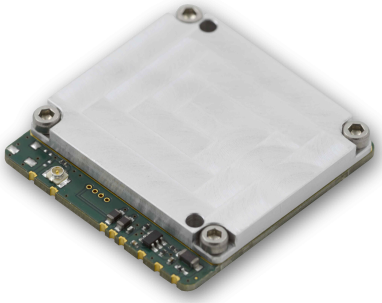
UHF Protocol Guide

for metraTec UHF Readers and Modules



Date: June 2016

Version: 2.8
For Firmware Version: 03.13
Customer Edition

Table of Contents

Introduction

1. General Information

This document describes the metraTec firmware protocol for all metraTec UHF RFID readers. This includes the DeskID UHF, DwarfG2, DwarfG2_XR, DwarfG2_Mini and the PulsarMX reader. The UMG2 is the OEM version of the DeskID_UHF. When a DeskID is named in this document this always includes the UMG2. All UHF readers by metraTec use a similar firmware with the same Reader-PC-Communication protocol. However, there are some differences concerning the functions supported (GPIOs for example), the default parameter settings (number of tags best read, power saving mode), the reader's RF power and supported functions. Especially the DwarfG2_Mini has some more differences (reduced functions, differing hardware [chipset, modem...], bootloader).

The target audience for this document are programmers, who need to communicate with the reader via low level protocol. An alternative to a low level protocol is to use our free .NET DLL on MS Windows systems. For all who want to write their own software for this task using the programming language of their choice not supported by Visual Studio (such as Java, Delphi, ANSI-C, …) or need to program for a different operating system such as Linux or even an embedded system or SPS, this documentation should give the needed information for low level protocol commands. It might also make sense to use this low-level protocol if your goal is maximum speed and you cannot wait for a DLL to process your data stream.

The reader firmware offers an ASCII based programming interface. The instructions are identified by an easy to remember, three character string usually followed by mandatory parameters and/or optional parameters. The response format depends on the type and result of an instruction.

Instructions (as well as this document) are divided into two main groups:

- Reader Instructions, divided into
  - Reader Control Instructions
  - Reader Configuration Instructions
- Tag Manipulation Instructions

All Instructions have Error Codes that are described in Chapter 5, *Error Codes*. In default mode (corresponding to Verbosity Level (**VBL**)=1) any data sent to the reader causes some answer (including parsing errors, communication errors etc.).

2. Basic Gen2 Memory Information

UHF protocol of our readers is according to the EPC UHF Class 1 Gen 2 Specification. This chapter summarizes the most important information. For further read please refer to the EPC Specification.

A Gen2 tag has up to 4 memory areas called memory banks. These 4 memory banks are:

- EPC (Electronic Product Code)
- TID (Tag ID)
- USR (USer Reserved memory)
- RES (REServed (for passwords))

All memory banks may contain additional data following any mandatory data if specified. This is manufacturer specific.

The EPC bank contains a 1 word (16 bits) CRC (Cyclic Redundancy Check) block, a 1 word PC (Protocol Control) block and a number of words as EPC (Electronic Product Code). The CRC is a CRC 16, computed over PC and EPC. The complete memory bank does not necessarily need to be used as EPC. The length of the EPC is saved in the 5 MSBs of the PC in words. The 5 bit can define a maximum of 31 words or 62 bytes of EPC data. A length of zero will be an empty EPC. The PC can be written by the user as can the EPC and the CRC. Additionally, the CRC is recalculated after every re-powering of the tag. Keep in mind that a wrong CRC will cause problems, so when writing the EPC memory bank also change the CRC or re-power the tag (remove it from field or switch the RF field off and on).

CRC	PC			EPC
WORD 0	WORD 1			WORDS 2: (1+Length)
CRC 16	BIT 0:4 EPC Length in WORD. MAX value: 31	BITS 5:6 Reserved (00)	BITS 7:F Numbering System Identifier (NSI)	EPC

Table 1. EPC memory bank structure according to EPC UHF Class 1 Gen 2 Specification

The TID (Tag IDentifier) memory bank contains an ID for the tag. They are usually non-writeable, often unique but do not need to be unique.

Bits 0:7 (1 Byte)	Bits 8:31 (3 Byte)	Vendor specific Bytes
ISO/IEC 15963 allocation class identifier	Additional identification to uniquely identify custom commands and optional features	These bytes are vendor specific and fully optional

Table 2. TID memory bank structure according to EPC UHF Class 1 Gen 2 Specification

The USR (user) memory bank can be used to store data on the tag. It is optional, so not every UHF-tag has usable memory in this bank. Its size is unlimited by the EPC Class 1 Gen 2 Protocol but the current firmware „only“ supports addressing up to eight Gbyte (unsigned32 variable, 2 bytes per level). Later versions might change this, if needed.

The RES (reserved) memory bank contains the Access Password (2 words) and the Kill Password (2 words). Passwords can be written to and read from the tag. They can be protected separately by the lock command. If required, they can become permanently unwriteable and unreadable.

Bits 0x00:0x1F (2 WORDs)	Bits 0x20:0x3F (2 WORDs)
Kill Password	Access Password

Table 3. RES memory bank structure according to EPC UHF Class 1 Gen 2 Specification

3. Further Documentation

To fully understand all commands and the response of the UHF tag it might be helpful to read the EPC UHF Class 1 Gen 2 Specification, which can be downloaded for free at http://www.gs1.org/sites/default/files/docs/epc/uhfc1g2_1_2_0-standard-20080511.pdf

Note

Note: The above linked EPC UHF Class 1 Gen 2 Specification is not be the most current version but the base for this documentation.

Additionally, it is recommended to read the respective tag IC datasheet concerning tag IC specific advanced features defined by the manufacturer.

Typographic Conventions

Special typographic conventions and highlightings are used in metraTec protocol guides and other documents to streamline content that would otherwise be hard to express (e.g. syntax descriptions) and in order to provide a consistent look across metraTec documentation.

The following table summarizes typographic conventions and their descriptions:

Convention	Description
`COMMAND`	A command name, i.e. the *literal* name of a command in a metraTec protocol. For instance, **RST** would correspond to the literal characters of a command that could be sent to a metraTec device.
`Literal`	Highlights a *literal value* directly representing the literal characters that have to be used (e.g. in a protocol). For instance `UCO` would correspond to the literal characters as they could be returned by a metraTec device.
`Token`	This convention highlights a replaceable (abstract) *`Token`* that in contrast to a literal token is a placeholder for some other value that must be substituted by the user. The abstract *`Token`* is usually documented in more detail.
<LITERAL>	Represents a literal character that cannot be printed as such or needs to be highlighted specifically and is therefore formatted as an abstract identifier. Examples include "*<CR>*" — representing the carriage return character (ASCII 13) — and "*<SPACE>*" — representing one or more space characters (ASCII 32). This special formatting is used both in syntax descriptions, command and response examples.
`{` *`Construct`* `}`	This convention highlights that a *`Construct`* is required. It is most commonly used in syntax descriptions to highlight that a parameter *must* be specified in the position that this construct is used.
`[` *`Construct`* `]`	Highlights that a *`Construct`* is optional. It is most commonly used in syntax descriptions to highlight that a parameter *may* be specified in the position that this construct is used.
`Construct` `...`	Highlights that a *`Construct`* may be repeated many times.
...*Name*...	The horizontal ellipsis "..." may be used in syntax descriptions to represent arbitrary characters. The arbitrary character field may be given a *`Name`* in order to document it in more detail.
`Alternative`$_1$ | *`Alternative`*$_2$ `...` | *`Alternative`*$_n$	Highlights that in the position of this construct one of *n* alternatives may be used.

`Literal Line 1`
`Literal Line 2`
`Literal Line 3`
	A literal block of text. It is often used to document example protocol exchanges or code examples. In the former case, literal placeholders like "*<CR>*" may be included in the code block to express that lines are separated by carriage return. In the latter case, programming language source

Convention	Description
	code may be syntax highlighted. These literal blocks of code may also contain callout graphics or line annotations to document each line of text.
» `Command` « `Response`	In examples of a command-response exchange, the literal examples of the `Command` and `Response` may be highlighted differently. Otherwise these literal blocks of text are formatted the same as described above.
ℹ **Note** Paragraph	A paragraph set off from the text to highlight noteworthy information.
⚠ **Warning** Paragraph	A paragraph set off from the text to highlight information necessary to prevent harm to electronic devices or persons.

Chapter 1. Communication Principle

1.1. General

The communication between the device and the host system is based on ASCII strings. Each string is terminated with a carriage-return (0x0D), *not* the null byte or line feed (0x0A), and will be transmitted with the most significant byte first.

The communication from the reader to the host system (i.e. the response) is the same as above but in some cases the response from the reader comprises more than one line.

If you use a serial connection to communicate with the reader (USB via Com-Port-Emulator, RS232, etc.) you need to use a baudrate of 115.200 baud, 8 data bits, no parity, 1 stop bit (8,N,1). Older firmware versions might use 460.800 baud instead (DeskID <=1.6, Dwarf<=1.4, Pulsar=1.0)

Please make sure that you really send a carriage-return character as the last character – not more and not less. Many programs (including Hyperterm and some Unix/Linux programs) use carriage return + line feed as end of line character which leads to problems after the first command, since the line feed character is treated as the first character of the following command which results in the error code „Unknown COmmand" (UCO) or Command Receive Timeout (CRT). For highest comfort use the free metraTerm application that can be found on the metraTec website: http://www.metratec.com/en/support/downloads/software.html .

1.2. Host to Reader Communication Details

General syntax:

```
{ Instruction } [ <SPACE> Parameter ...] <CR>
```

```
» REV<CR>
```

Example 1. Command without Parameter

```
char Rev[4] = {'R','E','V','\r'};
```

Example 2. Command without Parameter in ANSI C

The first value which will be sent in the above examples is R (0x52), followed by 0x45, 0x56, 0x0D. Some instructions may be specified with parameters, which are separated by a space (0x20).

```
» INV<SPACE>SSL<CR>
```

Example 3. Command with Parameter

```
char Inv[8] = {'I','N','V',' ','S','S','L','\r'};
```

Example 4. Command with Parameter in ANSI C

1.3. Helpful Tools

For debugging purposes it is very helpful to use a program to record the communication between the host and the reader. Depending on the type of communication and hardware you use, this could be:

- If you communicate via a (real or virtual) COM-Port: a Com-Port Monitor (several free version available in the net for WindowsXP / 7)
- If you use Ethernet or other TCP/IP-based communication, like WiFi: a packet sniffing tool, e.g. Wireshark/Ethereal [http://www.wireshark.org/], which is available for almost every platform
- If you use a direct UART connection or something at a similar low level: a hardware logic analyzer
- To send ASCII data via a serial connection or even Ethernet, you can use the free metraTerm terminal software, also available on our website. Example command files for almost all metraTec devices and protocols are available.

Chapter 2. Reader Instructions

This list gives an overview of all available instructions directed at the reader. The commands often have several possible answers, some of which indicate an error. Some of the error codes (URE, CRT, BOD...) can appear irrespective of the user software. If used correctly, other errors like parsing errors (UCO, UPA...) should not occur. The types of parsing errors possible depend on the command and its parameters.

Command	Name	Description
RST	Reset	Resets the device
REV	Revision	Returns device type, hardware architecture version and firmware version
RHR	Read Hardware Revision	Return the hardware revision
RFW	Read Firmware Revision	Returns firmware name and version
RHW	Read Hardware	Returns hardware name and version
RSN	Read Serial Number	Returns the Serial Number of the reader
STB	Standby	Sends the reader into standby/sleep mode to save power
WAK	Wake Up	Ends standby/sleep mode
ECH	Echo	Echos up to 16 characters
HBT	Heartbeat	Sends a "HBT" every given number of seconds
RIP	Read Input Pin	Reads the state of an input pin
WOP	Write Output Pin	Writes the state of an output pin
MOD	Outdated MOD Command	Formerly used to set the communication standard
STD	Set RFID Standard to Use	Sets the communication standard
SRI	Set RF Interface	Switches RF power
SAP	Set Antenna Port	Controls 4 output pins of a PulsarMX, DwarfG2, DwarfG2_XR or DwarfG2_Mini at once to control a multiplexer.
RAP	Read Antenna Port	Give the multiplexer state
RRP	Read Reflected Power	Gives you information regarding antenna mismatch, broken cables as well as missing antennas.
CFG	Configuration	Configuration of RFID frontend parameters
SET	Set Parameter	Configuring reader behaviour
SQV	Set Q Value	Sets the Q Value (see EPC Gen2)
RQV	Read Q Value	Reads the Q Value (see EPC Gen2)
SIR	Set Inventory Retry	Sets the Inventory Retry (IR) Value
RIR	Read Inventory Retry Value	Reads the IR value (see EPC Gen2)
CON	Cyclic Redundancy Check On	Turns on CRC checking of host / reader communication
COF	Cyclic Redundancy Check Off	Turns off CRC checking of host / reader communication

Command	Name	Description
EOF	End Of Frame Mode	Adds a *<LF>* at the end of every answer (not necessarily every line).
NEF	No End of Frame Mode	Turns off the End of Frame delimiter *<LF>*.
VBL	Verbosity Level	Sets the verbosity level
SUC	Start Up Commands	A set of commands that are executed automatically on **RST** or power up
RSC	Read Start Up Commands	Read the set of commands that are executed automatically on **RST** or power up.
GCT	Get Core Temperature	Gets the temperature of the IC core. Only supported on DwarfG2_Mini.
CNR	Continuous Retry	Executes the given command until a break condition
BRK	Break	Stops a **CNR** command operation

Table 4. Overview of Reader Instructions

2.1. Reset (RST)

The Reset command resets the reader. It has no parameters. After sending the **RST** command and receiving the answer `OK!` the reader will behave like after (re-)powering. The device is set to its default state and any applicable start up commands are executed. *Reset* works even in sleep mode (**STB** and **CNR** (Continuous Retry) mode. Any configurations will be forgotten. Changes committed to the internal non volatile memory (EEPROM) have to be reloaded (for example **CFG**, **SET** `IHC`) if not otherwise stated (for example **SUC**).

The startup process (from the time the `OK!` is received until new commands to the reader are accepted) takes about 200ms. This time will be increased in case of applicable startup commands (**SUC**).

```
» RST<CR>
```

```
« OK!<CR>
```

*Example 5. **RST** command and answer*

Instruction

RST *<CR>*

Return Values in Case of Success

`OK!` *<CR>*

Return Values in Case of Failure

"BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.2. Revision (REV)

On the revision command the reader returns its device type, its hardware architecture version and its firmware revision.

As the DwarfG2_Mini uses the 2nd-Gen metraTec bootloader the **REV** command is still supported but deprecated for this device. Use **RFW** and **RHW** instead.

The Revision command has no parameters.

»
```
REV<CR>
```

«
```
PULSAR_MX      01000313<CR>
```

or

«
```
DWARFG2_MINI   01010313<CR>
```

...

*Example 6. **REV** command and answer*

Instruction

REV *<CR>*

Return Values in Case of Success

Product revision

15 Bytes product name (filled with spaces) + 4 bytes HW-Architecture-Revision (ASCII) + 4 Bytes Firmware-Revision (ASCII) + *<CR>*

Return Values in Case of Failure

"BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.3. Read Hardware Revision (RHR)

The **RHR** command returns the hardware revision of the reader which corresponds to the PCB layout version printed on the board. The number is an ASCII string of four characters. It might be required for providing product support and has the form MMSS (2 bytes main version, 2 bytes sub version).

Old readers might not have a coded hardware revision. The answer is “ÿÿÿÿ” (0xFFFFFFFF) in this case.

As the DwarfG2_Mini uses the 2nd-Gen metraTec bootloader the **RHR** command is still supported but deprecated for this device. Use **RHW** instead.

