

TWN4

AppBlaster Config Cards User Guide

DocRev10, May 3, 2019



Elatec GmbH

Contents

1	Overview	4
2	System Requirements	5
3	Installation	6
4	Starting AppBlaster	7
5	Programming Firmware Image	8
6	Read Version of Connected USB Devices	9
7	Programming Firmware of BLE Module	10
8	Configurable Project	12
8.1	Step 1: Select an Application Template	13
8.2	Step 2: Add Transponder Types	14
8.2.1	Remove Transponder Types	15
8.3	Step 2a: Specify Data Source	16
8.4	Step 2b: Specify Bit Manipulation	17
8.5	Step 2c: Specify Output Format	17
8.6	Step 3: Specify Prefix, Delimiter, Suffix (Optional)	18
8.7	Step 4: Signalling/Behaviour (Optional)	19
8.8	Step 5: Specify Options (Optional)	20
8.9	Step 6: Specify Version Information (Recommended)	21
8.10	Step 7: Create Image	22
8.11	Step 8: Program Image	23
8.12	Step 9: Save Project (Recommended)	24
9	Configurable Project (OSDP)	25
9.1	OSDP Reader Model	25
9.2	OSDP Standard Properties	26
9.3	OSDP Custom Properties	26
10	Source Code Project	27
10.1	Step 1: Specify Type of USB	28
10.2	Step 2: Specify Source Code Module	28
10.3	Step 3: Specify Version Information (Recommended)	28
10.4	Step 4: Create Image	29
10.5	Step 5: Program Image	29
10.6	Step 6: Save Project (Recommended)	29
11	TWN4 CONFIG / Configuration Cards	30
11.1	Introduction	30
11.2	Requirements	31
11.3	TWN4 CONFIG Card	32
11.4	TWN4 CONFIG Programmer	32
11.5	Creating a Configuration Card	33
11.5.1	Step 1: Provide Configuration Information	33
11.5.2	Step 2: Connect TWN4 CONFIG Programmer	33
11.5.3	Step 3: Program a TWN4 CONFIG Card	34
11.6	Configuring TWN4 with a Configuration Card	35
11.7	Reading Information from the Card	36
12	Creation of Apps with make	37

13	Setting Up AppBlaster	38
14	Disclaimer	39

1 Overview

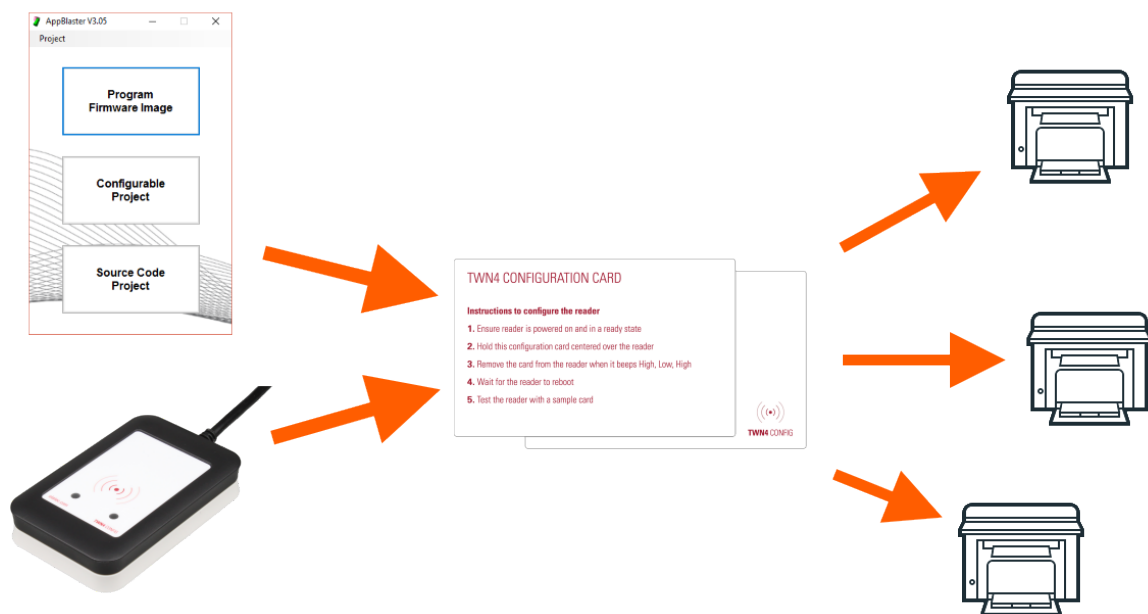
You received the AppBlaster as part of the developer pack for TWN4, which is distributed as a zip file.

AppBlaster is a program, which allows to configure and program TWN4 for a specific application.

There are three ways to prepare TWN4 for operation:

1. Directly program an appropriate firmware image into TWN4
2. Interactive configuration of TWN4 with AppBlaster
3. Write an App for TWN4 in programming language C

Moreover, TWN4 now can be configured with configuration cards via RF interface:



The following sections describe in detail, how this can be achieved.

2 System Requirements

There 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

3 Installation

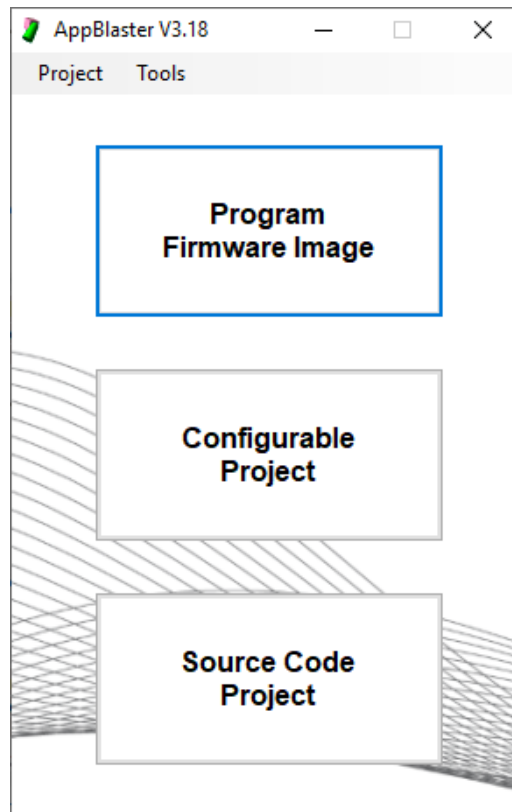
You received AppBlaster as part of the developer pack for TWN4, which is distributed as a zip file.

In order to install the package, please follow these steps:

- Create an empty directory on your hard disk
- Unzip the entire content of the zip file into this empty directory
- The program AppBlaster can be found in the top directory of the TWN4 developer pack.

4 Starting AppBlaster

In order to start AppBlaster, move to base directory of the installed TWN4 development pack and execute program AppBlaster.exe. The start dialog appears:



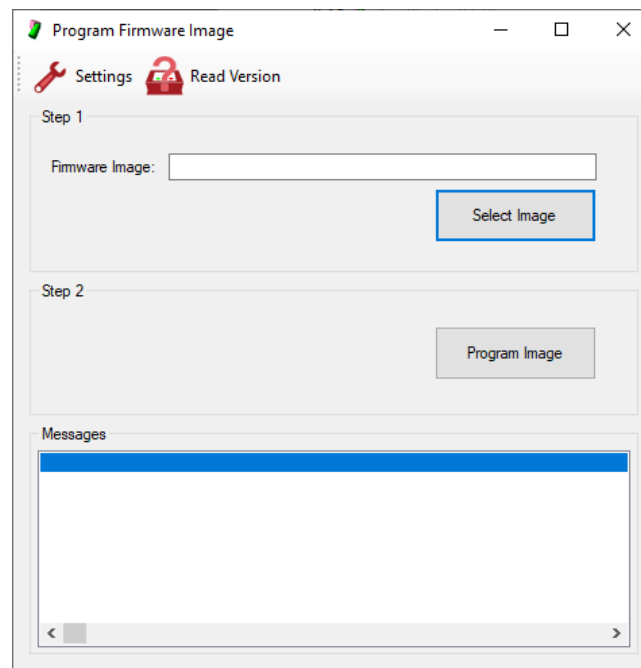
The start dialog offers three choices. In addition, an existing project can be loaded from disk via selection in the menu.

Depending on the task being achieved, press appropriate button:

- Button "Program Firmware Image": Program TWN4 with a firmware image
- Button "Configurable Project": Start a new project for TWN4, where setup is done via an interactive configuration.
- Button "Source Code Project": Write your App for TWN4 in C programming language.
- Menu "Project - Load": Load a previously saved project. This might be either a configurable project or a source code project.

