Idesco EPC Anticollision

User Manual

Version	Date	Author	Description
1.00	28.06.2011	Pah	First version

Idesco Oy | Teknologiantie 9, FIN-90590 OULU | Tel +358 20 743 4175 | info@idesco.fi | www.idesco.fi

Idesco Oy | Teknologiantie 9, FIN-90590 OULU | Tel +358 20 743 4175 | info@idesco.fi | www.idesco.fi

Idesco Oy | Teknologiantie 9, FIN-90590 OULU | Tel +358 20 743 4175 | info@idesco.fi | www.idesco.fi

1. Purpose of This Document

The purpose of this document is to give a comprehensive understanding of usage, installation and features of Idesco EPC Anticollision reader.

Please read this manual carefully before installing and using Idesco EPC Anticollision readers.

Do not use Idesco EPC Anticollision readers without selected antenna / antennas! If used without a selected antenna, the receiver part of the reader electronics may suffer damage.

2. Using Idesco EPC Anticollision Reader

2.1. Brief Operational Description

Idesco EPC Anticollision reader is a multipurpose reader designed for vehicle identification, automation and logistics applications. The reader can read and write data to EPC Gen2 tags. The reader isn't polling automatically, but requires commands from the host.

The reader supports RS-232 and optional Ethernet interfaces.

When the reader is switched on, the red and yellow LED's flash during start-up. After initialization and during normal operation the LED is switched off.

Green or red LED can be switched on any time by grounding corresponding input pin. However, by grounding both input pins simultaneously yellow LED is switched on.

Buzzer can also be switched on any time by grounding the buzzer input pin. Please see the installation instructions for wiring details.

LED's and FET output can be controlled also by commands described in chapter 5.5.

2.2. Default Settings for RS-232 and Ethernet connection

Settings for RS-232 serial communications are: 9600 bauds, no parity, 8 data bits, 1 start bit, stop bit. These settings cannot be changed.

The Ethernet connection is carried out by a Lantronix Xport ® converter module, which converts the Idesco EPC protocol sent by the Idesco EPC Anticollision reader from RS-232 into TCP/IP format. As default Ethernet module obtains IP address automatically.

As a default, the reader uses only its internal antenna and transmitting power setting is 28.95 dBm for both - read and write commands. Real transmitting power can be calculated by adding 6.15 dB to the value set by commands, so actual default transmitting power value is 35.1 dBm.

(6.15 dBm has to be added to the value set by commands, because the antenna gain needs to be taken into account when calculating actual transmitting power.)

2.3. Installation of Idesco EPC Anticollision Reader and Transponders

The reader can be installed using installation kit provided with the reader. The installation kit can be used for installation on a pole (pole sizes 1.75"-3") or mounting on a wall or ceiling. The installation angle of the reader can be adjusted by installation kit in both horizontal and vertical directions.

As the maximum reading distance between the tags and the reader is achieved directly in the front of the reader, the installation angle of the reader should be adjusted so that the tag is placed directly in front of the reader aligned on the same plane with the reader.

The tags should be installed on the material they are designed to work on. For example the Idesco EPC Windshield Label is designed to work properly when installed on glass. As a material, plastic is quite close to glass when looking at electrical properties, so the Idesco EPC Windshield Label works also when installed on plastic.

The inseparable 3 meter cable of the reader includes power, communication and LED/Buzzer control wires. The colors of the wires in the cable are described in separate Installation Instructions document. Installation Instructions are available from Idesco website or from support@idesco.fi.

3. Using optional Ethernet connection

3.1. Ethernet Interface

The Ethernet connection is carried out by a Lantronix Xport ® converter module, which converts the Idesco EPC protocol from RS-232 into TCP/IP format and vice versa. Usage of this connection requires the Xport software package. The needed software can be downloaded from these web links:

Device Installer:

http://www.lantronix.com/device-networking/utilities-tools/device-installer.html

Com Port Redirector:

http://www.lantronix.com/device-networking/utilities-tools/com-port-redirector.html

There is a class IP67 Ethernet connector in the backside of the Idesco EPC Anticollision reader. There is also a cable clamp for Ethernet cable to make the connection thoroughly protected against harsh environments.

3.2. Ethernet Cable Specification

The cable used connecting the Idesco EPC Anticollision reader to a host system via Ethernet must comply with the following specifications.

The cable must be of type Cat5e and the connectors must be of type RJ45. Depending on the host system, the cable should be either a straight-through or a crossover cable. The Ethernet interface signals are described on Table 1.

Signal name	DIR	Contact	Primary Function
TX+	Out	1	Transmit Data +
TX-	Out	2	Transmit Data -
RX+	In	3	Receive Data +
RX-	In	6	Receive Data -
Not Used		4	Terminated
Not Used		5	Terminated
Not Used		7	Terminated
Not Used		8	Terminated
SHIELD			Chassis Ground

Table 1. Ethernet interface signals.

4. Establishing Ethernet Connection

This chapter gives you an introduction in setting up communication between the Idesco EPC Anticollision reader and a PC. It covers a basic configuration of Lantronix XPort® that is used in Idesco EPC Anticollision reader and provides with screenshots of a functioning configuration. If needed, more information on XPort® can be found on www.lantronix.com.

4.1. Installing PC Software

Two PC programs should be installed on a PC. Device Installer is needed for configuring readers and Com Port Redirector (CPR) for setting up virtual COM ports. With the CPR you are enabled to run any terminal software that uses COM ports to operate with the Idesco EPC Anticollision reader.

Log in to Windows operating system with administrator rights and install the Device Installer. After this, install the Com Port Redirector. During the installation you may need an Internet connection.

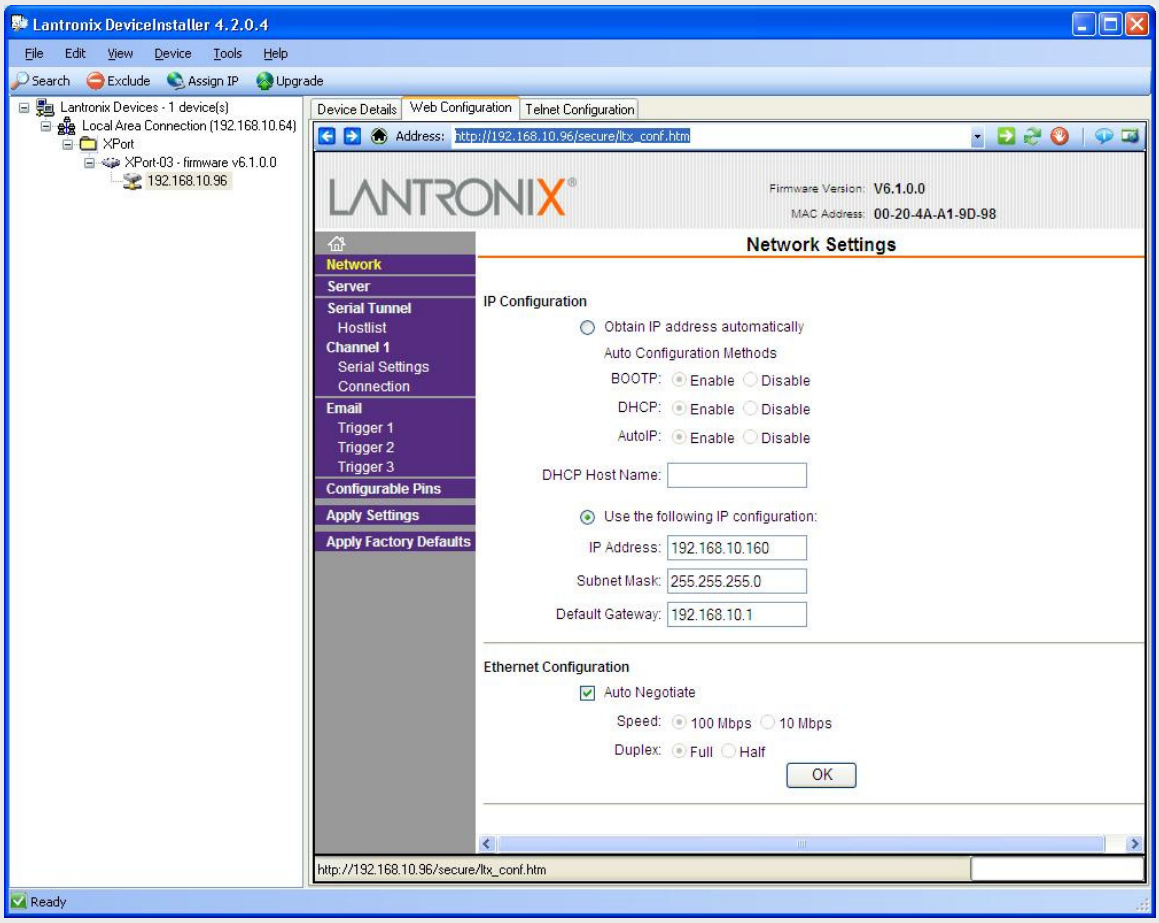
4.2. Configuring Idesco EPC Anticollision reader

4.2.1. Connecting Idesco EPC Anticollision Reader to a PC

Connect the devices with a crossover cable to each other and power up the reader. Alternatively, you can connect the PC and the reader to a local area network (LAN) or a corporate network, usually with a straight-thru cable.

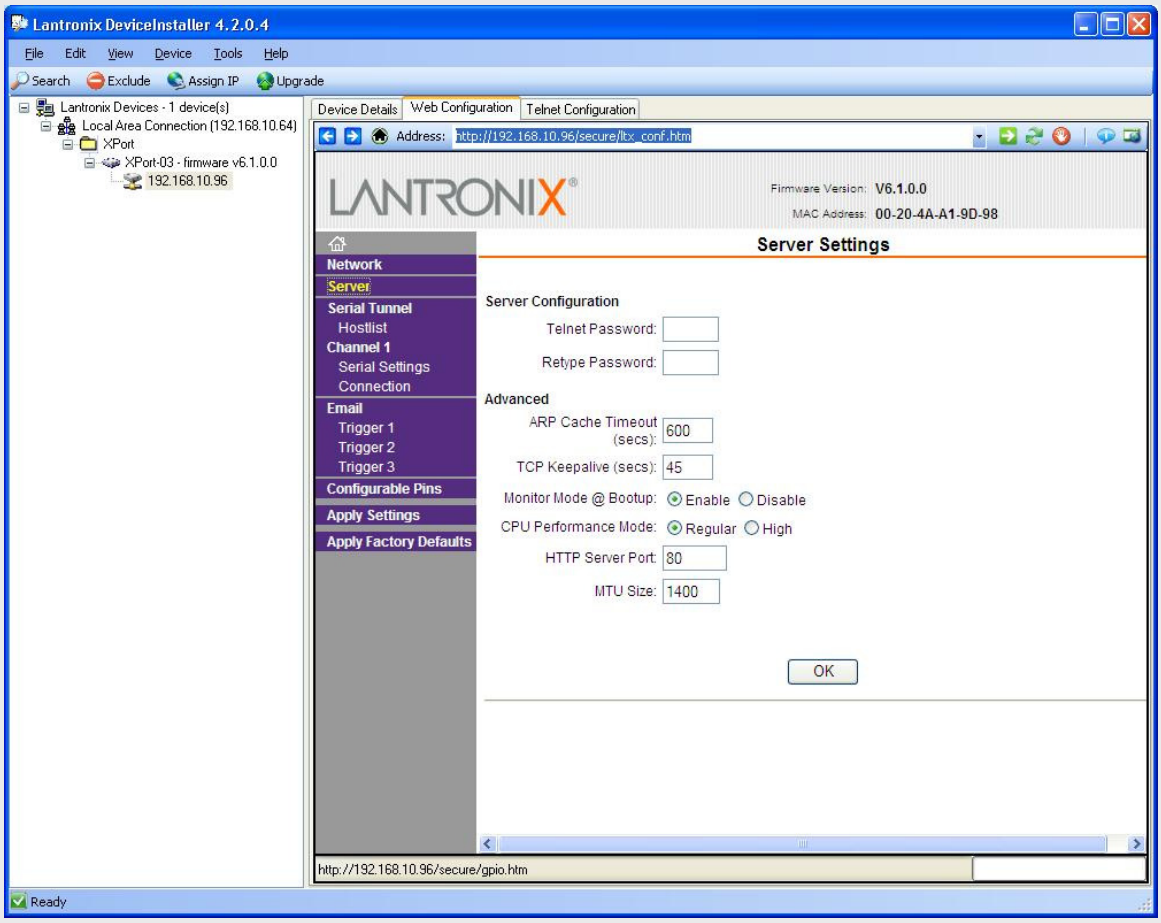
4.2.2. Changing Settings of Idesco EPC Anticollision Reader

1. Run Device Installer and click Search.
2. When a device is found, click it, and then click “Web Configuration” and “Go”.
3. Click OK without entering a user name or password.
4. Check that the reader settings are as described in pictures 1-4 below.
5. On Network Settings screen (Picture 1) select “use the following IP configuration”. Enter the chosen reader’s IP address, Subnet Mask and Default Gateway in the fields. A static IP address is needed if you are going to use the virtual serial port in PC. If you use local area network or a corporate network for establishing the connection, you can leave “Obtain IP address automatically” selected.
6. On Connection screen (Picture 4) the local port should be set to 10001.
7. After all settings are correct, click “Apply Settings”. If you want to improve safety, you can set the user name and password, so unauthorized users are not able to change the configuration.



