TWN4

Director User Guide

DocRev1, February 11, 2013

Elatec GmbH

Contents

1 Introduction

This document describes all features and the usage of the software Director V1.00.
Director is an application that runs on Microsoft Windows. The purpose of this software is to easily test all the internal firmware functions and powerful APIs of the TWN4, a state of the art RFID reader made by Elatec GmbH, Germany.

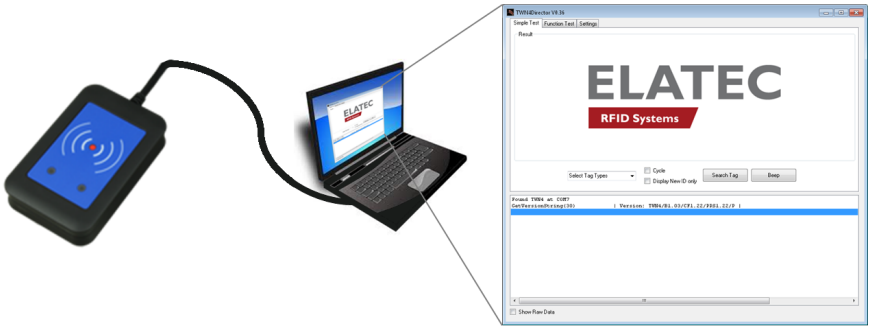
The behavior of TWN4 can be controlled by the user with so called apps. These apps are little programs which are mainly written in C, a common programming language. They have access to several APIs which makes it easy to develop complex RFID applications.

The Director can be used to test the firmware functions and APIs of TWN4 without writing a single line of code. It communicates with the reader over a serial interface like USB or RS232 with an ASCII protocol, which is called simple protocol.

1.1 System requirements

Host:	Director needs Microsoft Windows with .NET Framework 3.5 installed.
TWN4:	Firmware 1.22 or higher with app_proto (PRS1.22 or higher) e.g. TWN4_Cx122_PRS122.bix

2 Usage of Director



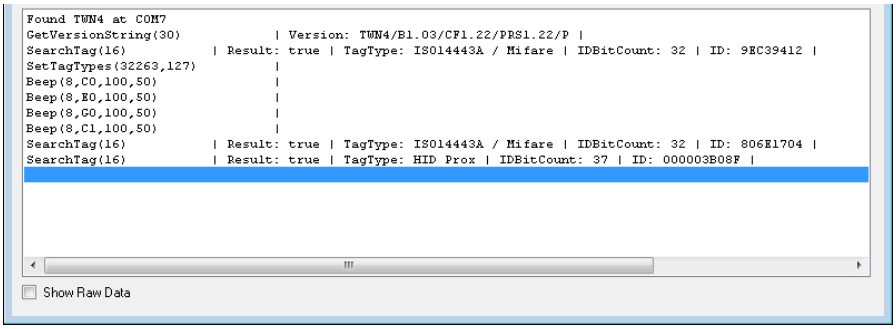
2.1 Startup

At startup, TWN4Director searches for a TWN4, first on USB and then on all other serial connections. After connecting, the version string will be displayed in the log. If there is no correct response from the reader, a message will be written in the log box (`Please connect a TWN4 with app_proto installed`).
This procedure will also take place if the reader is disconnected and connected again.

The automatic connection can be disabled on the tab sheet *Settings*, see chapter 2.5.

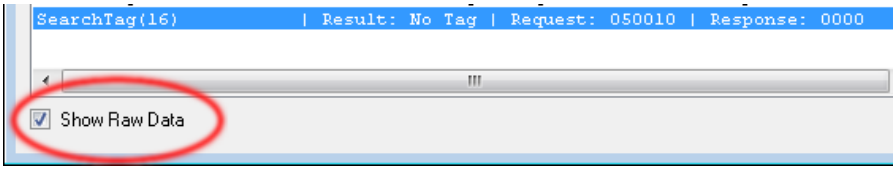
2.2 Log

Every function call and its result are logged in the log box at the bottom of the application. Also every message will be displayed here.



There is a context menu available for the log box which allows the user either to clear or to copy all items to the clipboard.