5 Programming Firmware Image

If you got a compiled firmware image, there is the possibility to program such files from within AppBlaster into a TWN4 device. In order to do this, click button "Program Firmware Image" in the start dialog. Another dialog opens:



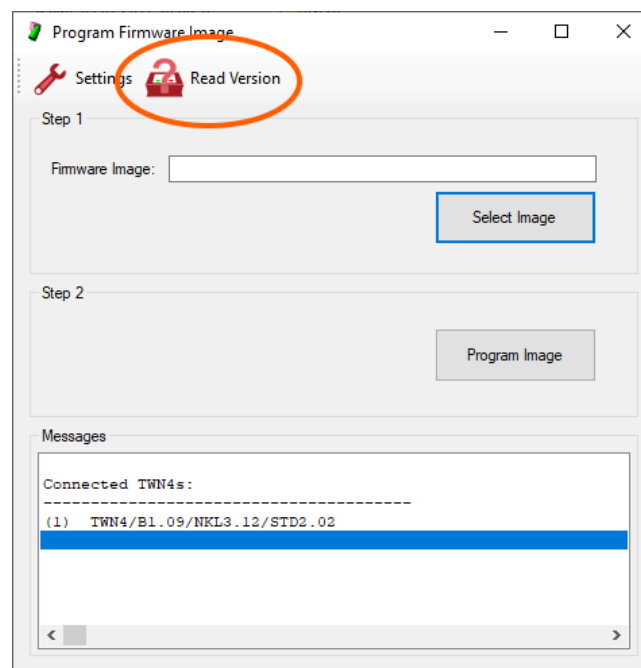
Follow these steps:

1. In "Step 1" choose the file of interest by clicking button "Select Image". Another way is to enter the known path and file name directly into the text box. The file extension for appropriate firmware images is ".bix".
2. In "Step 2" click button "Program Image". This will immediately start the programming sequence. After a few seconds, the step will be completed.

6 Read Version of Connected USB Devices

Following steps:

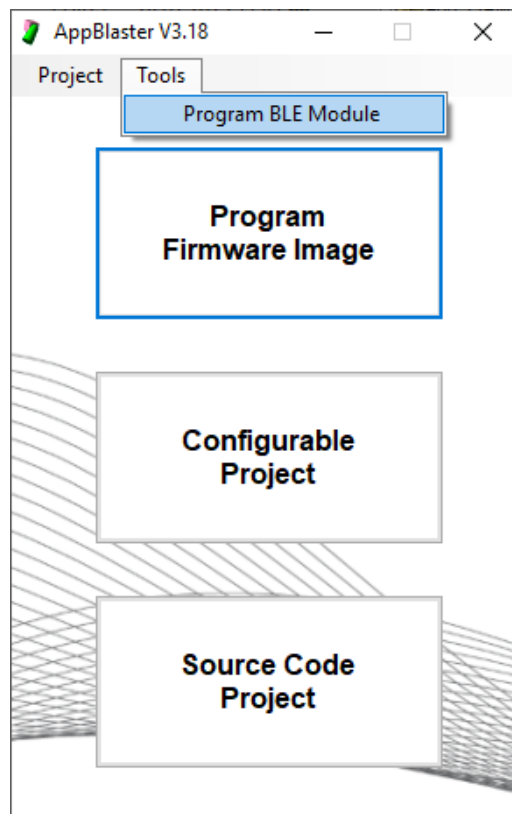
1. In the start dialog click button "Program Firmware Image". The dialog "Program Firmware Image" appears.
2. Click "Read Version" from the tool bar. AppBlaster lists all TWN4, which can be found on the USB bus.



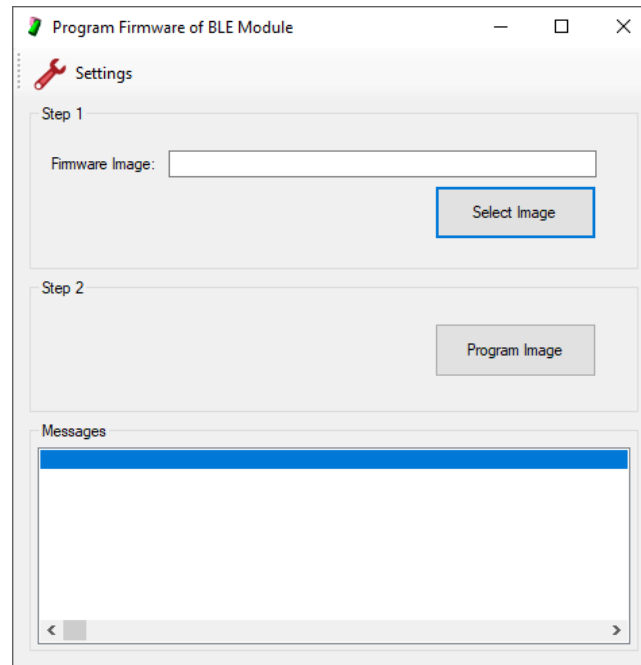
7 Programming Firmware of BLE Module

Following steps:

- In the start dialog select Tools - Program BLE Module. A dialog opens, which guides through the next steps.



- In "Step 1" choose the file of interest by clicking button "Select Image". Another way is to enter the known path and file name directly into the text box. The file extension for appropriate BLE firmware images is ".gbl".
- In "Step 2" click button "Program Image". This will immediately start the programming sequence. After a few seconds, the step will be completed.

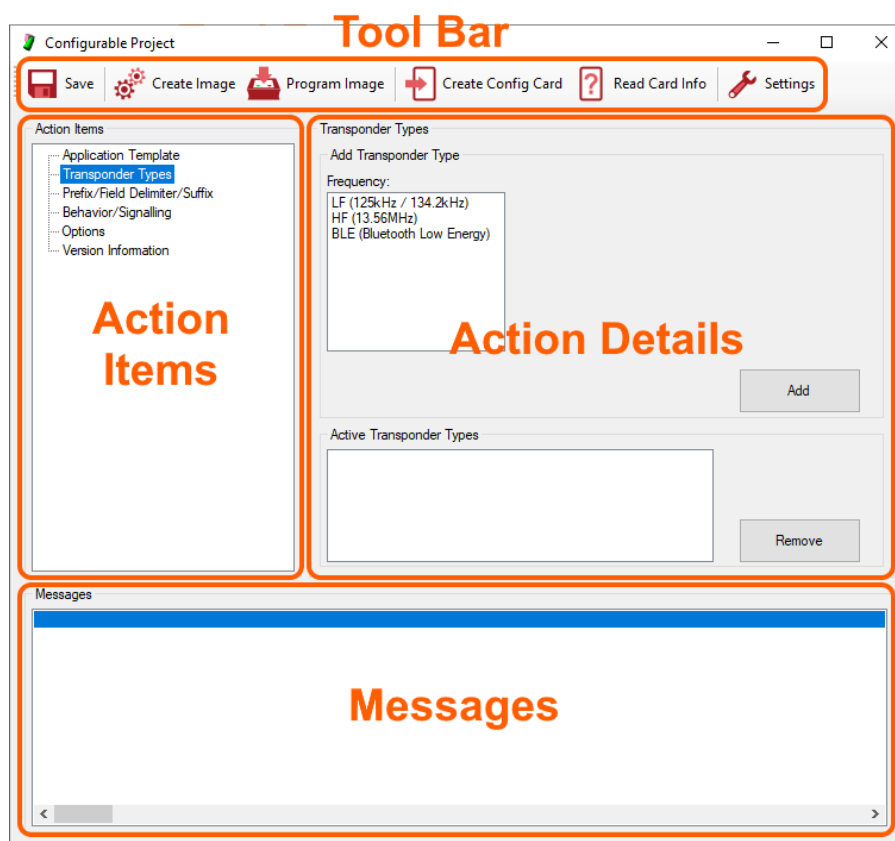


8 Configurable Project

In order to create a configurable project, click button "Configurable Project" in the start dialog. Another dialog appears, the project dialog.

This project dialog is separated into several sections, which allow user input and do output from the program:

- Tool Bar: Buttons in the tool bar let you start specific actions at any time
- Action Items: There are several action items, which can or must be done to complete a project
- Action Details: For every action item, there are details which can or must be entered
- Messages: AppBlaster is giving feedback on actions, which are started from the tool bar. Furthermore, error messages are displayed in this area.

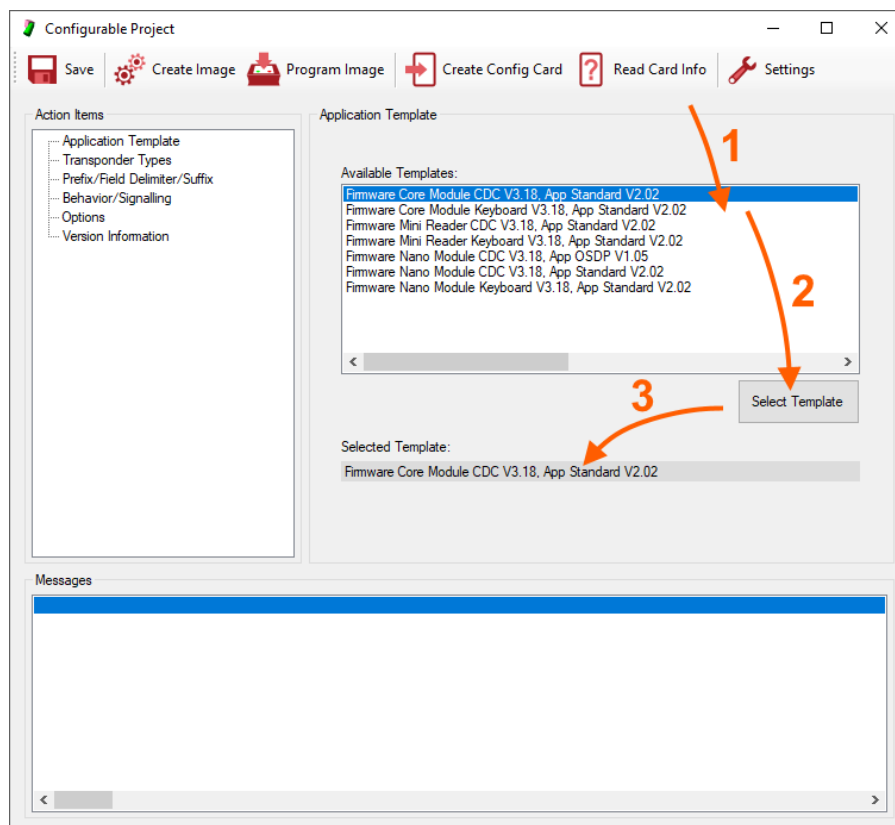


Once a configured project has been started, there are several steps, which must or can be followed.