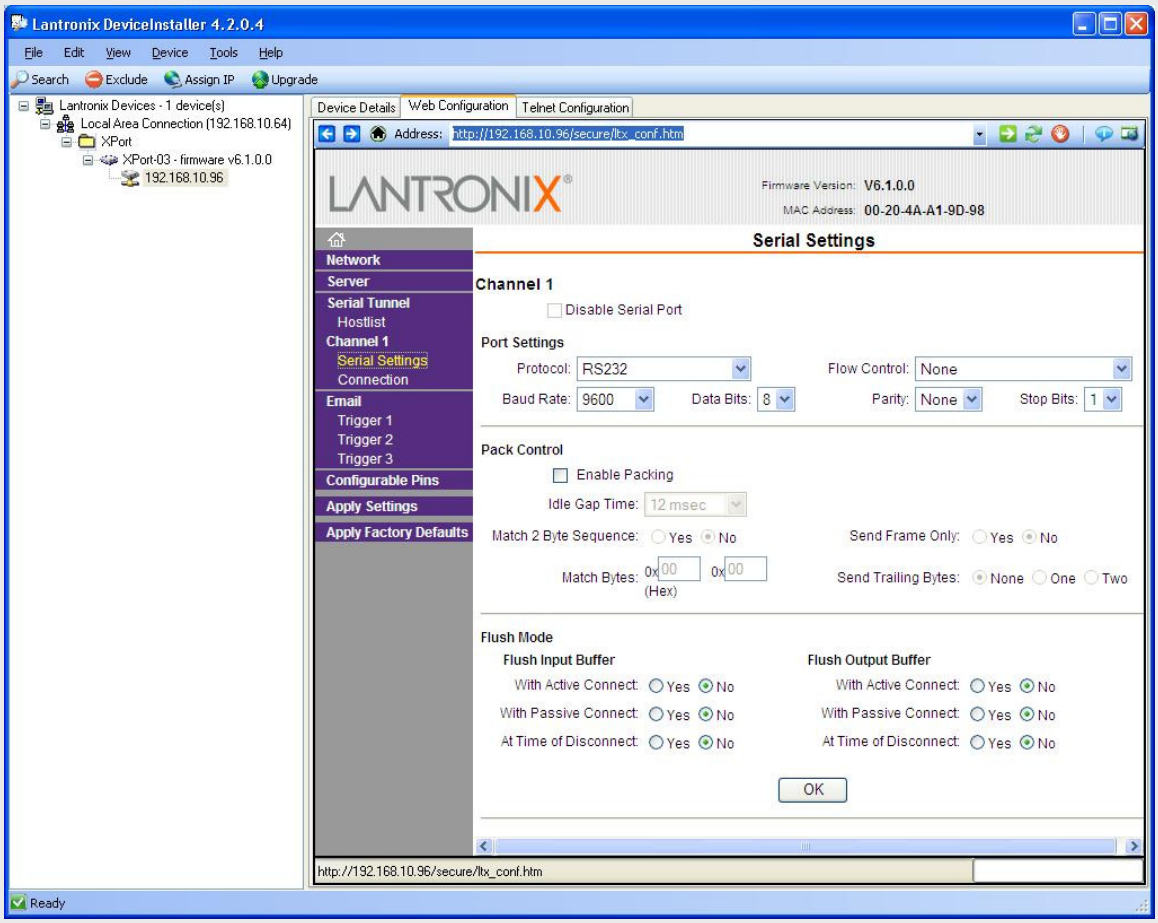
Picture 1. Example of network settings

Idesco Oy | Teknologiantie 9, FIN-90590 OULU | Tel +358 20 743 4175 | info@idesco.fi | www.idesco.fi



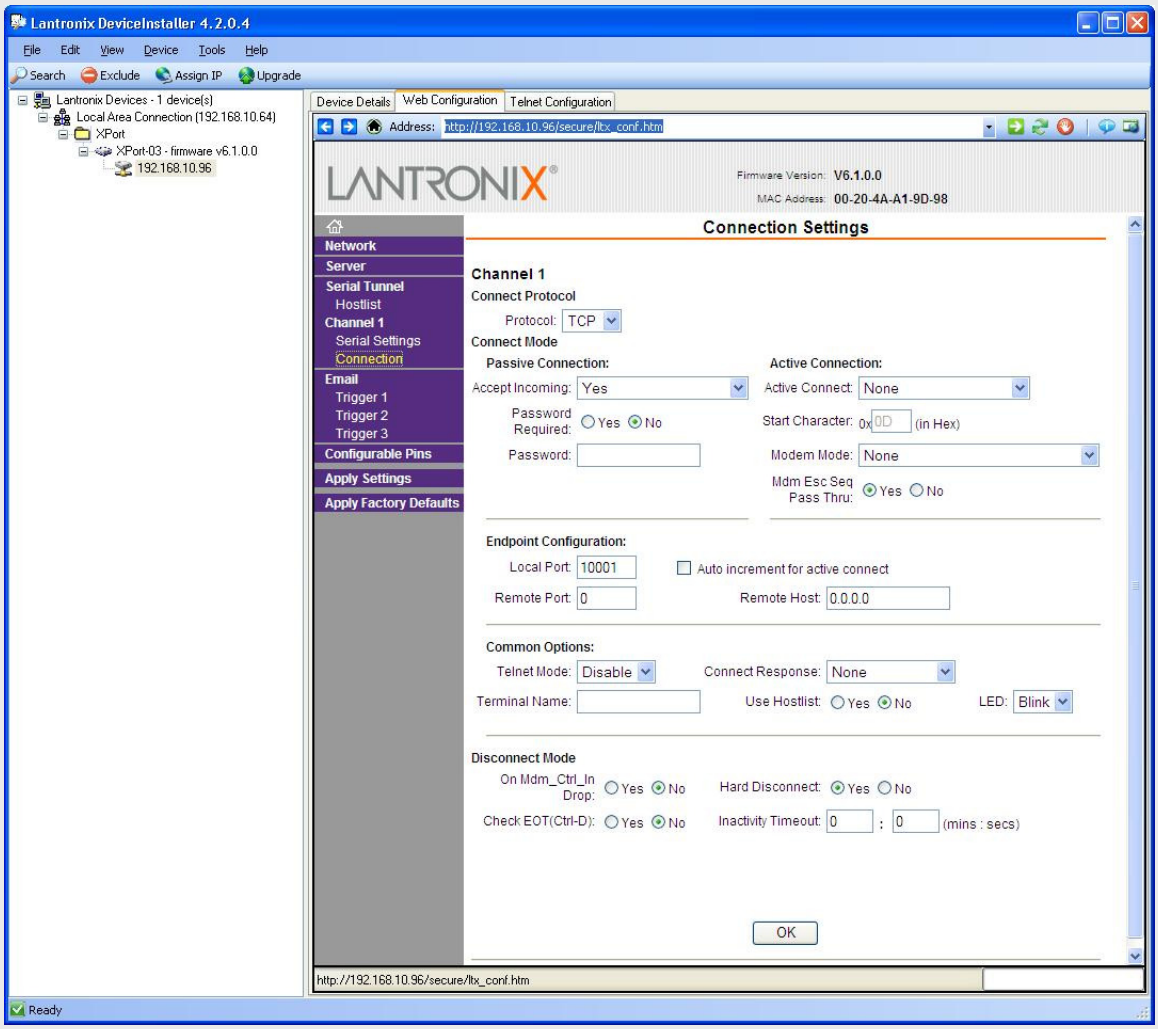
Picture 2. Server settings

Idesco Oy | Teknologiantie 9, FIN-90590 OULU | Tel +358 20 743 4175 | info@idesco.fi | www.idesco.fi



Picture 3. Serial settings

Idesco Oy | Teknologiantie 9, FIN-90590 OULU | Tel +358 20 743 4175 | info@idesco.fi | www.idesco.fi



Picture 4. Connection settings

4.3. Configuring PC

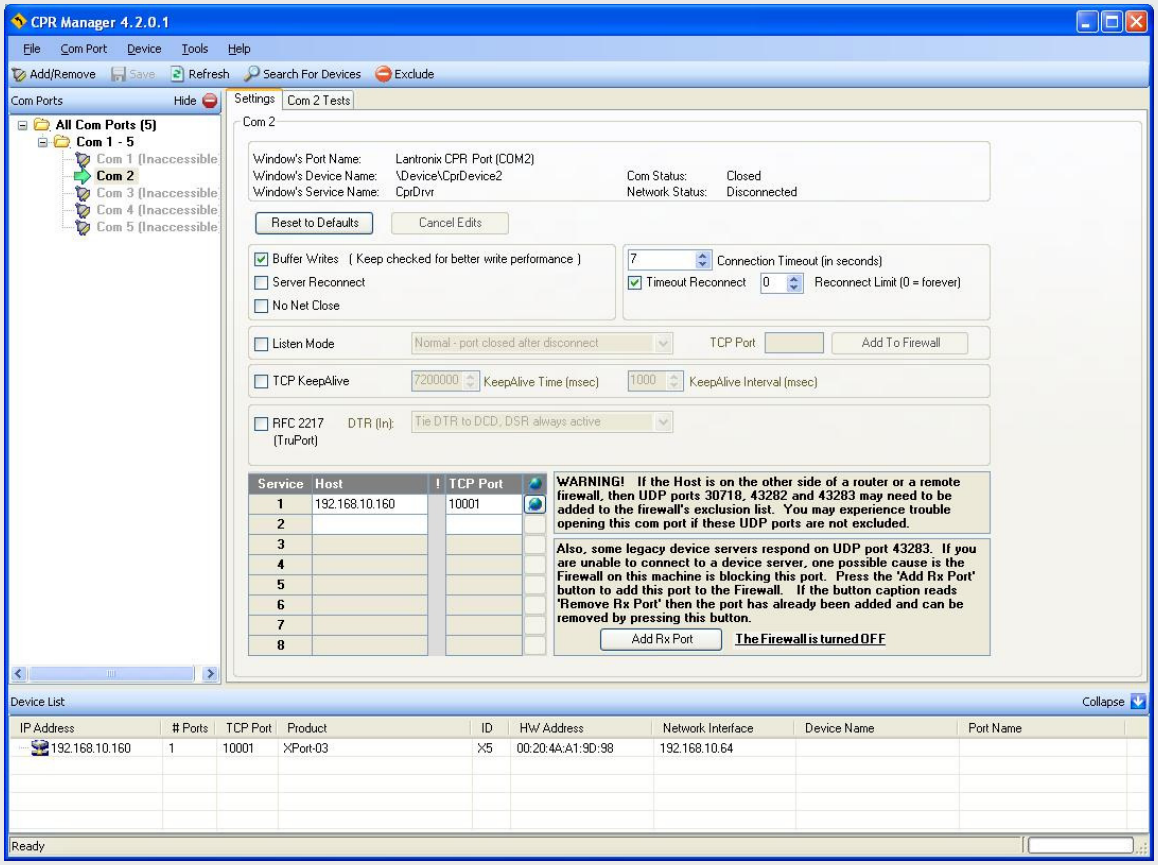
Creating and configuring the virtual serial port

Please note: You need to have the administrator rights when using CPR Manager. In addition you must have correct settings in your PC's Local Area Connections. The IP address of your computer should be in the same subnet as the Idesco EPC Anticollision reader. (For example TCP/IP: IP address (192.168.10.10), subnet mask (255.255.255.0) and default gateway (192.168.10.1).

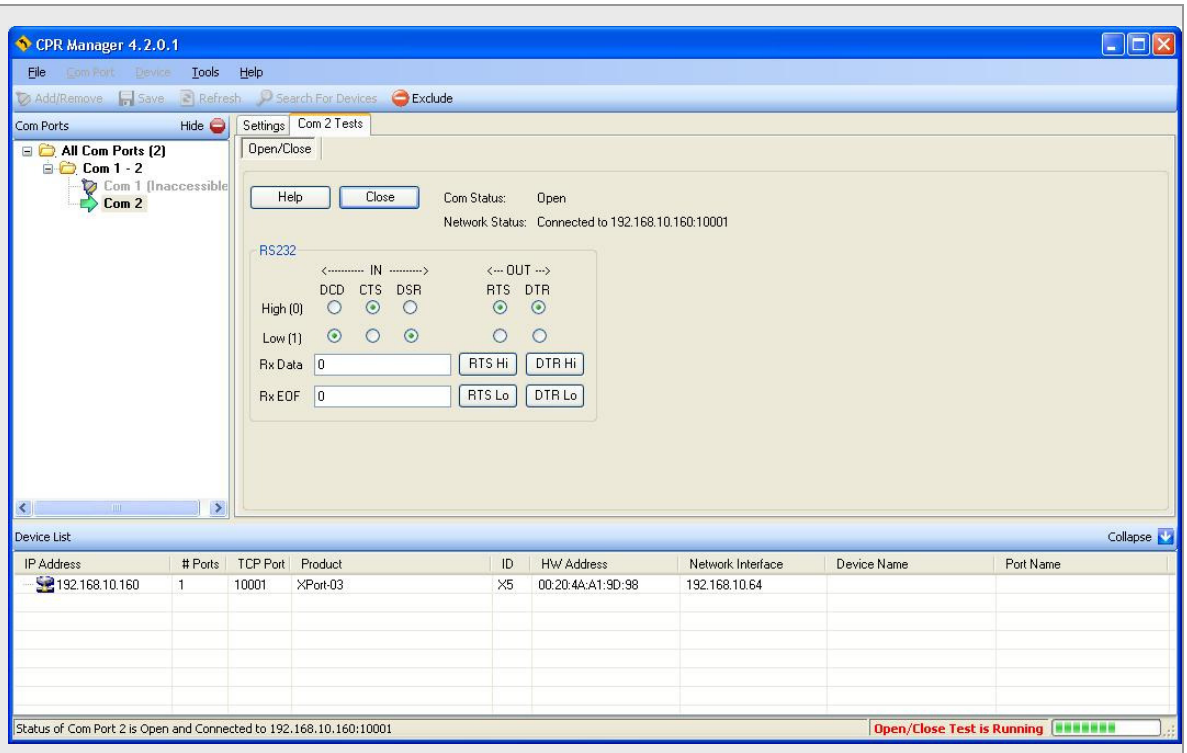
8. Run CPR Manager and Click "Search for Devices".
9. When devices are found, click "Add and Remove COM Ports".
10. Select the COM port you want to use and click Ok button.
11. Go to Settings (Picture 5). Write the IP address of your reader in the Host field and 10001 in the TCP Port field. Then click "Save Settings".
12. After this you can check if the port works by clicking "Open" in the port's test menu (Picture 6). When the port opens successfully it is properly configured and ready for use (the reader must be

connected and power must be switched on – and the IP of the reader must match the IP of COM Port Redirector).