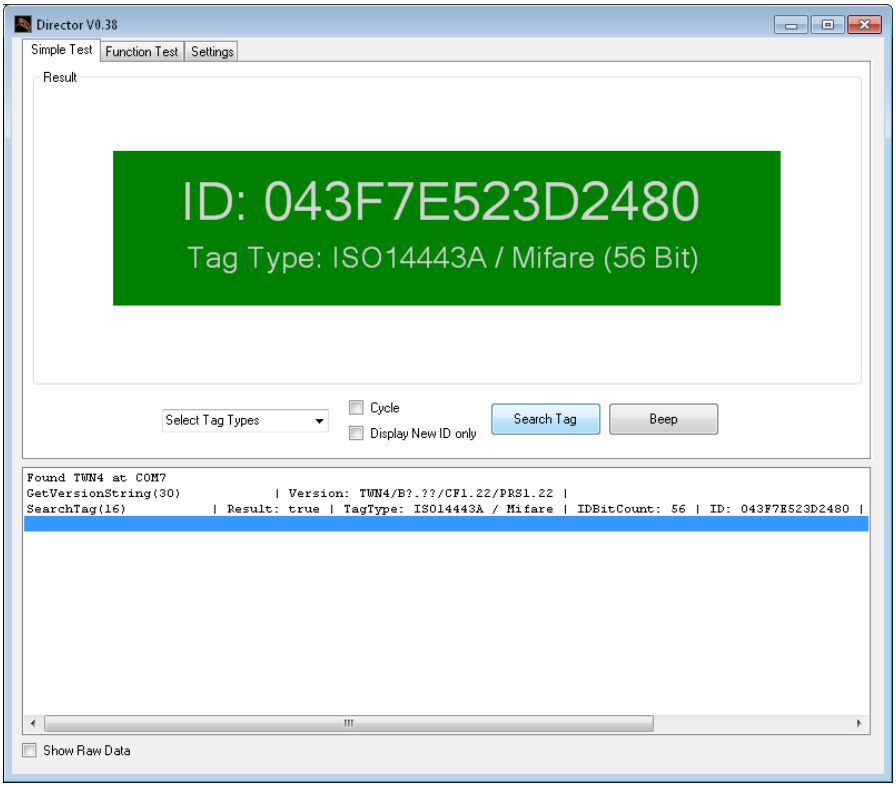
Beneath the log is a checkbox called *Show Raw Data*. If checked, each function call will also log the request / response which are sent to / from the reader in raw format.



2.3 Simple Test

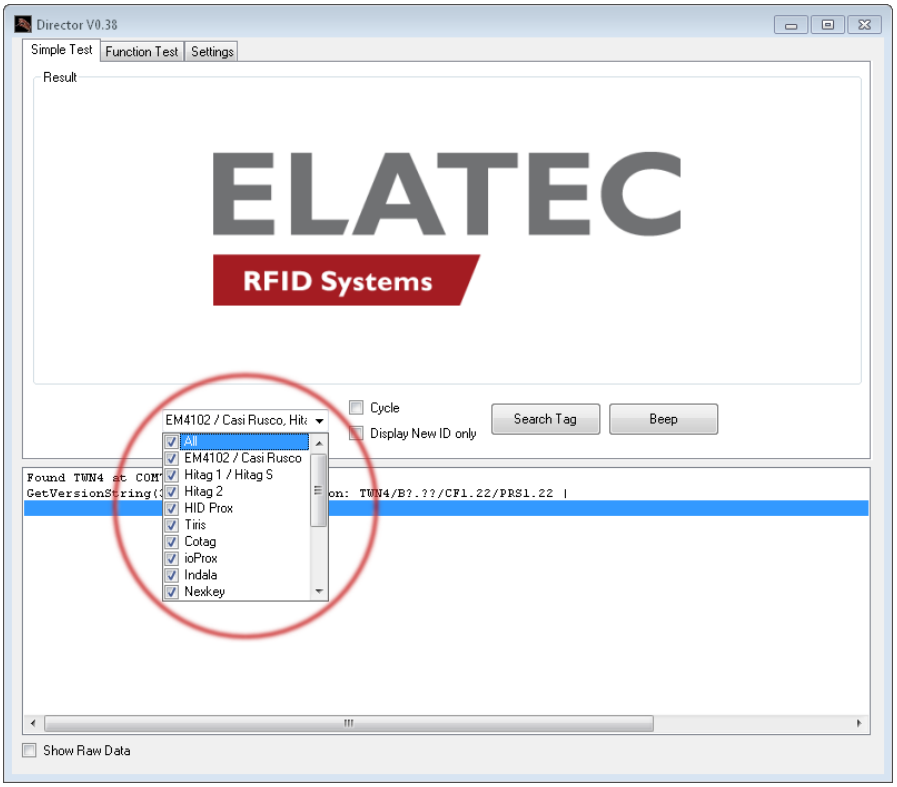
The first tab sheet has two buttons, *Search Tag* and *Beep*.

