TWN4

System Overview

DocRev1, February 11, 2013

Elatec GmbH

Contents

1 What is TWN4?

TWN4 is the name of Elatec's most powerful and versatile series of RFID readers and writers. Here are some of the outstanding features:

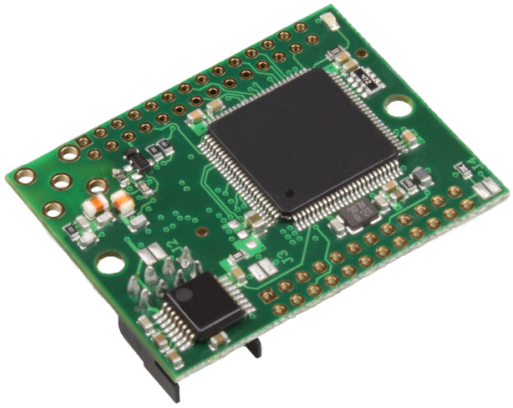
- Operation in two frequencies bands: 13.56 MHz (HF) and 125 kHz / 134.2 kHz (LF)
- Modular concept consisting of core modules, carrier boards, antennas and complete devices in housing.
- Security features such as slots for secure access modules or cryptogrphic functions.
- Possibility to write programs which are running on TWN4 itself (Apps).
- Standalone or host-based operation.

2 Products

2.1 TWN4 Core Module

The TWN4 family of RFID readers/writers is built around the TWN4 Core Module.

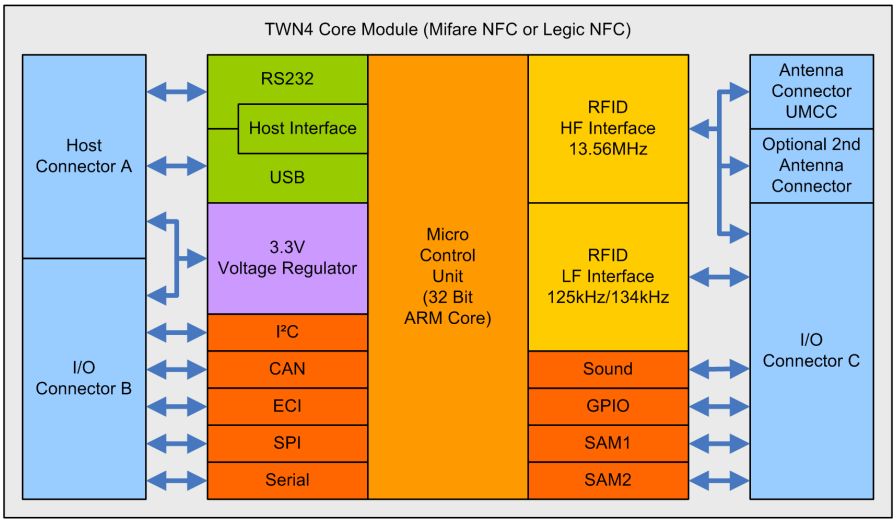
Here is a picture of the TWN4 Core Module Mifare NFC:



There are two different core modules available:

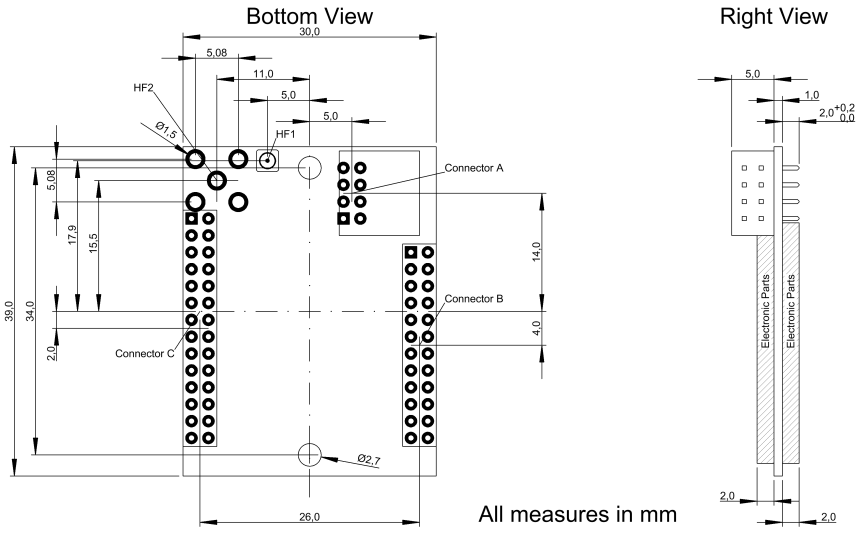
- TWN4 Core Module Mifare NFC
- TWN4 Core Module Legic NFC

The core module contains voltage regulator, control unit, RFID front ends and communication interfaces.



2.1.1 Dimensions

The dimensions of TWN4 Core Module are as follows.:



2.1.2 Connectors

The TWN4 Core Module has several connectors on-board. There are three connectors (A, B and C) which connect either to a carrier board or to a host. Furthermore, there is a antenna connector and a optional position for placing a coaxial connector.