13. After these steps, close the port and the CPR Manager and you can start using your application software.



Picture 5. CPR Manager's COM Port settings



Picture 6. CPR Manager's Port test menu

5. Communication Protocol

The Idesco EPC Anticollision reader doesn't read the tags automatically as all other Idesco EPC readers, but it requires commands from host system to be able to communicate with tags. In this chapter, the communication protocol between the host and the reader is described.

5.1. Overview of the Communication Protocol

The serial communication between a computer (host) and the reader is based on a synchronized command-response/master-slave mechanism. Whenever the host sends a message to the reader, it cannot send another message until after it receives a response. The reader never initiates a communication session; only the host initiates a communication session.

This protocol allows for each command to have its own timeout because some commands require more time to execute than others. The host manages retries, if necessary. The host keeps track of the state of the intended reader if it reissues a command.

5.1.1. Host-to-Reader Communication

Host-to-reader communication is packetized according to the following diagram. The reader can only accept one command at a time, and commands are executed serially, so the host waits for a reader-to-host response before issuing another host-to-reader command packet.

The fields are summarized in the following table:

Field	Length	Description
Header	1 byte	Defines the start of the packet. Equal to 0xFF

[1]**Length of data field**	1 byte	Defines the length, *N*, of the data field contained in the packet.
Command	1 byte	Specifies the command that the reader is to execute.
Data	N bytes (0 to 250)	Defines the binary data required by the reader for use with a command. This could, for example, represent transponder data to be written. The length, *N*, can vary between 0 and 250 bytes.
CRC-16 Checksum (CRC HI, CRC LO)	2 bytes	CRC-16 checksum (high order byte first). CRC polynomial is CCITT CRC-16, with a preload of 0xFFFF. This does not fully specify the operation of the CRC; see CCITT CRC-16 Calculation.

1. Minimum packet length is 5 bytes; the maximum packet length is 255 bytes.

5.1.2. Reader-to-Host Communication

The following diagram defines the format of the generic Response Packet sent from the reader to the host. The Response Packet is different in format from the Request Packet.

The fields are summarized in the following table:

Field	Length	Description
Header	1 byte	Defines the start of the packet. Equal to 0xFF
[1]**Length of data field**	1 byte	Defines the length, M, of the data field contained in the packet. Length can be 0-248 bytes.
[2]**Command**	1 byte	Command code of the last command received
[3]**Status word**	2 bytes	Specifies the status of the last command, Successful = 0x0000, else it contains a fault code.
Data	M bytes (0 to 248)	Defines the binary data returned by the reader in response to a command. This could, for example, represent data read from a transponder. Data length, *M*, can be a minimum of 0 and a maximum of 248 bytes.
CRC-16 Checksum (CRC HI, CRC LO)	2 bytes	CRC-16 checksum (high order byte first). CRC polynomial is CCITT CRC-16, with a preload of 0xFFFF. This does not fully specify the operation of the CRC; see CCITT CRC-16 Calculation.

1. The minimum packet length is 7 bytes and the maximum packet length is 255 bytes.
2. Each host command receives a response from the reader. In the response packet, the Header, Data Length, Command, Data, and Checksum are functionally similar to the command packet.
3. The only difference is the addition of the Status Word field. The Status Word has two types of values. A Status Word value of 0 (Zero) means the command received was successful. Any other value represents a fault.

5.1.3. CCITT CRC-16 Calculation

The same CRC calculation is performed on all serial communications between the host and the reader. The CRC is calculated on the Data Length, Command, Status Word, and Data bytes. The header (SOH, 0xFF) is not included in the CRC.

A sample implementation of the CCITT CRC-16 algorithm is shown in this section. The CRC_calcCrc8() function is written to calculate the CRC one byte at a time, with the calculated value stored in crc_calc. The crc_calc value must be pre-loaded the first time the CRC_calcCrc8() function is called with 0xFFFF to initialize the calculated CRC. The final value of crc_calc is sent as the 16-bit CRC at the end of the message.

An example implementation of CRC calculation, taken from the Arbser source CrcUtils.c, is shown here:

```
/** @fn void CRC_calcCrc8(u16 *crcReg, u16 poly, u16 u8Data)
```

```
* @ Standard CRC calculation on an 8-bit piece of data. To make it
* CCITT-16, use poly=0x1021 and an initial crcReg=0xFFFF.
*
* Note: This function allows one to call it repeatedly to continue
* calculating a CRC. Thus, the first time it's called, it
* should have an initial crcReg of 0xFFFF, after which it
* can be called with its own result.
*
* @param *crcRegPointer to current CRC register.
* @param poly Polynomial to apply.
* @param u8Datau8 data to perform CRC on.
* @return None.
*/
void CRC_calcCrc8(u16 *crcReg, u16 poly, u16 u8Data)
{
u16 i;
u16 xorFlag;
u16 bit;
u16 dcdBitMask = 0x80;
for(i=0; i<8; i++)
{
// Get the carry bit. This determines if the polynomial should be
// xor'd with the CRC register.
xorFlag = *crcReg & 0x8000;
// Shift the bits over by one.
*crcReg <<= 1;
// Shift in the next bit in the data byte
bit = ((u8Data & dcdBitMask) == dcdBitMask);
*crcReg |= bit;
// XOR the polynomial
if(xorFlag)
{
*crcReg = *crcReg ^ poly;
}
// Shift over the dcd mask
dcdBitMask >>= 1;
}
}
```

5.2. Reply Messages (From Reader to Host)

There are three different types of replies that the Idesco EPC Anticollision reader can make to the host:

- Acknowledge that the command was properly processed (ACK)
- Return a fault code
- Provide data that is requested by the host

This section describes each of these three types.

Unless otherwise specified, all commands return, as part of the Reply message, a status word with an ACK or a Fault code. Those commands that return a Data Reply message are clearly shown.

5.2.1. ACK Message

Many of the commands require the reader to perform a function, but do not require the reader to send data back to the host. However, since the host cannot send a message until the reader replies, an ACK is sent.

The ACK message contains no data. It returns the same Command Code that was sent originally to the reader, sets the Status Word to 0x0000 (zero) and the Data Length to 0x00 (zero).

The following shows an example of an ACK message to a Set Read TX Power command.

FF 00 92 00 00 27 3B

Data	Length	Description
FF	1 byte	Header
00	1 byte	Length of data field
92	1 byte	Command code
00 00	2 bytes	Status
27 3B	2 bytes	CRC

The value in the Command Code field (0x92) is the same as the Set Read TX Power Command Code, 0x92.

5.2.2. Fault Reply Message

If a problem occurs during the execution of a command, the reader returns a non-zero status value. Although this usually implies a fault or error, sometimes the nonzero status simply indicates a condition of the system. For instance, when executing a Read Tag Single command, if no tags are found, a status code of 0x0400 is returned. An example of a fault reply message is shown for the Set Read TX Power command.

FF 00 92 01 06 26 3D

Data	Length	Description
FF	1 byte	Header
00	1 byte	Length of data field
92	1 byte	Command code
01 06	2 bytes	Status
26 3D	2 bytes	CRC

A list of error codes is included in Chapter 3.5: Error Messages. Refer to this list when encountering any non-zero status codes.

5.2.3. Data Reply Message

If the requested command requires that the reader returns data, then the reader creates a message similar to the ACK Message with the data length set to a non-zero value. Since this command does not require a data field, the length field is set to Zero.

FF 00 92 00 00 27 3B

Data	Length	Description
FF	1 byte	Header
00	1 byte	Length of data field
92	1 byte	Command code

00 00	2 bytes	Status
27 3B	2 bytes	CRC

Here is an example of a Reply message with Data Field Length not zero. This message happens to be a successful reply to Read Tag Single command.

FF 0A 21 00 00 C8 05 07 A8 00 84 C4 FF 9E E0 F7 25

Data	Length	Description
FF	1 byte	Header
0A	1 byte	Length of data field
21	1 byte	Command code
00 00	2 bytes	Status
C8 05 07 A8 00 84 C4 FF	8 bytes	Tag ID
9E E0	2 bytes	Tag CRC
F7 25	2 bytes	CRC

5.3. Application Tag Commands

The application tag commands are used to interact with RFID tags in the field. These commands can have slightly different behavior based upon the current protocol selected in the system.

Note! Application tag commands include 16bit timeout value in milliseconds to specify maximum duration of wanted operation. Some of application tag commands can last for whole specified timeout, but some commands can last even less if they were able to perform their action in shorter time. Application tag commands shouldn't be repeated more often than timeout value specifies unless reader wasn't able to perform its action in shorter time (in this case reader replies by message).

5.3.1. Tag Singulation / Select Functionality

Many of the EPCGen2 tag commands now support the ability to singulate a specific tag or inventory only tags matching a defined criteria, i.e. matching on values in the EPC, TID and User Memory banks.

The commands currently supporting tag singulation through Select are:

- Read Tag Single (21h)
- Read Tag Multiple (22h)
- Write Tag Data (24h)
- Lock Tag (25h)
- Kill Tag (26h)
- Read Tag Data (28h)

The addition of this functionality has added several (some conditional) fields to these commands.

The following fields have been added to all the specified commands. Please check each command for exact order, and any exceptions, as they may not all correspond with the order below.

Tag Singulation Fields

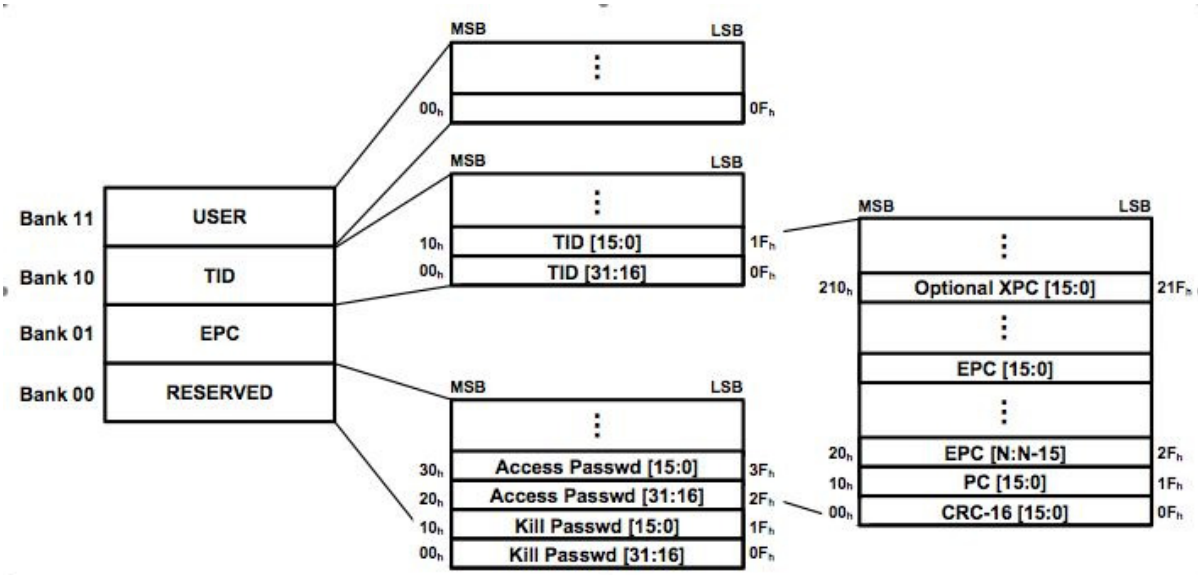
Field		Values	Description
Select options	[1]Select Contents (Bits 0,1, 2)	0x00	Select functionality is disabled. First tag found will be the tag operated on. No other Tag Singulation Fields should be specified. Option field must always be specified.

			Note: When Select is disabled commands do not support an access password. Use Select Option=0x05 to send a password without Select.
		0x01	Bit 3 of the Read Tag Multiple Search Flags must be set indicating this request contains embedded command(s).
		0x02	Select on the value of the EPC. Requires all fields except the *Select Address* field.
		0x03	Select on contents of User Memory memory bank (Gen2 bank 0x03). Requires all fields.
		0x04	Select on contents of the EPC memory bank (Gen2 bank 0x01). Requires all fields.
		0x05	Use this option when you need to specify an access password to operation on locked data but don't want to perform a Select. When this option is used do not pass any Select Criteria.
	[1]Select Invert (Bit 3)	0x08	Sets Invert Flag. This results in tags NOT matching the specified Tag Singulation Fields will be returned.
	[1]Extended Select Data (Bit 5)	0x20	Changes *Select Data Length* to 2 bytes, allowing *Select Data* to be greater than 255 bits.
The Select Options field is typically followed by command specific fields. After the command specific fields the following Tag Singulation fields should be specified as appropriate for the Select Contents specified.			
Select Address		4 bytes	Contains the offset, in bits, within the memory bank, specified by the *Option* value, at which the comparison is to start. **Note:** specifying *Option=0x04* and *Select Address=0x20* is the equivalent, for Gen2 v1 tags, of specifying *Option=0x01*, both specify a comparison against the tag EPC ID data. **Note:** Addresses are always zero-based. Specifying 0x00 indicates starting at the first address location.
Select Data Length		1 byte (*2 bytes if Extended Data enabled*)	Contains the length of the data (*Select Data*) to be compared, in bits, to the EPC when *Option*=0x01, or to the data beginning at *Select Address* for other options.
Select Data		M bytes	Contains the data to be compared against the specified tag data (memory bank and address, or EPC as specified by the *Option* value) **Note:** 1- The Select Options field contains multiple sub fields which
Note: 1- The Select Options field contains multiple sub fields which must be combined into a single Select Options value. This means the final Select Options value is a result of Select Invert + Select Contents.			

When performing a Tag Singulation/Select most of the criteria specify values of data in certain locations in a Gen2 tag's memory map. The following is a logical view of the Gen2 memory map from the Generation2 Protocol v1.2 that can be used for reference when trying to determine the memory address you are trying to match on:

Gen2 Tag Memory Map

Idesco Oy | Teknologiantie 9, FIN-90590 OULU | Tel +358 20 743 4175 | info@idesco.fi | www.idesco.fi



Note: The address values specified in the memory map are hexadecimal, zero-based bit offset within each memory bank (i.e. the PC section of the EPC memory bank starts at bit 0x10, decimal 16, and runs to bit 0x1F, decimal 31). It is important to note the units used in various command fields for address locations. In some cases the address is specified in words (16 bit chunks), sometimes bytes (8 bit chunks) and sometimes bits.