»
```
RHR<CR>
```

«
```
0100<CR>
```

Example 7. ***RHR*** *command and answer*

Instruction

RHR *<CR>*

Return Values in Case of Success

Hardware Revision

Return Values in Case of Failure

“BOD *<CR>*”, “BOF *<CR>*”, “CCE *<CR>*”, “CRT *<CR>*”, “PFE *<CR>*”, “PLE *<CR>*”, “SRT *<CR>*”, “UCO *<CR>*”, “UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*”, “UPA *<CR>*” or “URE *<CR>*”

2.4. Read Firmware Revision (RFW)

RFW is only supported on the DwarfG2_Mini.

On **RFW** the device returns its firmware name and version in the format `16 Bytes product name (filled with spaces) + 4 Bytes Firmware-Revision(ASCII) +` *<CR>* .

The **RFW** command has no parameters.

»
```
RFW<CR>
```

«
```
DWARFG2_MINI    0313<CR>
```

Example 8. ***RFW*** *command and answer*

Instruction

RFW *<CR>*

Return Values in Case of Success

Firmware revision

16 Bytes product name (filled with spaces) + 4 Bytes firmware version (ASCII) + *<CR>*

Return Values in Case of Failure

"NOS *<CR>*", "BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.5. Read Hardware (RHW)

This command returns the name of the hardware and the version of the hardware - both identical to the name and version printed on the circuit board.

This command is only supported by DwarfG2_Mini.

Instruction

RHW *<CR>*

Examples

```
RHW<CR>
```

Example 9. Only supported by DwarfG2_Mini: Read hardware name and revision from reader

Return Values in Case of Success

[16Byte Hardwarename][4 Bytes Revision]<CR>

Return Values in Case of Failure

"NOS *<CR>*", "BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.6. Read Serial Number (RSN)

The **RSN** command returns the reader's serial number. The serial number may be useful when requesting product support.

```
» RSN<CR>
« 2015100808325100<CR>
```

*Example 10. **RSN** command and answer*

Instruction

RSN *<CR>*

Return Values in Case of Success

Serial Number

Return Values in Case of Failure

"BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.7. Standby (STB)

The standby command sends the reader into power save mode. The RF power is turned off. This means that all tags that might be in the field will also be depowered. The RF power state is saved and will be reset on wake up. If successful it returns GN8 ("Good Night"). Except for DwarfG2_Mini, the reader will not accept any commands except reset (**RST**) until a Wake Up command (**WAK**) is received. The DwarfG2_Mini will wake up on any command line. Standby has no parameters.

```
» STB<CR>
« GN8<CR>
» STB<CR>
«
```

*Example 11. **STB** command and answer (normal and already in standby mode (no answer at all))*

Instruction

STB *<CR>*

Return Values in Case of Success

GN8 *<CR>*

Return Values in Case of Failure

"BOD <*CR*>", "BOF <*CR*>", "CCE <*CR*>", "CRT <*CR*>", "PFE <*CR*>", "PLE <*CR*>", "SRT <*CR*>", "UCO <*CR*>", "UER[<*SPACE*> {*Two Digit Hex Code*}] <*CR*>", "UPA <*CR*>" or "URE <*CR*>"

2.8. Wake Up (WAK)

The wake up command ends the power save mode. The reader will restore its last state prior to the standby command. If successful it returns GMO ("Good Morning"). Wake up has no parameters.

»
```
WAK<CR>
```

«
```
GMO<CR>
```

»
```
WAK<CR>
```

«
```
DNS<CR>
```

Example 12. ***WAK*** *command and answer*

Instruction

WAK <*CR*>

Return Values in Case of Success

GMO <*CR*>

Return Values in Case of Failure

"DNS <*CR*>", "BOD <*CR*>", "BOF <*CR*>", "CCE <*CR*>", "CRT <*CR*>", "PFE <*CR*>", "PLE <*CR*>", "SRT <*CR*>", "UCO <*CR*>", "UER[<*SPACE*> {*Two Digit Hex Code*}] <*CR*>", "UPA <*CR*>" or "URE <*CR*>"

2.9. Echo (ECH)

This command takes whatever the host gives as parameter and echoes that parameter. The echo is just normally added with <*CR*> and, if activated, <*LF*>. Any value may be echoed except <*CR*> which marks the command end and <*LF*> which is ignored by the command receiver. Lower case data will be echoed upper case.

»
```
ECH HELLO<CR>
```

```
« HELLO<CR>
```

```
» ECH hello<SPACE><SPACE>2<CR>
```

```
« HELLO<SPACE><SPACE>2<CR>
```

*Example 13. **ECH** command and answer*

Instruction

ECH <*SPACE*> {…*String to echo*…} <*CR*>

Return Values in Case of Success

ECHO <*CR*>

Return Values in Case of Failure

"WDL <*CR*>", "BOD <*CR*>", "BOF <*CR*>", "CCE <*CR*>", "CRT <*CR*>", "PFE <*CR*>", "PLE <*CR*>", "SRT <*CR*>", "UCO <*CR*>", "UER[<*SPACE*> {*Two Digit Hex Code*}] <*CR*>", "UPA <*CR*>" or "URE <*CR*>"

2.10. Heartbeat (HBT)

The heartbeat command enables or disables the heartbeat. If enabled the device will send a HBT every given number of seconds. A HBT without any parameter or with SHW will give you the state meaning either OFF or the number of seconds.

```
» HBT SHW<CR>
```

```
« OFF<CR>
```

```
» HBT<CR>
```

```
« OFF<CR>
```

```
» HBT 1<CR>
```

```
« OK!<CR>
```

```
« HBT<CR>
```

one second delay

```
« HBT<CR>
```

```
» HBT 20<CR>
```

```
« OK!<CR>
```

```
« HBT<CR>
```

20 seconds delay

```
« HBT<CR>
```

20 seconds delay

```
« HBT<CR>
```

```
» HBT SHW<CR>
```

```
« 20<CR>
```

```
» HBT OFF<CR>
```

```
« OK!<CR>
```

Example 14. ***HBT*** *command and answer*

2.10.1. Get heartbeat parameter state (SHW)

Instruction

`HBT [<SPACE> SHW ] <CR>`

Parameters

Name	Type	Description
SHW	Optional Keyword	The parameter is the `SWH` keyword. It's use is optional.

2.10.2. HBT OFF (OFF)

`OFF` will switch the heartbeat off.

Instruction

`HBT <SPACE> OFF <CR>`

Parameters

Name	Type	Description
OFF	Keyword	The parameter is the OFF keyword

2.10.3. HBT ON (INT)

Any number (1-300) will switch the heartbeat on with for the specified time in seconds as time distance.

Instruction

HBT *<SPACE>* {*Heartbeat time*} *<CR>*

Parameters

Name	Type
Heartbeat time	Decimal Integer (1 ≤ x ≤ 300)

Return Values in Case of Success

OK! *<CR>*

Return Values in Case of Failure

"NOR *<CR>*", "BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.11. Read Input Pin (RIP)

This command is used to read the current state of an input pin. It takes one parameter, which is the hex-coded, zero-based number of the input pin to be read. The possible parameter range depends on the number of inputs the hardware has. The PulsarMX accepts 0 and 1 as pin numbers. DwarfG2 and DwarfG2_XR accept 0 to 7 as input pin numbers. The DwarfG2_Mini accepts 0 to 3 as input pin numbers. The DeskID_UHF and UMG2 do not have input pins.

If successful, it returns either HI! or LOW depending on whether the input pin is high or low.

Warning

In case of the DwarfG2, DwarfG2_XR, and the DwarfG2_Mini the input pins can also be used as output pins (General Purpose Inputs / Outputs - GPIOs). When calling **RIP** the direction the pin is being used in is changed to being an input pin. Please make sure that the hardware connected to the pin is meant to be operated this way before calling **RIP** as this can destroy the hardware.

»
```
RIP 0<CR>
```

«
```
HI!<CR>
```

»
```
RIP 7<CR>
```

```
« LOW<CR>
```

*Example 15. **RIP** command and answer*

Instruction

`RIP <SPACE> {Input Pin Number} <CR>`

Parameters

Name	Type
Input Pin Number	Hexadecimal Integer ($0_{16} \le x \le 7_{16}$)

Return Values in Case of Success

"`HI! <CR>`" or "`LOW <CR>`"

Return Values in Case of Failure

`NOS <CR>`

(Not Supported (DeskID_UHF and UMG2 have no GPIOs))

`NOR <CR>`

(Number Out of Range (PulsarMX has 2 inputs, DwarfG2 and DwarfG2_XR have 8 GPIOs, DwarfG2_Mini has 4 GPIOs))

"`EHX <CR>`", "`BOD <CR>`", "`BOF <CR>`", "`CCE <CR>`", "`CRT <CR>`", "`PFE <CR>`", "`PLE <CR>`", "`SRT <CR>`", "`UCO <CR>`", "`UER[<SPACE> {Two Digit Hex Code}] <CR>`", "`UPA <CR>`" or "`URE <CR>`"

2.12. Write Output Pin (WOP)

This command is used to set the state of an output pin either to high or to low. It takes two parameters. The first parameter is the zero-based number of the output pin to be written to. The second parameter is either "HI" or "LOW" to set the according pin to high or low respectively. The possible parameter range depends on the number of output pins the hardware has. PulsarMX and DwarfG2_Mini accept 0 to 3 as output pin numbers. DwarfG2 and DwarfG2_XR accept 0 to 7 as output pin numbers. The DeskID_UHF and UMG2 do not have output pins.

Warning

In case of the DwarfG2, DwarfG2_XR, and DwarfG2_Mini the output pins can also be used as input pins (General Purpose Inputs / Outputs - GPIOs). When calling **WOP** the direction the pin is being used in is changed to being an output pin. Please make sure that the hardware connected to the pin will not exceed the pin's maximum limits in output mode before calling **WOP** as this can destroy the reader.

For addressing the four lowest pins at once e.g. in case of a multiplexer please refer to the **SAP** command.

» `WOP 0 HI<CR>`

« `OK!<CR>`

» `WOP 7 LOW<CR>`

« `OK!<CR>`

*Example 16. **WOP** command and answer*

Instruction

`WOP` `<SPACE> {Output Pin Number} <SPACE> { HI | LOW } <CR>`

Parameters

Name	Type	Description
Output Pin Number	Hexadecimal Integer ($0_{16} \le x \le 7_{16}$)	The first parameter is the hex-coded, zero-based number of the output pin to be written to.
Output Value	Enumeration (`HI` or `LOW`)	The second parameter is the output value

Return Values in Case of Success

`OK! <CR>`

Return Values in Case of Failure

`NOS <CR>`
(Not Supported (DeskID_UHF and UMG2 have no GPIOs))

`NOR <CR>`
(Number Out of Range (PulsarMX has 4 outputs, DwarfG2 and DwarfG2_XR have 8 GPIOs, DwarfG2_Mini has 4 GPIOs))

“`EHX <CR>`”, “`BOD <CR>`”, “`BOF <CR>`”, “`CCE <CR>`”, “`CRT <CR>`”, “`PFE <CR>`”, “`PLE <CR>`”, “`SRT <CR>`”, “`UCO <CR>`”, “`UER`[`<SPACE>` {`Two Digit Hex Code`}] `<CR>`”, “`UPA <CR>`” or “`URE <CR>`”

2.13. Outdated MOD Command (MOD)

The **`MOD`** command is deprecated. The **`MOD`** `STD` command is replaced by the single **`STD`**. See documentation of the new command **`STD`**.

Instruction

`MOD` `<SPACE> STD <SPACE> { ETSI | ETS | ISR | FCC } <CR>`

Parameters

Name	Type
Standard	Enumeration (ETSI, ETS, ISR or FCC)

Return Values in Case of Success

OK! *<CR>*

Return Values in Case of Failure

“NOS *<CR>*”, “ARH *<CR>*”, “BOD *<CR>*”, “BOF *<CR>*”, “CCE *<CR>*”, “CRT *<CR>*”, “PFE *<CR>*”, “PLE *<CR>*”, “SRT *<CR>*”, “UCO *<CR>*”, “UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*”, “UPA *<CR>*” or “URE *<CR>*”

2.14. Set RFID Standard to Use (STD)

The **STD** command allows the setting of the RFID communication standard. Depending on the firmware used (shown in the name) there is normally only one flag allowed depending on the region the device is sold to. In EU: ETS (for ETSI), in Israel ISR (not supported on standard releases), in the USA FCC (supported since revision 03.08 for DwarfG2, DwarfG2_XR, and PulsarMX).

Note

ETS uses frequencies band of 865-868 MHz. FCC uses 902-915 MHz (out of the allowed 902-928 MHz band). Make sure an antenna is attached and matching the frequency range, especially on DwarfG2 and DwarfG2_Mini as they have no reflectivity measurement implemented.

If successful, the **STD** command also causes a channel hopping according to the set rules.

For the PulsarMX and the DwarfG2_XR, it also performs an antenna check. The antenna check can be disabled by using the AWI flag. If the antenna check failes the answer is the error message ARH and the power LED starts to blink. During this time, turning on HF power (**SRI** ON) is impossible. The LED will stop blinking after a successful **STD**.

```
» STD ETS<CR>

« OK!<CR>

» STD FCC<CR>

« OK!<CR>

» STD FCC<CR>
```

```
« NOS<CR>
```

*Example 17. **STD** command and answer*

Instruction

STD *<SPACE>* { ETS | ISR | FCC } [*<SPACE>* AWI] *<CR>*

Parameters

Name	Type
Standard	Enumeration (ETS, ISR or FCC)

Return Values in Case of Success

OK! *<CR>*

Return Values in Case of Failure

“NOS *<CR>*”, “ARH *<CR>*”, “BOD *<CR>*”, “BOF *<CR>*”, “CCE *<CR>*”, “CRT *<CR>*”, “PFE *<CR>*”, “PLE *<CR>*”, “SRT *<CR>*”, “UCO *<CR>*”, “UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*”, “UPA *<CR>*” or “URE *<CR>*”

2.15. Set RF Interface (SRI)

The **SRI** command is used to control the RF output of the reader. This command is used to switch the RF field ON or OFF (e.g. to reset tags or save power).

Note

The tag manipulation command **INV** will switch on RF power for execution regardless whether RF power was set to ON by using **SRI** ON.

Before using **SRI** the standard has to be set by the Set RFID Standard (**STD**) command.

```
» SRI ON<CR>
```

```
« OK!<CR>
```

```
» SRI OFF<CR>
```

```
« OK!<CR>
```

```
» SRI TIM 500<CR>
```

```
« OK!<CR>
```

»
```
SRI SPM ON<CR>
```

«
```
OK!<CR>
```

Example 18. ***SRI*** *command and answer*

2.15.1. RF ON (ON)

`ON` will switch the RF power on.

Instruction

```
SRI <SPACE> ON <CR>
```

Parameters

Name	Type	Description
ON	Keyword	The parameter is the ON keyword

2.15.2. RF OFF (OFF)

`OFF` will switch the RF power off.

Instruction

```
SRI <SPACE> OFF <CR>
```

Parameters

Name	Type	Description
OFF	Keyword	The parameter is the OFF keyword

2.15.3. Timed OFF (TIM)

`TIM` will switch the RF power off for the specified time.

Instruction

```
SRI <SPACE> TIM <SPACE> {Time} <CR>
```

Parameters

Name	Type	Description
TIM	Keyword	This parameter is the TIM keyword
Time	Decimal Integer ($10 \leq x \leq 10000$)	This parameter is the time in milliseconds to stay disabled.

2.15.4. Set Powersaving Mode (SPM)

`SPM` will either activate or deactivate the save power mode. When activated the RF field is disabled between commands. This will save energy, keep the device cool and reduce noise to other devices. This setting will not have much effect in **CNR** (Continuous Retry) mode as the reader is continuously reading (and powered on at the time).