The connectors A, B and C have identical pitch. Following type of connector is recommended:

- Dual row header
- Pitch 2.0mm
- Pin shape square
- Pin width 0.5mm x 0.5mm
- Length of pins appropriate to custom requirements

2.1.2.1 Connector A

The connector A is intended for connecting a cable to the TWN4 core module, which allows communication with a host. Typically, such a cable is either type USB or RS232. Elatec delivers various appropriate USB and RS232 cables from stock.

There is a right-angle connector mounted on-board, which is type Hirose, DF11 series.

Pin	Pin Name	Function
A1	UGND	USB Ground
A2	USB_D+	USB Data +
A3	UVCC	USB VCC
A4	USB_D-	USB Data -
A5	V24_RXD	RS232 RXD (Input)
A6	GND	Ground
A7	V24_TXD	RS232 TXD (Output)
A8	MGND	Cable Sense

Note:

If pin A6 and A8 are connected to each other this has two effects:

1. The firmware of TWN4 changes host channel to RS232
2. The integrated circuit, which is interfacing to voltage levels of RS232 is powered up.

2.1.2.2 Connector B

Pin	Pin Name	Function
B1	GND	Ground
B2	VIN	Unregulated input to on-board voltage regulator
B3	VREG	3.3V output from on-board voltage regulator
B4	VCC	3.3V power supply input
B5	RESET-	Low active TTL input with internal pull-up resistor for hard reset.
B6	PWRDWN-	Low active TTL input with internal pull-up resistor for turning off the voltage regulator.
B7	COM1_RX-	Low active TTL input with internal pull-up resistor of asynchronous RXD to COM1.
B8	COM1_TX-	Low active TTL output (push/pull) of asynchronous TXD from COM1.
B9	I2C_SDA	Data pin of I2C interface. No internal pull up.
B10	I2C_SCI	Clock pin of I2C interface. No internal pull up.
B11	CAN_RX	TTL RX pin of CAN interface. A external interface circuit is required.
B12	CAN_TX	TTL TX pin of CAN interface. A external interface circuit is required.
B13	ECI_MOSI	Pin MOSI of ECI
B14	ECI_MISO	Pin MISO of ECI.
B15	ECI_CLK	Pin CLK of ECI.
B16	ECI_ATTN-	Pin ATTN- of ECI.
B17	GND	Ground
B18	Res.	This pin is reserved for future purposes.
B19	SPI_MOSI	Pin MOSI of SPI interface
B20	SPI_MISO	Pin MISO of SPI interface
B21	SPI_SCK	Pin SCK of SPI interface
B22	SPI_SS-	Pin SS- of SPI interface
B23	Res.	This pin is reserved for future purposes.
B24	Res.	This pin is reserved for future purposes.

2.1.2.3 Connector C

Pin	Pin Name	Function
C1	GND	Ground
C2	ANT_HF	Together with pin C1, this pin builds a 50 ohm output for connecting external 13.56MHz antennas
C3	ANT_LF1	Output 1 for connecting external 125 kHz / 134.2 kHz antennas.
C4	ANT_LF2	Output 2 for connecting external 125 kHz / 134.2 kHz antennas.
C5	Res.	This pin is reserved for future purposes.
C6	SPK+	Digitally modulated output for a speaker. Second connection for the speaker is ground. The impedance of the speaker should be greater than 24 ohm.
C7	IO0	GPIO0, I/O pin for general purposes.
C8	IO1	GPIO1, I/O pin for general purposes.
C9	IO2	GPIO2, I/O pin for general purposes.
C10	IO3	GPIO3, I/O pin for general purposes.
C11	IO4	GPIO4, I/O pin for general purposes.
C12	IO5	GPIO5, I/O pin for general purposes.
C13	IO6	GPIO6, I/O pin for general purposes.
C14	IO7	GPIO7, I/O pin for general purposes.
C15	SAM1_CLK	Clock output for SAM1
C16	GND	Ground
C17	SAM1_IO	I/O line for SAM1
C18	SAM1_RST	Reset output for SAM1
C19	SAM2_CLK	Clock output for SAM2
C20	GND	Ground
C21	SAM2_IO	I/O line for SAM2
C22	SAM2_RST	Reset output for SAM2

continued on next page. . .

Pin	Pin Name	Function
C23	COM2_RX-	Low active TTL input with internal pull-up resistor of asynchronous RXD to COM2.
C24	COM2_TX-	Low active TTL output (push/pull) of asynchronous TXD from COM2.
C25	Res.	This pin is reserved for future purposes.
C26	Res.	This pin is reserved for future purposes.
C27	Res.	This pin is reserved for future purposes.
C28	VCC	3.3V power supply input or output for supplying external components. Internally connected to B4.

Note:

- The nominal inductance for an external 125 kHz/134.2 kHz antenna is 490 µH. The series resistance of the antenna should be lower than 10 ohms.

2.1.2.4 Connector HF1

HF connector 1 is a UMCC series 50-ohms output, which is connected internally in parallel to pins C1 and C2.

2.1.2.5 Connector HF2

Position of HF2 offers the possibility to place another 50-ohm connector. It is connected internally in parallel to pins C1 and C2. There are several series of RF connectors, which can be used for position HF2, like SMA, SMB, SMC, MCX.