5.3.2. Read Tag Single (21h)

The Read Tag Single command will search for a tag for the specified timeout or until a single tag is found, whichever comes first. The search criterion is specified using the Tag Singulation Fields. If Option=0x00 is specified it will return data from the first tag it finds, otherwise it will only return Success and the found EPC if a tag matching the specified criterion is found. If no tag is read, a fault code is returned.

In addition to the Tag Singulation Fields the basic Read Tag Single command takes a 16-bit timeout value in milliseconds. The command will return after a tag is found or the timeout expires, whichever happens first.

The basic syntax which returns only the tag EPC data is defined in Get Tag EPC. With additional Option bits Read Tag Single can also return additional information (called here as Meta Data) using the syntax in Get Tag EPC and Meta Data.

5.3.2.1. Get Tag EPC

The following example shows a search requesting a tag matching the following criterion for a max timeout of 1000 ms. This example uses the Tag Singulation/Select Functionality with Option=0x03, indicating Tag Selection based on the contents of User Memory,
Specifically:
Memory Bank = User Memory.
Starting Address = bit 32
Select Data = 0x1234

FF 0A 21 03 E8 03 00 00 00 20 10 12 34 D7 DF

Data	Length	Description
FF	1 byte	Header

0A	1 byte	Length of data field
21	1 byte	Command code
03 E8	2 bytes	Timeout (ms)
03	1 byte	Option
00 00 00 20	4 bytes	Select address
10	1 byte	Select data length
12 34	2 bytes	Select data
D7 DF	2 bytes	CRC

If Option=0x00 or 0x01 were used then the unused Tag Singulation Fields must be removed from the request.

The response to this command varies depending upon the number of bits in the tag EPC data of the tag found. The general response format is shown here:

FF M+3 21 00 00 03 M bytes ?? ?? ?? ??

Data	Length	Description
FF	1 byte	Header
M+3	1 byte	Length of data field
21	1 byte	Command code
00 00	2 bytes	Status
03	1 byte	Option
M Bytes	M bytes	EPC data
?? ??	2 bytes	Tag CRC
?? ??	2 bytes	CRC

5.3.2.2. Get Tag EPC and Meta Data

In addition to getting the tag EPC data returned you can also get Tag Read Meta Data for the found tag. This version of Read Tag Single requires bit 4 of the Option flag to be set and takes an additional Meta Data Flags field which defines what Meta Data will be returned. The following table lists the supported values for these fields.

Read Tag Single – Get Tag EPC and Meta Data Request Fields

Field	Value	Description
Option	Bit 4=0 (0x0X)	No Meta Data flags are specified and Meta Data will not be returned. This is the Get Tag EPC syntax. The lower bits (X) are specified as defined by Tag Singulation/Select Functionality.
	Bit 4=1 (0x1X)	Indicates that Meta Data flags are to follow and the corresponding Meta Data shall be returned with the tag EPC. The lower bits (X) are specified as defined by Tag Singulation/Select Functionality.
Meta Data Flags (to specify more than one OR the values together)	0x0000	When no flags are set no Meta Data will be returned, only the tag EPC (including PC bits and tag CRC)
	0x0001	When bit 0 is set the Read Count will be returned
	0x0002	When bit 1 is set the LQI/RSSI will be returned
	0x0004	When bit 2 is set the Antenna ID will be returned
	0x0008	When bit 3 is set the Frequency will be returned
	0x0010	When bit 4 is set the Timestamp will be returned
	0x0020	When bit 5 is set the RFU will be returned
	0x0040	When bit 6 is set the Protocol ID will be returned.

	0x0080	When bit 7 is set Tag Data information will be returned. (0x0000 is always returned for Read Tag Single)
These fields are followed by the Tag Singulation/Select Functionality, used the same as defined in the Get Tag EPC syntax, if necessary.		

A response can contain the following information:

Read Tag Single Get EPC and Meta Data Response Fields

Field	Length	Value
Header	1 byte	0xFF
Length of data field	1 byte	Based on data returned
Command code	1 byte	0x21
Status	2 bytes	Status of command
Option	1 byte	As sent in request
Meta Data flags	2 bytes	As sent in request
Read Count$_1$	1 byte	Tag EPC/antenna read count
RSSI$_1$	1 byte	Return signal strength indicator
Antenna ID$_1$	1 byte	Antenna ID, 4 MSBs for TX and 4 LSBs for RX
Frequency$_1$	3 bytes	Frequency in kHz
Timestamp$_1$	4 bytes	RTC timestamp
RFU$_1$	2 bytes	Reserved for future use
Protocol ID	1 byte	Protocol ID of tag (always 0x05)
Tag Data Length	2 bytes	Size of tag data to follow. Always 0x0000 for read tag single
EPC ID	N bytes	Tag EPC
Tag CRC	2 bytes	Tag EPC CRC
CRC	2 bytes	Message CRC

1 - Conditionally returned depending on the Meta Data Fields specified in the request.

5.3.2.3. Examples

An example command requesting Antenna ID and Timestamp

Meta Data Flags ⇒ 0x0004 and 0x0010 = 0x0014

With no tag singulation criterion, just return the first tag found, is as follows:

FF 05 21 01 E8 10 00 14 2F 6D

Data	Length	Description
FF	1 byte	Header
05	1 byte	Length of data field
21	1 byte	Command code
01 E8	2 bytes	Timeout (ms)
10	1 byte	Option
00 14	2 bytes	Meta Data flags
2F 6D	2 bytes	CRC

Here is an example response to the example request specified above. The response contains the tag EPC info of the found tag and the requested tag read Meta Data (Antenna ID and Timestamp):

FF 16 21 00 00 10 00 14 11 00 BB 5F 04 01 23 45 67 89 AB CD EF 01 23 45 67 E6 C8 9D D1

Data	Length	Description
FF	1 byte	Header
16	1 byte	Length of data field
21	1 byte	Command code
00 00	2 bytes	Status
10	1 byte	Option
00 14	2 bytes	Meta Data flags
11	1 byte	Antenna ID
00 BB 5F 04	4 bytes	Time Stamp
01 23 45 67 89 AB CD EF 01 23 45 67	12 bytes	Tag EPC
E6 C8	2 bytes	Tag CRC
9D D1	2 bytes	CRC

Here is another example request and response showing the use of Tag Singulation/Select Functionality and getting the tag Meta Data for the specified tag. Note the Options Field includes the Select Options and bit 4 is set indicating the Meta Data flags follow. This command requests the same tag read Meta Data as the previous example (Antenna ID and Timestamp) except now it is selecting a tag with a specific EPC value (EPC=0x111122223333444455556666), which requires adding the appropriate Tag Singulation Fields after the Read Tag Single Get EPC and Meta Data Request Fields along with updating the Option field to set the appropriate flag for the tag singulation based on EPC value

FF 12 21 01 E8 11 00 14 60 11 11 22 22 33 33 44 44 55 55 66 66 9F CE

Data	Length	Description
FF	1 byte	Header
12	1 byte	Length of data field
21	1 byte	Command code
01 E8	2 bytes	Timeout
11	1 byte	Option
00 14	2 bytes	Meta Data flags
60	1 byte	Select data length
11 11 22 22 33 33 44 44 55 55 66 66	12 bytes	Select data (EPC)
9F CE	2 bytes	CRC

The response contains the requested Meta Data and the tag EPC matching the requested tag EPC:

FF 16 21 00 00 11 00 14 22 0F C8 CD B7 11 11 22 22 33 33 44 44 55 55 66 66 18 35 FE 7D

Data	Length	Description
FF	1 byte	Header
16	1 byte	Length of data field
21	1 byte	Command code
00 00	2 bytes	Status
11	1 byte	Option
00 14	2 bytes	Meta Data flags
22	1 byte	Antenna ID
0F C8 CD B7	4 bytes	Time Stamp
11 11 22 22 33 33 44 44 55 55 66 66	12 bytes	Tag EPC
18 35	2 bytes	Tag CRC

FE 7D	2 bytes	CRC

5.3.3. Read Tag Multiple (22h)

The Read Tag Multiple command supports several different levels of functionality. In addition to performing a Basic Search Operation for all tags in the field it can also perform advanced searching and perform operations on the tags found. The different syntax for Read Tag Multiple is defined as follows:

- Basic Tag Inventory - Searches for and returns all tags in the field.
- Tag Inventory with Select - Searches for and returns all tags in the field meeting Select criterion as defined by Tag Singulation Fields specified.
- Tag Inventory With Embedded Operations - Allows for operations (Write Tag Data, Lock Tag, Kill Tag, Read Tag Data) to be performed on each tag inventoried.

Note
A Read Tag Multiple command will return early if it fills up the Tag Buffer before the timeout has expired. This will not result in an error. If a Read Tag Multiple command is issued with an already full Tag Buffer an error will be returned.

5.3.3.1. Basic Tag Inventory

The Read Tag Multiple command performs a search for the specified period of time then returns the number of tags that have been found. Afterwards, multiple Get Tag Buffer commands can be sent to receive the found tag EPCs along with tag read Meta Data, including the antenna the tag was read on. The command allows the user to specify the method to use when multiple antennas are configured and connected along with indicating the command contains embedded commands:

Read Tag Multiple Search Flags

[2]**Flag Value**		**Description**
Antenna usage (bits 0, 1, 2)	0x0000	Use single antenna as configured by the most recent Set Antenna command.
	[1]0x0001	Automatically search on both antennas (internal and external), starting with Antenna 1 (external). The search cycles through antennas moving to the next antenna when no more tags are found on the current antenna. It stops when the search timeout expires.
	[1]0x0002	Automatically search on both antennas (internal and external), starting with Antenna 2 (internal). The search cycles through antennas moving to the next antenna when no more tags are found on the current antenna. It stops when the search timeout expires.
	[1]0x0003	Automatically searches on all configured logical antennas, using the Search Order defined using Set Multi-Antenna Search Configuration. The search cycles through antennas, in the order specified, moving to the next antenna when no more tags are found on the current antenna. It stops when the search timeout expires.
Embedded command (bit3)	0x0004	An embedded command is specified in the request and will be executed on each inventoried tag. Note: This bit should only be set when using the Tag Inventory With Embedded Operations syntax.
Note: 1- Only one of these flags should be set since both Antenna 1 and 2 cannot be the starting antenna. 2 -Multiple Flags can be set (perform a binary OR) to specify different behaviors. Example: Search Flags = 0x0006 indicates a multiple antenna search starting on antenna2 is to be performed and the embedded command specified is to be executed on each tag.		

5.3.3.2. Examples

For example, the syntax for a Read Tag Multiple with automatic multi-antenna search starting with antenna 1 (external antenna) is:

FF 04 22 00 01 03 E8 3F 8E

Data	Length	Description
FF	1 byte	Header
04	1 byte	Length of data field
22	1 byte	Command code
00 01	2 bytes	Antenna flag
03 E8	2 bytes	Timeout (ms)
3F 8E	2 bytes	CRC

The response format for both is the following:

FF 01 22 00 00 02 46 BA

Data	Length	Description
FF	1 byte	Header
01	1 byte	Length of data field
22	1 byte	Command code
00 00	2 bytes	Status
02	1 byte	# Tag IDs found
46 BA	2 bytes	CRC

5.3.3.3. Tag Inventory with Select

If you want to inventory only tags meeting a specific criteria this syntax should be used. The search criterion is specified using the Tag Singulation Fields. If Option=0x00 is specified it will perform the same search as the Basic Tag Inventory syntax. Otherwise, it will return the number of tags found matching the specified criteria. The tag EPCs and Meta Data will be available in the Tag Buffer. If no tags are found, a fault code is returned. The required fields are as follows:

Read Tag Multiple with Select Fields

Field	Length	Description
Select options	1 byte	The Options value of the Tag Singulation Fields
Search flags	2 bytes	Read Tag Multiple Search Flags indicating antenna usage. Bit 3 must be 0, no embedded commands.
Timeout	2 bytes	Indicates how long the command should spend searching.
Access password	4 bytes	The Access Password is only used with Tag Inventory With Embedded Operations for the embedded command. With this syntax it should be specified as 0x00000000. **Note:** If Select Options=0x00 this field should be omitted.
Tag singulation fields		The remaining, appropriate fields depending on the value of Select Options.

5.3.3.4. Examples

Here is an example request and response showing the use of Tag Singulation/Select Functionality to inventory tags which meet a specific criterion: This command will inventory all tags with an EPC val-

ue ending in 0x66, which requires adding the appropriate Tag Singulation Fields to the Basic Tag Inventory syntax.

FF 0F 22 04 00 00 03 E8 00 00 00 00 00 00 00 78 08 66 DE C0

Data	Length	Description
FF	1 byte	Header
0F	1 byte	Length of data field
22	1 byte	Command code
04	1 byte	Options (EPC memory)
00 00	2 bytes	Search flags
03 E8	2 bytes	Timeout (ms)
00 00 00 00	4 bytes	Access password
00 00 00 78	4 bytes	Select data address (bits)
08	1 byte	Select data length (bits)
66	1 byte	Select data
DE C0	2 bytes	CRC

Note:
The Select Options field of the Tag Singulation Fields in the request is specified at the beginning of the command followed by the Search Flags, Timeout, Access Password then the rest of the Tag Singulation Fields. This is different than the typical format for Select fields. Also, this syntax always requires an Access Password be specified. Since only Reserved Memory can be read locked and Reserved Memory cannot be used for singulation the Access Password must be 0x00000000.