Clicking the button *Search Tag* lets the TWN4 search for a transponder. If one is found, the ID, the tag type and the bit count are displayed in the result box. Also the reader sounds a short beep .To search for transponders continuously, just check the *Cycle* checkbox.



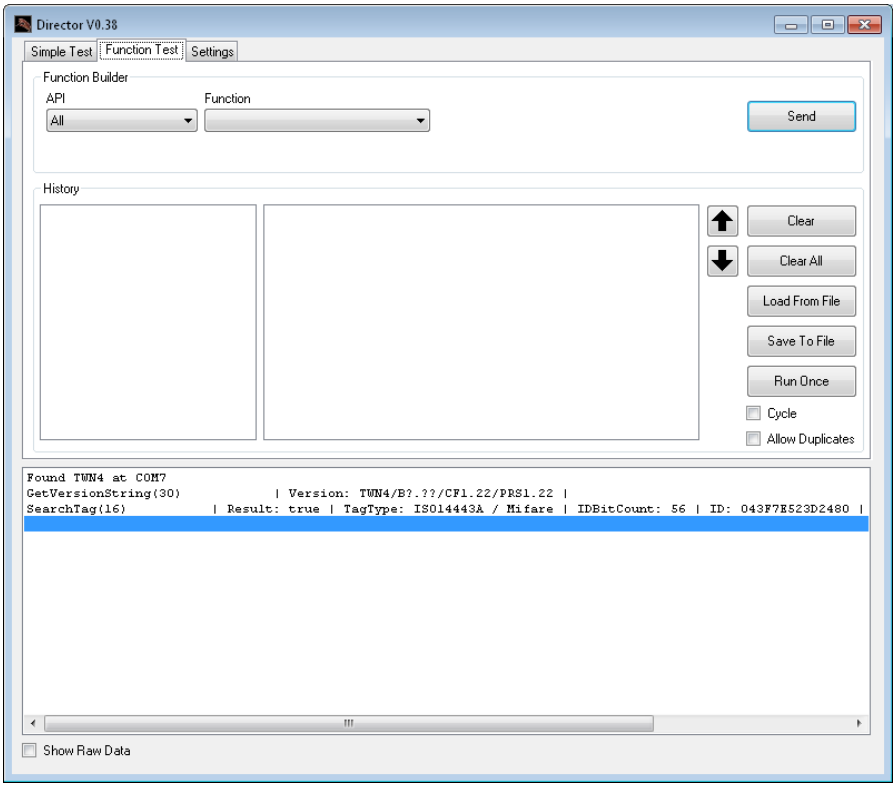
The button *Beep* results in a sequence of 4 beeps of the reader (it's a c major chord, just to let you know).

There is also a combo box *Select Tag Types* which lets you comfortably enable or disable the tag types the reader is searching for. This is also possible in cyclic operation (checkbox *Cycle* is checked).



2.4 Function Test

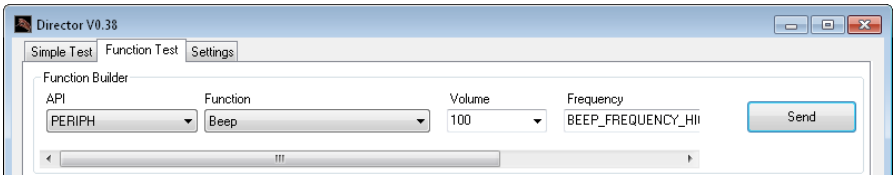
This is the section for advanced users and is the heart of the application. Here you have access to nearly all firmware functions of TWN4.