2.1.3 Jumpers

There are several jumpers on-board of the TWN4 Core Module. Depending on the requirements these jumpers can be soldered together.

Jumper	Function
J1	The function is identical to pins A6 and A8. If J1 is closed, the RS232 interface is turned on and the host channel is assumed to be RS232.
J2	Same function like J1 but other side of PCB.
J3	This jumper must be closed, if TWN4 Core Module is powered via connector A, e.g. from USB. It connects VCC from the Core Module to the on-board voltage regulator, which is supplied from connector A. If TWN4 Core Module is mounted on a carrier board, this connection can be avoided by connecting pins B3 and B4 at the carrier board, which results in exactly the same functionality.
J4	This jumper is for internal purposes only.

2.2 TWN4 OEM PCBs

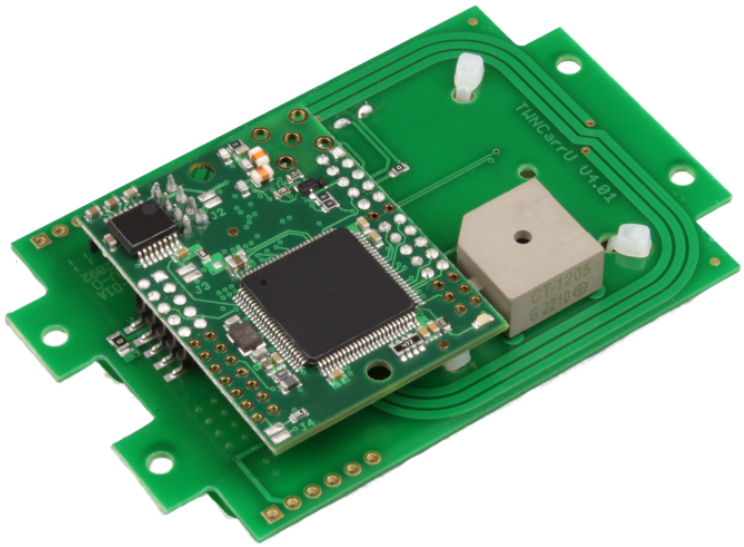
An OEM PCB combines all components including antennas and core module, which are required to build a full functional device. A OEM PCB consists of a carrier board, where all other parts ar placed. Only cable and power supply is required to bring device into operation.

Following OEM PCBs are available:

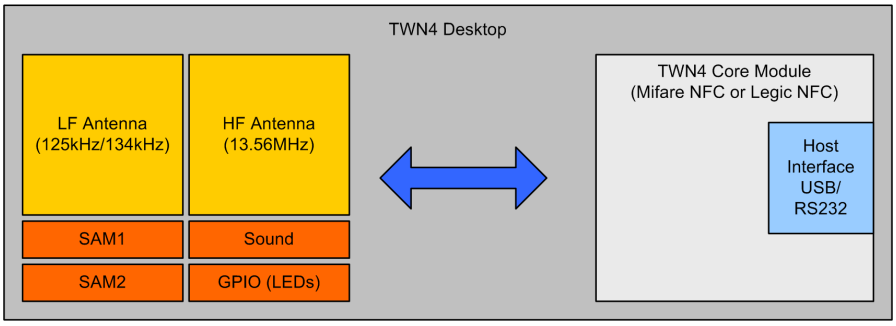
- TWN4 Mifare NFC OEM PCB (Desktop)
- TWN4 Legic NFC OEM PCB (Desktop)
- TWN4 Mifare NFC OEM PCB (Panel)
- TWN4 Legic NFC OEM PCB (Panel)

2.2.1 TWN4 Desktop OEM PCB

Here is a picture of a TWN4 Desktop OEM PCB (it is a Mifare NFC):



The block diagram looks as follows:



2.2.1.1 Dimensions & Pinout

The dimensions and pinout of TWN4 OEM PCB as follows.:

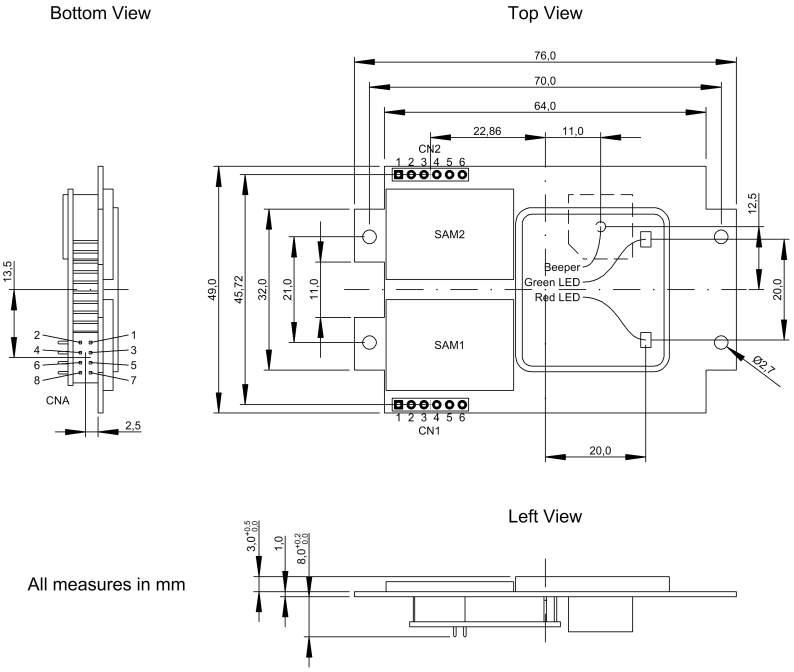
Connector CNA*	
Pin	Signal
1	USB/Supply Ground (black Wire)
2	USB Data+ (green Wire)
3	USB/Supply 5V (red Wire)
4	USB Data- (white Wire)
5	RX232 RxD (Input)
6	J2-1**
7	RX232 TxD (Output)
8	J2-2**