The response contains the number of tags found matching the Select criteria specified. Use Get Tag Buffer (29h) to access the tag EPCs and Tag Read Meta Data:

FF 04 22 00 00 04 00 00 02 B7 6E

Data	Length	Description
FF	1 byte	Header
04	1 byte	Length of data field
22	1 byte	Command code
00 00	2 bytes	Status
04	1 byte	Options (EPC memory)
00 00	2 bytes	Search flags
02	1 byte	# Tag IDs found
B7 6E	2 bytes	CRC

5.3.3.5. Tag Inventory with Embedded Operations

In addition to inventorying tags, Read Tag Multiple can be used to perform an operation on each tag in a population of tags. Starting with the Tag Inventory with Select syntax to define the population of tags the operation is to be performed on, the Search Flag bit 3 (0x0004) can be set to indicate embedded commands are to be performed on the inventoried commands. The required fields are as follows:

Read Tag Multiple Embedded Command Fields

Field	Length	Description
Select options	1 byte	The Options value of the Tag Singulation Fields
Search flags	2 bytes	Bit 3 of the Read Tag Multiple Search Flags must be set in-

		dicating this request contains embedded command(s).
Timeout	2 bytes	Indicates how long the command should spend searching AND performing the embedded command. It may be desirable to specify a longer timeout if a large number of tags are likely to get the embedded command executed on them.
Access password	4 bytes	The Access Password of the tags expected to be inventoried for the embedded command, if they are locked. If the tags are not locked specify 0x00000000. **Note:** If operating on locked tags, only tags which meet the Select criteria and matching passwords will get a successful execution of the embedded command. **Note:** If Select Options=0x00 this field should be omitted.
Tag singulation fields		The remaining, appropriate fields depending on the value of Select Options.
Embedded command count	1 byte	The number of embedded commands to follow. (Currently only supports one)
Embedded command length	1 byte	Length of embedded commands. Follows standard Length value calculation: number of bytes after Command code.
Embedded command code	1 byte	The Command code of the embedded command. Currently supports: • Write Tag Data (24h) • Lock Tag (25h) • Kill Tag (26h) • Read Tag Data (28h) **Note:** When embedding Read Tag Data the complete set of data requested is returned for the first tag that responds with the command response, but the first 4 bytes of data requested is returned for every tag that responds and is stored in the tag buffer with the other metadata. See Example with Embedded Read Tag Data below for details.
The fields and values required by the embedded command.		**Note:** The embedded commands do not support Tag Singulation as it is already performed during the inventory operation. The Options field for the embedded command must be 0x00. **Note:** The Timeout field for embedded commands must be 0x0000.

Read Tag Multiple Embedded Response Fields

Field	Length	Description
Status	2 bytes	Error Code if command failed, otherwise 0x0000 for Success
Select options	1 byte	Options set in the Request Command
Search flags	2 bytes	Search Flags set in the Request Command
# Tags found	1 byte	Number of tags found and added to the Tag Buffer matching Select criterion. Follows the standard criteria of adding tags the Tag Buffer. If the Tag Buffer already has tags in it they will not be counted towards Tags Found. However, if they match the Select Criteria they will have the operation performed on them. For this reason it is important to always make sure the Tag Buffer is clear before any Read Tag Multiple execution to insure accurate response information.
Embedded command count	1 byte	The number of embedded commands to follow. (Currently only supports one)
Embedded command code	1 byte	The Command code of the embedded command as specified in the request command.

Operations succeeded	2 bytes	Number of Embedded command operations which succeeded. **Note:** Depending on the Gen2 Session/User Mode used the Operations Succeeded/Failed counts can be misleading since in Session 0, for example, the tag may respond many times during an inventory round and the command may be attempted many times. This would result in counts higher than the actual number of tags the operation succeeded or failed on.
Operations failed	2 bytes	Number of Embedded command operations which failed. **Note:** As noted above this number can be indicating the command failed multiple times on the same tag. These values should be used in combination with Tag Found and checking the Tag Buffer to insure the operation was completed on the desired tags.
The fields and values returned by the embedded command.		

5.3.3.6. Example with Embedded Write Tag Data

Here is an example request and response showing the use of Tag Singulation/Select Functionality to inventory tags which meet a specific criterion and then setting the Access password on each using Write Tag Data (24h) as an embedded command:

This command will inventory all tags with an EPC value ending in 0x34, which requires adding the appropriate Tag Singulation Fields to the Basic Tag Inventory syntax then will use Write Tag Data (24h) to write 0x12345678 into the Reserved Memory Bank starting at Word address 0x00000002 (Access Password)

FF 1E 22 04 00 04 03 E8 00 00 00 00 00 00 00 78 08 34 01 0C 24 00 00 00 00 00 00 02 00 12 34 56 78 AF 29

Data	Length	Description
FF	1 byte	Header
1E	1 byte	Length of data field
22	1 byte	Command code
04	1 byte	Options (EPC memory)
00 04	2 bytes	Search flags
03 E8	2 bytes	Timeout (ms)
00 00 00 00	4 bytes	Access password
00 00 00 78	4 bytes	Select data address (bits)
08	1 byte	Select data length (bits)
34	1 byte	Select data
01	1 byte	Embedded command count
0C	1 byte	Embedded command length
24	1 byte	Embedded command code
00 00	2 bytes	Embedded command timeout (not used)
00	1 byte	Embedded command options (must be 0x00)
00 00 00 02	4 bytes	Write address (words)
00	1 byte	Write memory bank
12 34 56 78	4 bytes	Write data
AF 29	2 bytes	CRC

The response contains the number of tags found matching the Select criteria specified and the number of embedded command operations which succeeded and failed. Use Get Tag Buffer (29h) to access the tag EPCs and Tag Read Meta Data for the Tags Found. Tags in the buffer may or may not have had successful execution of the embedded command on them:

FF 0A 22 00 00 04 00 04 02 01 24 00 02 00 00 FF 5E

Data	Length	Description
FF	1 byte	Header
0A	1 byte	Length of data field
22	1 byte	Command code
00 00	2 bytes	Status
04	1 byte	Option
00 04	2 bytes	Search flags
02	1 byte	# Tags found
01	1 byte	Embedded command count
24	1 byte	Embedded command code
00 02	2 bytes	# Operations succeeded
00 00	2 bytes	# Operations failed
FF 5E	2 bytes	CRC

Note:
Depending on the Gen2 Session/User Mode used the Operations Succeeded/Failed counts can be misleading since in Session 0, for example, the tag may respond many times during an inventory round and the command may be attempted many times. This would result in counts higher than the actual number of tags the operation succeeded or failed on. The above commands were run in User Mode = Portal.

Note:
When embedding write operations, including Lock and Kill, in Read Tag Multiple, the Read TX Power will be used for the entire operations: inventory and the write. The power will not switch to the Write TX Power for each Write operation.

5.3.3.7. Example with Embedded Read Tag Data

When using Read Tag Multiple with an embedded Read Tag Data command an extra field containing the requested Data from the first tag which responded matching the Select criteria is added to the response. Up to 4 bytes of tag data for the other tags responding, if any, is available as Tag Read Meta Data. This command provides the added benefit over simply using just a Read Tag Data with Select of being able to identify other tags matching the criteria which still may need to be written or which you didn't expect to match. An example of sending an embedded Read Tag Data and its response is as follows:

This command will inventory all tags with an EPC value ending in 0x34, which requires adding the appropriate Tag Singulation Fields to the Basic Tag Inventory syntax then will use Read Tag Data (28h) to verify that 0x12345678 was written into the Reserved Memory Bank starting at Word address 0x00000002 (Access Password):

FF 1B 22 04 00 04 03 E8 00 00 00 00 00 00 00 78 08 34 01 09 28 03 E8 00 00 00 00 00 02 8C 7F

Data	Length	Description
FF	1 byte	Header
1B	1 byte	Length of data field
22	1 byte	Command code
04	1 byte	Options (EPC memory)
00 04	2 bytes	Search flags
03 E8	2 bytes	Timeout (ms)
00 00 00 00	4 bytes	Access password

00 00 00 78	4 bytes	Select data address (bits)
08	1 byte	Select data length (bits)
34	1 byte	Select data
01	1 byte	Embedded command count
09	1 byte	Embedded command length
28	1 byte	Embedded command code
03 E8	2 bytes	Embedded command timeout (not used)
00	1 byte	Embedded command options (must be 0x00)
00	1 byte	Read memory bank (0x00=Reserved Membank)
00 00 00 02	4 bytes	Read address (words)
02	1 byte	Read word count
8C 7F	2 bytes	CRC

The response contains the number of tags found matching the Select criteria specified and the number of embedded command operations which succeeded and failed. Use Get Tag Buffer (29h) to access the tag EPCs and Tag Read Meta Data for the Tags Found. Tags in the buffer may or may not have had successful execution of the embedded command on them:

FF 0E 22 00 00 04 00 04 03 01 28 00 01 00 00 11 22 33 44 DE 02

Data	Length	Description
FF	1 byte	Header
0E	1 byte	Length of data field
22	1 byte	Command code
00 00	2 bytes	Status
04	1 byte	Option
00 04	2 bytes	Search flags
03	1 byte	# Tags found
01	1 byte	Embedded command count
28	1 byte	Embedded command code
00 01	2 bytes	# Operations succeeded
00 00	2 bytes	# Operations failed
11 22 33 44	4 bytes	Data read
DE 02	2 bytes	CRC

5.3.3.8. Example with Embedded Kill Tag

This command will inventory all tags with an EPC value with the first 88bits equal to 0x30940425C4C1967400004E, which requires adding the appropriate Tag Singulation Fields to the Basic Tag Inventory syntax then will use Kill Tag (26h) to kill all matching tags, assuming they have their Kill password set to 0x12345678:

FF 20 22 01 00 04 00 FA 87 65 43 21 58 30 94 04 25 C4 C1 96 74 00 00 4E 01 08 26 00 00 00 12 34 56 78 00 5F C6

Data	Length	Description
FF	1 byte	Header
20	1 byte	Length of data field
22	1 byte	Command code
01	1 byte	Options (EPC memory)
00 04	2 bytes	Search flags
00 FA	2 bytes	Timeout (ms)
87 65 43 21	4 bytes	Access password

58	1 byte	Select data length (bits)
30 94 04 25 C4 C1 96 74 00 00 4E	11 bytes	Select data
01	1 byte	Embedded command count
08	1 byte	Embedded command length
26	1 byte	Embedded command code
00 00	2 bytes	Embedded command timeout (not used)
00	1 byte	Embedded command options (must be 0x00)
12 34 56 78	4 bytes	Kill Password
00	1 byte	RFU
5F C6	2 bytes	CRC

Note:
If the tags being inventoried and operated on with the embedded command have their Access Password set then the Access Password field must be set accordingly, otherwise it can be left as zero.

The response contains the number of tags found matching the Select criteria specified and the number of embedded command operations which succeeded and failed. Use Get Tag Buffer (29h) to access the tag EPCs for the Tags Found. Tags in the buffer may or may not have had successful execution of the embedded command on them:

FF 0A 22 00 00 01 00 04 01 01 26 00 02 00 00 37 52

Data	Length	Description
FF	1 byte	Header
0A	1 byte	Length of data field
22	1 byte	Command code
00 00	2 bytes	Status
01	1 byte	Option
00 04	2 bytes	Search flags
01	1 byte	# Tags found
01	1 byte	Embedded command count
26	1 byte	Embedded command code
00 02	2 bytes	# Operations succeeded
00 00	2 bytes	# Operations failed
37 52	2 bytes	CRC

Note:
In this example it shows the number of Tags Found lower than the Operations Succeeded. This is because, as with any Read Tag Multiple execution, if the Tag Buffer already has a specific tag in it, it will not be counted towards Tags Found. However, because it matches the Select Criteria it will have the operation performed on it. For this reason it is important to always make sure the Tag Buffer is clear before any Read Tag Multiple execution.

5.3.4. Write Tag EPC (23h)

The Write Tag EPC command should be used when updating the EPC value of a tag. It is preferred over using Write Tag Data (24h) because Write Tag EPC will automatically lengthen or shorten the EPC ID, by modifying the PC bits, according to the Tag EPC specified. If Write Tag Data is used, the specified data will be modified but the EPC ID length will not be modified.

The Write Tag EPC command takes the following data elements:

FF M+4 23 03 E8 00 00 M bytes ?? ??

Data	Length	Description
FF	1 byte	Header
M+4	1 byte	Length of data field
23	1 byte	Command code
03 E8	2 bytes	[1]Timeout (ms)
00	1 byte	[2]RFU
00	1 byte	[3]RFU
M Bytes	M bytes	[4]Tag EPC data (M bytes)
?? ??	2 bytes	CRC

1. 16-bit timeout value in milliseconds. Due to tag difference some tags may require more time to write than others. Experimentation may be required to determine the optimal timeout.
2. Reserved for Future Use, this field is required but ignored.
3. Reserved for Future Use, this field is required but ignored
4. Up to 496-bit (Depending on EPC Length parameter setting) tag ID to write to the Tag.

The reader sends a Fault Code / ACK response back to the host.

An example of Write Tag EPC command sequence of events and format is shown next:
1. Starts a timer on the reader.
2. Wakes the tag.
3. Programs the tag
4. Reads the tag and verifies if the write succeeded
   Note: The verify operation uses the same power level as the write operation. It does not change to the Read power level.
5. Sends back an ACK if OK or a fault code for timeout or other faults.

FF 0C 23 03 E8 00 00 11 22 33 44 55 66 77 88 5D D8

Data	Length	Description
FF	1 byte	Header
0C	1 byte	Length of data field
23	1 byte	Command code
03 E8	2 bytes	Timeout (ms)
00	1 byte	RFU
00	1 byte	RFU
11 22 33 44 55 66 77 88	8 bytes	Tag EPC data
5D D8	1 byte	CRC

Bytes 6 and 7, identified as Reserved for Future Use (RFU) are ignored, although still required in the command. EPC values up to 496-bits are supported, depending on EPC Length parameter setting.

5.3.5. Write Tag Data (24h)

The Write Tag Data command writes to the specified memory bank and data address location within that memory bank of a tag. The tag which will be written to can be specified using the Tag Singulation Fields or, if Option=0x00 of the Tag Singulation Fields is specified, it will attempt to write to the first tag it finds. If no tag is in the field, the memory location doesn't exist or is un-writable, or the Select criteria cannot be satisfied a fault code is returned.

In addition to the Tag Singulation Fields the Write Tag Data command takes several fields which specify the data which will be written to the tag. These fields are:

Write Tag Data Fields

Field	Length	Description
Write address	4 bytes	The Address field is the offset in the specified Memory Bank, in 16-bit words, where the content of the Data field is written. It corresponds to the *WordPtr* argument in the Gen2 specification. **Note:** Addresses are always zero-based. Specifying 0x00 indicates starting at the first address location.
Write membank	1 byte	The MemBank field specifies which of the tag's memory banks the data is to be written to. The values correspond to the Memory Bank values as specified in the *Gen2* specification. They are: 0x00 = Reserved 0x01 = EPC 0x02= TID 0x03 = User Memory
Write data	N bytes	The data to be written to the tag in Memory bank [MemBank] starting at address [Address].
Access password	4 bytes	The Access Password for the tag, if the tag is locked. For an unlocked tag AccessPwd=0x00000000. **Note:** When Option=0x00 is specified the Access Password is not used.