8.1 Step 1: Select an Application Template

TWN4DevPack and AppBlaster is delivered with several Application Templates. An Application Templates defines the type of Application to be created. It has an influence on:

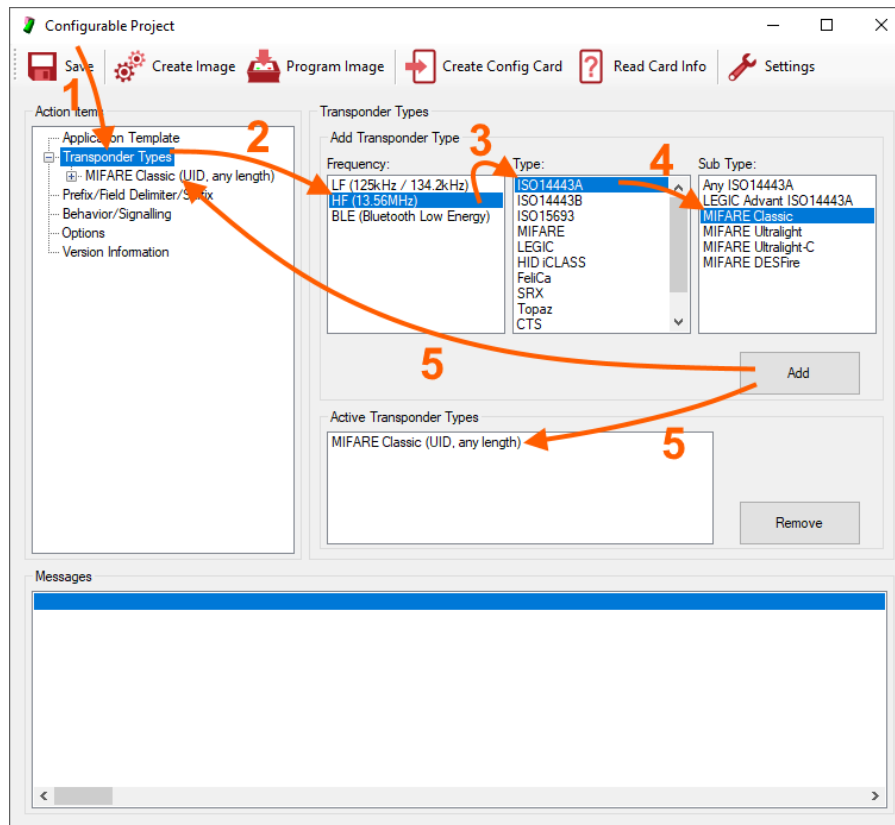
- Type of hardware (Core, Mini, Nano). Core is related to TWN4 Core Module, TWN4 MultiTech. Mini is related to TWN4 Mini. Nano is related to TWN4 Nano Module, TWN4 MultiTech 2, TWN4 MultiTech 3, TWN4 Palon.
- Type of USB (USB HID Keyboard or USB CDC)
- Type of application (Standard, OSDP).



1. Select appropriate template in the list box "Available Templates". Currently, there are six templates available: For TWN4 Core, TWN4 Nano and TWN4 Mini, either USB CDC (virtual COM port) and USB keyboard. If you would like to operate TWN4 via RS232, it makes no difference if you choose CDC or keyboard. You at least have to select one of both.
2. Double click template or click button "Select Template"
3. The selected template is displayed in a text box.

8.2 Step 2: Add Transponder Types

In your application, one or more types of transponders might be involved. Here is the path through the dialog to add one type of transponder:



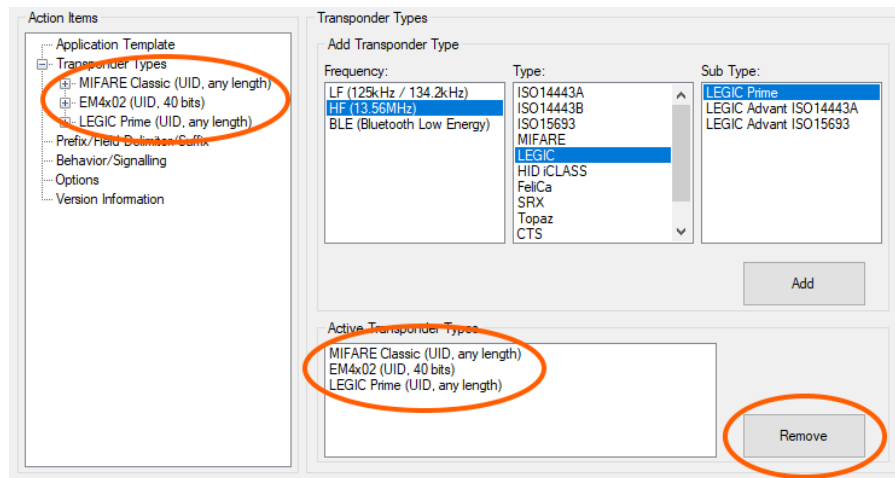
Following (sub-)steps:

1. Select "Transponder Types" from "Action Items".
2. In the list box "Frequency" select, if the transponder type is LF (125kHz or 134.2kHz) or HF (13.56MHz). A selection of the frequency uncovers another list box, which displays according types of transponders in the selected frequency band.
3. Select according type of transponder in the list box.
4. In many cases, there is a need to select a specific sub type. In these cases, another list box is displayed, which allows selection of appropriate sub type.
5. Now, button "Add" can be clicked. The selected type of transponder is added at two places in the dialog: 1. in the list of "Active Transponder Types" and 2. in "Action Items"

Continue with item 2, if the application requires more types of transponder to deal with.

8.2.1 Remove Transponder Types

If a transponder type was added unintentionally, it can be removed.



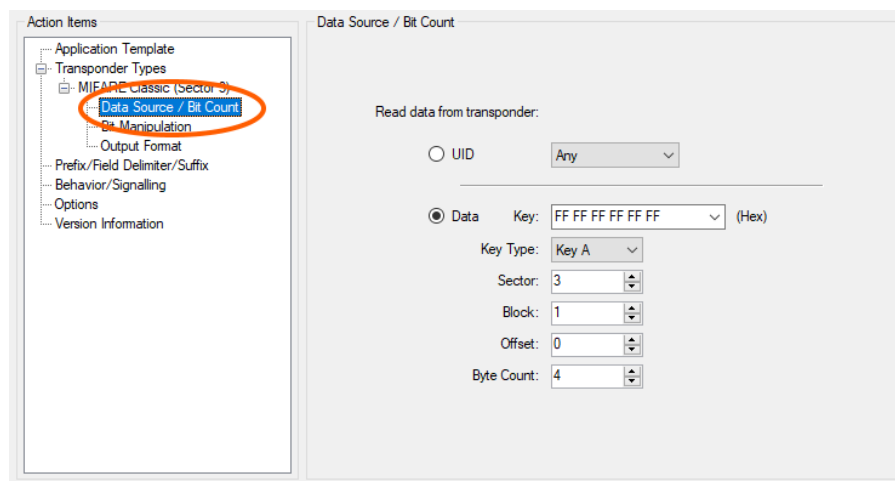
1. Select "Transponder Types" from "Action Items".
2. In the list box "Active Transponder Types" select transponder type to be removed.
3. Finally, click button "Remove" to remove the transponder type in question.

8.3 Step 2a: Specify Data Source

The data source specifies the place on a transponder, where data in question is stored. Depending on the type of transponder such storage places are called UID, sector, block, segment, page, file or something similar. In many cases, simply the UID should be read ("data source") from the transponder. It is the default setup, once a transponder type is added.

The specification of the data source depends on the selected type of transponder. In many cases and by default, the UID is treated as data source.

Here is an example, how a block within a sector of MIFARE Classic can be used as data source:

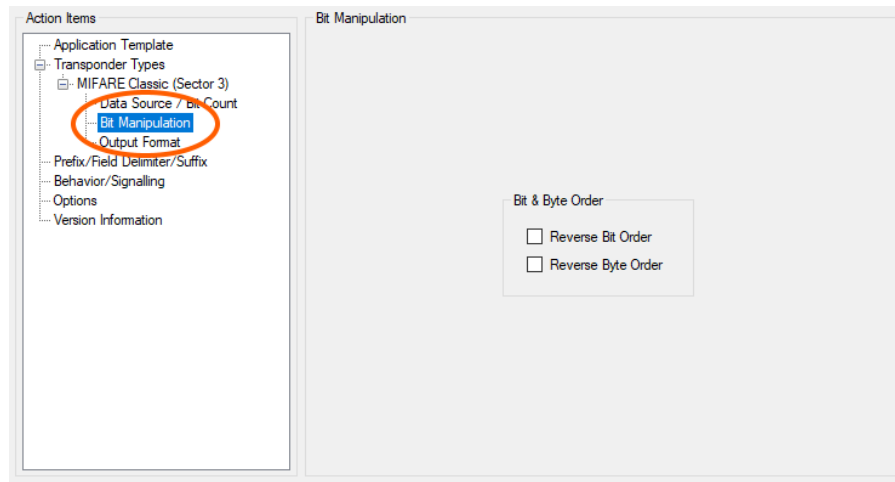


- Instead of reading UID choose "Data", which means read data from a specific block in a sector of MIFARE Classic.
- Specify the key, which gains access to the sector in question. There are some well-known default keys, which can be selected directly from the combo box.
- Select the sector number (0 to 15) in question.
- Select the block number (0 to 3) within the sector.
- Select offset within the block, where the data is stored.
- Finally, select the number of bytes, which data in question occupies. Depending on the specified offset, this can be up to 16 bytes (the size of a MIFARE block).