* Hirose DF11 series, 2mm pitch

** RS232 operation is achieved by either closing jumper J2 on the PCB or by connecting pin 6 and pin 8 of CNA.

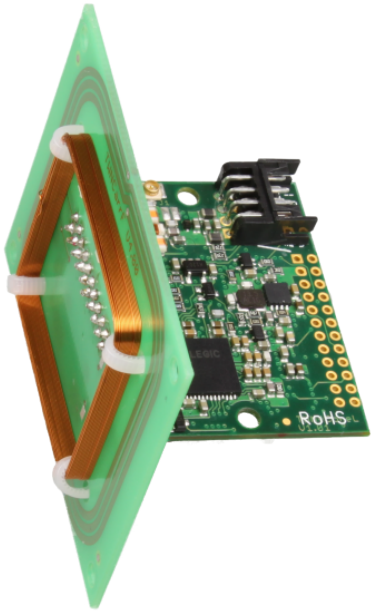
Connector CN1	
Pin	Signal
1	Supply Ground
2	5V Supply
3	Logic Input RxD (active low)
4	Logic Output TxD (active low)
5	GPIO2
6	GPIO3

Connector CN2	
Pin	Signal
1	Supply Ground
2	3.3V Supply
3	SCL (I²C)
4	SDA (I²C)
5	RESET (active low)
6	POWERDOWN (active low)

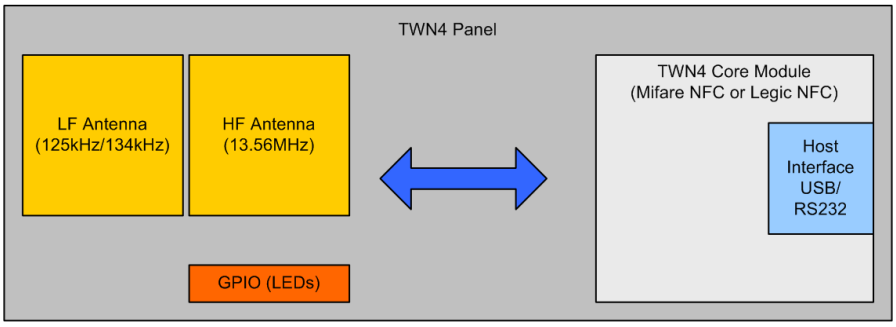


2.2.2 TWN4 Panel OEM PCB

Here is a picture of TWN4 Panel OEM PCB:



The block diagram looks as follows:



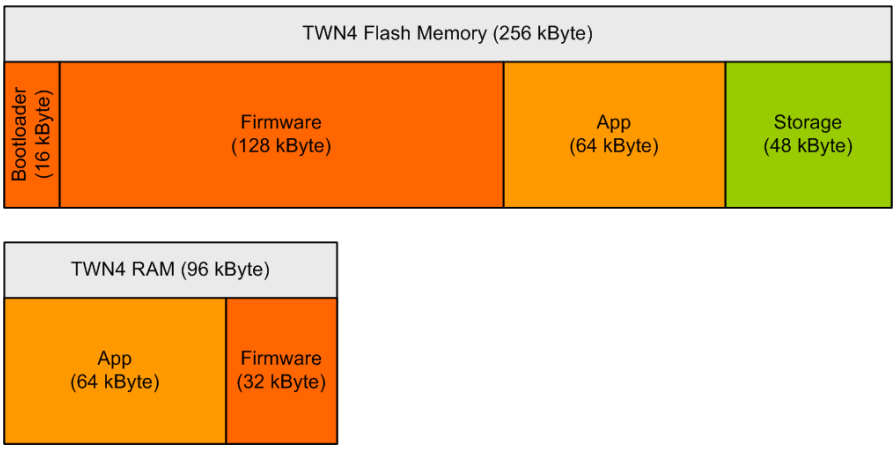
2.3 TWN4 Desktop

A TWN4 Desktop is a device, which contains TWN4 Desktop OEM PCB, housing and USB or RS232 cable for host connection. It is available either in black or white.

3 Firmware

3.1 Memory View

The TWN4 Core Module has internal 256 kBytes of flash and 96 kBytes of RAM. The memory is devided into several sections as shown in the following diagram:



3.1.1 Boot Loader

The boot loader is the entry point for the firmware after powering up TWN4 or after a reset.

Only the boot loader provides functions for programming new firmware or Apps. This means in order to program either a new firmware or another App, the boot loader must be entered.