5.3.5.1. Examples

The following example will attempt to write to Reserved Memory to set the Kill password=0x11112222. It will write this data to a tag matching the following criteria for a max timeout of 1000 ms.

Memory Bank = User Memory.
Starting Address = bit 32
Select Data = 0x1234
The Reserved Memory bank is not locked so the Access Password is zero

FF 17 24 03 E8 03 00 00 00 00 00 00 00 00 00 00 00 20 10 12 34 11 11 22 22 3E 71

Data	Length	Description
FF	1 byte	Header
17	1 byte	Length of data field
24	1 byte	Command code
03 E8	2 bytes	Timeout (ms)
03	1 byte	Option
00 00 00 00	4 bytes	Write address
00	1 byte	Write membank (0x00=Reserved Membank)
00 00 00 00	4 bytes	Access password
00 00 00 20	4 bytes	Select address
10	1 byte	Select data length
12 34	2 bytes	Select data
11 11 22 22	4 bytes	Write data
3E 71	2 bytes	CRC

If Option=0x00 or 0x01 is used then the unused Tag Singulation Fields must be removed from the request.

Note: If Option=00, the write tag data is performed without any select criteria resulting the first tag found being written. In that case there is no way to determine what tag gets written to unless there is only one tag in the RFfield.

The next example will attempt to write 4 words to User Memory starting at the second word. It will write this data to a tag matching the following criteria for a max timeout of 1000 ms.

EPC ID= 0x0123456789ABCDEF01234567

The User Memory bank is not locked so the Access Password is zero

FF 21 24 03 E8 01 00 00 00 02 03 00 00 00 00 60 01 23 45 67 89 AB CD EF 01 23 45 67 11 11 22 22 00 00 00 00 27 03

Data	Length	Description
FF	1 byte	Header
21	1 byte	Length of data field
24	1 byte	Command code
03 E8	2 bytes	Timeout (ms)
01	1 byte	Option
00 00 00 02	4 bytes	Write address
03	1 byte	Write membank
00 00 00 00	4 bytes	Access password
60	1 byte	Select data length
01 23 45 67 89 AB CD EF 01 23 45 67	12 bytes	Select data
11 11 22 22 00 00 00 00	8 bytes	Write data
27 03	2 bytes	CRC

Note: Try reading back the value written in this example with the Read Tag Data Examples showing reading from the same tag.

5.3.6. Lock Tag (25h)

The Lock Tag command locks the specified memory bank of a tag. The tag which will be locked can be specified using the Tag Singulation Fields or, if Option=0x00 of the Tag Singulation Fields is specified, it will attempt to lock the first tag it finds. If no tag is in the field, the memory location doesn't exist or is unlockable, or the Select criteria cannot be satisfied a fault code is returned.

In addition to the Tag Singulation Fields the Lock Tag command takes several fields which specify how the tag is to be locked. These fields are:

Lock Tag Fields

Field	Length	Description
AccessPwd	4 bytes	The Access Password for the tag. In order to lock a tag the Access Password must be non-zero. To set the access password using the Write Tag Data (24h) command.
Mask $Bits_1$	2 bytes	The Mask bits specify which fields should be modified ac-

		cording to the Actions bits. When a Mask Bit =0 the corresponding Action bit is not applied and the current lock setting is retained. When a Mask Bit =1 the corresponding Action bit is applied and the new lock setting is implemented.
Action Bits[1]	2 bytes	The Action bits specify whether to assert or deassert a lock behavior for the associated memory location. Action Bits are only applied if the corresponding Mask Bits =1.

1-The Mask and Action bits correspond to the identically named fields described in Section 6.3.2.10.3.5 of the Gen2 specification.

The values of the Mask and Action bits indicate how a tag is to be locked. The 10 Least Significant Bits of each 16-bit argument are used to indicate the lock behavior for each memory bank (Action Bits) and which of those behaviors to apply (Mask Bits). These bits and their corresponding behaviors are:

	First Byte								Second Byte							
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Unused						Kill Pwd		Access Pwd		EPC Mem		TID Mem		User Mem	
Mask	x	x	x	x	x	x	Set?	Set?	Set?	Set?	Set?	Set?	Set?	Set?	Set?	Set?
Action	x	x	x	x	x	x	R/W	Perm	R/W	Perm	W	Perm	W	Perm	W	Perm

For each bit in the Mask field where Set?=1 the corresponding Action bit will be applied and the specified lock setting (R/W, W. Permanent) will be asserted (1) or de-asserted (0). Please see the Gen2 specification for more information on Lock Action functionality.

Note: Operations to lock/unlock memory banks can be combined by passing Mask and Action fields with multiple changes, but each bank must exist and be lockable independently. For example, passing 0xFFFF in the Mask field and 0x0000 in the **Action** field unlocks all the memory banks as long as all memory banks exist and are lockable.

Example: The following example shows an attempt to apply a temporary (not permanent) Write lock on the EPC memory (Option=0x01) of a tag whose EPC ID=0x111122223333444455556666 and whose access password=0x11223344:

FF 18 25 03 E8 01 11 22 33 44 00 20 00 20 60 11 11 22 22 33 33 44 44 55 55 66 66 9E 7A

Data	Length	Description
FF	1 byte	Header
18	1 byte	Length of data field
25	1 byte	Command code
03 E8	2 bytes	Timeout (ms)
01	1 byte	Option
11 22 33 44	4 bytes	Access Password
00 20	2 bytes	Mask Bits
00 20	2 bytes	Action Bits
60	1 byte	Select data length
11 11 22 22 33 33 44 44 55 55 66 66	12 bytes	Select EPC ID
9E 7A	2 bytes	CRC

5.3.7. Kill Tag (26h)

The Kill Tag command kills the specified tag. The tag which will be killed can be specified using the Tag Singulation Fields or, if Option=0x00 of the Tag Singulation Fields is specified, it will attempt to

kill the first tag it finds. If no tag is in the field, the kill password is zero, or the Select criteria cannot be satisfied a fault code is returned.

In addition to the Tag Singulation Fields the Kill command takes the tag's Kill Password and an extra 1 byte field for future use (RFU).

Example: The following example shows an attempt to kill a tag whose EPC ID=0x112233445566778899AA and whose Kill password=0x11112222:

FF 13 26 03 E8 01 11 11 22 22 00 50 11 22 33 44 55 66 77 88 99 AA B9 69

Data	Length	Description
FF	1 byte	Header
13	1 byte	Length of data field
26	1 byte	Command code
03 E8	2 bytes	Timeout (ms)
01	1 byte	Option
11 11 22 22	4 bytes	Kill Password
00	1 byte	Reserved for future use
50	1 byte	Select data length
11 22 33 44 55 66 77 88 99 AA	10 bytes	Select EPC ID
B9 69	2 bytes	CRC

Note: If the tag's kill password is set to 0, the protocol does not allow the tag to be killed. A non-zero kill password must be set (using the Write Tag Data command) before the kill command succeeds. The RFU field should be set to 0.

5.3.8. Read Tag Data (28h)

The Read Tag Data command reads the specified memory bank at data address location within that memory bank of a tag. The tag which will be read can be specified using the Tag Singulation Fields or, if Option=0x00 of the Tag Singulation Fields is specified, it will attempt to read from the first tag it finds. If no tag is in the field, the memory location doesn't exist or is read locked, or the Select criteria cannot be satisfied a fault code is returned.

In addition to the Tag Singulation Fields the Read Tag Data command takes several fields which specify the data which will be read from the tag. These fields are:

Read Tag Data Fields

Field	Length	Description
Read membank	1 byte	The MemBank field specifies which of the tag's memory banks the data is to be read from. The values correspond to the Memory Bank values as specified in the *Gen2* specification. They are: 0x00 = Reserved 0x01 = EPC 0x02= TID 0x03 = User Memory
Read address	4 bytes	The Address field is the offset in the specified Memory Bank, in 16-bit words, to start reading from. It corresponds to the *WordPtr* argument in the Gen2 specification. **Note:** Addresses are always zero-based. Specifying 0x00 indicates starting at the first address location.

Word count	1 byte	The number of words (16 bit chunks) of data to read from memory bank [MemBank] starting at offset [ReadAddress].
Access password	4 bytes	The Access Password for the tag, if the tag is read locked. For an unlocked tag AccessPwd=0x00000000.

The Basic syntax which returns only the requested Tag Data is defined in Get Tag Data. With additional Option bits set, Read Tag Data can also return Tag Read Meta Data using the syntax in Get Tag Data and Meta Data.

5.3.8.1. Get Tag Data

The following example will attempt to read the Kill Password (the first 2 words) from Reserved Memory. It will read this data from a tag matching the following criteria for a max timeout of 1000 ms.

Memory Bank = User Memory.
Starting Address = bit 32
Select Data = 0x1234
The Reserved Memory bank is not locked so the Access Password is zero.

FF 14 28 03 E8 03 00 00 00 00 00 02 00 00 00 00 00 00 00 20 10 12 34 D1 E7

Data	Length	Description
FF	1 byte	Header
14	1 byte	Length of data field
28	1 byte	Command code
03 E8	2 bytes	Timeout (ms)
03	1 byte	Option
00	1 byte	Read membank
00 00 00 00	4 bytes	Read address
02	1 byte	Word count
00 00 00 00	4 bytes	Access password
00 00 00 20	4 bytes	Select address
10	1 byte	Select data length
12 34	2 bytes	Select data
D1 E7	2 bytes	CRC

If Option=0x00 or 0x01 is used then the unused Tag Singulation Fields and the Access Password (for Option=0x00 only) must be removed from the request.

The response to this Read Data command example is:

FF 05 28 00 00 03 11 11 22 22 10 BF

Data	Length	Description
FF	1 byte	Header
05	1 byte	Length of data field
28	1 byte	Command code
00 00	2 bytes	Status
03	1 byte	Option
11 11 22 22	4 bytes	Data
10 BF	2 bytes	CRC

The next example will attempt to read four words from User Memory starting at the third word (0x0002). It will read this data from a tag matching the following criteria for a max timeout of 1000 ms. Note the Options field, no Meta Data is requested unlike the previous example.

EPC ID = 0x0123456789ABCDEF01234567.

The User Memory bank is not locked so the Access Password is zero

FF 1A 28 03 E8 01 03 00 00 00 02 04 00 00 00 00 60 01 23 45 67 89 AB CD EF 01 23 45 67 7A C1

Data	Length	Description
FF	1 byte	Header
1A	1 byte	Length of data field
28	1 byte	Command code
03 E8	2 bytes	Timeout (ms)
01	1 byte	Option
03	1 byte	Read membank
00 00 00 02	4 bytes	Read address
04	1 byte	Word count
00 00 00 00	4 bytes	Access password
60	1 byte	Select data length
01 23 45 67 89 AB CD EF 01 23 45 67	12 bytes	Select data
7A C1	2 bytes	CRC

The response to this Read Data command example is:

FF 09 28 00 00 01 AA BB CC DD 00 00 00 00 E7 54

Data	Length	Description
FF	1 byte	Header
09	1 byte	Length of data field
28	1 byte	Command code
00 00	2 bytes	Status
01	1 byte	Option
AA BB CC DD 00 00 00 00	8 bytes	Data read
E7 54	2 bytes	CRC

Note: Try changing the value at this memory location with the Write Tag Data Examples showing writing to the same tag.

5.3.8.2. Get Tag Data and Meta Data

In addition to getting the tag data returned you can also get Tag Read Meta Data for the found tag. This version of Read Tag Data requires bit 4 of the Option flag to be set and takes an additional Meta Data Flags field which defines what Meta Data will be returned.

The following table lists the supported values for these fields.

Read Tag Data – Get Data and Meta Data Request Fields

Field	Value	Description
Option	Bit 4=0 (0x0X)	No Meta Data flags are specified and Meta Data will not be returned. This is the Get Tag Data syntax. The lower bits (X)

		are specified as defined by Tag Singulation/Select Functionality.
	Bit 4=1 (0x1X)	Indicates that Meta Data flags are to follow and the corresponding Meta Data shall be returned with the tag data. The lower bits (X) are specified as defined by Tag Singulation/Select Functionality.
Metadata Flags (to specify more than one OR the values together)	0x0000	When no flags are set no Meta Data will be returned, only the tag data.
	0x0001	When bit 0 is set the Read Count will be returned
	0x0002	When bit 1 is set the LQI/RSSI will be returned
	0x0004	When bit 2 is set the Antenna ID will be returned
	0x0008	When bit 3 is set the Frequency will be returned
	0x0010	When bit 4 is set the Timestamp will be returned
	0x0020	When bit 5 is set the RFU will be returned
	0x0040	When bit 6 is set the Protocol ID will be returned.
	0x0080	When bit 7 is set Tag Data information will be returned. Tag Data is always returned for Read Tag Data, this field cause an extra 2 bytes, always 0x0000, to be returned for Tag Data Length
These fields are followed by the Read Tag Data Fields then the Tag Singulation/Select Functionality (set appropriate bits in the Option field defined above, do not specified an additional Option field), used the same as defined in the Get Tag Data syntax, as necessary.		

A response can contain the following information:

Read Tag Data – Get Data and Meta Data Response Fields

Field	Length	Value
SOH	1 byte	0xFF
Length of data field	1 byte	Based on data returned
Command code	1 byte	0x28
Status	2 bytes	0x0000 for success. Otherwise see Chapter 3.5: Error Messages
Options	1 byte	As sent in request
Meta Data flags	2 bytes	As sent in request
Read Count$_1$	1 byte	Tag EPC/Antenna Read Count
RSSI$_1$	1 byte	Return Signal Strength Indicator
Antenna ID$_1$	1 byte	Antenna ID, 4 MSBs for TX and 4 LSBs for RX
Frequency$_1$	3 bytes	Frequency in kHz
Timestamp$_1$	4 bytes	RTC Timestamp
RFU$_1$	2 bytes	Reserved for Future Use
Protocol ID$_1$	1 byte	Protocol ID of tag (always 0x05)
Tag data length$_1$	2 bytes	N/A - always zero for Read Tag Data.
Tag data	N bytes	N/A - always zero for Read Tag Data.
CRC	2 bytes	Message CRC

1 - Conditionally returned depending on the Meta Data Fields specified in the request.