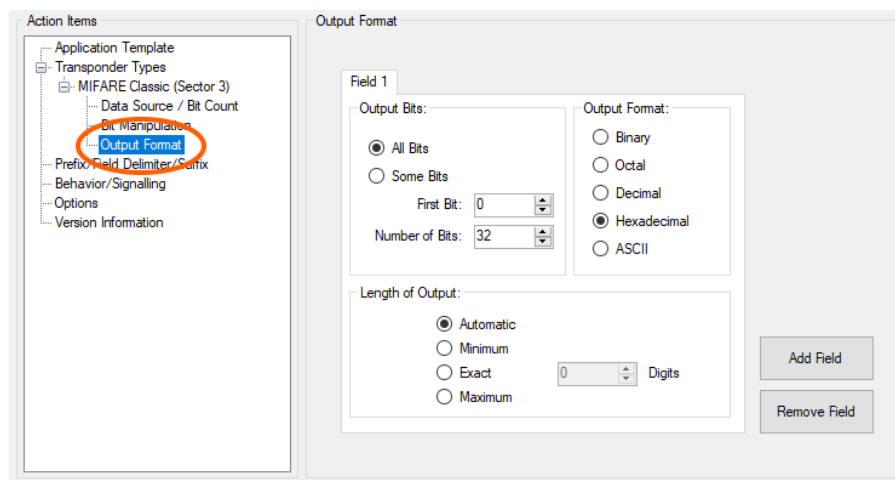
Notes:

- For other types of transponders, the specification of the data source can be different but works in similar manner.
- Many transponders do allow specification of UID as the only source of data.

8.4 Step 2b: Specify Bit Manipulation

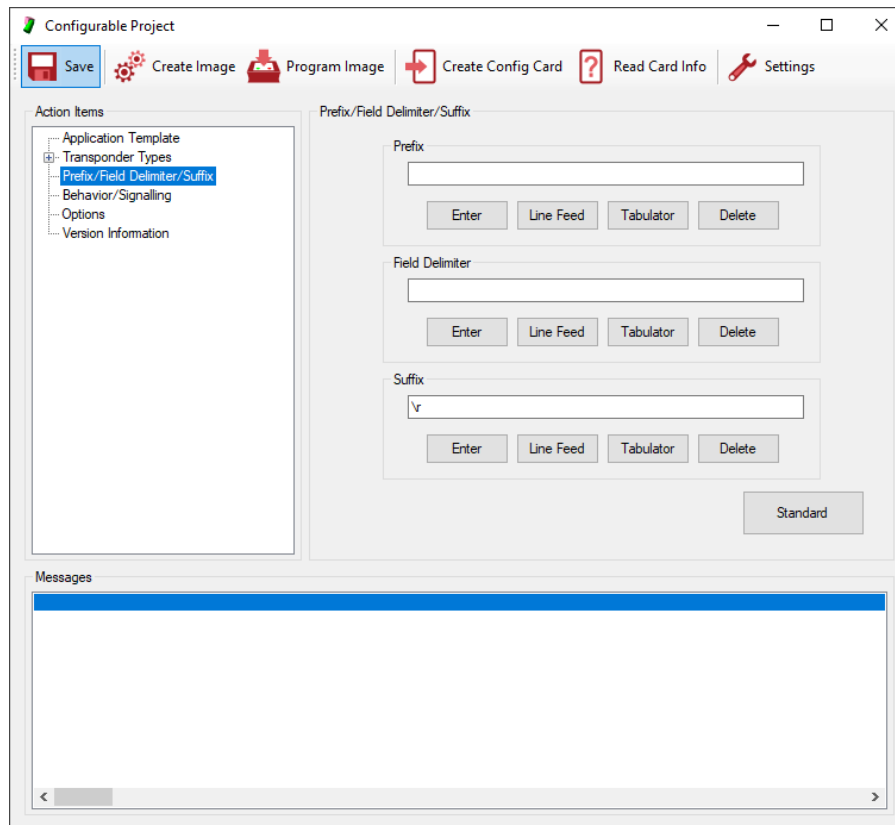


8.5 Step 2c: Specify Output Format



8.6 Step 3: Specify Prefix, Delimiter, Suffix (Optional)

AppBlaster allows to specify a prefix, delimiters and suffix for the output data. There are several special characters available, which might be helpful for the specific project. By default, there is no prefix sent, no delimiters and "Enter" is sent as suffix. Delimiters separate fields specified in the output format. Two or more fields are needed for a delimiter being sent.



8.7 Step 4: Signalling/Behaviour (Optional)

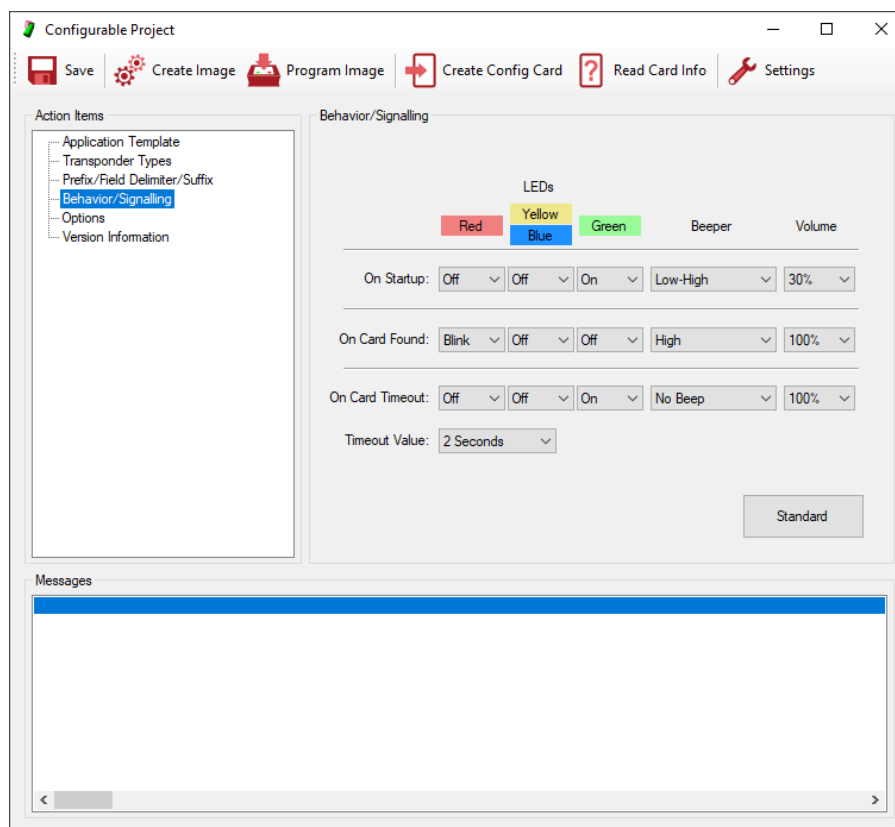
During operation of TWN4, there are several types of events, where action can take place. These events are

- **Startup**
When TWN4 is turned on or performs a reset, the event occurs.
- **Found Card/Transponder**
When a transponder enters the RF field of TWN4, the event occurs.
- **Card/Transponder Timeout**
When a transponder leaves the RF field for a specific time, the event occurs. The timeout value can be adjusted.

By default, following actions take place:

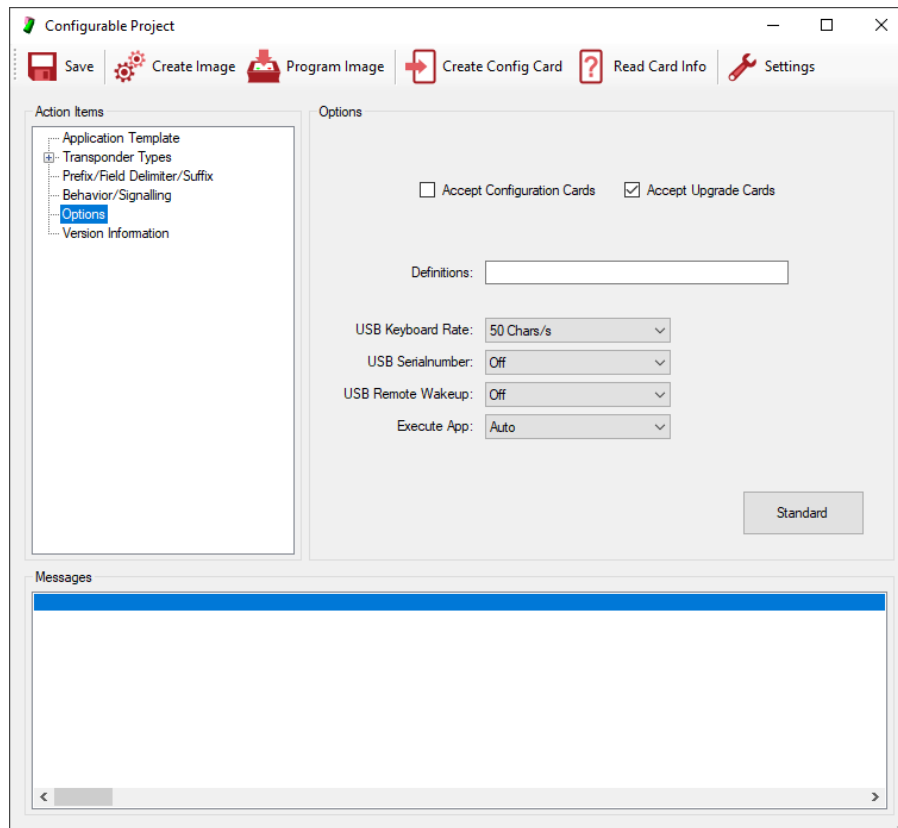
- **Startup**
Turn off red LED, turn on green LED, sound a low/high beep at 30% volume
- **Found Card/Transponder**
Let red LED blink, turn off green LED, sound a high beep at 100% volume
- **Card/Transponder Timeout**
Turn off red LED, turn on green LED. The timeout value is set to 2 seconds.