3.1.2 Firmware

The firmware occupies most space in flash memory. It provides functions for accessing IO or doing RFID operations. Furthermore, the execution of an App is controlled by the firmware.

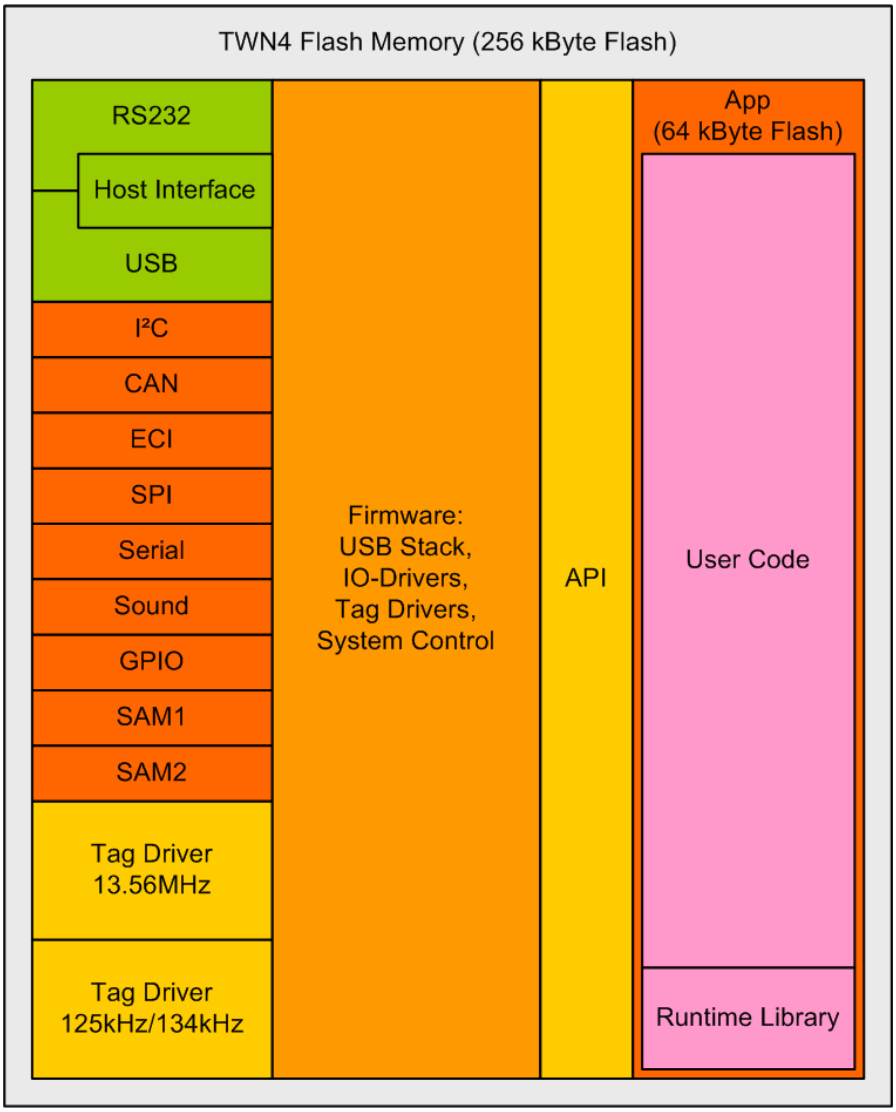
The firmware cannot be read back from a TWN4.

3.1.3 App

The App is the part of flash memory, which specifies the behaviour of a TWN4. Due to this, the programmer of the App has full control over the behaviour of the final application. A App can be programmed by the customer. In order to do so, there is a developer pack provided by Elatec.

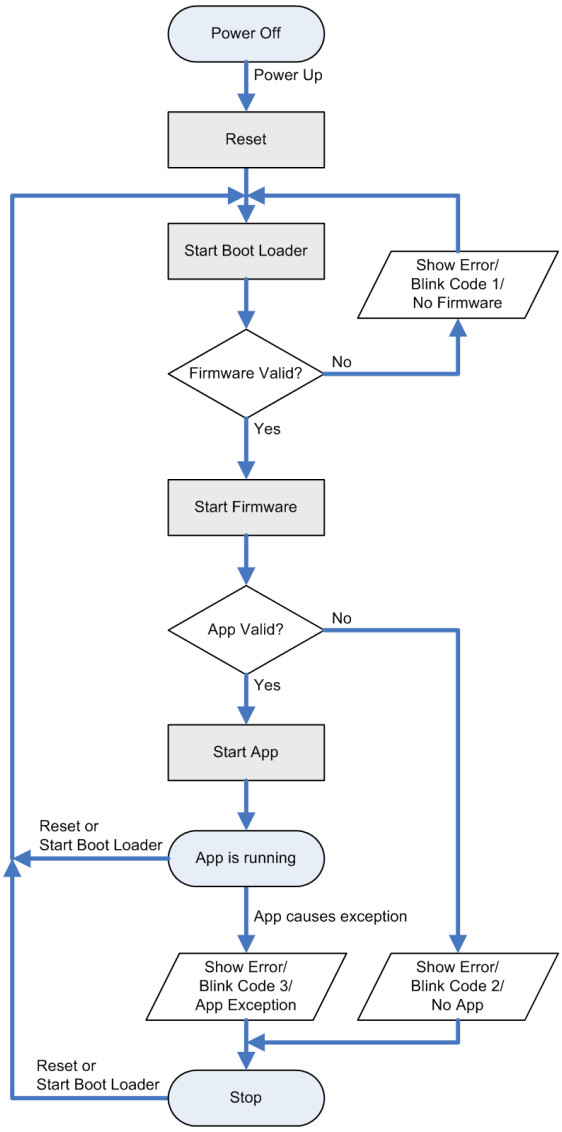
An App cannot be read back from a TWN4. This allows to store secret keys and other cryptographic functionality as part of an App. Furthermore, the possibility to clone a device is avoided and the intellectual property is protected.