The following example will attempt to read the Access Password (the last 2 words) from Reserved Memory. It will read this data from a tag matching the following criteria for a max timeout of 1000 ms.

Memory Bank = EPC

Starting Address = bit 120 (beginning of the last byte of the EPC value: 16 PC bits + 16 CRC bits + 88 EPC bits)

Select Data = 0x34

The Reserved Memory bank is not locked so the Access Password is zero

FF 15 28 03 E8 14 00 14 00 00 00 00 02 02 00 00 00 00 00 00 00 78 08 34 9C 0E

Data	Length	Description
FF	1 byte	Header
15	1 byte	Length of data field
28	1 byte	Command code
03 E8	2 bytes	Timeout (ms)
14	1 byte	Option
00 14	2 bytes	Meta Data flags
00	1 byte	Read membank
00 00 00 02	4 bytes	Read address
02	1 byte	Word count
00 00 00 00	4 bytes	Access password
00 00 00 78	4 bytes	Select address
08	1 byte	Select data length
34	1 byte	Select data
9C 0E	2 bytes	CRC

The response to this Read Data command example is:

FF 0C 28 00 00 14 00 14 22 00 00 00 15 12 34 56 78 B2 B4

Data	Length	Description
FF	1 byte	Header
0C	1 byte	Length of data field
28	1 byte	Command code
00 00	2 bytes	Status
14	1 byte	Option
00 14	2 bytes	Meta Data flags
22	1 byte	Antenna ID
00 00 00 15	4 bytes	Timestamp
12 34 56 78	4 bytes	Tag read data
B2 B4	2 bytes	CRC

5.3.9. Get Tag Buffer (29h)

After a Read Tag Multiple command is executed, the found tags are stored in an internal Tag Buffer. The Get Tag Buffer command can perform several different operations depending on the syntax used. These operations are:

- Get tags remaining in the tag buffer
- Get tag EPCs
- Get tag EPCs and their Tag Read Meta Data.

5.3.9.1. Get Tags Remaining

To determine the number of tags remaining in the buffer, send the Get Tag Buffer command with a data length of zero:

FF 00 29 1D 26

Data	Length	Description
FF	1 byte	Header
00	1 byte	Length of data field
29	1 byte	Command code
1D 26	2 bytes	CRC

This command returns the current read index, the location of the next tag to be read, and the current write index, the location where the next tag will be written. These two numbers can be used to get the number of tags left in the tag buffer:

Tags Left = WriteIndex - ReadIndex

The following response shows there are three tags left in the buffer, and the first one has already been read (the read index parameter starts counting from 0.):

FF 04 29 00 00 00 01 00 04 87 72

Data	Length	Description
FF	1 byte	Header
04	1 byte	Length of data field
29	1 byte	Command code
00 00	2 bytes	Status
00 01	2 bytes	Read Index
00 04	2 bytes	Write Index
87 72	2 bytes	CRC

5.3.9.2. Get Tag EPCs

When you want to get the tag EPCs out of the buffer and don't care about the tag read Meta Data, the options available with this syntax maybe useful. This syntax reads out the requested number of tags from the buffer. A maximum of 13 tags are read at a time, less if Max EPC Length is set to 496 bits. Multiple Get Tag Buffer commands may be required to obtain the complete results from a single Read Tag Multiple command. If more tags are requested than remain in the buffer an error will be returned.

All tag buffer indexes are encoded as 16-bit unsigned integers. The indexes start counting from 0. Thus, a start index of 0 indicates tag #1, and a start index of 10 would indicate tag #11.

There are two ways to read tag EPCs out of the buffer when you only want EPC values. The first way is to send a Get Tag Buffer command with the desired number of tags, *n.* This retrieves the next *n* tags, starting from the current read index. In the previous example, the read index was 1, indicating that one tag was already read. To read the next two tags, set the number of tags parameters to 2. This returns tags number 2 and 3 in the tag buffer:

FF 02 29 00 02 57 EB

Data	Length	Description
FF	1 byte	Header
02	1 byte	Length of data field
29	1 byte	Command code
00 02	2 bytes	# Tag IDs to return
57 EB	2 bytes	CRC

When using the Get Tag Buffer command in this way, the read index is automatically incremented internally. Thus, if the read index was '1' before getting two tags, then it is incremented to '3' at the end of this command.

Another way to get the second and third tags is to explicitly send the start and end indexes of the tags to read. This is used to retrieve any contiguous block of tag buffer entries at any time.

FF 04 29 00 01 00 03 CC 94

Data	Length	Description
FF	1 byte	Header
04	1 byte	Length of data field
29	1 byte	Command code
00 01	2 bytes	Start Index
00 03	2 bytes	End Index
CC 94	2 bytes	CRC

Using the start and end index method does not affect the read index that is internally stored in the reader. The above request returns a message with two tags. The length of each tag EPC record returned is defined based on the max EPC length configured with Set Reader Configuration, regardless of the size of a specific tag's EPC. The sizes and fields of the EPC portion of a tag buffer entry are defined in the Tag Buffer Entry table.

For example:

- A 64-bit tag has the length set to 0x60 (16-bit PC + 64-bit tag EPC + 16-bit tag CRC), but only the first 12 bytes of the tag record is filled in, the trailing bytes, after the CRC are padded with zeros.
- A 96-bit GEN2 tag, the length is 0x80 (16-bit PC (Protocol Control) word + 96-bit tag EPC + 16-bit tag EPC CRC).

In the response below, two tag EPCs are returned. Tag #1 is a 96-bit EPC tag and tag #2 is a 64-bit EPC tag.

FF 24 29 00 00 00 80 30 00 11 11 22 22 33 33 44 44 55 55 66 66 18 35 00 60 20 00 11 11 22 22 33 33 44 44 C2 41 00 00 00 00 D3 32

Data	Length	Description
FF	1 byte	Header
24	1 byte	Length of data field
29	1 byte	Command code
00 00	2 bytes	Status
00 80	2 bytes	EPC Length
30 00	2 bytes	PC Word
11 11 22 22 33 33 44 44 55 55 66 66	12 bytes	Tag EPC
18 35	2 bytes	Tag CRC
00 60	2 bytes	EPC Length
20 00	2 bytes	PC Word
11 11 22 22 33 33 44 44	8 bytes	Tag EPC
C2 41	2 bytes	Tag CRC
00 00 00 00	4 bytes	Padding
D3 32	2 bytes	CRC

Note: Using this syntax the number of tags in the response is not specified. It must be determined by the message length.

5.3.9.3. Get Tag EPCs and Meta Data

Using this syntax for getting information from the tag buffer provides several benefits:

- When at tag EPC is returned there is no padding if the actual tag EPC is shorter than the configured max EPC length. This helps minimize the amount of data returned.
- Any or all fields of the Tag Read Meta Data can be returned.
- In the event of a communication error the last Get EPC and Meta Data request can be repeated.

When data is requested using this syntax the response will contain as many tags as can be fit in the response packet. The response will indicate how many tags were returned and should be processed accordingly.

This version of Get Tag Buffer takes two additional fields: the Meta Data Flags which defines what Meta Data will be returned and the Read Options which specifies special read functionality. The following table lists the supported values for these fields.

Get EPC and Meta Data Request Fields

Field	Value	Description
Meta Data Flags (to specify more than one OR the values together)	0x0000	When no flags are set no Meta Data will be returned, only the tag EPC (including PC bits and tag CRC).
	0x0001	When bit 0 is set the Read Count will be returned
	0x0002	When bit 1 is set the LQI/RSSI will be returned
	0x0004	When bit 2 is set the Antenna ID will be returned
	0x0008	When bit 3 is set the Frequency will be returned
	0x0010	When bit 4 is set the Timestamp will be returned
	0x0020	When bit 5 is set the RFU will be returned
	0x0040	When bit 6 is set the Protocol ID will be returned.
	0x0080	When bit 7 is set up to 4 bytes of tag data will be returned. **Note:** This data is only available if the tag buffer entries are from a Read Tag Multiple (22h) with embedded Read Tag Data (28h).
Read Option	0x00	Read the next set of tags from the Tag Buffer
	0x01	Re-send the last set of tags. **Note:** When setting this option the tags will be resent starting at the same ReadIndex as the last command. If the Meta Data requested is different than the last request the number of tags returned might be different. To get those 'missing' tags, additional Get Tag Buffer commands should be sent without the re-send option; otherwise it will reset the ReadIndex again.

An example command requesting Read Count, Antenna ID and Timestamp

Meta Data Flags = 0x0001 OR 0x0004 OR 0x0010 = 0x0015

is as follows:

FF 03 29 00 15 00 E1 22

Idesco Oy | Teknologiantie 9, FIN-90590 OULU | Tel +358 20 743 4175 | info@idesco.fi | www.idesco.fi

Data	Length	Description
FF	1 byte	Header
03	1 byte	Length of data field
29	1 byte	Command code
00 15	2 bytes	Meta Data flags
00	1 byte	Read options
E1 22	2 bytes	CRC

A response contains the following information:

Get EPC and Meta Data Response Fields

Field	Length	Value
SOH	1 byte	0xFF
Length of data field	1 byte	Based on data returned
Command code	1 byte	0x29
Status	2 bytes	0x0000 for success. Otherwise see Chapter 3.5: Error Messages
Meta Data flags	2 bytes	As sent in request
Read Options	1 byte	As sent in request
Tag Count	1 byte	Number of tags in response
Read Count$_1$	1 byte	Tag EPC/Antenna Read Count
RSSI$_1$	1 byte	Return Signal Strength Indicator
Antenna ID$_1$	1 byte	Antenna ID, 4 MSBs for TX and 4 LSBs for RX
Frequency$_1$	3 bytes	Frequency in kHz
Timestamp$_1$	4 bytes	RTC Timestamp
RFU$_1$	2 bytes	Reserved for Future Use
Protocol ID$_1$	1 byte	Protocol ID of tag (always 0x05)
Tag data length$_1$	2 bytes	Length, in bits, of the tag data read for this tag. This value indicates how many bytes (ceiling [bits/8]), up to 4, will follow. *Example:* if the value is 0x1D (29) then 4 bytes will follow: 29/4 = 3.625 -- ceiling (3.625) = **4**.
Tag data$_1$	N bytes	Number of bytes of tag data as specified in *Tag Data Length*
EPC Length	2 bytes	Number of bits in EPC including PC and CRC bits
PC Word	2 bytes	Tag EPC Protocol Control bits
EPC ID	N bytes	Tag EPC
Tag CRC	2 bytes	Tag EPC CRC
Repeat fields starting at *Read Count* for remaining tags in message as defined by *Tag Count*		
CRC	2 bytes	Message CRC

1 - Conditionally returned depending on the Meta Data Fields specified in the request.

Here is an example response to the example request specified above. The response contains two tags as specified in the Tag Count Field each with its EPC info and requested tag read Meta Data: Read Count, Antenna ID and Timestamp:

FF 34 29 00 00 00 15 00 02 22 11 02 50 CE F8 00 80 31 C1 11 11 22 22 33 33 44 44 55 55 66 66 FB 15 0E 11 04 1D 3D 3C 00 80 30 00 05 00 00 00 00 00 00 00 00 00 23 54 4A C8 BA 75

Data	Length	Description
FF	1 byte	Header
34	1 byte	Length of data field
29	1 byte	Command code

00 00	2 bytes	Status
00 15	2 bytes	Meta Data flags
00	1 byte	Read Options
02	1 byte	Tag Count
22	1 byte	Read Count
11	1 byte	Antenna ID
02 50 CE F6	4 bytes	Timestamp
00 80	2 bytes	EPC Length
31 C1	2 bytes	PC Word
11 11 22 22 33 33 44 44 55 55 66 66	12 bytes	Tag EPC
FB 15	2 bytes	Tag CRC
0E	1 byte	Read Count
11	1 byte	Antenna ID
04 1D 3D 3C	4 bytes	Timestamp
00 80	2 bytes	EPC Length
30 00	2 bytes	PC Word
05 00 00 00 00 00 00 00 00 23 54	12 bytes	Tag EPC
4A C8	2 bytes	Tag CRC
BA 75	2 bytes	CRC

5.3.10. Clear Tag Buffer (2Ah)

The Clear Tag Buffer command resets the tag buffer. This clears the buffer of any current tags and reset the read index to 0. A Read Tag Multiple command must be issued to load new tags into the buffer.

FF 00 2A 1D 25

Data	Length	Description
FF	1 byte	Header
00	1 byte	Length of data field
2A	1 byte	Command code
1D 25	2 bytes	CRC

If Clear Tag Buffer or Get Tag Buffer is not used to remove all tags after a Read Tag Multiple command, old tags will be left in the buffer. This means that if tags that have not been removed from the buffer are read again they will not be added to the buffer as second time, only their Read Count will be incremented.

5.4. Set Application Commands

The Set application commands are used to set configurable values in the firmware. Since the values are not stored in flash, these values are reset to the default values whenever the application firmware is restarted. The application responds with a fault code / ACK to all commands.

5.4.1. Set Antenna Port (91h)

As a default, readers use only internal antenna for reading and writing tags. If internal antenna is wanted to use, none of these commands has to be sent to the reader. Still, antenna configuration for single tag operations (Read Tag Single (21h), Write Tag Data (24h) etc.) can be changed using these commands.

Please note: Do not select an antenna port that is not connected to an antenna. By activating a not connected antenna port, the reader's receiver part might suffer damage!