Instruction

SRI *<SPACE>* SPM *<SPACE>* { ON | OFF } *<CR>*

Parameters

Name	Type	Description
SPM	Keyword	This parameter is the SPM keyword
ON/OFF	Enumeration (ON or OFF)	Turns the mode on/off

Return Values in Case of Success

"OK! *<CR>*" or "NRF *<CR>*"

Return Values in Case of Failure

"EDX *<CR>*", "NOR *<CR>*", "NSS *<CR>*", "NOS *<CR>*", "BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>*{*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.16. Set Antenna Port (SAP)

This command is used to set the 4 lowest outputs of the device at once so that a metraTec multiplexer connected to the reader will directly activate the correct antenna port (port numbers starting at 0). The antenna port number is represented by the four pins in binary code with the pins being set to high or low accordingly. It replaces a sequence of **WOP** commands that would be needed to set the individual outputs sequentially.

Note

Please remember that this command will set the 4 lowest outputs of the reader at once. In case you are only using some of the outputs for controlling a multiplexer and using other outputs for something else it might be better to switch the multiplexer using the **WOP** command.

Warning

In case of the DwarfG2, DwarfG2_XR, and DwarfG2_Mini the output pins can also be used as input pins (General Purpose Inputs / Outputs - GPIOs). When calling **SAP** the direction the pins is being used in is changed to being an output pin. Please make sure that the hardware connected to the pin will not exceed the pins' maximum limits in output mode before calling **SAP** as this can destroy the reader.

2.16.1. Manual mode (MAN)

Manual mode ist used to activate a specific antenna port. The supplied number is the antenna port number, with 0 being the first.

For legacy reasons, the `MAN` flag is optional. Not specifying `MAN` will result in the same result, but not using it is deprecated.

» `SAP MAN 1<CR>`

« `OK!<CR>`

Example 19. ***SAP*** *`MAN` command and answer*

Instruction

SAP `<SPACE> MAN <SPACE> {Antenna Port} <CR>`

Parameters

Name	Type	Description
MAN	Keyword	Manual mode
Antenna Port	Decimal Integer (0 ≤ x ≤ 15)	Number of antenna port to activate

2.16.2. Automatic Switching Mode (AUT)

In case you want to automatically switch between multiple antennas (e.g. trying to find all tags in a search area that can only be searched using multiple antennas) you can use this automatic switching mode.

The number of antennas to use has to be specified after AUT.

Note

Please note that for this parameter the number given is the counted number of participating antennas, not the antenna port numbers, thus stating a number "X" would stand for "X antennas participating".

Switching always starts with the lowest antenna port (0). Switching to the next antenna port occurs automatically with the start of every tag manipulation command. No pin state is changed until the first tag manipulation command.

The automatic mode can be stopped by using **SAP** `AUT OFF`. Setting **SAP** `AUT 0` will do the same as setting `AUT OFF`.

Note

SAP `AUT 1` will set the used antenna to only one, but with time-consuming overhead. If you want to address one antenna only, do not use this but **SAP** `MAN 0`, instead!

» `SAP AUT 0<CR>`

« `OK!<CR>`

» `SAP AUT OFF<CR>`

```
« OK!<CR>
```

```
» SAP AUT 4<CR>
```

```
« OK!<CR>
```

*Example 20. **SAP** AUT command and answer*

Instruction

SAP `<SPACE> AUT <SPACE> {No. Antennas} <CR>`

Parameters

Name	Type	Description
AUT	Keyword	Automatic switching mode
No. Antennas	Decimal Integer (0 ≤ x ≤ 16)	Number of antennas connected

2.16.3. Antenna Report Control (ARP)

ARP controls the antenna reporting. If enabled every tag manipulation command answer will write out the antenna port before giving the number of antennas found (`IVF XXX`), with "0" being the first antenna port. The format is a 2 digit decimal.

AUT provides the active antenna port value after tag command answers. For separately calling antenna information please refer to the Read Antenna Port (**RAP**) command.

```
» SAP AUT 4<CR>
```

```
« OK!<CR>
```

```
» SAP ARP ON<CR>
```

```
« OK!<CR>
```

```
» INV<CR>
```

```
« AABBCCDD<CR>
  ARP 00<CR>
  IVF 001<CR>
```

```
» INV<CR>
```

```
« AABBCCDD<CR>
  ARP 01<CR>
  IVF 001<CR>
```

```
» SAP ARP OFF<CR>
```

```
« OK!<CR>
```

*Example 21. **SAP** ARP command and answer*

Instruction

SAP *<SPACE>* ARP *<SPACE>* { ON | OFF } *<CR>*

Parameters

Name	Type	Description
ARP	Keyword	Antenna reporting
Enable/Disable	Enumeration (ON or OFF)	Enable or disable

Return Values in Case of Success

OK! *<CR>*

Return Values in Case of Failure

NOS *<CR>*

Not Supported for DeskID_UHF and UMG2

"EDX *<CR>*", "NOR *<CR>*", "BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.17. Read Antenna Port (RAP)

RAP will inform about the multiplexing state. It is only active in case the reader's lowest four output pins are set according to the **SAP** command. If one of the lowest 4 pins is configured as input the answer for **RAP** would be OFF.

If the pins are all outputs the corresponding values are given. These are :

- In case of manual mode: MAN {active antenna port}
- In case of automatic mode: AUT {currently active antenna port} {total count of antennas for switching}

All cases add ARP if the reporting is active.

Note

If **RAP** is called directly after using **SAP** AUT, the answered port number of the active antenna will be the total count of antennas for switching. Since port numbers start with "0", this number is higher than the highest possible port number. This is for signaling that the switching did not start yet.

»
```
RAP<CR>
```

«
```
OFF<CR>
```

*Example 22. **RAP** command and answer after **RIP** 0*

»
```
RAP<CR>
```

«
```
MAN 1<CR>
```

*Example 23. **RAP** command and answer after **SAP** MAN 1*

»
```
RAP<CR>
```

«
```
AUT 2 8 ARP<CR>
```

*Example 24. **RAP** command and answer after **SAP** AUT 8 and 3 times **INV***

»
```
RAP<CR>
```

«
```
AUT 5 5 ARP<CR>
```

*Example 25. **RAP** command and answer after **SAP** AUT 5: active antenna port "5" is higher than the possible "4" (switching did not start yet)*

Instruction

RAP *<CR>*

Return Values in Case of Success

MODE [<SPACE> {active antenna} [<SPACE> {number of antennas}]] [<SPACE> ARP]<CR>

Return Values in Case of Failure

"NOS *<CR>*", "BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.18. Read Reflected Power (RRP)

This command is used to read the reflected power value for the antenna connected. A well matched antenna that is connected correctly to the reader should not reflect much power back to the reader. The command returns a complex number as two decimal coded values (A-Channel and B-Channel), which correspond to the I and Q value of the reflected power. To use the command a radio standard needs to have been set (see **STD**). The value range of A and B is -127 to +127.

The reflected power Pin is computed as follows:

Pin (dBm) = 20*log(sqrt(A^2+B^2)/G) dBm

In this equation, G depends on the hardware of the reader. For the PulsarMX it is about 16, for the DeskID UHF and UMG2 about 6, for the DwarfG2_XR about XX and for the DwarfG2 about 3. The transmitted RF power level at which this measurement is taken is about 12 dBm.

For example, if the values returned by a PulsarMX are A=-36 and B=05, the reflected power is 7,1dBm. As we used a transmitted power level of 12 dBm and got 7.1 dBm reflected back from the antenna the reflectivity of the antenna is S11 = 7,1 dBm – 12 dBm= -4,9 dB. As this is a very poor value which would drastically limit the reading range the antenna, cables and connections should be checked. During initialization by the STD command this antenna check is performed automatically for the PulsarMX and DwarfG2_XR. If the value for the reflected power is above -6 dB (25% reflected power), the **ARH** alarm is triggered and RF is not turned on.

RRP is not supported by DwarfG2_Mini.

```
» RRP<CR>

« 20 30<CR>

» RRP<CR>

« +127 -127<CR>
```

*Example 26. **RRP** command and answer*

Instruction

RRP *<CR>*

Return Values in Case of Success

A <SPACE> B

Return Values in Case of Failure

"NOS *<CR>*", "NSS *<CR>*", "BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.19. Configuration (CFG)

For the advanced programmer, commands are provided for detailed configuration of the reader with regard to communication parameters. These options can be used to get the optimal device's performance for each type of tag, antenna, and general situation. The sum of all settings define a profile.

Warning

The configuration commands can seriously off-tune the reader-to-tag communication and are meant for advanced programmers only. Do not use these if you are not sure what you are doing.

These configuration options can be used to get the optimal device's performance for each type of tag, antenna, and general situation. The sum of all settings define a profile.

The usage of **CFG** depends on the device. For Pulsar_MX, DwarfG2, DwarfG2_XR, DeskID_UHF, and UMG2 the command and parameter are as described. They only differ in value range.

DwarfG2_Mini differs much more. Some parameters are not supported (you will get UPA) for example DHS. To get all supported parameters use PRP once (except TRC which is just reported). RXG supports AGC as value (which is also default) with a value range of 0 to 70.

Note

The main Gen2 settings are used as a set. Some of them are computed from the set values of the given parameters. Setting is possible for DR8, TAR, LKF and MIL. The TRC value is computed from these values and the RTcal and other internal values, too. If the set is not Gen2 compliant the device will answer NOS.

The value is saved but not used so you can change the other values needed for your setting. When a set is valid the answer is OK! and the setting are used from now on.

Up to eight profiles may be stored in the internal non volatile memory (EEPROM) of the reader. The profiles stored in EEPROM can be changed using the SPE (Save Profile to EEPROM) command. This saves the current profile. LPE loads them from EEPROM. Since the settings are always reset to default values after resetting the reader, using a startup command (**SUC**) is needed to automatically get the desired setting activated after reboot.

»

```
CFG PRP<CR>
```

«

```
OK!<CR>
DR8 ON<CR>
DR8 ON<CR>
MIL 3<CR>
TAR 1<CR>
LKF 6<CR>
TRC 500<CR>
RWT 65<CR>
RWL 65<CR>
PWR 11<CR>
RXG AGC<CR>
```

*Example 27. **CFG** PRP command and answer on DwarfG2_Mini*

»

```
CFG PRP<CR>
```

«
```
OK!<CR>
DMG OFF<CR>
MIA ON<CR>
SSU OFF<CR>
PAS OFF<CR>
DR8 OFF<CR>
MIL 3<CR>
TAR 1<CR>
LKF 12<CR>
TRC 667<CR>
NRW 15<CR>
RWT 0<CR>
LPF 4<CR>
HPF 0<CR>
PWR 27<CR>
DHS 0<CR>
RXG -0<CR>
```

*Example 28. **CFG** PRP command and answer on other devices*

»
```
CFG MIL 2<CR>
```

«
```
OK!<CR>
```

»
```
CFG DR8 ON<CR>
```

«
```
OK!<CR>
```

»
```
CFG DR8 ON<CR>
```

«
```
NOS<CR>
```

»
```
CFG RXG AGC<CR>
```

«
```
OK!<CR>
```

»
```
CFG RXG AGC<CR>
```

«
```
EDX<CR>
```

*Example 29. **CFG** setting command and answer*

2.19.1. Print Profile (PRP)

PRP will print the whole profile. The elements match the following subcommands. The values are given on the same base or format as described at parameter description.

Instruction

CFG *<SPACE>* PRP *<CR>*

Parameters

Name	Type	Description
PRP	Keyword	The first parameter is the PRP keyword

2.19.2. Differential Mixer Gain (DMG)

DMG is a hardware value

Instruction

```
CFG <SPACE> DMG <SPACE> { ON | OFF } <CR>
```

Parameters

Name	Type	Description
DMG	Keyword	The first parameter is the name of the variable to change
Value	Enumeration (ON or OFF)	The second parameter is the value. 10 dB Gain if ON

2.19.3. Mixer Input Attenuation (MIA)

MIA is a hardware value

Instruction

```
CFG <SPACE> MIA <SPACE> { ON | OFF } <CR>
```

Parameters

Name	Type	Description
MIA	Keyword	The first parameter is the name of the variable to change
Value	Enumeration (ON or OFF)	The second parameter is the value. 5dB Attenuation on DwarfG2, DeskID UHF and UMG2, 8 dB on PulsarMX and DwarfG2_XR when ON

2.19.4. RX Settling Speed Up (SSU)

SSU should make reading a bit faster for most tags

Instruction

```
CFG <SPACE> SSU <SPACE> { ON | OFF } <CR>
```

Parameters

Name	Type	Description
SSU	Keyword	The first parameter is the name of the variable to change

Name	Type	Description
Value	Enumeration (ON or OFF)	The second parameter is the value.

2.19.5. PR-ASK (PAS)

PR-ASK is an abbreviation for "Phase Reversal Amplitude shift keyed" - a coding method. If deactivated the default DSB-ASK (double sideband ASK) is used.

"PR-ASK is a modulation that can minimize the carrier to noise requirement in a narrowband environment while maximizing the power transport to the tag. This modulation has carrier to noise and bandwidth requirements more closely matching PSK than DSB-ASK, making it attractive for narrowband and longer-range applications. DSB-ASK is the least bandwidth efficient modulation, but the easiest to produce by On and Off Keying (OOK) of the carrier signal." (Source: http://www.eetimes.com/document.asp?doc_id=1276402)

Instruction

```
CFG <SPACE> PAS <SPACE> { ON | OFF } <CR>
```

Parameters

Name	Type	Description
PAS	Keyword	The first parameter is the name of the variable to change
Value	Enumeration (ON or OFF)	The second parameter is the value. Uses PR-ASK if ON, else DSB-ASK

2.19.6. Divide Ratio = 8 (DR8)

DR8 sets the DR (divide ratio) defined in the EPC Gen2 standard

Instruction

```
CFG <SPACE> DR8 <SPACE> { ON | OFF } <CR>
```

Parameters

Name	Type	Description
DR8	Keyword	The first parameter is the name of the variable to change
Value	Enumeration (ON or OFF)	The second parameter is the value. DR = 8 if ON, else DR = 64/3

2.19.7. RX Encoding (MIL)

MIL is short for miller encoding and sets the encoding of the tag answer.

MILLER8 is usually most reliable on tags supporting MILLER8 but Miller4 is much faster

Instruction

```
CFG <SPACE> MIL <SPACE> {Value} <CR>
```

Parameters

Name	Type	Description
MIL	Keyword	The first parameter is the name of the variable to change
Value	Decimal Integer (0 ≤ x ≤ 3)	The second parameter is the value 0: FMO, 1: MILLER2, 2: MILLER4, 3: MILLER8

2.19.8. Tari (TAR)

TAR is the Tari value defined in the EPC Gen2 standard.

Instruction

```
CFG <SPACE> TAR <SPACE> {Value} <CR>
```

Parameters

Name	Type	Description
TAR	Keyword	The first parameter is the name of the variable to change
Value	Decimal Integer (1 ≤ x ≤ 2)	The second parameter is the value. 1: Tari=12.5µs , 2: Tari=25µs

2.19.9. RX Link Frequency (LKF)

LKF is the link frequency defined in the EPC Gen2 standard. The following values can be set:

0: 40kHz, 6: 160kHz, 9: 256kHz, 12: 320kHz, 15: 640kHz

Instruction

```
CFG <SPACE> LKF <SPACE> { 0 | 6 | 9 | 12 | 15 } <CR>
```

Parameters

Name	Type	Description
LKF	Keyword	The first parameter is the name of the variable to change
Value	Enumeration (0, 6, 9, 12 or 15)	The second parameter is the value. 0: 40kHz, 6: 160kHz, 9: 256kHz, 12: 320kHz, 15: 640kHz

2.19.10. TRcal (TRC)

TRC is the TRcal value defined in the EPC Gen2 standard. Default is 66,7µs (value=667).