2.4.1 Function selection

First we need to build our function call, which consists of the functions name and its parameters. The function can be selected by the drop down list *Function*. The huge amount of available functions can be filtered by the drop down list *API*, which lists all APIs.

If a function is selected, a combo box appears for each parameter (if any). The combo boxes are hopefully filled with some default values. Then you can just click the send button on the right and see what happens. If a default value is missing, you have to fill the box by yourself. Of course you can also overwrite the default values.



2.4.2 Formats for parameter values

The parameter values can be typed in different formats which are listed below.

Type	Format	Example
Decimal value	Number	`110`
Hexadecimal	Prefix 0x	`0x6e, 0x6E`
Binary	Prefix 0b	`0b01101110`
ASCII character	Quotes	`‘n‘`
ASCII string	Double quotes	`"A string"`

Table 2.1: Formats for parameter values

Each single number or character can be combined to a list of values, if a parameter needs more than one. Just separate the values with a space. Each value can have a different format. It is also possible to type in a list of hexadecimal values as one block (see *Byte array* in table above)

For example,

```
0x41 0x20 0x73 0x74 0x72 0x69 0x6E 0x67
```

is the same like

```
‘A‘ 0x20 0x73 0x74 0x72 0x69 0b01101110 147
```

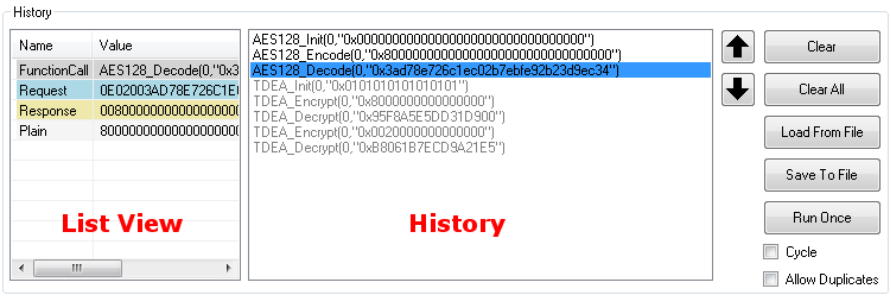
If the mouse hovers over a parameter field, a small hint appears which type and how many values are expected.

When a function is executed by clicking the *Send* button or just hitting the Enter key, the request is sent to the reader. Hopefully we get a response from the reader, which is then parsed and split into its return values. The function call with its parameters and the return values are displayed in the log box, see chapter 2.2

2.4.3 History

Each executed function with its request and corresponding response will be stored in the history box. If a function call with same parameters is already stored, it will be overwritten with the currently executed function. If identical function calls should be stored, the checkbox *Allow Duplicates* must be checked. Double clicking a stored function will execute it again with same parameters.

If a function has (not yet) got a response, it is displayed in gray color, otherwise it is black.



When a function call is selected by mouse click, the function builder shows the function with its parameters, to make it easy to start the function again with the same or with changed parameters. When a function call is selected by mouse or keyboard, the function call and the return values are displayed in the list view on the left of the history box.

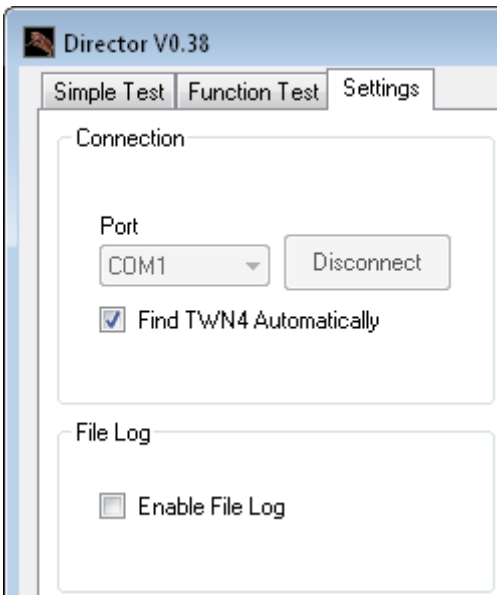
The position of a function call in the history box can be changed by clicking the buttons with the up or down arrow.