The behaviour can be adjusted to specific requirements coming from the application. A simple modification could be to suppress any output from the beeper. In this case, select "No Beep" for all three types of events.



8.8 Step 5: Specify Options (Optional)

In this section some special requirements for the application can be set up.



8.9 Step 6: Specify Version Information (Recommended)

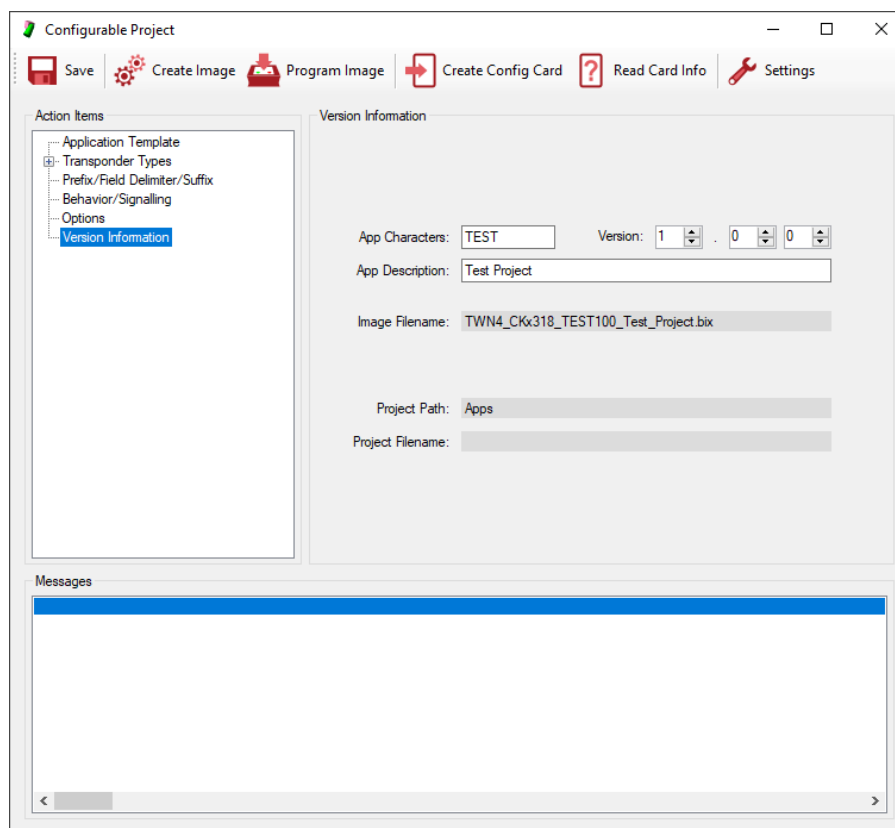
It is recommended to use a standard naming scheme for the firmware image. The information contained in the file name of the firmware image will be used at several places and allows recognition of correct version of a firmware image or TWN4.

Version information of TWN4 is therefore identical under following conditions:

- The name of the firmware image file itself.
- Information contained on the label of TWN4.
- Version information returned by the application running on TWN4. This is applicable for firmware, which allows request for the firmware version and is achieved via the system function `GetString`.
- Version information returned by USB as product string

AppBlaster supports the naming scheme by offering a dialog, where appropriate information can be entered:

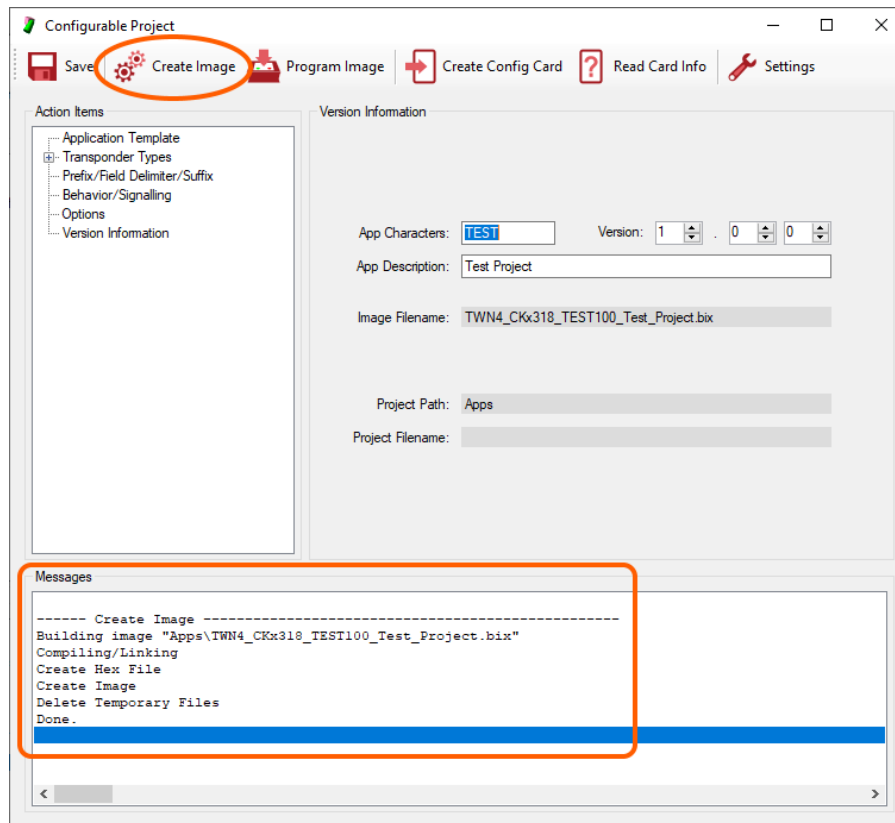
- App Characters: These characters are 1 to 4 letters ('A'-'Z') or digits ('0'-'9'). Please use at least 2 characters which identify the project.
- Version: The version always consists out of 3 digits, which can be set up here.
- App Description: You may do a short description of the project here.



8.10 Step 7: Create Image

Before a configuration can be tested, the firmware image must be created.

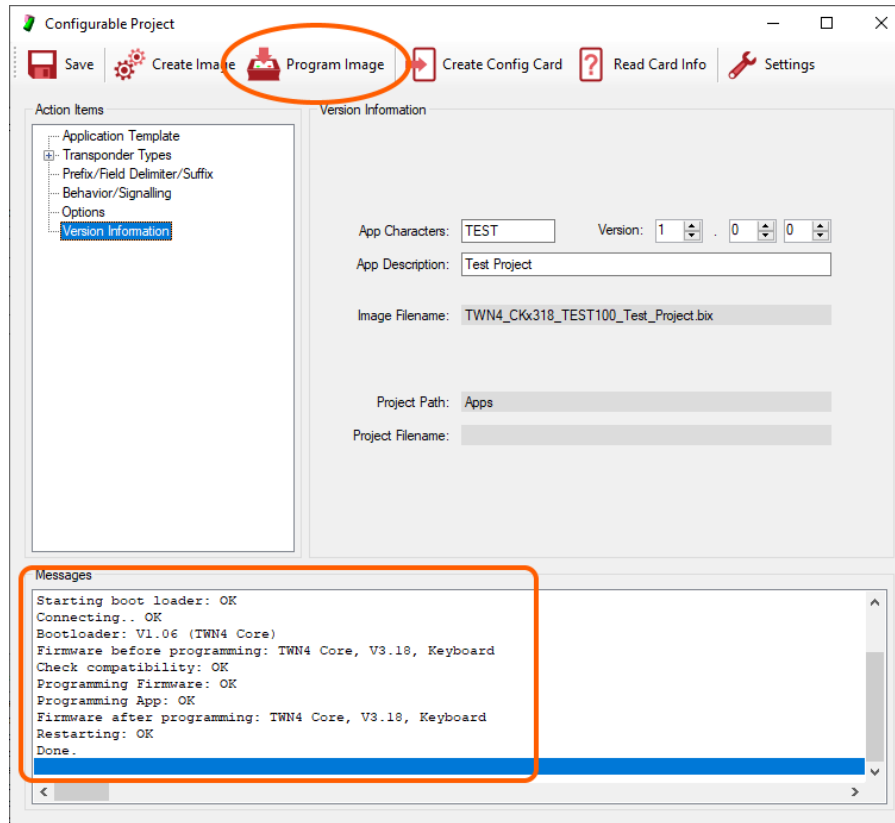
In order to create the firmware image, click button "Create Image". If the project never was saved before, the project will be created in the sub directory "Apps". Otherwise, it will be created, where the project file resides.



8.11 Step 8: Program Image

If the firmware image was created successfully it can be programmed into TWN4.