Instruction

```
CFG <SPACE> TRC <SPACE> {Value} <CR>
```

Parameters

Name	Type	Description
TRC	Keyword	The first parameter is the name of the variable to change
Value	Decimal Integer (0 ≤ x ≤ 4095)	The second parameter is the value to set in 0.1 µs

2.19.11. No Response Wait Time (NRW)

NRW is the time to wait before a tag is considered to be "not answering" so the next slot will be used. This is tag dependent and should be as low as possible for the tag(s) in use (low setting speeds up tag read rate).

Instruction

```
CFG <SPACE> NRW <SPACE> {Value} <CR>
```

Parameters

Name	Type	Description
NRW	Keyword	The first parameter is the name of the variable to change
Value	Decimal Integer (0 ≤ x ≤ 255)	The second parameter is the value to set in 25.6 µs steps

2.19.12. RX Wait Time (RWT)

RWT is the time to wait before the receiver is activated. Multiplier is 1µs for DwarfG2_Mini and 6.4 µs for every other device. On DwarfG2_Mini this is only used for fast answering commands, RWL for any other.

Instruction

```
CFG <SPACE> RWT <SPACE> {Value} <CR>
```

Parameters

Name	Type	Description
RWT	Keyword	The first parameter is the name of the variable to change
Value	Decimal Integer (0 ≤ x ≤ 255)	The second parameter is the value to set in 6.4 µs / 1µs steps

2.19.13. RX Wait Time Long (RWL)

Many tags need much more time to execute a WDT or other long commands than RWT provides. On DwarfG2_Mini, RWL is the time to wait before the receiver is activated like RWT above, but for commands with long answer time. These are **WDT**, **LCK** and **KIL**. The GEN2 commands used for these commands have a maximal answer time of 20ms instead of just some microseconds like all other commands. The answer performance for these commands may increase significantly with an RWL value close to the real answer time.

Note

Keep in mind: The answer time will often differ for **WDT**, **LCK**, **KIL**. It will also depend on the executed action. **WDT** will answer much faster if the tag supports no writing to the set area and only answers an error code. Therefore, change the value between commands.

RWL is only supported on the DwarfG2_Mini.

Instruction

```
CFG <SPACE> RWL <SPACE> {Value} <CR>
```

Parameters

Name	Type	Description
RWL	Keyword	The first parameter is the name of the variable to change
Value	Decimal Integer (0 ≤ x ≤ 20000)	The second parameter is the value to set in 1 µs steps

2.19.14. Low Pass Frequency (LPF)

Sets the low pass frequency. Ideal value depends on Link Frequency. Suggested values are 0 for 640kHz, 4 for 320 kHz (default), 6 for 256 kHz and 7 for 160kHz and 40kHz

Instruction

```
CFG <SPACE> LPF <SPACE> {Value} <CR>
```

Parameters

Name	Type	Description
LPF	Keyword	The first parameter is the name of the variable to change
Value	Decimal Integer (0 ≤ x ≤ 7)	The second parameter is the value to set.

2.19.15. High Pass Frequency (HPF)

Sets the low pass frequency. Ideal value depends on Link Frequency. For 320 kHz values from 0 to 4 work best for most tags

Instruction

```
CFG <SPACE> HPF <SPACE> {Value} <CR>
```

Parameters

Name	Type	Description
HPF	Keyword	The first parameter is the name of the variable to change
Value	Decimal Integer (0 ≤ x ≤ 7)	The second parameter is the value to set.

2.19.16. Transmitter Power (PWR)

PWR is the transmit power level given in dBm. It supports MIN or MAX also (see PWR enums below). The power level limits are device type dependent. The values given are examples. Use MIN / MAX to set and CFG PRP to get the real values of MIN or MAX.

Instruction

```
CFG <SPACE> PWR <SPACE> {Value} <CR>
```

Parameters

Name	Type	Description
PWR	Keyword	The first parameter is the name of the variable to change
Value	Decimal Integer (-2 ≤ x ≤ 27)	The second parameter is the value in dBm

2.19.17. Transmitter Power (PWR)

PWR sets the transmit power level. The enums can set the minimal and maximal values. PWR supports the power level as a decimal number, too (see PWR decimal above). The power level limits are device type dependent.

Instruction

```
CFG <SPACE> PWR <SPACE> { MIN | MAX } <CR>
```

Parameters

Name	Type	Description
PWR	Keyword	The first parameter is the name of the variable to change
Value	Enumeration (MIN or MAX)	The second parameter sets the value to MIN or MAX

2.19.18. Digitizer Hysteresis Setting (DHS)

DHS sets the hysteresis value of the digitizer.

Instruction

```
CFG <SPACE> DHS <SPACE> {Value} <CR>
```

Parameters

Name	Type	Description
DHS	Keyword	The first parameter is the name of the variable to change
Value	Decimal Integer (0 ≤ x ≤ 7)	The second parameter is the value to set in 3dB steps

2.19.19. Receiver(RX) Gain (RXG)

RXG is the amplifier gain setting for the signal received. Too high settings add too much noise, too low settings reduce reading range as the signal is too weak.

Instruction

```
CFG <SPACE> RXG <SPACE> {Value} <CR>
```

Parameters

Name	Type	Description
RXG	Keyword	The first parameter is the name of the variable to change
Value	Decimal Integer (-3 ≤ x ≤ 3)	The second parameter is the value to set in 3dB steps

2.19.20. Save Profile to EEPROM (SPE)

SPE will save the current settings to the profile number given in non volatile memory (EEPROM). It can be reloaded by using **CFG** LPE .

Instruction

```
CFG <SPACE> SPE <SPACE> {Value} <CR>
```

Parameters

Name	Type	Description
SPE	Keyword	The first parameter activates saving
Value	Decimal Integer (0 ≤ x ≤ 7)	The second parameter is the profile number.

2.19.21. Load Profile from EEPROM (LPE)

LPE will override current settings with the settings in the give profile number in non volatile memory. If the profile you are trying to load has not been saved before, the values loaded would be undefined and an error is cast.

Instruction

```
CFG <SPACE> LPE <SPACE> {Value} <CR>
```

Parameters

Name	Type	Description
LPE	Keyword	The first parameter activates loading
Value	Decimal Integer (0 ≤ x ≤ 7)	The second parameter is the profile number.

Return Values in Case of Success

Answer *<CR>*

All **CFG** commands answer with

OK!

except for **CFG**PRP witch gives the current values of the parameters. **CFG**PRP always starts with OK! followed by the parameters always consisting of the parameter short form and the value like NRW 15.

Return Values in Case of Failure

"NOR *<CR>*", "EDX *<CR>*", "EHX *<CR>*", "NOS *<CR>*", "BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>*{*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.20. Set Parameter (SET)

Using the **SET** command, the reader can be configured to give additional information. The **SET** command also allows the configuration of device behaviour like masking or setting passwords. All settings are set back to default upon reset of the reader.

2.20.1. Set Masking Paramters (MSK)

Most tag manipulation commands can be limited to a population of tags with certain data values, e.g. tags that start with a certain Electronic Product Code (EPC), a certain Tag ID (TID) or even contain certain data in the user memory (USR). This is done via a mask. Using this feature you can interact with certain tags in the field by directly addressing each tag via its TID or EPC.

This is especially handy if you only want to read from or write to tags of a certain product type (coded in the EPC).

To use this feature you have to tell the reader by which data field you want to filter and what the starting address should be to compare your mask. The mask is always given in full bytes. By default the mask is not set, so all tags will answer to your commands.

The EPC itself starts at address 0x20 (in bit), not zero. The parts before 0x20 should not be used for masking.

To stop any masking use **SET** MSK OFF as described below.

»
```
SET MSK EPC AABBCCDD 20<CR>
```

```
« OK!<CR>
```

```
» SET MSK TID 1122<CR>
```

```
« OK!<CR>
```

```
» SET MSK USR A0 0 4<CR>
```

```
« OK!<CR>
```

*Example 30. **SET** MSK command and answer*

Instruction

```
SET <SPACE> MSK <SPACE> { EPC | TID | USR } <SPACE> {Mask} [<SPACE>
{Mask offset} [<SPACE> {Mask length}]] <CR>
```

Parameters

Name	Type	Description
MSK	Keyword	The first parameter defines that you are setting a mask
Memory bank	Enumeration (EPC, TID or USR)	Sets the memorybank to mask (or the flag to disable masking, see MSK OFF.
Mask	Hexadecimal String (between 1 and 31 bytes)	The mask value is a hexadecimal bytestream of up to 31 bytes. It may not be used with the OFF flag but is required for the EPC, TID and USR flags.
Mask offset	Optional Hexadecimal Integer (x $\geq 0_{16}$)	Start Address in bit (hex encoded, unsigned32). Default = 0
Mask length	Optional Hexadecimal Integer ($0_{16} \leq x \leq F8_{16}$)	Optional: Length in Bit (hex coded. Maximum is 31 Byte=0xF8 Bits). Default = Length of Mask Value (full Nibbles). If a Length is given the [Length] first bits will be used. This is only useful if you want to use a mask that is not a full word length.

2.20.2. Disable Masking (MSK)

If no masking is needed the masking can be disabled by sending **SET** MSK OFF. The masking is disabled for default. It is enabled by using a **SET**MSK command.

```
» SET MSK OFF<CR>
```

```
« OK!<CR>
```

*Example 31. **SET** MSK OFF command and answer*

Instruction

```
SET <SPACE> MSK <SPACE> OFF <CR>
```

Parameters

Name	Type	Description
MSK	Keyword	The first parameter defines that you are setting a mask
OFF	Keyword	Sets the flag to disable masking

2.20.3. Set Access Password (ACP)

Use this parameter to set the access password to use.

```
» SET ACP ACCEC0DE<CR>
```

```
« OK!<CR>
```

*Example 32. **SET** ACP command and answer*

Instruction

```
SET <SPACE> ACP <SPACE> {Access code} <CR>
```

Parameters

Name	Type	Description
ACP	Keyword	Sets Access Password in RAM to be used for authenticated access. **Note** Using an access password causes the possibility to find tags more than once in one round, with the maximum number of multiple detection depending on the inventory retry value **SIR**). In addition to the obvious reason of communication errors, this will also happen with high probability if using a wrong password.
Access code	Hexadecimal String (4 bytes)	Value of 32bit Access code

2.20.4. Disable Access Password (ACP)

If the access password is set to OFF all tags with an access password other than zero will stay in the open state (instead of the secured state) so some commands might not work. For further details on this topic please refer to the EPC Gen 2 Protocol Description and the lock command.

```
» SET ACP OFF<CR>
```

```
« OK!<CR>
```

*Example 33. **SET** ACP command and answer*

Instruction

```
SET <SPACE> ACP <SPACE> { OFF } <CR>
```

Parameters

Name	Type	Description
ACP	Keyword	Resets Access Password in RAM
Off code	Enumeration (OFF)	Disable Access Password

2.20.5. Access Password Save (APS)

Use this command to save an access password to non volatile memory in the reader

```
» SET APS ACCECODE 0<CR>
```

```
« OK!<CR>
```

*Example 34. **SET** APS command and answer*

SET APS is not supported by DwarfG2_Mini.

Instruction

```
SET <SPACE> APS <SPACE> {Access code} <SPACE> {Slot} <CR>
```

Parameters

Name	Type	Description
APS	Keyword	Writes Access Password to non volatile memory for later use (so you do not have to transmit it over an unsafe line later). There are 8 slots to save access passwords to. Together with **SET** APL it replaces **SET** ACP.
Access code	Hexadecimal String (4 bytes)	Value of 32bit Access code
Slot	Decimal Integer ($0 \leq x \leq 7$)	Slot to save the data to

2.20.6. Access Password Load (APL)

Use this command to load a saved access password from a non volatile slot. The EEPROM load command is also useful for higher security as the password is not send via an unsafe line. There are 8 slots to load access passwords from.

Note

Using an access password causes the possibility to find tags more than once in one round, with the maximum number of multiple detection depending on the inventory retry value **SIR**). In addition to the obvious reason of communication errors, this will also happen with high probability if using a wrong password.

SET APL is not supported by DwarfG2_Mini.

```
» SET APL 1<CR>
```

```
« OK!<CR>
```

*Example 35. **SET** APL command and answer*

Instruction

SET *<SPACE>* APL *<SPACE>* {*Slot*} *<CR>*

Parameters

Name	Type	Description
APL	Keyword	Sets Access Password by loading it from EEPROM to RAM
Slot	Decimal Integer (0 ≤ x ≤ 7)	Slot to load the data from

2.20.7. Set Kill Password (KLP)

Use this parameters to set the kill password. For further details on this topic please refer to the EPC Gen 2 Protocol Description and the kill command. The default kill password is `00000000`

```
» SET KLP 00000000<CR>
```

```
« OK!<CR>
```

```
» SET KLP DEADC0DE<CR>
```

```
« OK!<CR>
```

*Example 36. **SET** KLP command and answer*

Instruction

SET *<SPACE>* KLP *<SPACE>* {*Kill password*} *<CR>*

Parameters

Name	Type	Description
KLP	Keyword	Sets Kill Password in RAM
Kill password	Hexadecimal String (4 bytes)	Value of 32bit kill password

2.20.8. Kill Password Save (KPS)

Use this command to save a kill password to a non volatile memory slot in the reader. There are 8 slots to save kill passwords to.

```
» SET KPS DEADC0DE 0<CR>
```

```
« OK!<CR>
```

*Example 37. **SET** KPS command and answer*

SET KPS is not supported by DwarfG2_Mini.

Instruction

```
SET <SPACE> KPS <SPACE> {Kill password} <SPACE> {Slot} <CR>
```

Parameters

Name	Type	Description
KPS	Keyword	Writes kill password in EEPROM of reader for later use (so you do not have to transmit it over an unsecure line later)
Kill password	Hexadecimal String (4 bytes)	Value of 32bit kill password
Slot	Decimal Integer (0 ≤ x ≤ 7)	Slot to save the data to

2.20.9. Access Password Load (KPL)

Use this command to load a saved kill password from a non volatile memory slot of the reader. The EEPROM load command is also useful for higher security as the password is not send via an unsafe line. There are 8 slots to load kill passwords from.

SET KPL is not supported by DwarfG2_Mini.

```
» SET KPL 0<CR>
```

```
« OK!<CR>
```

*Example 38. **SET** KPL command and answer*

Instruction

```
SET <SPACE> KPL <SPACE> {Slot} <CR>
```

Parameters

Name	Type	Description
KPL	Keyword	Sets kill password by loading it from EEPROM to RAM
Slot	Decimal Integer (0 ≤ x ≤ 7)	Slot to load the data from

2.20.10. Set Additional EPC Sending Mode (EPC)

With this parameter the Electronic Product Code (EPC) of the responding tag will be added to every answer to a tag manipulation command. In case of active **SET** TRS, the EPC will be displayed before the RSSI value.

This makes it easier to identify which tag is actually responding to a command like e.g. Write Data (**WDT**) or Read Data (**RDT**).

»
```
SET EPC ON<CR>
```

«
```
OK!<CR>
```

*Example 39. **SET** EPC command and answer*

Instruction

```
SET <SPACE> EPC <SPACE> { ON | OFF } <CR>
```

Parameters

Name	Type	Description
EPC	Keyword	Parameter to set the EPC mode
Mode	Enumeration (ON or OFF)	Enable or disable addition of EPC to tag answer

2.20.11. Set Global TimeOut (GTO)

Any tag manipulation command starts a global timeout timer. If the command's function does not terminate – either successfully or by returning an error code – the function will be killed and the TOE error code will be printed. **SET** GTO changes the time value until timeout. It is given in decimal notation in milliseconds.