Selecting external antenna only:

FF 02 91 01 01 70 3B

Data	Length	Description
FF	1 byte	Header
02	1 byte	Length of data field
91	1 byte	Command code
01 01	2 bytes	TX antenna number and RX antenna number
70 3B	2 bytes	CRC

Selecting internal antenna only:

FF 02 91 02 02 73 38

Data	Length	Description
FF	1 byte	Header
02	1 byte	Length of data field
91	1 byte	Command code
02 02	2 bytes	TX antenna number and RX antenna number
73 38	2 bytes	CRC

Selecting external antenna for transmitting and internal antenna for receiving gives a little bit better reading distance compared to a reader using only one same antenna for both purposes. Improved reading distance is achieved by smaller disturbance leaking from transmitter to receiver in this bistatic antenna mode. This configuration can be set using following command:

FF 02 91 01 02 70 38

Data	Length	Description
FF	1 byte	Header
02	1 byte	Length of data field
91	1 byte	Command code
01 02	2 bytes	TX antenna number and RX antenna number
70 38	2 bytes	CRC

To every antenna configuration command, the reader gives this reply message, when antenna port has been successfully changed:

FF 00 91 00 00 17 58

Data	Length	Description
FF	1 byte	Header
00	1 byte	Length of data field
91	1 byte	Command code
00 00	2 bytes	Status code, 0x0000 means successful data transmission
17 58	2 bytes	CRC

5.4.2. Set Read TX Power (92h)

The Set Read TX Power command sets the default, reader-wide, power level to be used for reading tags. The power is specified as a 16-bit value in centi-dBm. For instance, a power of 25 dBm is 2500 centi-dBm, which is 0x09C4.

FF 02 92 09 C4 48 9D

Data	Length	Description
FF	1 byte	Header
02	1 byte	Length of data field
92	1 byte	Command code
09 C4	2 bytes	Power in centi-dBm
48 9D	2 bytes	CRC

Maximum value for both Read and Write TX Power setting is 30dBm and minimum value is 5 dBm. Still, it is not allowed to exceed the maximum allowed transmitting power set by European regulations.

The maximum value for real transmitting power according European regulations is 35.1 dBm and so the maximum value set for the internal antenna of the reader is 28.95 dBm (See chapter 2.2 for more detailed information).

When using external antenna, the transmitting power needs to be adjusted so, that the maximum allowed transmitting power 35.1 dBm is not exceeded. Antenna gain and antenna cable attenuation need to be taken into account, when calculating correct value for maximum transmitting power setting.

5.4.3. Set Write TX Power (94h)

The Set Write TX Power command sets the default, reader-wide, power level to use for writing to tags. This is necessary because a write operation may require a different RF power setting than a read command. The format and arguments of this command are identical to the Set Read TX Power command:

FF 02 94 09 C4 28 5B

Data	Length	Description
FF	1 byte	Header
02	1 byte	Length of data field
94	1 byte	Command code
09 C4	2 bytes	Power in centi-dBm
28 5B	2 bytes	CRC

Maximum value for both Read and Write TX Power setting is 30dBm and minimum value is 5 dBm. Still, it is not allowed to exceed the maximum allowed transmitting power set by European regulations.

The maximum value for real transmitting power according European regulations is 35.1 dBm and so the maximum value set for the internal antenna of the reader is 28.95 dBm (See chapter 2.2 for more detailed information).

When using external antenna, the transmitting power needs to be adjusted so, that the maximum allowed transmitting power 35.1 dBm is not exceeded. Antenna gain and antenna cable attenuation

need to be taken into account, when calculating correct value for maximum transmitting power setting.

5.5. Output control messages

LEDs and buzzer can be switched on any time by grounding corresponding input pin.

Additionally LED's and FET output can be controlled by commands, which are described here in this chapter.

Please note that output control messages differ from all other commands by their header, message structure and reply messages.

5.5.1. LED control messages

F0 00 00 00 14 A0 88 sets RED LED on. Reply message from the reader is **C0 00 00 10**. This setting is only valid until reset/power down.

F0 00 00 00 14 A0 84 sets GREEN LED on. Reply message from the reader is **C0 00 00 10**. This setting is only valid until reset/power down.

F0 00 00 00 14 A0 8C sets YELLOW LED on. Reply message from the reader is **C0 00 00 10**. This setting is only valid until reset/power down.

F0 00 00 00 14 A0 80 sets LEDs off. Reply message from the reader is **C0 00 00 10**.

5.5.2. FET output control message

F0 00 00 00 14 B0 00 sets FET output to the ground. Reply message from the reader is **C0 00 00 10**. This setting is only valid until reset/power down.

F0 00 00 00 14 B0 01 sets FET output out from the ground. Reply message from the reader is **C0 00 00 10**. This setting is only valid until reset/power down.

5.6. Error Messages

5.6.1. Common Error Messages

The following table lists the common faults discussed in this section.

Fault Message	Code
Wrong Number of Data	100h
Invalid Command code	101h
Unimplemented Command Code	102h
Power Too High	103h
Invalid Parameter Value	105h
Power Too Low	106h
Unimplemented Feature	109h

5.6.1.1. Wrong Number of Data (100h)

Cause: If the data length in any of the Host-to-Reader messages is less than or more than the number of arguments in the message, the reader returns this message.

Solution: Make sure the number of arguments matches the data length.

5.6.1.2. Invalid Command Code (101h)

Cause: The Command code received is invalid or not supported in the currently running program or is not supported in the current version of code.

Solution: Check the following:

- Make sure the command is supported in the currently running program.
- Check the documentation for the Command code the host sent and make sure it is correct and supported.
- Check the previous reader responses for a System error (0x7F0X) which will reset the reader.

5.6.1.3. Unimplemented Command Code (102h)

Cause: Some of the reserved commands might return this error code.

Solution: Check the documentation for the Command code the host sent to the reader and make sure it is supported.

5.6.1.4. Power Too High (103h)

Cause: A message was sent to set the read or write power to a level that is higher than the current HW supports.

Solution: Check the HW specifications for the supported powers and insure that the level is not exceeded. Idesco EPC readers support power setting values from 5 dBm to 30 dBm.

5.6.1.5. Invalid Parameter Value (105h)

Cause: The reader received a valid command with an unsupported or invalid value within this command. For example, currently the reader supports two antennas, 1 and 2. If the reader receives a message with an antenna value other than 1 or 2, it returns this error.

Solution: Make sure the host sets all the values in a command according to the values published in this document.

5.6.1.6. Power Too Low (106h)

Cause: A message was received to set the read or write power to a level that is lower than the current HW supports.

Solution: Check the HW specifications for the supported powers and insure that level is not exceeded. Idesco EPC Readers support power setting values between 5 and 30 dBm.

5.6.1.7. Unimplemented Feature (109h)

Cause: Attempting to invoke a command not supported on this firmware or hardware.

Solution: Check the command being invoked against the documentation.

5.6.2. Protocol Error Messages

The following table lists the common faults discussed in this section.

Fault Message	Code
No Tags Found	400h
No Protocol Defined	401h
Invalid Protocol Defined	402h
Write Passed Lock Failed	403h
No Data Read	404h
AFE Not ON	405h
Write Failed	406h
Not Implemented For This Protocol	407h
Invalid Write Data	408h
Invalid Address	409h
General Tag Error	40Ah
Data Too Large	40Bh
Bit Decoding Failed	40Fh
Invalid EPC	410h
Invalid Num Data	411h
Other Error	420h
Memory Overrun Bad PC	423h
Memory Locked	424h
Insufficient Power	42Bh
Non Specific Error	42Fh
Unknown Error	430h

5.6.2.1. No Tags Found (400h)

Cause: A command was received (such as like read, write, or lock) but the operation failed. There are many reasons that can cause this error to occur.

Here is a list of possible reasons that could be causing this error:

- No tag in the RF field
- Read/write power too low
- Antenna not connected
- Tag is weak or dead

Solution: Make sure there is a good tag in the field and all parameters are set up correctly. The best way to check this is to try few tags of the same type to rule out a weak tag. If none passed, then it could be SW configuration such as protocol value, antenna, and so forth, or a placement configuration like a tag location.

5.6.2.2. No Protocol Defined (401h)

Cause: A command was received to perform a protocol command but no protocol was initially set. The reader powers up with no protocols set.

Solution: A Set Current Tag Protocol command must be sent followed by resending the desired command.

5.6.2.3. Invalid Protocol Defined (402h)

Cause: A Set Current Tag Protocol command was received for a protocol value that is not supported with the current version of SW.

Solution: This value is invalid or this version of SW does not support the protocol value. Check the documentation for the correct values for the protocols in use.

5.6.2.4. Write Passed Lock Failed (403h)

Cause: During a Write Tag Data for ISO18000-6B or UCODE, if the lock fails, this error is returned. The write command passed but the lock did not. This could be a bad tag.

Solution: Try to write a few other tags and make sure that they are placed in the RF field.

5.6.2.5. No Data Read (404h)

Cause: A Read Tag ID or Data command was sent but did not succeed.

Solution: The tag used has failed or does not have the correct CRC. Try to read a few others to check the HW/SW configuration.

5.6.2.6. AFE Not ON (405h)

Cause: A command was received for an operation, like read or write, but the AFE was in the off state.

Solution: Start the AFE and then rerun the command.

5.6.2.7. Write Failed (406h)

Cause: This fault can occur when an operation such as write, lock, kill, set password, or initialize, fails. There are many reasons for failure.

Solution: Check that the tag is good and try another operation on a few more tags.

5.6.2.8. Not Implemented For This Protocol (407h)

Cause: A command was received which is not supported by a protocol.

Solution: Check the documentation for the supported commands and protocols.

5.6.2.9. Invalid Write Data (408h)

Cause: In EPC0+, the first two bits determine the tag ID length. If the first two bits are 0b00, then the tag ID must be 96-bits. Otherwise the tag ID is 64 bits.

Solution: Make sure that the first two bit have the correct values depending in the Tag ID length.

5.6.2.10. Invalid Address (409h)

Cause: A command was received attempting to access an invalid address in the tag data address space.

Solution: Make sure that the address specified is within the scope of the tag data address space and available for the specific operation. The protocol specifications contain information about the supported addresses.

5.6.2.11. General Tag Error (40Ah)

Cause: This error is used by the Idesco EPC Reader. This fault can occur if the read, write, lock, or kill command fails. This error can be internal or functional.

Solution: Make a note of the operations you were performing and contact Idesco Support at support@idesco.fi.

5.6.2.12. Data Too Large (40Bh)

Cause: A command was received to Read Tag Data with a data value larger than expected or it is not the correct size.

Solution: Check the size of the data value in the message sent to the reader.

5.6.2.13. Invalid Kill Password (40Ch)

Cause: An incorrect kill password was received as part of the Kill Tag (26h) command.

Solution: Check the password.

5.6.2.14. Kill Failed (40Eh)

Cause: Attempt to kill a tag failed for an unknown reason

Solution: Check tag is in RF field and the kill password.

5.6.2.15. Bit Decoding Failed (40Fh)

Cause: Attempt to operate on a tag with an EPC length greater than the Maximum EPC length setting.

Solution: Contact Idesco Support (support@idesco.fi) to get Set Reader Configuration command to set the Max EPC to 496 bits.

5.6.2.16. Invalid EPC (410h)

Cause: This error is used by the Idesco EPC reader indicating an invalid EPC value has been specified for an operation. This fault can occur if the read, write, lock, or kill command fails.

Solution: Check the EPC value that is being passed in the command resulting in this error.

5.6.2.17. Invalid Num Data

Cause: This error is used by the Idesco EPC reader indicating invalid data has been specified for an operation. This fault can occur if the read, write, lock, or kill command fails.

Solution: Check the data that is being passed in the command resulting in this error.

5.6.3. Hardware Error Messages

The following table lists the common faults discussed in this section.

Fault Message	Code
Antenna Not Connected	503h
Temperature Exceeds Limits	504h

High Return Loss	505h

5.6.3.1. Antenna Not Connected (503h)

Cause: An attempt was made to transmit on an antenna which did not pass the antenna detection when antenna detection was turned on.

Solution: Connect a detectable antenna (antenna must have some DC resistance).

5.6.3.2. Temperature Exceeds Limits (504h)

Cause: The reader has exceeded the maximum or minimum operating temperature and will not allow an RF operation until it is back in range.

Solution: Take steps to resolve thermal issues with reader:

- Reduce duty cycle
- Add heat sink
- Use Low Power Mode (contact Idesco Support support@idesco.fi for proper command)

5.6.3.3. High Return Loss (505h)

Cause: The reader has detected a high return loss and has ended RF operation to avoid reader damage.

Solution: Take steps to resolve high return loss on receiver:

- Make sure antenna VSWR is within reader specifications
- Make sure antennas are correctly attached before transmitting
- Check environment to ensure no occurrences of high signal reflection back at antennas.

5.6.4. Tag ID Buffer Faults

The following table lists the common faults discussed in this section.

Fault Message	Code
Not enough tags available	600h
Buffer Full	601h
Repeated Tag ID	602h
Number of tags too large	603h

5.6.4.1. Not Enough Tags Available (600h)

Cause: A command was received to get a certain number of tag ids from the tag id buffer. The reader contains less tag ids stored in its tag id buffer than the number the host is sending.

Solution: Send a Get Tag ID Buffer command to ascertain how many tags are in the buffer. You can get the exact number of tags as long as they are less than or equal to the number returned by the previous command.

5.6.4.2. Buffer Full (601h)

Cause: The tag id buffer is full.

Solution: Send Clear Tag ID Buffer or Get Tag ID Buffer command with the number of tags to read, to get more space See Get Tag Buffer (29h) for more information.

5.6.4.3. Repeated Tag ID (602h)

Cause: The reader has an internal error. One of the protocols is trying to add an existing Tag ID to the buffer.

Solution: Report this problem to Idesco Oy at support@idesco.fi.

5.6.4.4. Number of Tags Too Large (603h)

Cause: The reader received a request to retrieve more tags than is supported by the current version of the software.

Solution: Locate the maximum number of supported tags in the Tag ID buffer in this document Get Tag Buffer (29h).

5.6.5. System Errors

5.6.5.1. Unknown Errors (7F00h and 7F01h)

Cause: An unexpected Internal Error has occurred.

Solution: Make note of the operations you were executing and contact Idesco Oy at support@idesco.fi.

Idesco Oy | Teknologiantie 9, FIN-90590 OULU | Tel +358 20 743 4175 | info@idesco.fi | www.idesco.fi