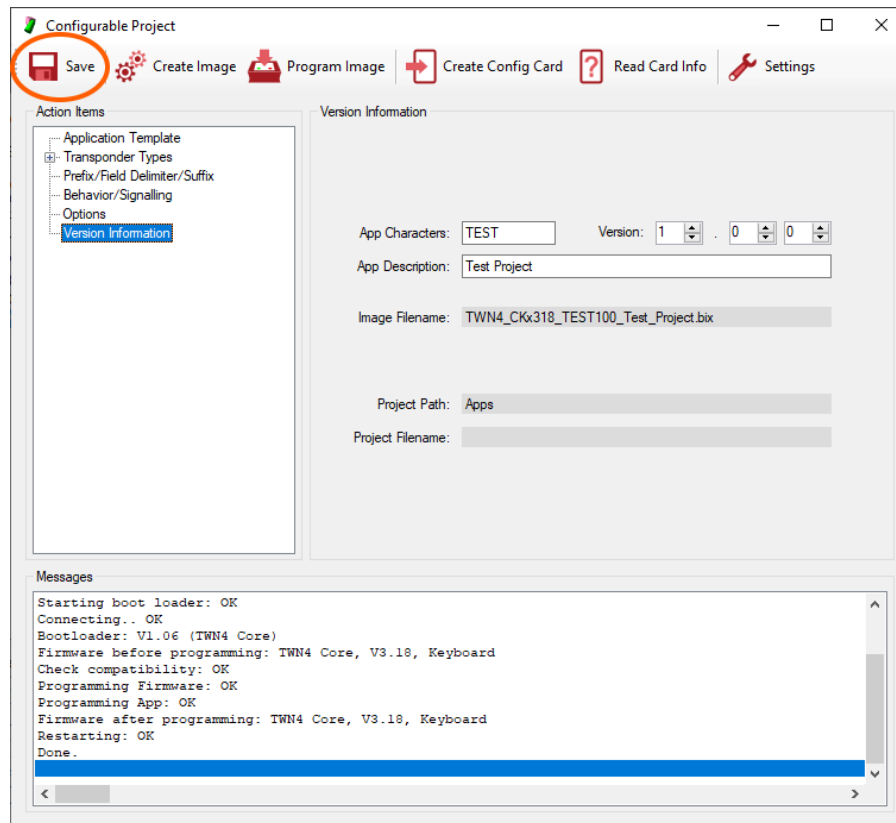
In order to program firmware image, click button "Program Image".



8.12 Step 9: Save Project (Recommended)

After finishing work, you may save entire information in a project file. Compared to the production image, the project file allows to modify settings of the configuration later, the production image does not.

As a suggestion, AppBlaster will choose the name for the project file identical to the name of the production image.



9 Configurable Project (OSDP)

The chapter above explained, how to configure a standard App. Additionally, there is the possibility to create a OSDP App.

There are some difference between standard App and OSDP App:

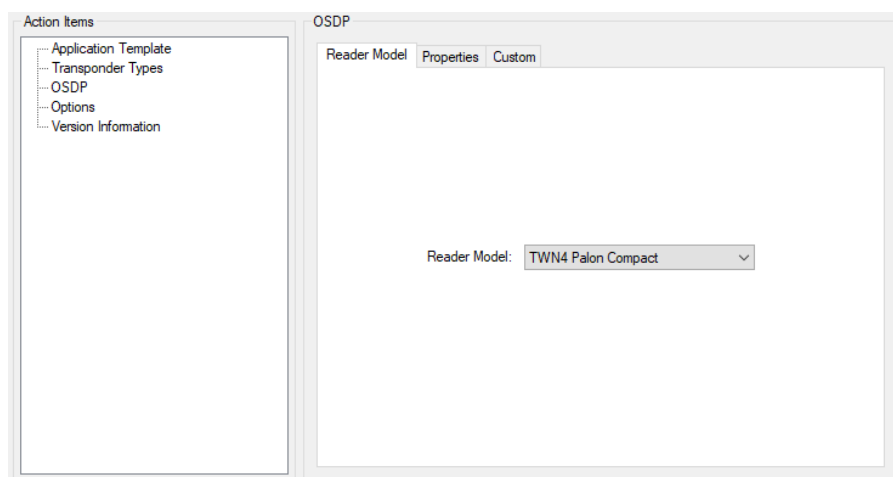
- While the standard App allows to configure signalling and behaviour, this is not possible in a OSDP App, because behaviour and signalling is fully under control of the host, the TWN4 is connected to.
- There are only limited possibilities to configure the output format of a OSDP App, because data is transferred binary. On the other hand, this simplifies configuration of an OSDP App.
- For the same reason, prefix, delimiter and suffix can not be configured in a OSDP App.

For the reasons mentioned above, selection of prefix, delimiter and suffix and signalling/behaviour is replaced by another sheet called "OSDP", which allows to select OSDP-specific options. The OSDP-options are separated into three tab sheets:

Please note: Even though, it is possible to install an OSDP Application on other reader models, it is only supported by devices, which are part of the TWN4 Palon family.

9.1 OSDP Reader Model

Select the appropriate reader model:



9.2 OSDP Standard Properties

Select appropriate options for your Application:

The screenshot shows the 'OSDP' configuration window with the 'Properties' tab selected. On the left, the 'Action Items' pane lists: Application Template, Transponder Types, OSDP, Options, and Version Information. The main area contains the following settings:

- Address: Selected by DIP-switch
- Baud Rate: Selected by DIP-switch
- RS485 Bias: Selected by DIP-switch
- Tamper: Enabled
- Card Timeout: 2 Seconds
- LED Style: Ring
- System Timeout: ☒
- Active/Idle Beep: ☒

9.3 OSDP Custom Properties

In case of hardware, which is designed by customer, it is possible to do more specific selections:

The screenshot shows the 'OSDP' configuration window with the 'Custom' tab selected. On the left, the 'Action Items' pane lists: Application Template, Transponder Types, OSDP, Options, and Version Information. The main area contains the following settings:

- OSDP Channel: COM1
- LED Type: GPIO
- LED Num: 1
- Tamper Type: None
- Tamper GPIO:
 - GPIO: 3
 - Active High: ☒
- RS-485 TX GPIO: ☐ Activated
- RS-485 TX GPIO:
 - GPIO: 4
- Have IPE: ☐
- Have GPIO Extender: ☐
- Host: ☐

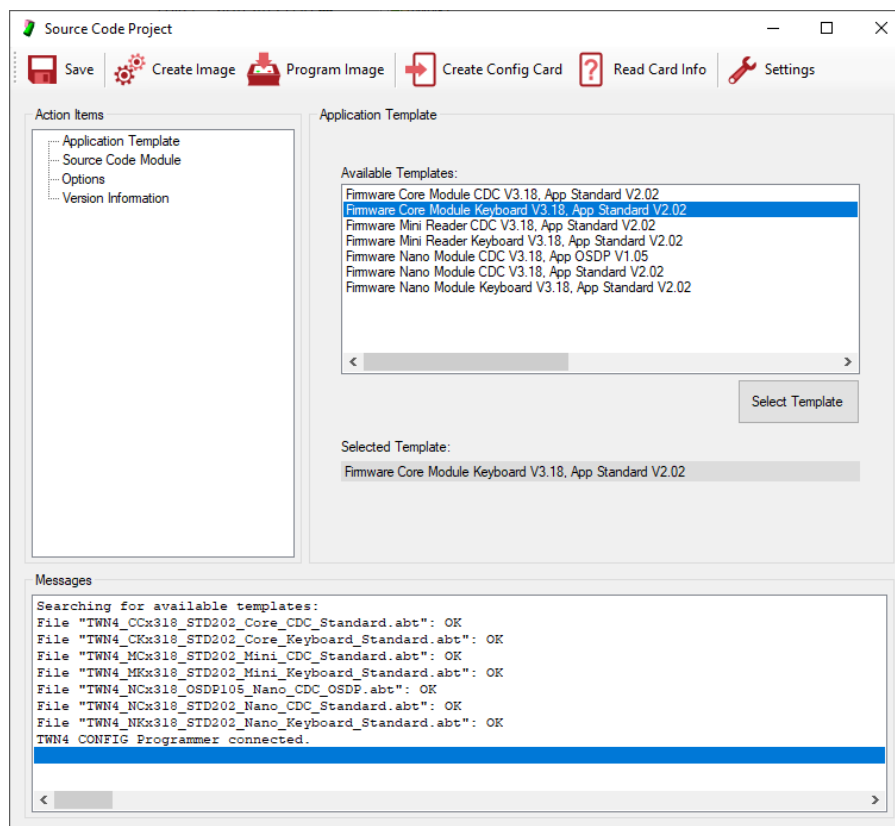
10 Source Code Project

For several reasons, there might be a need for creating a configuration, which is not covered by a configurable project:

- 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 some tricky data section of the transponder or some kind of special 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 way out of this situation, which is offered by AppBlaster, is to create a project based on custom source code. This is called source code project.

In order to create a source code project, click button "Source Code Project" in the start dialog. The project dialog appears in a somewhat reduced version because - compared to a configurable project - most of setup is not needed here.