»
```
SET GTO 2000<CR>
```

«
```
OK!<CR>
```

*Example 40. **SET** GTO command and answer*

Instruction

```
SET <SPACE> GTO <SPACE> {Timeout} <CR>
```

Parameters

Name	Type	Description
GTO	Keyword	Parameter to set the timeout value
Timeout	Decimal Integer ($0 \leq x \leq 4000$)	Time value until timeout in milliseconds

2.20.12. Show Transponder Receive Strength (TRS)

Sometimes you want to know the received signal strength when communicating with a transponder. With the TRS setting the reader will automatically add the RSSI value to responses from a tag to any tag manipulation command. The value is always negative in a range from

-25 to -70 with -25 being the best case (strongest signal). The value will be printed in a new line following the answer of the tag operation for each tag and the EPC if "**SET** `EPC ON`" is set.

This command is not supported by DwarfG2_Mini.

»
```
SET TRS ON<CR>
```

«
```
OK!<CR>
```

*Example 41. **SET** TRS command and answer*

Instruction

SET *<SPACE>* `TRS` *<SPACE>* `{ ON | OFF }` *<CR>*

Parameters

Name	Type	Description
TRS	Keyword	Parameter to set TRS reporting
State	Enumeration (`ON` or `OFF`)	Enable or disable addition of RSSI value to tag answer

2.20.13. Turn High On Tag Mode Off (`HOT`)

This parameter is only usable on DwarfG2, DwarfG2_Mini, DwarfG2_XR and PulsarMX (DeskID and UMG2 have no IOs). This command disables the `HOT` mode (see below).

»
```
SET HOT OFF<CR>
```

«
```
OK!<CR>
```

*Example 42. **SET** `HOT OFF` command and answer*

Instruction

SET *<SPACE>* `HOT` *<SPACE>* `{ OFF }` *<CR>*

Parameters

Name	Type	Description
HOT	Keyword	Parameter set the `HOT` mode
State	Enumeration (`OFF`)	Disable HighOnTag mode

2.20.14. Set High On Tag Mode (`HOT`)

This parameter is only usable on readers that have output pins. It makes the reader set an output pin to high state and reset it to low after a tag manipulation command finds a tag (i.e. number of tags found `IVF XXX` with XXX >0).

Note

Please note that a correct error message from the tag will also initiate this response. Also note, that in case of a write request it does not show a successful write but just that a tag was found.

The default output pin being temporarily set to high state is GPIO 7 for the DwarfG2 and DwarfG2_XR, GPIO 3 for DwarfG2_Mini, and output 0 for the PulsarMX. Any output pin can be used by using the pin usage mask. Also, more than one pin can be used to power more than one device (like for example an LED and a buzzer).

The time set is in ms. Any command is prolonged by the high and low time so no overlapping is possible.

Warning

In case of the DwarfG2, DwarfG2_XR, and DwarfG2_Mini the output pins can also be used as input pins (General Purpose Inputs / Outputs - GPIOs). When calling **SET** HOT and every time a tag is found the direction the pin is being used in is changed to being an output pin. Please make sure that the hardware connected to the pin is compatible to the pin's maximum limits in output mode before calling **SET** HOT as this can otherwise destroy the reader or connected hardware.

```
» SET HOT 100 50<CR>
```

```
« OK!<CR>
```

```
» SET HOT 100 50 18<CR>
```

```
« OK!<CR>
```

*Example 43. **SET** HOT command and answer*

Instruction

SET *<SPACE>* HOT *<SPACE>* {*High time value in ms*} *<SPACE>* {*Low time value in ms*} [*<SPACE>* {*Output mask*}] *<CR>*

Parameters

Name	Type	Description
HOT	Keyword	Parameter to set the high-on-tag behaviour
High time value in ms	Decimal Integer ($0 \le x \le 255$)	
Low time value in ms	Decimal Integer ($0 \le x \le 255$)	
Output mask	Optional Hexadecimal Integer ($0_{16} \le x \le FF_{16}$)	The mask is hexadecimal and codes which outputs are set after finding the tag(s). "0" means no one is used so it's identical to SET HOT OFF. "1" is bit 0, "2" is bit 1, "4" is bit 2, "80" is bit 7. It is possible to use more than one at the same time. "FF" means all GPIOs (on

Name	Type	Description
		DwarfG2(_XR)), for the PulsarMX and DwarfG2_Mini F is the max value (all 4 outputs). "A1" are bits 0,5,7 used.

2.20.15. Set Input High Commands (`IHC`)

This parameter is only supported by DwarfG2, DwarfG2_XR, and PulsarMX. It allows defining a set of commands that is to be executed on a falling edge of an input pin - e.g. when a light barrier triggers. Supported input pins are 0 and 1 (Input 0/1 on PulsarMX, GPIO 0/1 on DwarfG2 and DwarfG2_XR) and for each pin a different set of commands can be saved. Besides the pin that is being monitored, the command expects a second parameter. This can either be a flag (turning the behaviour `ON`, `OFF` or with `SHW` showing the command set being used, see below) or it expects the command set to be used with individual commands separated by ';'. The ';'s are replaced by *<CR>* on execution.

For more info please refer to the description of (**`SET`**`IHC` flags).

Warning

In case of the DwarfG2 and DwarfG2_XR the input pins can also be used as output pins (General Purpose Inputs / Outputs - GPIOs). When calling **`SET`** `IHC` the direction the pin is being used in is changed to being an input pin. Please make sure that the hardware connected to the pin is compatible to the pin's maximum limits in input mode before calling **`SET`** `IHC` as this can otherwise destroy the reader or connected hardware.

Note

Please also note that using **`WOP`** on the pin sets the pin to output mode and deactivates `IHC` mode.

`SET` `IHC` is not supported by DwarfG2_Mini.

```
» SET IHC 0 RST<CR>
« OK!<CR>
» SET IHC 1 STD ETS;SRI ON<CR>
« OK!<CR>
» SET IHC 0 ON<CR>
« OK!<CR>
» SET IHC 1 OFF<CR>
« OK!<CR>
» SET IHC 1 SHW<CR>
```

«
```
STD ETS<CR>SRI ON<CR>
```

»
```
SET IHC<CR>
```

«
```
OK!<CR>
```

*Example 44. **SET** `IHC` command and answer*

Instruction

SET *<SPACE>* `IHC` *<SPACE>* `{Pin}` *<SPACE>* `{…Commands…}` *<CR>*

Parameters

Name	Type	Description
IHC	Keyword	IHC flag
Pin	Decimal Integer (0 ≤ x ≤ 1)	Pin to use
Commands	Any String	Sets commands to execute on falling edge. All commands are allowed. The command sequences are stored in non volatile memory.

2.20.16. Configure Input High Commands (IHC)

This parameter is only supported by DwarfG2, DwarfG2_XR, and PulsarMX. It allows defining a set of commands that is to be executed on a falling edge of an input pin. For more info please refer to the description for Set Input High ((**IHC**)) commands. The `ON`/`OFF` flag allows turning this input response behaviour on and off. The `ON`/`OFF` flag is reset to `OFF` when the reader is reset - if this is not desired please use a startup command (**SUC**).

`SHW` allows reading the commands set for the pin.

SET `IHC` is not supported by DwarfG2_Mini.

Instruction

SET *<SPACE>* `IHC` *<SPACE>* `{Pin}` *<SPACE>* `{ ON | OFF | SHW }` *<CR>*

Parameters

Name	Type	Description
IHC	Keyword	IHC flag
Pin	Decimal Integer (0 ≤ x ≤ 1)	Pin to use / show
Operation	Enumeration (`ON`, `OFF` or `SHW`)	The SHW flag shows the commands saved for that pin. It uses the *<CR>* during reporting, not ';' in case of multiple commands.

Return Values in Case of Success

`OK!` *<CR>*

Return Values in Case of Failure

"EDX *<CR>*", "WDL *<CR>*", "NOR *<CR>*", "NOS *<CR>*", "BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>*{*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.21. Set Q Value (SQV)

The **SQV** command is used to change the Q value used by the reader. The Q value defines the number of slots the reader uses for the anticollision sequence during every tag manipulation command. It is useful in cases where the maximum number of expected tags is known, as lower Q values mean less time spent in every round and therefore speed up the search.

The Q value is given as decimal number. The number of slots is 2^Q, so Q=0 would be used for only one tag (see also **INV** SSL, which will set Q=0 temporarily). For 2 tags Q=2 will be fine (in general: the number of channels should be much higher (about two times higher) than the expected number of tags. The maximum value is 15.

Tag manipulation commands implement the SSL parameter which will set the Q-value =0 temporarily.

Note

Using Q=15 will result most probably in a timeout error if timeout is not adjusted (see **SET** GTO command).

The default value is four (16 Slots) for DeskID_UHF, UMG2, DwarfG2_Mini and DwarfG2 and 6 (64 slots) for Pulsar_MX and DwarfG2_XR, which is fine for up to 8 tags or 50 tags, respectively.

```
» SQV 5<CR>
```

```
« OK!<CR>
```

*Example 45. **SQV** command and answer*

Instruction

SQV *<SPACE>* {*Q Value*} *<CR>*

Parameters

Name	Type
Q Value	Decimal Integer (0 ≤ x ≤ 15)

Examples

```
SQV 5<CR>
```

Example 46. Set Q to 5 (equals 2^5 = 32 slots)

Return Values in Case of Success

OK! *<CR>*

Return Values in Case of Failure

"NOR *<CR>*", "EDX *<CR>*", "BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.22. Read Q Value (RQV)

The **RQV** command gets the Q value set by **SQV** command as two-digit decimal number. For more information see the Set Q Value (**SQV**) command.

» RQV*<CR>*

« 5*<CR>*

*Example 47. **RQV** command and answer*

Instruction

RQV *<CR>*

Return Values in Case of Success

Q-Value

Return Values in Case of Failure

"BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.23. Set Inventory Retry (SIR)

Sets the maximum number of retries in the tag searching algorithm as specified in the EPC Gen2.

Depending on the number of tags in the field and the number of slots used (defined by the Q value) there is the chance of a tag being undetected although it is in detectable state (i.e. arbitrate according to EPC Gen2). This could be due to tag collision or a corrupted tag detection communication sequence. In a new round the tag might be detected correctly.

The Inventory Retry (IR) value defines how often the reader will start a new round if an undetected tag is expected. It will not be started if there is no sign of an undetected tag at all. For every repeated round the Q value is adapted internally for this inventory round only, independent on the value set via **SQV**.

The default value is 2 for DwarfG2, DwarfG2_Mini, UMG2, and DeskID_UHF. For Pulsar_MX and DwarfG2_XR it is 3.

Tag manipulation commands implement the SSL parameter which will set the IR value =0 temporarily.

»
```
SIR 1<CR>
```

«
```
OK!<CR>
```

*Example 48. **SIR** command and answer*

Instruction

SIR *<SPACE>* {*IR-Value*} *<CR>*

Parameters

Name	Type
IR-Value	Decimal Integer (0 ≤ x ≤ 10)

Return Values in Case of Success

OK! *<CR>*

Return Values in Case of Failure

"NOR *<CR>*", "EDX *<CR>*", "BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.24. Read Inventory Retry Value (RIR)

The **RIR** command returns the Inventory Retry (IR) value set by the Set Inventory Retry (**SIR**) command as a two-digit decimal number. For more information look at the **SIR** command.

»
```
RIR<CR>
```

«
```
03<CR>
```

*Example 49. **RIR** command and answer*

Instruction

RIR *<CR>*

Return Values in Case of Success

IR value

Return Values in Case of Failure

"BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.25. Cyclic Redundancy Check On (CON)

The **CON** command turns on the Cyclic Redundancy Check (CRC) of the host to reader communication. This can be used to detect transmission errors between the reader and the host (computer, embedded PC, microcontroller, etc.). In general enabling this feature is not necessary except in scenarios where you have lots of noise on the communication bus (e.g. when using USB communication in the vicinity of electric motors) or if you encounter any other problems with communication errors.

Note

This has nothing to do with the reader to tag communication CRC.

If this feature is activated (default is off), the reader firmware expects a CRC16 (4 digit hexadecimal number) between each command to the reader and the respective *<CR>*. Between the command and the CRC there is a space character which is included into the CRC calculation. All answers from the reader will also be extended accordingly. The CRC calculation uses the 0x8408 polynomial, starting value is 0xFFFF. The **CON** command itself will work with or without the (optional) CRC (819E).

If successful the command returns `OK!` plus the according CRC of "OK! ".

Note

Please keep in mind: In case of errors that cause the reader to reset you get an error code after the reset that is without the CRC because of the reset.

The Appendix includes a function in C, C# and Java to calculate the correct CRC16.

```
» CON<CR>
« OK! 9356<CR>
» CON<CR>
« CCE C095<CR>
» CON 819E<CR>
```

```
« OK! 9356<CR>
```

*Example 50. **CON** command and answer. Once the CRC of host to reader communication is active, the CRC is expected for all commands (including CON)*

Instruction

`CON` `<CR>`

Return Values in Case of Success

`OK! 9356 <CR>`

Return Values in Case of Failure

"`BOD <CR>`", "`BOF <CR>`", "`CCE <CR>`", "`CRT <CR>`", "`PFE <CR>`", "`PLE <CR>`", "`SRT <CR>`", "`UCO <CR>`", "`UER[<SPACE> {Two Digit Hex Code}] <CR>`", "`UPA <CR>`" or "`URE <CR>`"

2.26. Cyclic Redundancy Check Off (COF)

The **COF** command turns off the Cyclic Redundancy Check (CRC) of the host-to-reader communication. This is the default setting. The command will only work in CRC mode if the correct CRC (4F5E) is added. The **COF** command is accepted with or without the CRC even when CRC mode is not active.

The command returns `OK!` without a CRC.

```
» COF<CR>
« CCE C095<CR>
» COF 4F5E<CR>
« OK!<CR>
» COF<CR>
« OK!<CR>
» COF 4F5E<CR>
« OK!<CR>
```

*Example 51. **COF** command and answer (starting with CRC active)*

Instruction

`COF` `[<SPACE> {CRC}] <CR>`

Parameters

Name	Type
CRC	Optional Hexadecimal Integer (x = $4F5E_{16}$)

Return Values in Case of Success

`OK! <CR>`

Return Values in Case of Failure

"BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.27. End Of Frame Mode (EOF)

The **EOF** command turns on the End of Frame Delimiter (EOF). This means that after every complete message (frame) the last *<CR>* will be followed by an additional *<LF>* (0x0A). This allows the user to build a simpler parser since it is clear when not to expect any further message from the reader.

In case the command being executed was called using the **CNR** (Continuous Retry) mode for repetitive / continuous execution every complete answer of a single iteration will be appended with an additional line feed.

Note

Please keep in mind that asynchronous errors that reset the reader will lead to the error code being reported after the reader has been reset (and the EOF delimiter deactivated as is the default setting). Thus the error code will not be terminated by the line feed.

»
```
EOF<CR>
```

«
```
OK!<CR>
<LF>
```

»
```
INV<CR>
```

«
```
AABBCCDD<CR>
ABCD1234<CR>
IVF 002<CR>
<LF>
```

*Example 52. **EOF** command and answer*

Instruction

EOF *<CR>*

Return Values in Case of Success

OK! *<CR>*

Return Values in Case of Failure

“BOD *<CR>*”, “BOF *<CR>*”, “CCE *<CR>*”, “CRT *<CR>*”, “PFE *<CR>*”, “PLE *<CR>*”, “SRT *<CR>*”, “UCO *<CR>*”, “UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*”, “UPA *<CR>*” or “URE *<CR>*”

2.28. No End of Frame Mode (NEF)

The **NEF** command will turn off the End of Frame **EOF** delimiter. The answer to the command will already not include it.

```
» NEF<CR>
```

```
« OK!<CR>
```

*Example 53. **NEF** command and answer*

Instruction

NEF *<CR>*

Return Values in Case of Success

OK! *<CR>*

Return Values in Case of Failure

“BOD *<CR>*”, “BOF *<CR>*”, “CCE *<CR>*”, “CRT *<CR>*”, “PFE *<CR>*”, “PLE *<CR>*”, “SRT *<CR>*”, “UCO *<CR>*”, “UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*”, “UPA *<CR>*” or “URE *<CR>*”

2.29. Verbosity Level (VBL)

This command allows the user to adjust the amount of communication coming from the reader. The default value of the verbosity level is 1. For 0 the information received is reduced, for 2 it is increased. The verbosity level has no influence on shown parsing and hardware errors.