The *Clear* button deletes a selected item from the list while the *Clear All* button deletes the complete list.

The items of the history box can be saved to a text file and loaded again with the buttons *Save To File* and *Load From File*.

Providing that one or more items are in the history, they can be executed one time in sequence by clicking the button *Run Once* or in a cycle by checking the *Cycle* checkbox.

2.5 Settings



2.5.1 Connection

The automatic connection to a TWN4 can be disabled by de-checking the checkbox *Find TWN4 Automatically*. Then a port can be chosen and the connection process started manually by clicking the button *Connect*. If the connection was successful, the name of the button switches from *Connect* to *Disconnect* and the connection can be closed by clicking the button again.

2.5.2 File log

If the checkbox *Enable File Log* is checked, the log (see chapter 2.2) will also be written to the file `log.txt` in the same directory as the application.

2.5.3 Constants

There are a lot of predefined constants that can be used for function parameters (you can either type in the constant or the respective value). Many of these constants are also used

in the TWN4 firmware and have the same value there.

Constant	Value
CHANNEL_NONE	0
CHANNEL_USB	1
CHANNEL_COM1	2
CHANNEL_COM2	3
DIR_IN	1
DIR_OUT	0
COM_WORDLENGTH_8	8
COM_PARITY_NONE	0
COM_PARITY_ODD	1
COM_PARITY_EVEN	2
COM_STOPBITS_0_5	1
COM_STOPBITS_1	2
COM_STOPBITS_1_5	3
COM_STOPBITS_2	4
COM_FLOWCONTROL_NONE	0
ALLTAGS	0xFFFFF
REDLED	0x01
GREENLED	0x02
YELLOWLED	0x04
ALLLEDS	0x07
GPIO0	0x01
GPIO1	0x02
GPIO2	0x04
GPIO3	0x08
GPIO4	0x10
GPIO5	0x20
GPIO6	0x40
GPIO7	0x80

continued on next page. . .

Constant	Value
GPIO_PUPD_NOPULL	0
GPIO_PUPD_PULLUP	1
GPIO_PUPD_PULLDOWN	2
GPIO_OTYPE_PUSHPULL	0
GPIO_OTYPE_OPENDRAIN	1
BEEP_FREQUENCY_HIGH	2400
BEEP_FREQUENCY_LOW	2057
C0	523
C#0	554
D0	587
D#0	622
E0	659
F0	698
F#0	740
G0	784
G#0	831
A0	880
Bb0	932
H0	988
C1	1047
C#1	1109
D1	1175
D#1	1245
E1	1319
F1	1397
F#1	1480
G1	1568
G#1	1661
A1	1760
Bb1	1865

continued on next page. . .

Constant	Value
H1	1976
I2CMODE_MASTER	0x0000
I2CMODE_SLAVE	0x1000
I2CMODE_CHANNEL	0x2000
I2CMODE_ADDRESS_MASK	0x007F
KEYA	0
KEYB	1
DESF_AUTHMODE_COMPATIBLE	0
DESF_AUTHMODE_EV1	1
DESF_COMMSET_PLAIN	0
DESF_COMMSET_PLAIN_MACED	1
DESF_COMMSET_FULLY_ENC	3
DESF_KEYTYPE_3DES	0
DESF_KEYTYPE_3K3DES	1
DESF_KEYTYPE_AES	2
DESF_KEYLEN_3DES	16
DESF_KEYLEN_3K3DES	24
DESF_KEYLEN_AES	16
DESF_FILETYPE_STDDATAFILE	0
DESF_FILETYPE_BACKUPDATAFILE	1
DESF_FILETYPE_VALUEFILE	2
DESF_FILETYPE_LINEARRECORDFILE	3
DESF_FILETYPE_CYCLICRECORDFILE	4

Table 2.2: Constants used in TWN4Director