3.2 Functional Units



3.3 Firmware Startup Sequence

The diagram below is showing the sequence of how boot loader, firmware and App are started:



3.4 Firmware Error Conditions

There are several reasons, because the firmware may run into a unwanted condition. If this happens, the condition is shown by a on-board diagnostic red LED of the TWN4 Core Module. The LED is signalling the error code by a number of flashes separated by a pause.

This signalling is called blink code. Following blink codes are defined:

- Flash 1 time: There is no valid firmware installed on TWN4. This might be caused, if programming a new firmware onto TWN4 is interrupted by a power failure. In this case, the programming must be started from the beginning.
- Flash 2 times: There is no valid App installed on TWN4. This might be caused, if programming a new App onto TWN4 is interrupted by a power failure. In this case, the programming must be started from the beginning.
- Flash 3 times: The running App caused an exception. A exception is a invalid memory access or invalid program instruction. An App is allowed to access it's own memory space only (64kByte ROM/64kByte RAM).

3.5 Backdoor for Starting the Boot Loader

During development of new Apps and under undefined circumstances, the situation might arise, the starting the boot loader is not possible anymore. In such a situation, it is useful to start the boot loader manually. This can be achieved by connecting two pins of the TWN4 core module together and do a power cycle or reset. The two pins, which have to be connected together are C25 and C28 of the TWN4 Core Module.

3.6 App & Firmware Images

Elatec provides App and firmware images to the customer. App and firmware images can be distginguished by the extension of their filename.

3.6.1 App Images

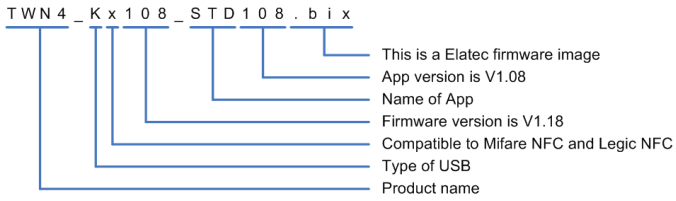
An App image has the extension .twn4.app. After compiling source code, the result is a App image with that extension. It can be programmed using AppBlaster into TWN4

3.6.2 Firmware Images

A firmware image has the extension .bix. Normally, such a file contains a firmware for TWN4 and a appropriate App for the intended purpose.

3.6.2.1 Naming Scheme

There is a standard naming scheme for firmware images, which are given to the customer. This is how the name of a firmware image is constructed:



There are several types of USB stacks available:

- 'K': USB HID device class (keyboard)
- 'C': USB CDC device class (virtual COM port)

A USB stack might be combined with following Apps as follows:

- "STD": Standard App, TWN4 is searching for transponder and forwarding the ID to the host
- "PRS": Simple Protocol, TWN4 is running under the control by a host. TWN4 executes commands, sent by the host and returns response. In this way, nearly all TWN4 system functions can be execute remotely by the host. There is separate documentation and software available for using this mode of operation.

The mentioned USB types and Apps lead to the following images, which are shipped as part of the developer pack:

- "TWN4_Kx118_STD108.bix": Keyboard emulator, which beeps, blinks LEDs and sends ID to the host.
- "TWN4_Cx118_STD108.bix": Virtual COM port, which beeps, blinks LEDs and sends ID to the host.
- "TWN4_Cx118_PRS108.bix": Virtual COM port with command interface "Simple Protocol"

4 TWN4 Developer Pack

The TWN4 developer pack contains all software and documents necessary to operate, program and configure TWN4.

4.1 Installation

You received the TWN4 developer pack as zip file. In order to install the package, please follow these steps:

- Create a empty directory on your hard disk
- Unzip the entire content of the zip file into this empty directory
- You're done!

4.1.1 System Requirements

These are the minimum system requirements for a serious use of the TWN4 Developer Pack:

- Operating system: Microsoft Windows XP or later, 32 or 64 bit
- Microsoft .NET Framework 3.5
- Processor (CPU): 2 GHz
- Hard Disk: 200 MB
- RAM: 2 GB

4.2 Starting Point: AppBlaster

You already installed the developer pack. Therefore, you can find the program AppBlaster.exe on the top directory of the TWN4 developer pack. Start AppBlaster now.