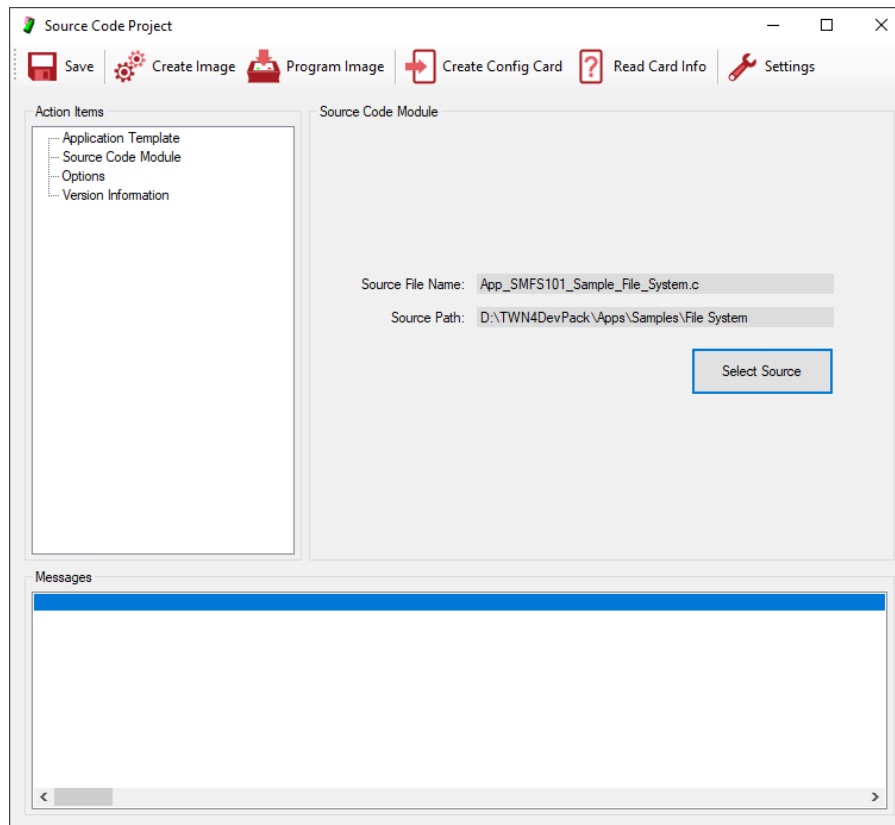


10.1 Step 1: Specify Type of USB

TWN4 can be programmed to act as several different type of USB. Select action item "Type of USB (Template)" in order to select appropriate operating mode.

10.2 Step 2: Specify Source Code Module

Click button "Select Source" to select the file, which contains the source code in question.



10.3 Step 3: Specify Version Information (Recommended)

It is recommended to use a standard naming scheme for the firmware image. The information contained in the file name of the firmware image will be used at several places and allows recognition of correct version of a firmware image or TWN4.

10.4 Step 4: Create Image

Before a configuration can be tested, the firmware image must be created.

In order to create the firmware image, click button "Create Image". If the project never was saved before, the project will be created in the sub directory "Apps". Otherwise, it will be created, where the project file resides.

10.5 Step 5: Program Image

If the firmware image was created successfully it can be programmed into TWN4.

In order to program firmware image, click button "Program Image".

10.6 Step 6: Save Project (Recommended)

After finishing work, you may save entire information in a project file. Compared to the production image, the project file allows to modify settings of the configuration later, the production image does not.

As a suggestion, AppBlaster will choose the name for the project file identical to the name of the production image.

11 TWN4 CONFIG / Configuration Cards

11.1 Introduction

With AppBlaster V3, a new method is introduced, how TWN4 can be configured:

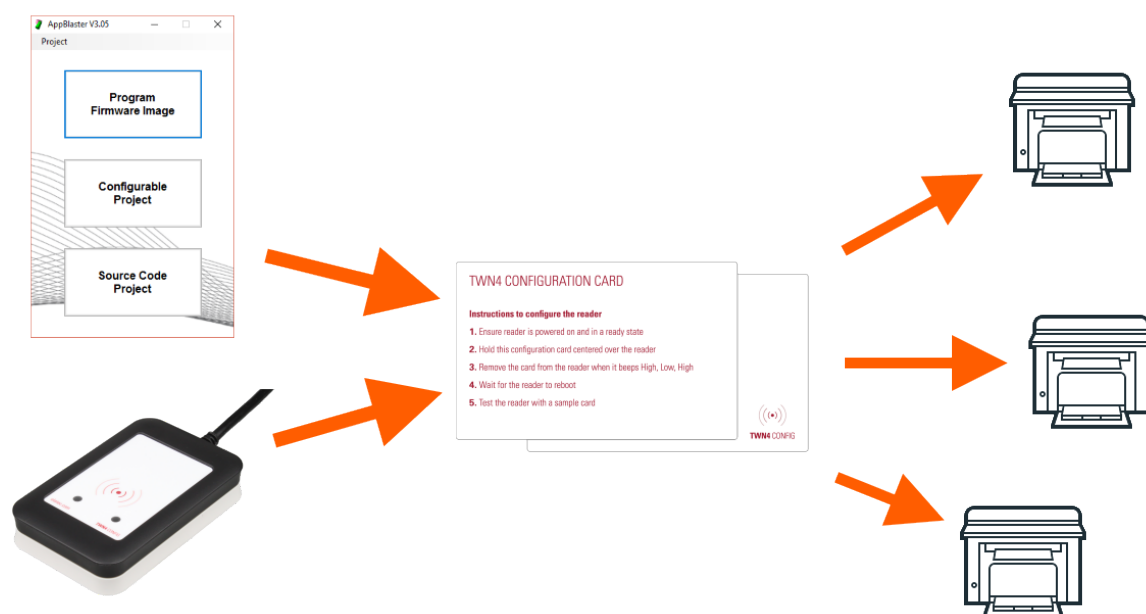
The well-known way is to program a firmware image into TWN4 via host interface.

The newly introduced way is to configure TWN4 via the RF interface just by swiping a configuration card. This system is called TWN4 CONFIG.

There are several advantages by using TWN4 CONFIG Cards:

- There is no need to unplug TWN4 from the host in order to configure it.
- There is no need to disassemble the device wherein TWN4 is integrated.
- Updates can be done very quickly without need of additional tools or even the click of a button.

See below the resulting workflow by using a configuration card:



With AppBlaster and TWN4 CONFIG Programmer, a TWN4 CONFIG Card is programmed. TWN4 in the field can be re-configured directly with that programmed TWN4 CONFIG Card.

Please note:

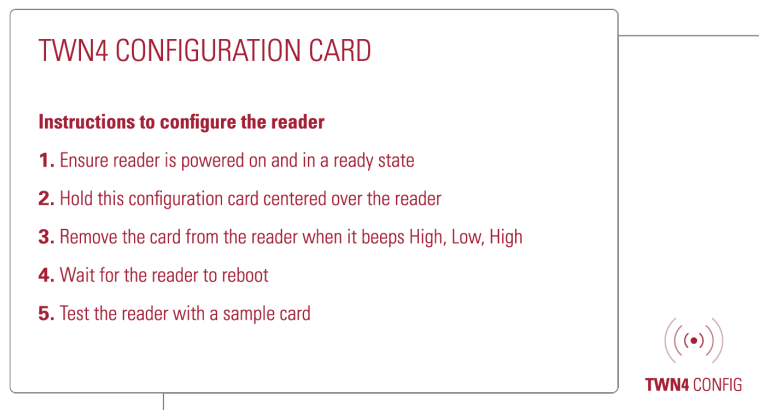
A TWN4 CONFIG Card contains the App-part of a firmware image. This means it is possible to exchange the App within a TWN4 but not the firmware itself.

11.2 Requirements

There are a few requirements to configure TWN4 with a configuration card:

- On TWN4, the firmware V3.04 (or later) must be installed
- TWN4 must be configured to accept TWN4 CONFIG Cards
- A TWN4 CONFIG Card is needed (see description below)
- A TWN4 CONFIG Programmer is needed to program a configuration onto a TWN4 CONFIG Card (see description below)
- AppBlaster V3 or later

11.3 TWN4 CONFIG Card



The picture above is showing front and rear side of a TWN4 CONFIG Card.

The memory size of a TWN4 CONFIG Card is 8 kBytes. It can be re-written.

The rear side of the TWN4 CONFIG Card offers plenty of white space, where additional project-specific labelling can be done. It is strongly recommended to do so!

Notes:

- TWN4 CONFIG Cards can be purchased pre-formatted and ready-to-use
- In order to store configurations on a TWN4 CONFIG Card, a TWN4 CONFIG Programmer is needed

11.4 TWN4 CONFIG Programmer



In order to program a configuration card, a TWN4 CONFIG Programmer is needed. A TWN4 CONFIG Programmer can be purchased separately or as part of a TWN4 CONFIG Kit. A TWN4 CONFIG Programmer can be recognized by its unique black/white colour scheme.

11.5 Creating a Configuration Card

Creating a configuration card is similar to dealing with a normal project (configurable or source code).

The result of a project is an image file which contains the firmware and the App. Instead of programming a image directly into a TWN4, the App section of the image file is programmed onto the configuration card.

