035	15693		Inventory Read block (8 bytes) Write block (8 bytes) Lock block Get system info
STM	LRI 64	15693		Inventory Read block (1 byte) Write block (1 byte) Get system info
STM	LRI 512	15693		Inventory Read block (4 bytes) Write block (4 bytes) Lock block Get system info
FUJITSU	MB89R118	15693		Inventory Read block (8 bytes) Write block (8 bytes) Get system info
INFINEON	MY-D 02P	15693	SRF 55V02P	Inventory Read block (4 bytes) Write block (4 bytes) Lock block
INFINEON	MY-D 10P	15693	SRF 55V10P	Inventory Read block (4 bytes) Write block (4 bytes)

Type coding for supported transponders:

Type Code	Part	UID Length
0x20	Generic ISO 15693	8 bytes
0x21	ICODE2	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