4.3 Creating Apps

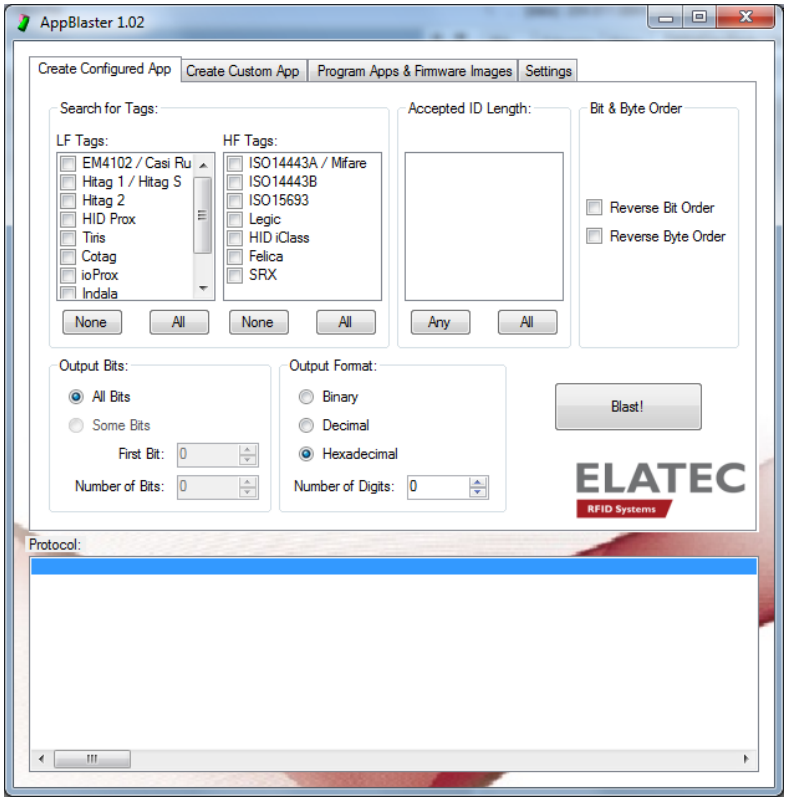
There are several ways to create an App for TWN4. The simplest way is using AppBlaster, which provides several ways on its own depending on the requirements of the user.

4.3.1 Reference: The Standard App

TWN4 is delivered with a default App. This App is called standard App. The source code of this App is part of the developer pack and can be found in the subdirectory Apps\app_standard.c. The source code of this App is a good starting point for doing first experiments in App programming of TWN4.

4.3.2 Method 1: Create Configured App

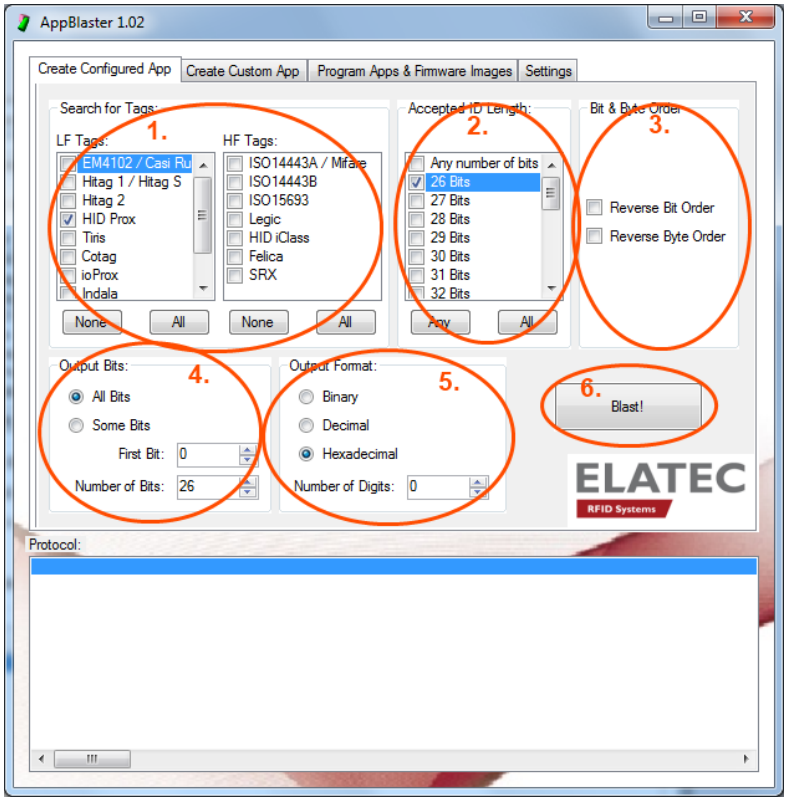
The easiest way to modify behaviour of TWN4 is to create a configured application. To do so select tab sheet "Create Configured App".



Creating a configured App allows you to modify the output data and format of TWN4 according to your requirements. Select required options as follows:

1. Select the type of transponder or transponders in the two lists of LF tags and HF tags. You may combine all tags as needed in your application.
2. Select the number of bits, which you are accepting in your application. Example: A Mifare transponder could deliver 32 or 56 bits depending on the built-in chip type. If you are using 32-bit-transponders only you can shrink the number of accepted transponders to the types, which are in use by your application.