11.5.1 Step 1: Provide Configuration Information

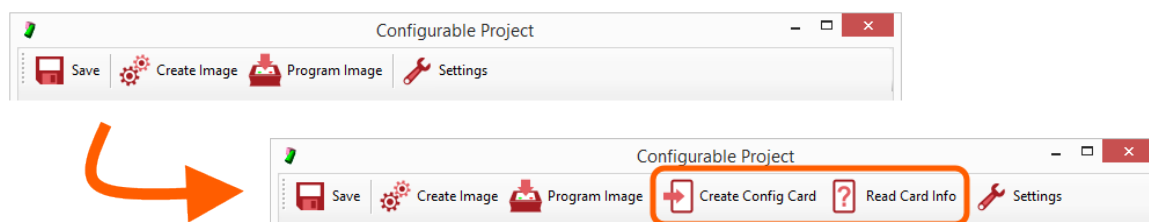
Depending on if you are working on a configurable project or a source code project provide all needed configuration specific information like described:

In a configurable project, follow steps 1 to 6

In a source code project, follow steps 1 to 4

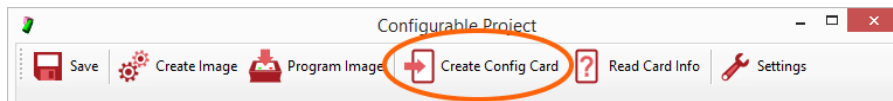
11.5.2 Step 2: Connect TWN4 CONFIG Programmer

Connect TWN4 CONFIG Programmer to your PC. Once a TWN4 CONFIG Programmer is connected to the host, the tool unhides two additional buttons like shown below:

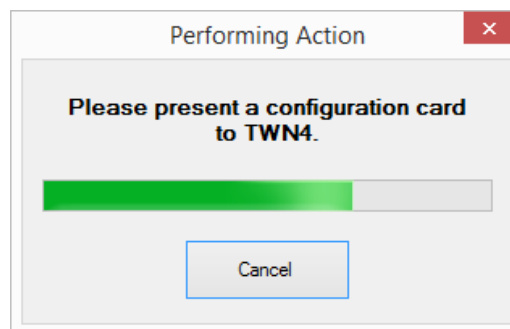


11.5.3 Step 3: Program a TWN4 CONFIG Card

In order to program a configuration card, click the button "Create Config Card".



Once the button "Create Config Card" was clicked, the firmware image is created (like button "Create Image"). As soon as the firmware image is created following dialog appears:



The dialog asks the user to now present a configuration card to the TWN4 card programmer.

A progress bar illustrates remaining time to follow this request. If no card is presented or the user clicks the button "Cancel", the operation is cancelled.

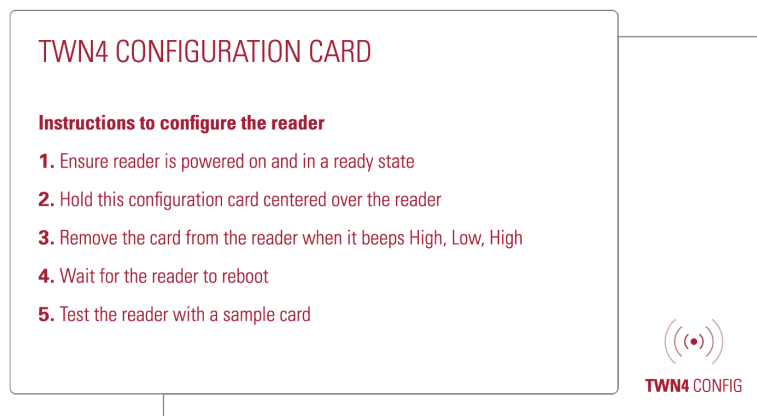
If a configuration card is detected, it is programmed immediately with the newly created App section of the firmware image.

Note:

If the card already was programmed with a configuration before, another dialog appears, which asks to overwrite the configuration card.

11.6 Configuring TWN4 with a Configuration Card

The instructions on how to use a configuration are printed on the card itself:

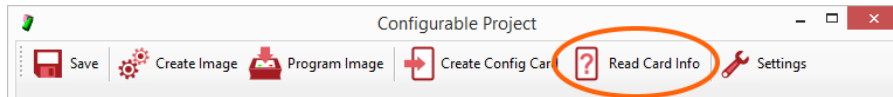


Here are a some more details about this process:

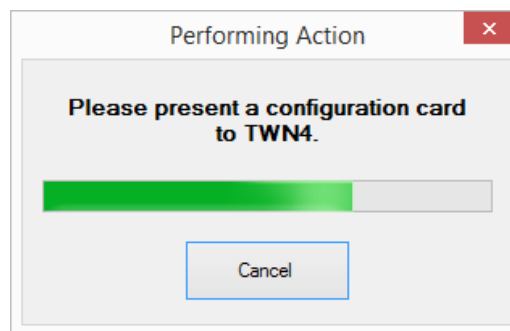
1. Ensure reader is powered on and ready state
Comment: The ready state is typically reached immediately after reader was connected to the host or powered on.
2. Hold the configuration card centered over the reader
Comment: Especially, hold the configuration card in read range of the reader.
3. Remove the card from the reader when it beeps High, Low, High
Comment: If reading the configuration card fails, it beeps Low, Low, Low. Depending on progress of the configuration, the reader may do a reboot as well.
4. Wait for the reader to reboot
Comment: The completion of the reboot normally can be recognized as another beep sequence. Most often, it is Low, High at a lower volume.
5. Test the reader with a sample card

11.7 Reading Information from the Card

While reading the configuration from a configuration card is only possible by TWN4, there is an additional section on the configuration card, which stores some general information. This information is public accessible and can be read by a click on button "Read Card Info" in the tool bar:



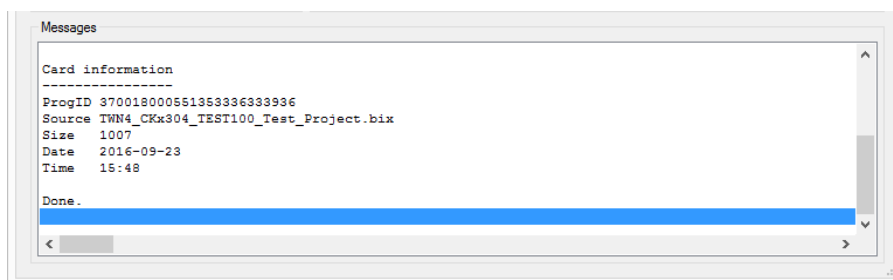
Once button "Read Card Info" is clicked, following dialog appears:



The dialog asks the user to now present a configuration card to the TWN4 card programmer.

A progress bar illustrates remaining time to follow this request. If no card is presented or the user clicks the button "Cancel", the operation is cancelled.

If a configuration card is detected, information is immediately read from the card and displayed in the message-section:



12 Creation of Apps with *make*

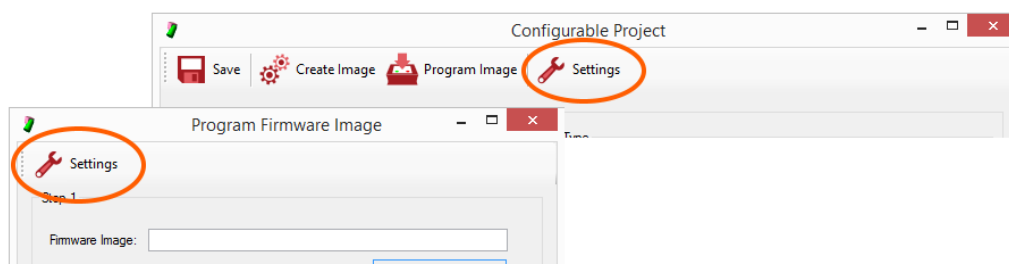
Within AppBlaster there are limitations, which might not be acceptable by advanced users, such as:

- 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.

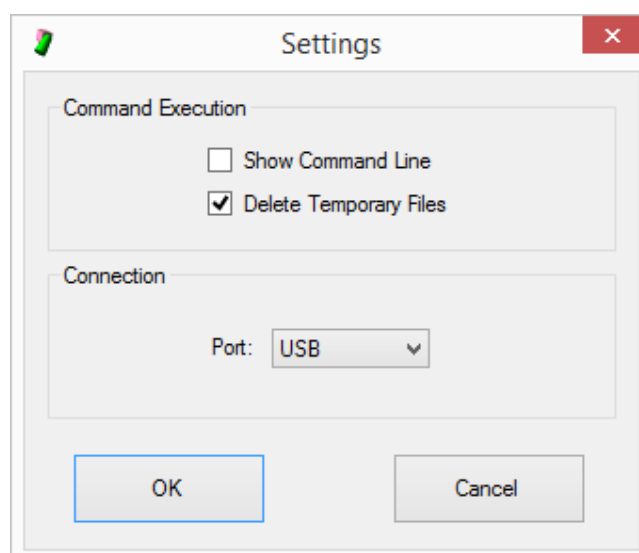
The tools for creation of an App from a command line are part of the TWN4 developer pack. AppBlaster is a graphical user interface (GUI) for the tool chain which resides in the subdirectory `Tools`. In order to become more convenient with how an App is built, there is an option to display the command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

13 Setting Up AppBlaster

There are a few settings, which can be made for AppBlaster. The settings can be accessed from both a project dialog or dialog for programming firmware:



- "Command Execution - Show Command Line": By activating this option, the commands, which are used for creation of images are displayed in the protocol section. This is helpful, if you plan to set up a project using make.
- "Command Execution - Delete Temporary Files": Turn this option off, if you are interested in temporary files, which are part of the compilation process.
- "Connection - Port": Use USB as long as TWN4 is connected via USB to the host (even virtual COM port). If you are using TWN4 RS232, please select appropriate COM port here.



14 Disclaimer

Elatec reserves the right to change any information or data in this document without prior notice. The distribution and the update of this document is not controlled. Elatec declines all responsibility for the use of product with any other specifications but the ones mentioned above. Any additional requirement for a specific custom application has to be validated by the customer himself at his own responsibility. Where application information is given, it is only advisory and does not form part of the specification.

All referenced brands, product names, service names and trademarks mentioned in this document are the property of their respective owners.