The different possible levels are:

0: Any Tag answer except the correct answer is suppressed. No tag manipulation error codes are given. The number of tags (`IVF XXX`) is not sent.

1: In case of the default setting of one the answers correspond to what is shown in this documentation. Secured tag communication errors are shown.

2: All default values, and additionally all tag communication answers are displayed. This includes `RXE` and `CER` normally indicating a collision or `TNR` (Tag Not Responding, indication that no tag in this slot was detected) for all slots in every round. Also, the answers to NAK and Select as specified in the EPC Gen2 are printed. For high Q values or high IR (Inventory Retry) values the number of answers might be more than 1000 for a single tag manipulation command.

Note

Due to the possible high numbers of answers, **VBL** = 2 should only be used for debugging purposes.

Also note, that some error codes printed for **VBL** = 2 might not be mentioned in the command description of this guide.

»
```
VBL 0<CR>
```

«
```
OK!<CR>
```

*Example 54. **VBL** command and answer*

Instruction

VBL *<SPACE>* {*Mode*} *<CR>*

Parameters

Name	Type
Mode	Decimal Integer (0 ≤ x ≤ 2)

Return Values in Case of Success

`OK!` *<CR>*

Return Values in Case of Failure

"`EDX` *<CR>*", "`NOR` *<CR>*", "`BOD` *<CR>*", "`BOF` *<CR>*", "`CCE` *<CR>*", "`CRT` *<CR>*", "`PFE` *<CR>*", "`PLE` *<CR>*", "`SRT` *<CR>*", "`UCO` *<CR>*", "`UER`[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "`UPA` *<CR>*" or "`URE` *<CR>*"

2.30. Start Up Commands (SUC)

Any time the reader is reset - either by using the **RST** command or by toggling the power - the reader will set all parameters to their default settings. In case you want your setup to be retained after a reader reset, the **SUC** command allows setting up a set of commands to be executed at start up of the device, e.g. to (re-)apply any settings you want automatically. These commands are persistently stored in EEPROM or FLASH memory. Upon start up of the device the commands are loaded and executed as if they had been sent to the reader at that time. The only difference is that the responses to the commands are suppressed.

Note

Please note that any error messages are suppressed, too. Therefore, errors in the commands normally causing UCO or UPA are not detected. Due to this, please check the commands for correct spelling!

Multiple commands are separated by ";" (semicolon). **RST** is not accepted. If the **CNR** command is used the command will be executed continuously and will show results (not for the first execution, but afterwards). As usual, continuous execution of commands can be terminated by the **BRK** command.

SUC OFF disables the execution of the startup command sequence. **SUC** ON reenables the start up command sequence using the previously set command sequence. Setting a new command will set **SUC** to enabled, setting an empty **SUC** command (**SUC**) will disable and delete any Start Up commands.

```
» SUC STD ETS;SRI ON<CR>

« OK!<CR>

» SUC OFF<CR>

« OK!<CR>

» SUC ON<CR>

« OK!<CR>
```

*Example 55. **SUC** command and answer*

2.30.1. Commands

Instruction

SUC *<SPACE>* {…} *<CR>*

2.30.2. Enable (ON)

Instruction

SUC *<SPACE>* ON *<CR>*

2.30.3. Disable (OFF)

Instruction

SUC *<SPACE>* OFF *<CR>*

Return Values in Case of Success

`OK!` ***<CR>***

Return Values in Case of Failure

"`WMO` ***<CR>***", "`BOD` ***<CR>***", "`BOF` ***<CR>***", "`CCE` ***<CR>***", "`CRT` ***<CR>***", "`PFE` ***<CR>***", "`PLE` ***<CR>***", "`SRT` ***<CR>***", "`UCO` ***<CR>***", "`UER`[***<SPACE>*** {*Two Digit Hex Code*}] ***<CR>***", "`UPA` ***<CR>***" or "`URE` ***<CR>***"

2.31. Read Start Up Commands (RSC)

This command will return the sequence of startup commands set via **SUC**. The answer to the command is a first line stating whether **SUC** mode is turned on or off and then one command of the command sequence is reported per line instead of using the formatting with ';' that was used when setting the command sequence. As a last line of the answer the command returns `OK!`.

»
```
RSC<CR>
```

«
```
ON<CR>
STD ETS<CR>
SRI ON<CR>
OK!<CR>
```

»
```
RSC<CR>
```

«
```
OFF<CR>
OK!<CR>
```

»
```
SUC ON<CR>
```

«
```
OK!<CR>
```

*Example 56. **RSC** command and answer*

Instruction

RSC ***<CR>***

Return Values in Case of Success

`OK!` ***<CR>***

Return Values in Case of Failure

"`BOD` ***<CR>***", "`BOF` ***<CR>***", "`CCE` ***<CR>***", "`CRT` ***<CR>***", "`PFE` ***<CR>***", "`PLE` ***<CR>***", "`SRT` ***<CR>***", "`UCO` ***<CR>***", "`UER`[***<SPACE>*** {*Two Digit Hex Code*}] ***<CR>***", "`UPA` ***<CR>***" or "`URE` ***<CR>***"

2.32. Get Core Temperature (GCT)

Gets the temperature of the IC core. Only supported on DwarfG2_Mini.

```
» GCT<CR>

« 35<CR>

» GTC<CR>

« 40<CR>
```

Example 57. ***GCT*** *command and answer*

Instruction

GCT *<CR>*

Return Values in Case of Success

Temperature in °C

Return Values in Case of Failure

"NOS <CR>", "BOD <CR>", "BOF <CR>", "CCE <CR>", "CRT <CR>", "PFE <CR>", "PLE <CR>", "SRT <CR>", "UCO <CR>", "UER[<SPACE> {*Two Digit Hex Code*}] <CR>", "UPA <CR>" or "URE <CR>"

2.33. Continuous Retry (CNR)

CNR allows executing a tag manipulation command indefinitely until a break condition is matched. As the commands do not need to be parsed multiple times and since the host communication time is saved this is a faster way to get the data you need.

Commands repeated by **CNR** will use the mask, the Q value and IR (Inventory Retry) value set every time they are executed. The respective command will be repeated indefinitely or until either the **BRK** command is sent, the **RST** command is sent or, with BAR appended, until a tag is found. All other commands are suppressed while in **CNR** mode. This is a very powerful mechanism for unassisted operations where the reader is initialized at the beginning (e.g. via **SUC**) and then repeats the command over and over. Examples for useful continuous operations are reading tag EPCs, reading data from tags or even writing and locking data on tags continuously, e.g. in a printer.

In some situations the reader is set to read continuously, expecting only rare reading events (e.g. an access application via tag). In the case of finding a tag, however, some operation has to be performed with the tag so that the continuous operation has to be interrupted. This is where the BAR postfix comes in. Any command that uses the **CNR** command to enter continuous scanning mode can be automatically ended once a tag is found using this postfix.

Note

In case the command used generates several answer lines (as e.g. **INV** that generates a whole inventory of tag EPCs with each command execution) you can actually get several answers even though **CNR** mode is terminated. Basically, the last command is completely executed.

»

```
CNR BAR INV<CR>
```

«

```
AABBCCDD<CR>
IVF 001<CR>
```

»

```
CNR INV ONT<CR>
```

«

```
AABBCCDD<CR>
IVF 001<CR>
IVF 000<CR>
IVF 000<CR>
...
```

*Example 58. **CNR** command and answer*

Instruction

CNR [*<SPACE>* BAR] *<SPACE>* {*…Command…*} *<CR>*

Parameters

Name	Type	Description
BAR	Flag	BAR is the (optional) Break At Read flag. If it is set the successful command will exit the continuous (CNR) mode if in any round at least one tag was found meaning the answer ends with IVF XXX with XXX > 0.

Return Values in Case of Success

{Multiple answers depending on the used command} *<CR>*

Return Values in Case of Failure

"BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

2.34. Break (BRK)

Stops a **CNR** command operation.

»
```
BRK<CR>
```

«
```
BRA<CR>
```

*Example 59. **BRK** command and answer*

Instruction

BRK *<CR>*

Return Values in Case of Success

BRA *<CR>*

Return Values in Case of Failure

“NCM *<CR>*”, “BOD *<CR>*”, “BOF *<CR>*”, “CCE *<CR>*”, “CRT *<CR>*”, “PFE *<CR>*”, “PLE *<CR>*”, “SRT *<CR>*”, “UCO *<CR>*”, “UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*”, “UPA *<CR>*” or “URE *<CR>*”

Chapter 3. Tag Manipulation Instructions

The difference between reader and tag manipulation instructions is whether the target is the reader or the tag itself. Since RFID is mostly about tags and the data stored on tags, the tag manipulation instructions are used extensively in almost any program. Any tag manipulation command can be combined with the **`CNR`** to repeat the command.

Every answer to a tag manipulation command will end with `IVF XXX`. IVF stands for Inventory Found and represents the number of tags found in this round. The XXX value is decimal coded, always 3 characters long and does not exceed 250. If a tag is found the detect LED is switched on at this moment for 500ms. The timer will restart if another tag is found.

Note

Please note that the number of tags found as given by the answer `IVF XXX` is not necessarily the same but can be higher than the number of tags with successful tag manipulations such as read, write, kill etc.

Note

Any command might cause a TimeOut (`TOE`) error in case the instruction takes too long to complete. This might happen if the instruction leads to wrong, missing, or unexpected answers in UHF communication steps or just takes longer to complete than the timeout setting. Examples of this might be an **`INV`** command with a high Q value and / or IR value. The time value until timeout can be adjusted accordingly using the **`SET`** `GTO` command.

Command	Name	Description
`INV`	Inventory	Searches tags and prints their Electronic Product Code (EPC)
`RDT`	Read Data from Tag	Read data from tag (EPC, user memory, TID, etc.)
`WDT`	Write Data to Tag	Write data to tag (EPC, user memory, TID, etc.)
`LCK`	Lock	Locks (or unlocks) a memory area
`KIL`	Kill	Kill a tag, it will answer never again
`DRC`	Direct Command	The Direct Command is used to communicate with tags with optional or manufacturer specific commands

Table 5. Overview of Tag Manipulation Instructions

3.1. Inventory (`INV`)

The inventory command **`INV`** is used to find tags and read their Electronic Product Codes (EPC). It is called inventory command because it allows finding all tags in the field using an anti-collision sequence defined by the EPC UHF Class 1 Gen 2 specification. This is the most common command for almost any UHF RFID application. Using the **CNR** command this command can be used to search for tags continuously.

The inventory command will return the tags' EPCs found one per line as hex-coded numbers with each line terminated by <*CR*>. After the EPCs of that inventory round have all been reported an additional line is reported back which consists of the keyword IVF followed by <*SPACE*> and a three digit number of tags found (e.g. 000 in case no tags were found or 008 in case 8 tags were found).

The length of the answer (the length of the EPC) is defined by the Protocol Control (PC) data field on the tag as defined by the EPC Class 1 Gen 2 Protocol.

Note

The **SET** EPC command gives the same information as the **INV** command. Using them both will give no additional information but double the reader-host communication line usage.

Note

INV will switch on RF power for execution regardless whether RF power was set to ON by using **SRI** ON. The reader will go back to power saving mode after execution of **INV** in case **SRI** SPM ON is set.

»
```
INV<CR>
```

«
```
AABBCCDD<CR>
ABCD1234<CR>
IVF 002<CR>
```

»
```
INV SSL<CR>
```

«
```
AABBCCDD<CR>
IVF 001<CR>
```

*Example 60. **INV** command and answer*

Instruction

```
INV [<SPACE> SSL] [<SPACE> ONT] [<SPACE> SEC] <CR>
```

Parameters

Name	Type	Description
SSL	Flag	Single Slot (sets Q and IR values to zero for this round)
ONT	Flag	This flag causes the reader to not reset the state of tags via a select command. Under normal conditions, this causes the tags to be found only once as long as they are not depowered or reset. Under certain conditions it is possible that tags are found multiple times within one inventory cycle. Use SEC if you need to be sure to find each tag only once.

Name	Type	Description
SEC	Flag	This flag causes the reader to bring the tag to secured / open mode as defined by the EPC Class 1 Gen 2 Protocol. Use this if you need to be sure to find each tag only once (as long as the tags stay powered). SEC is not supported on DwarfG2_Mini (since the DwarfG2_Mini brings the tag to secured or open state each time it is detected).

Return Values in Case of Success

```
IVF <SPACE> {Three digit number} <CR>
```

Return Values in Case of Failure

"NSS <CR>", "TMT <CR>", "FLE <CR>", "TOE <CR>", "WDL <CR>", "TCE <CR>", "RXE <CR>", "PDE <CR>", "CER <CR>", "TOR <CR>", "BOD <CR>", "BOF <CR>", "CCE <CR>", "CRT <CR>", "PFE <CR>", "PLE <CR>", "SRT <CR>", "UCO <CR>", "UER[<SPACE> {*Two Digit Hex Code*}] <CR>", "UPA <CR>" or "URE <CR>"

3.2. Read Data from Tag (RDT)

To read data from the tag this Read Data from Tag command is used. The data can be just the tag's EPC, some data from the tag's user memory if existent, or even the access or kill password (if not locked against reading). With each **RDT** command only one type of memory information can be called (EPC, TID, passwords, etc.). The answer is provided as multiples of two bytes (one word of 16 bit).

It is possible to mask this command (using the **SET** MSK command before calling **RDT**) to limit this command to a certain population of tags. This command can be combined with the **CNR** command, including its BAR option.

As usual the answer ends with an IVF XXX line, giving the number of tags found (not necessarily equivalent to the number of successful read operations) as is the case for e.g. the **INV** command. The XXX value is decimal coded, always 3 characters long and does not exceed 250.

ACP and KLP are allowed as the memory bank parameter, with respect to the according commands. This will give you the access password and kill password, respectively (from the RES memory bank) without the need to search for the right offset value.

```
» RDT USR 0 2<CR>
```

```
« 11223344<CR>
  55667788<CR>
  IVF 002<CR>
```

```
» RDT TID 0 0<CR>
```

```
« E2003412013FF200040164FF011500BA30045FFBFFFFDC50<CR>
  IVF 001<CR>
```

*Example 61. **RDT** command and answer*

Instruction

RDT [*<SPACE>* SSL] *<SPACE>* { EPC | RES | TID | USR | ACP | KLP } *<SPACE>* *{Offset in word}* *<SPACE>* *{Length in word}* *<CR>*

Parameters

Name	Type	Description
SSL	Flag	Single Slot (sets Q and IR values to zero for this round)
Memory bank	Enumeration (EPC, RES, TID, USR, ACP or KLP)	Memory area (bank) to read from
Offset in word	Hexadecimal Integer ($x \geq 0_{16}$)	Start address (in the memory bank)
Length in word	Hexadecimal Integer ($0_{16} \leq x \leq 20_{16}$)	The amount of data you are trying to read. In case zero is given, this is interpreted to mean you want to read the rest of the memory bank starting at Offset. A length of zero is not (properly) supported by the DwarfG2_Mini.

Return Values in Case of Success

Data

Return Values in Case of Failure

"NOR *<CR>*", "EHX *<CR>*", "WDL *<CR>*", "NRF *<CR>*", "TCE *<CR>*", "RXE *<CR>*", "PDE *<CR>*", "CER *<CR>*", "TOR *<CR>*", "TOE *<CR>*", "NSS *<CR>*", "TMT *<CR>*", "FLE *<CR>*", "HBE *<CR>*", "BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

3.3. Write Data to Tag (WDT)

To write data to the tag the Write Data to Tag command is used. The data can be just the tag's EPC, some data from the tag's user memory if existent, or even the access or kill password (if not locked against writing).