   If you select one specific number of bits as ID length, you can do some more sophisticated settings in step 4. Otherwise you skip step 4.
3. In order to maintain compatibility with 3rd-party products or existing content of a user database, you can optionally reverse the bit or byte order of a read ID.
4. This sections allows to take some specific bits of the ID only. A typical case is e.g. HID Prox, which often delivers an ID, let's say with 26 bits, where you might be interested in the 16-bit personal number only. In this case you would select "Some Bits" instead of "All Bits", set "First Bit" to 9 (the first bit of the personal number) and "Number of Bits" to 16.
5. After having selected the data of interest you now have to specify, how data is formatted. If you select "Binary" or "Decimal" and 0 for "Number of Digits", the number of digits depends on the number itself. If you select "Hexadecimal", the number of digits is a multiple of 2 and chosen in a way, that all bits of the ID fit into that number. E.g. 26 bits lead to 8 hexadecimal digits.
6. Now you are ready to program the selected configuration into a TWN4. Elatec calls this step "Blast!" due to the fact, that it is easy and fast. Easy and fast = Blast!



4.3.3 Method 2: Create Custom App

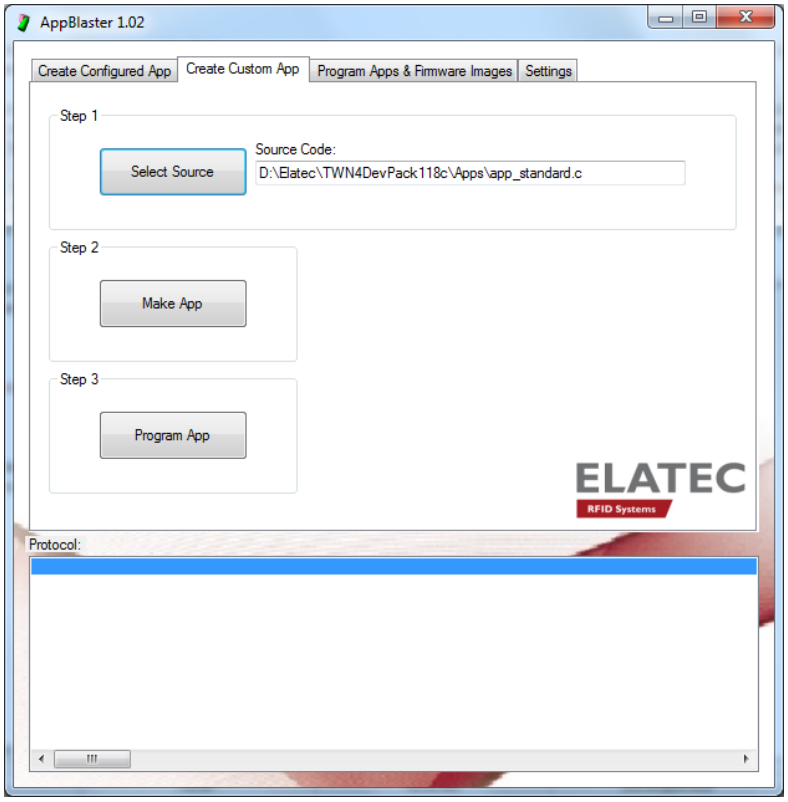
For several reasons, there might be a need for creating a custom App:

- There is some source code, you got from 3rd party.
- You have some specific requirements for reading data from a transponder, where data is in a data section of the transponder or some kind of authentication is required to do reading of data.
- There is a requirement to change entire behaviour of TWN4. This could be e.g. implementing some kind of host based communication, modifying behaviour of beeper, LEDs or doing input/output via other interfaces of TWN4.

The steps for creating a custom App from within AppBlaster are quite simple and self-explaining:

1. In "Step 1" choose the file containing the source code of your App.
2. In "Step 2" start creation of your App. This step will compile and link the source code and furthermore create an output file, which is compatible to the TWN4 tool chain. The result is a file located in the same directory as the source with the extension ".c" replaced by ".twn4.app". This, the result is a compiled App.

3. In the last "Step 3", you program this created App to a connected TWN4.



4.3.4 Method 3: make

Method 1 and method 2 do have limitations, which might not be acceptable by advanced users. These are:

- Only one file which contains the source code for the entire App. Especially bigger projects do have a demand for splitting source code into several source and header files.
- Requirement to compile entire source code even in case of small changes.
- AppBlaster is a solution, which is away from well-known programming environments such as Eclipse.

The tools for creation of an App from a command line are part of the TWN4 developer pack. AppBlaster is a graphical user interface (GUI) for the tool chain which resides in the subdirectory Tools\. In order to become more convenient with how an App is built, there is a option to display the command line parameters during creation of an App. This option can be found in "Settings" of AppBlaster.

4.4 Programming Apps & Firmware Images

If you have a compiled App or a firmware image, you have the possibility to program such files from within AppBlaster into a TWN4 device. In order to do this select tab sheet "Program Apps & Firmware Images" and follow these steps:

1. In "Step 1" choose the file of interest: This might be either a firmware image (file extension ".bix") or a compiled App for TWN4 (file extension ".twn4.app")
2. In "Step 2" click button "Program Image". This will immediately start the programming sequence. After a few seconds, the step should be completed.