Note

When writing to the EPC memory block do not forget to recalculate a new CRC16 by writing it to the tag or by re-powering the tag (move it out of the field, switch the field off, or use **SRI** TIM with sufficient time). Without this all following tag operations will result in the CRC error CCE.

Even though the reader firmware permits it, writing the TID memory bank is usually not supported by the tags. This memory bank is usually written once by the manufacturer of the tag IC and further writing is not permitted by the tag.

Note

All addressed tags in the field will execute the write command. Make sure to mask this command (using the **SET** MSK command before calling **WDT**) to limit this command to the population of tags you wish to address.

This command can be combined with the **CNR** command, including its BAR option.

After the memory type you have to specify the start address you want to start writing from in words (two bytes) in hexadecimal and zero-based notation. Please keep in mind to write complete words.

The data you are trying to write can be multiples of one word (one word equals two bytes as defined by the EPC Gen2). The reader will try to send all data word by word to the tag. Sending of every word will be repeated several times in case of an error. If an error occurs multiple times while trying to write your data the reader will stop writing and report an error.

In case of tags being unsufficiently powered some are not able to send a correct answer before the timeout expires. This will also lead to error messages for answers to **WDT** command calls, although the writing was executed (at least partially) successfully.

Note

With some tags even a **WDT** command call that returns an error code might have been (partially) successful. Please check the success of your **WDT** call in case an error code is reported by using the **RDT** command.

Some of the "Memory bank" parameters specified are just shortcuts for easier writing of special values:

- ACP sets the access password (RES memory bank). The data is 2 words long
- KLP sets the kill password (RES memory bank). The data is 2 words long
- LEN sets the EPC length (EPC memory bank). The data is a hex number of at max 31
- NSI sets the numbering system identifier value (EPC memory bank). The data is a hex value of 9 bits (max 0x1FF)

»

```
WDT USR 0 11223344<CR>
```

«

```
OK!<CR>
OK!<CR>
IVF 002<CR>
```

»

```
WDT EPC 2 AABBCCDD<CR>
```

«

```
OK!<CR>
IVF 001<CR>
```

»
```
WDT LEN 2<CR>
```

«
```
OK!<CR>
IVF 001<CR>
```

»
```
WDT RES 2 ACCEC0DE<CR>
```

«
```
OK!<CR>
IVF 001<CR>
```

»
```
WDT RES 0 DEADC0DE<CR>
```

«
```
OK!<CR>
IVF 001<CR>
```

»
```
WDT KLP DEADC0DE<CR>
```

«
```
OK!<CR>
IVF 001<CR>
```

Example 62. ***WDT*** *command and answer*

Instruction

`WDT [<SPACE> SSL] <SPACE> { EPC | RES | TID | USR | ACP | KLP | LEN | NSI } [<SPACE> {Starting Block} <SPACE> {…DATA…} ] <CR>`

Parameters

Name	Type	Description
SSL	Flag	Single Slot (sets Q and IR values to zero for this round)
Memory bank	Enumeration (EPC, RES, TID, USR, ACP, KLP, LEN or NSI)	Memory area (bank) to write to
Starting Block	Optional Hexadecimal Integer ($x \geq 0_{16}$)	Start address for writing in words

Return Values in Case of Success

`OK! <CR>`

Return Values in Case of Failure

"NOR <CR>", "EHX <CR>", "TCE <CR>", "WDL <CR>", "NRF <CR>", "RXE <CR>", "PDE <CR>", "TOR <CR>", "TOE <CR>", "NSS <CR>", "TMT <CR>", "FLE <CR>", "BOD <CR>", "BOF <CR>", "CCE <CR>", "CRT <CR>", "PFE <CR>", "PLE <CR>", "SRT <CR>", "UCO <CR>", "UER[<SPACE> {Two Digit Hex Code}] <CR>", "UPA <CR>" or "URE <CR>"

3.4. Lock (LCK)

The Lock command is used to set the access rights of the different data blocks, including the access password itself and the kill password. To use this command you have to be in the secured state (i.e. authenticated yourself with the correct password).

Note

All addressed tags in the field will execute the Lock command. Make sure to mask this command (using the **SET** MSK command before calling **LCK**) to limit this command to the population of tags you wish to address.

This command can be combined with the **CNR** command, including its BAR option.

For each type of memory (EPC, TID, USR, ACP, KLP) you can define the access mode. Their meaning differ depending on the memory type. (See EPC Gen 2 Specification for more details).

For the passwords (ACP and KLP):

Mode	Description
0	Password location is readable and writeable from either open or secured state
1	Password location is permanently readable and writeable from either the open or secured states and may never be locked
2	Password location is readable and writeable from secured state but not from open state
3	Password location is permanently not readable or writeable from any state

Table 6. Lock coding for ACP and KLP

For the memory banks (EPC, TID, USR):

Mode	Description
0	Memory bank is writeable from either open or secured states
1	Memory bank is permanently writeable from either the open or secured states and may never be locked
2	Memory bank is writeable from secured state but not from open state
3	Memory bank is permanently not writeable from any state

Table 7. Lock coding for EPC, TID and USR

»
```
LCK EPC 0<CR>
```

«
```
OK!<CR>
OK!<CR>
IVF 002<CR>
```

»
```
LCK KIL 3<CR>
```

«
```
OK!<CR>
IVF 001<CR>
```

*Example 63. **LCK** command and answer*

Instruction

```
LCK [<SPACE> SSL] <SPACE> { EPC | TID | USR | ACP | KLP } <SPACE>
{Lock level} <CR>
```

Parameters

Name	Type	Description
SSL	Flag	Single Slot (sets Q and IR values to zero for this round)
Memory area	Enumeration (EPC, TID, USR, ACP or KLP)	Memory area to lock. These are banks (EPC, TID, USR) and password areas (instead of RES bank)
Lock level	Hexadecimal Integer ($0_{16} \le x \le 3_{16}$)	The lock level according to EPC Gen2. Also described above

Return Values in Case of Success

```
OK! <CR>
```

Return Values in Case of Failure

“HBE <CR>”, “WDL <CR>”, “NRF <CR>”, “TCE <CR>”, “RXE <CR>”, “PDE <CR>”, “CER <CR>”, “TOR <CR>”, “TOE <CR>”, “NSS <CR>”, “TMT <CR>”, “FLE <CR>”, “BOD <CR>”, “BOF <CR>”, “CCE <CR>”, “CRT <CR>”, “PFE <CR>”, “PLE <CR>”, “SRT <CR>”, “UCO <CR>”, “UER[<SPACE> {*Two Digit Hex Code*}] <CR>”, “UPA <CR>” or “URE <CR>”

3.5. Kill (KIL)

The Kill command can be used to disable UHF Gen2 tags forever. To do this, the kill password is necessary (four bytes). The password is given or loaded from EEPROM via **SET** command.

Note

All addressed tags in the field will execute the Kill command. Make sure to mask this command (using the **SET** MSK command before calling **KIL**) to limit this command to the population of tags you wish to address.

This command can be combined with the **CNR** command, including its BAR option.

The **SET** EPC ON command might be helpful to keep track of tags killed.

Warning

If you use this command incorrectly (especially in combination with the **CNR** command) you can irreversibly kill a very big number of UHF tags in a very short time!

```
» KIL<CR>
```

«

```
OK!<CR>
OK!<CR>
IVF 002<CR>
```

»

```
KIL<CR>
```

«

```
IVF 000<CR>
```

Example 64. ***KIL*** *command and answer*

Instruction

KIL [*<SPACE>* SSL] *<CR>*

Parameters

Name	Type	Description
SSL	Flag	Single Slot (sets Q and IR values to zero for this round)

Return Values in Case of Success

OK! *<CR>*

Return Values in Case of Failure

"NRF *<CR>*", "TCE *<CR>*", "RXE *<CR>*", "PDE *<CR>*", "HBE *<CR>*", "CER *<CR>*", "TOR *<CR>*", "TOE *<CR>*", "NSS *<CR>*", "TMT *<CR>*", "FLE *<CR>*", "BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

3.6. Direct Command (DRC)

The Direct Command is used to communicate with tags with optional or manufacturer specific commands. The command should be used by advanced users only. A good understanding of EPC Gen2 Basics is required for correct use.

Note

All addressed tags in the field will execute the Direct Command. Make sure to mask this command (using the **SET** MSK command before calling **DRC**) to limit this command to the population of tags you wish to address.

This command can be combined with the **CNR** command, including its BAR option.

Warning

Using this command incorrectly might lead to setting of incorrectly or completely undefined states. It might not be possible to reset the tag afterwards. Please be sure that you know what you do!

DRC is not supported by DwarfG2_Mini.

```
» DRC CHB C2800040 26 H CRC<CR>
```

```
« E200FFA039C992<CR>
  IVF 001<CR>
```

*Example 65. **DRC** command and answer equalising RDT TID 0 1 with autoremoval of header bit. The first word are the data read, then 1 word RN16 (16-bit random or pseudo-random number according to EPC Gen2), last 1 word CRC*

Instruction

DRC [*<SPACE>* SSL] [*<SPACE>* FST] [*<SPACE>* CHB] *<SPACE>* {*Data1*} [*<SPACE>* {*Data1 length in bit*} [*<SPACE>* H [*<SPACE>* {*Data2*}]]] [*<SPACE>* CRC] *<CR>*

Parameters

Name	Type	Description
SSL	Flag	Single Slot (sets Q and IR values to zero for this round)
FST	Flag	If the command is expected to be fast in execution set this flag. The no response wait time (NRW) parameter will be used. Else, a 20ms timeout is used, which should be sufficient for most commands.
CHB	Flag	CHB is used to increase user readability of answers to commands with header bits. It interprets the header bit and removes it by shifting the data by one bit. If this parameter is set the first bit of the answer will be seen as a header bit. If the first bit of the answer is set (=1) as it is in case of an error, the answer will get a HBE (Header bit error) prefix.
Data1	Hexadecimal String	Data1 is mandatory. The data to be sent to the tag
Data1 length in bit	Optional Decimal Integer (x ≥ 1)	Data1 length is optional
H	Optional Key-word	H (Handle) is optional. The handle will be added at this position after Data1 (and before Data2 if Data2 exists). This is to allow the use of custom commands that require the tag handle as an argument which is usually not available outside of the firmware as it is generated by the tag as a random number.
Data2	Optional Hexadecimal String	Data 2 is optional and only allowed if handle is used
CRC	Flag	The CRC flag is optional but frequently needed as the CRC depends on the handle. If this flag is set, the reader automatically calculates and adds the CRC16 according to EPC Gen2.

Return Values in Case of Success

```
[Answer Data in ASCII hex code] <CR>
```

Return Values in Case of Failure

"NRF *<CR>*", "TCE *<CR>*", "RXE *<CR>*", "PDE *<CR>*", "HBE *<CR>*", "CER *<CR>*", "TOR *<CR>*", "TOE *<CR>*", "NSS *<CR>*", "TMT *<CR>*", "FLE *<CR>*", "BOD *<CR>*", "BOF *<CR>*", "CCE *<CR>*", "CRT *<CR>*", "PFE *<CR>*", "PLE *<CR>*", "SRT *<CR>*", "UCO *<CR>*", "UER[*<SPACE>* {*Two Digit Hex Code*}] *<CR>*", "UPA *<CR>*" or "URE *<CR>*"

Chapter 4. Precommands

Precommands may help parsing answers or detecting the answer of a device (if more than one device share a communication line).

The precommands always start and end with '#', contain a one character command and the following signs as (optional) parameters.

4.1. Command Answer Prefixes ('P')

This command sets a prefix of up to 16 characters which is sent before any answer line. To the prefix a space character (0x20) is appended. The prefix will stay active until it is deleted (using the command without following parameters (#P#)). This may be used to detect specific devices or to "tag" specific answers to specific commands.

»
```
#PDEVICE1#INV<CR>
```

«
```
DEVICE1 E200FFA039C992<CR>
DEVICE1 IVF 001<CR>
```

»
```
INV<CR>
```

«
```
DEVICE1 E200FFA039C992<CR>
DEVICE1 IVF 001<CR>
```

»
```
#P#INV<CR>
```

«
```
E200FFA039C992<CR>
IVF 001<CR>
```

Example 66. Prefix precommand and answers with INV as command

4.2. Answer Counter ('C')

This precommand causes the counter to be reset to zero and be formatted as given. The counter counts up every answer starting by 0 (not every line ending on *<CR>*, but every complete answer that would end with *<LF>* if the EOF command were activated). For this reason it may ease parsing. The prefix data may be given as hex or decimal numbers each one or two digits long. Please keep in mind that this leads to the possible counter overflows at 9->0, 99->0, F->0 or FF->0.

The counter is reset every time the C command is called.

The prefix is deactivated by giving '0' as parameter.

The format is set by the parameters `D1`, `D2`, `H1`, `H2` to define decimal (D) or hexadecimal (H) and one or two digits, respectively.

»
```
#CD1#INV<CR>
```

«
```
0 E200FFA039C992<CR>
0 IVF 001<CR>
```

»
```
INV<CR>
```

«
```
1 E200FFA039C992<CR>
1 IVF 001<CR>
```

»
```
#CD1#INV<CR>
```

«
```
0 E200FFA039C992<CR>
0 IVF 001<CR>
```

»
```
#C0#INV<CR>
```

«
```
E200FFA039C992<CR>
IVF 001<CR>
```

Example 67. Counter precommand and answers with INV as command

4.3. Using 'P' and 'C' together

Both precommands can be active at the same time. In this case, the prefixes have to be specified one after the another in the same command line. Setting or reenabling one if the other one is active is not defined. For the answers, the same prefix order will be used as during the command calls. The prefixes can be disabled independently (the prefix not disabled stays active). Setting of prefixes will be executed even if the command is rejected (`UCO`, `UPA`, etc.)

»
```
#PDEVICE1##CD1#INV<CR>
```

«
```
DEVICE1 0 E200FFA039C992<CR>
DEVICE1 0 IVF 001<CR>
```

»
```
INV<CR>
```

«
```
DEVICE1 1 E200FFA039C992<CR>
DEVICE1 1 IVF 001<CR>
```

»
```
#P#INV<CR>
```

«

```
2 E200FFA039C992<CR>
2 IVF 001<CR>
```

Example 68.

Chapter 5. Error Codes

Error Code	Name	Description
ACE	Access Error	An error occurred while trying to authenticate yourself to a tag (e.g. wrong password)
ARH	Antenna Reflectivity High	**STD** command checked the antenna reflection sending a continuous wave as output and measuring the analog input level. The ARH code occurred because the measured value was higher than it should be with a matching antenna. Often caused from missing or mismatched antennas.
BOD	Brownout Detected	The device detected a brownout. This is a hardware error. If it occurs more repeatedly please check your power supply or contact metraTec for support.
BOF	Buffer Overflow	A host-to-reader communication buffer received an overflow error. Send and receive buffer have 256 bytes each. The reason can be too fast answers or too fast command sending.
CCE	Communication CRC Error	A CRC error has been detected while receiving a line from the host system.
CER	CRC error	CRC from tag is wrong. Common error sources: • The tag left the field or is too far away • Another device disturbed the communication • A collision between tag answers happened • On DwarfG2_Mini: An error was reported by the tag causing an unexpected answer length and therefore the CER error (instead of HBE XX)
CRT	Command Receive Timeout	Parts of a command were received by the reader but the device never received a carriage return. Always close commands with *<CR>*.
DNS	Did Not Sleep	A **WAK** command was sent although the reader was not in sleep mode.
EDX	Error Decimal Expected	Parameter string cannot be interpreted as a valid decimal value. Common error source: Character other than '0' to '9'.
EHF	Error Hardware Failure	The RF interface chip does not match or is damaged. Please try a full reset (power reset) and/or contact support.

Error Code	Name	Description
`EHX`	Error Hexadecimal Expected	Parameter string cannot be interpreted as a valid hexadecimal value. Common error source: Character other than '0' to '9' or 'A' to 'F'.
`FLE`	FIFO Length Error	The FIFO contains too much data, data might be corrupted.
`HBE`[***<SPACE>*** {*Two Digit Hex Code*}]	Header Bit Error	`HBE` gives the error code received from the tag. See the EPC Gen 2 Specification for further information. If no error code follows, the answer containing the error code had an error (CRC or handle does not match) The error codes defined in the EPC Gen2 are: • 0x00: Other error (none matches) • 0x03: Memory location does not exist or PC value not supported • 0x04: Memory locked • 0x0B: Insufficient power to write • 0x0F: Nonspecific error (tag does not support error codes) • any other: manufacturer specific error code
`NCM`	Not in CNR Mode	A **`BRK`** command was sent although the reader was not in **`CNR`** mode.
`NOR`	Number Out of Range	Parameter value exceeds maximum value / minimum value.
`NOS`	Not Supported	This command is not supported by this hardware e.g. **`WOP`** on DeskID (which has no output pins).
`NRF`	No RF-Field Active	RF field was not yet turned on using the **`SRI`** `ON` command. Might also hint to an undetected restart of the device.
`NSS`	No Standard Selected	Regional standard was not set before starting the RF field. Use **`STD`** `ETS` before using **`SRI`** or **`INV`**.
`PDE`	Preamble Detect Error	Common error sources: • The tag left the field or is too far away • Another device disturbed the communication • A collision between two tags' answers happened

Error Code	Name	Description
		- The tag is too near, overloading the tag or mismatching the antenna. This occurs especially with the **WDT** command
PFE	Prefix Error	Prefix command is incorrectly formatted (data too long, ending '#' missing, unknown parameter or command, etc.)
PLE	PLL Error	PLL, RF level or crystal oscillator is not stable. The PLE error code is sent on **VBL**=1 after resetting the air interface hardware and getting a “not stable” again. With **VBL**=2 PLE is sent on the first occurrence, too. PLE can occur at every frequency hop so even if no command is sent or even if the RF is off. The reader will retry getting PLL stable at the next hop (about every 4s with ETSI). If PLE occurs again, try a hard restart (power off). If this does not work it is likely the hardware is damaged. In this case please call support.
RDL	Read Data too Long	The answer to an **RDT** command from the tag is longer than the memory buffer reserved for receiving (120 Byte).
RXE	Response Length not as Expected Error	The answer was longer or shorter than expected. Common error sources: - The tag left the field or is too far away - Another device disturbed the communication - A collision between two tag answers happened
SRT	Watchdog Reset	In case of a critical error this error might occur. If you get this error frequently, your hardware is probably damaged.
TCE	Tag Communication Error	General error during tag communication (but tag was found) Common error source: Write command returned wrong check (handle). Data might be corrupted.
TMT	Too Many Tags	The reader found more tags than it can handle (max. 250).
TNR	Tag Not Responding	No answer on query. This only occurs if **VBL** = 2
TOE	TimeOut Error	The command timed out meaning the timeout value has run out. This may be caused by an unexpected transceiver error or by too many tags / too long data. The timeout is 1000ms by default and can be changed using **SET** GTO. The IVF gives the number of tags found un-

Error Code	Name	Description
		til the timeout occurred. The command gets cancelled.
TOR	Tag Out of Range	The tag answered on query but the following command (for example read or lock) was not successful. This might indicate a range or access problem.
UCO	Unknown Command	An unknown command has been passed. Common error sources: ● You have a typo in your command string ● Wrong firmware version
UER[***<SPACE>*** {*Two Digit Hex Code*}]	Unknown Error	Internal error reached the API unintendedly. The error code is shown hex encoded With no hex error code: A bad interrupt occurred, unknown internal tag error code returned or another unexpected case occurred.
UPA	Unknown Parameter	An invalid or missing parameter has been passed to a function. Common error source: ● You have a typo in your command string ● A parameter is missing ● Wrong firmware version
URE	UART Receive Error	The hardware UART detected an error. Data might be corrupted. The whole packet will not be used. If the ***<CR>*** at the end of command was the corrupted byte the next command will be deleted, too.
WDL	Wrong Data Length	The data given is too long or short. This might occur on commands using data of variable length.
WMO	Wrong Mode	A command can not be executed because it is prohibited in a specific mode. For example setting **SUC** ON when no **SUC** command is saved.

Appendix A. Quick Start Guide and Examples

The previous chapters have given a thorough reference to the commands the metraTec UHF readers support. While this reference is necessary it also creates the impression that using the reader is somehow complicated which it is not. In most practical cases a user will only need to send a handful of strings to the reader to make it do everything that is needed. Only in special circumstances more is needed. In the following sections, you will find the sequence of commands to the reader that are needed in the most common cases.

A.1. Typical Reader Initialization Sequence

Before the reader can start reading tags, the RF field has to be activated. This example shows a typical initialization sequence to read UHF EPC Gen2 tags. These are probably the first strings you need to send to the reader.

```
» STD ETS<CR>
```

```
« OK!<CR>
```

```
» SRI ON<CR>
```

```
« OK!<CR>
```

*Example 69. Initialization by **STD** and **SRI***

This configures the device for ETSI compatible usage. If you are in an FCC covered country use FCC instead of ETS. Afterwards the reader is ready to search for tags and read from or write to most tags.

A.2. Reading the EPCs of a tag population

By far the most common operation done with UHF RFID tags is reading the EPC of a tag or of a tag population. In many cases this is the only thing needed from the tag in which case this is the third (and last) type of string you need to send to the reader. There are several possibilities to do this with a metraTec device, depending on what exactly you need to do. All operations however are based on the **INV** (inventory) command. The answer gives the tag EPC and the number of tags found.

To simply read the EPC of a low number of tags (a maximum of about 40 for PulsarMX and DwarfG2_XR and 10 for the other devices) in the field (using anti collision) the simple **INV** command is sufficient:

```
» INV<CR>
```

If no tag is found, the answer will look like in Example 70, “Inventory answer if no tag has been found”. If two tags are found the answer could look like in Example 71, “Inventory answer if two tags have been found”.

```
« IVF 00<CR>
```

Example 70. Inventory answer if no tag has been found

```
« E0040100078E3BB0<CR>
  E0040100078E3BB7<CR>
  IVF 02<CR>
```

Example 71. Inventory answer if two tags have been found

If you are sure that there will be only a single tag in the field, you can use the single slot (`SSL`) inventory. This disables the anti collision algorithms and makes the operation a bit faster and more reliable.

Instruction:

```
» INV SSL<CR>
```

Possible responses:

```
« IVF 000<CR>
```

Example 72. Answer to INV SSL if there is no tag

```
« E0040100078E3BB0<CR>
  IVF 001<CR>
```

*Example 73. Answer to **INV** SSL if there is exactly one tag with the EPC E0040100078E3BB0*

With `SSL` the answer never contains more than one tag.

If there is more than one tag in range the use of `SSL` is practicable in combination with masking the tags (**`MSK`**) so only one tag answers.

Note

If there is more than one tag in range while using **`INV`** `SSL` without masking the tags, the result is undetermined. In case of collision of answers, no tag could be detected. If one tag answers, it is usually the one with the better antenna coupling. In case of identical tags this could be the nearer one.

A.3. Reading EPCs continuously

All tag manipulation commands can be processed by the reader continuously by using the **`CNR`** command. With the help of this command it is possible to make the reader read the tag

EPCs of the tags in the field endlessly. It is also possible to adapt this example to read or write to the tags in the field (very useful in tag producing machines or automation scenarios).

Instruction and response with two tags in the field:

```
» CNR INV<CR>
```

```
« E0040100078E3BB0<CR>
  E0040100078E3BB7<CR>
  IVF 02<CR>
  E0040100078E3BB0<CR>
  E0040100078E3BB7<CR>
  IVF 02<CR>
  E0040100078E3BB7<CR>
  E0040100078E3BB0<CR>
  IVF 02<CR>
  ...<CR>                        the output will repeat, the answer order is
   (as always) random
```

You can stop the endless sequence by sending the **BRK** (break) command.

Instruction and response:

```
» BRK<CR>
```

```
« BRA<CR>
```

A.4. Example for writing and reading to and from UHF Gen2 tags

Next we show how to write and read data to and from a tag in unaddressed mode. Unaddressed mode means that you do not send the command to a specific tag so you do not need to supply the tag ID as part of the command. This of course means that the command is then executed by any tag in the field. If needed, please make sure to use a mask (**MSK**) to specifiy the tags that should be written to.

Instruction and response:

```
» WDT USR 03 11112222<CR>
```

```
« OK!<CR>                  Data written
  IVF 001<CR>              1 tag found
```

Example 74. Write 11112222 data to word 3 of user memory bank

To read the same data we just wrote to the tag, use:

```
» RDT USR 03 02<CR>
```

```
« 11112222<CR>    2 word read
  IVF 001<CR>     1 tag found
```

Example 75. Read the data (2 words) from word 3 of user memory bank

A.5. Example for changing EPCs of UHF Gen2 tags

The memory organisation on EPC Gen2 tags make the writing of EPCs (the answer to **INV**) a bit complicated. Please note the following if attempting to write EPCs:

- The EPC memory bank contains a precomputed CRC. For successfully changing the EPC, the user has to either compute the new CRC and make it part of the writing or has to repower the tag so the tag recomputes the CRC. **SRI** TIM 500 will do the repowering for many tags.
- The EPC length is not fixed. It can be changed to match the customers needs. To make the access to the length bits easy there is a LEN parameter to the **WDT** command.
- The EPC itself starts at address 0x20 (in bit), not zero. The parts before 0x20 should not be used for masking.

An example to check and then change the EPC:

```
» INV<CR>
```

```
« AABBCCDD<CR>    2 word read
  IVF 001<CR>     1 tag found
```

```
» MSK EPC AABBCCDD 20<CR>
```

```
« OK!<CR>
```

```
» WDT 1234<CR>
```

```
« OK!<CR>         write success
  IVF 001<CR>     1 tag found
```

```
» SRI TIM 500<CR>
```

```
« OK!<CR>         Disables the field for half a second
```

```
» INV<CR>         not needed, just to show the state
```

```
« 1234CCDD<CR>    2 word read
  IVF 001<CR>     1 tag found
```

```
» WDT LEN 1<CR>
```

```
« OK!<CR>         write success
  IVF 001<CR>     1 tag found
```

```
» SRI TIM 500<CR>
```

```
« OK!<CR>         Disables the field for half a second
```

```
» INV<CR>         not needed, just to show the state
```

```
« 1234<CR>        1 word read
  IVF 001<CR>     1 tag found
```

Example 76. Get EPC, Mask to it, change it to 1234

A.6. Configuring reader to automatically start reading tag IDs when powered

All metraTec readers will wait for commands when first powered. In some cases, however, the user wants the reader to automatically start searching for tags once it is powered and only start sending messages when it finds tags. To configure the reader to do this we use the **SUC** command and set the verbosity level to minimum so that the reader stays quiet until it finds tags.

```
» SUC STD ETS;SRI ON;VBL 0;CNR INV<CR>
```

```
« OK!<CR>
```

The reader will respond with `OK!` and will start performing in the way specified after it is reset or repowered. In case you want to end the continuous reading mode you will need to send the **BRK** command.

Appendix B. CRC Calculation

```c
/**
 * This function calculates a CRC16 over a unsigned char array
 * with LSB first.
 *
 * @param DataBuf Pointer to data to calculate CRC16 for.
 * @param SizeOfDataBuf Length of the data buffer (DataBuf)
 * @param Polynom Value of the generator polynom.
 *                0x8408 is recommended.
 * @param Initial_Value Initial value of CRC16.
 *                      0xFFFF is recommended for
 *                      host to reader communication.
 * @return Calculated CRC16
 */
unsigned short GetCrc(unsigned char *DataBuf,
                      unsigned char SizeOfDataBuf,
                      unsigned short Polynom,
                      unsigned short Initial_Value)
{
        unsigned short Crc16 = Initial_Value;
        unsigned char Byte_Counter, Bit_Counter;

        for (Byte_Counter = 0;
             Byte_Counter < SizeOfDataBuf;
             Byte_Counter++)
        {
                Crc16 ^= DataBuf[Byte_Counter];
                for (Bit_Counter = 0; Bit_Counter < 8; j++)
                {
                        if ((Crc16 & 0x0001) == 0)
                                Crc16 >>= 1;
                        else
                                Crc16 = (Crc16>>1)^Polynom;
                }
        }

        return (Crc16);
}
```

Version Control

Version	Change	By	Date
1.0	created	KD	16.10.2009
1.1	Major changes for FW Version 1.0	KD	17.9.2010
1.2	Some minor corrections and layout work	KD	06.10.2010
1.3	Added some clarifications	KD	15.10.2010
1.4	Profiles added added commands (KLP, KLS,APS,APL..) removed commands (HWO, OPS)	MK	22.02.2011
1.5	Errors removed added some clarifications	MK	04.04.2011
1.5.1	Added SQV RST clarification	MK	12.04.2011
1.6	Added BOD error code changed UART to 115200 baud (since Firmware Desk1.7,Dwarf1.5,Pulsar1.1	MK	28.04.2011
1.6.1	RRP constant changed for Pulsar+DwarfG2 added output power level Changed PulsarMX power range	MK	14.07.2011
1.7	Added DRC command	MK	16.09.2011
1.8	Added RQV, SIR, RIR Removed EPH Profile description changed some parameter max values changed removed parsing errors from command error descriptions Changed G-Value for RRP	MK	17.11.2011
1.9	NOS no longer possible error on SRI SPM	MK	28.11.2011
1.10	NRF now a possible error on SRI TIM	MK	14.06.2012
1.11	RRP clarified and G value for Pulsar changed	TM	30.07.2012
1.11	Added commands SUC, RSC and SAP	MK	21.11.2012
1.12	Added DR8 CFG parameters	MK	04.12.2012
1.13	Added SET HOT and SET IHC	MK	15.03.2013
1.14	Added MOD STD requirement to RRP	MK	14.06.2013
1.15	Changed a WDT example setting start block to 20 instead of 2	MK	05.09.2013

	Added RX Wait Time Offset to WDT command		
1.16	For Firmware 2.13 Added SUC ON Added Precommands Changed MOD STD to STD and ETSI to ETS Internal changes to command execution regarding SUC, IHC as well as user generated commands (timings may change)	MK	28.04.2014
2.0	For FW Version 3.0 Changed profiles Added INV SEC as backwards compatiblity to INV change MOD STD ETSI changed to STD ETS	MK	12.08.2014
2.1	For FW Version 3.0 Added SRI which was missing in rev 2.0	MK	12.08.2014
2.2	Corrected typos Some clarifications	FS	23.09.2014
2.3	Moved SET example Removed commands for internal use	MK	04.11.2014
2.4	Added internal commands to internal version Removed profiles (formerly deprecated)	MK	04.05.2015
2.5	Added internal commands to internal version Added MOD instruction (outdated but still supported) Added some minor text	MK	03.09.2015
2.6	Added STD FCC	MK	22.09.2015
2.7	Includes DwarfG2_Mini Reworked format Added the new **RAP** command and changed the reworked **SAP** command **STD** to reflect firmware change: DwarfG2_XR now checks antenna like PulsarMX does.	MK, CS	29.01.2016
2.8	Added the new **ECH** and **HBT** command	MK	27.06.2016

metraTec GmbH

Werner-Heisenberg-Str. 1
39106 Magdeburg
Germany

Tel.: +49 (0)391 251906-00
Fax: +49 (0)391 251906-01

Email: <support@metratec.com>
Web: http://www.metratec.com