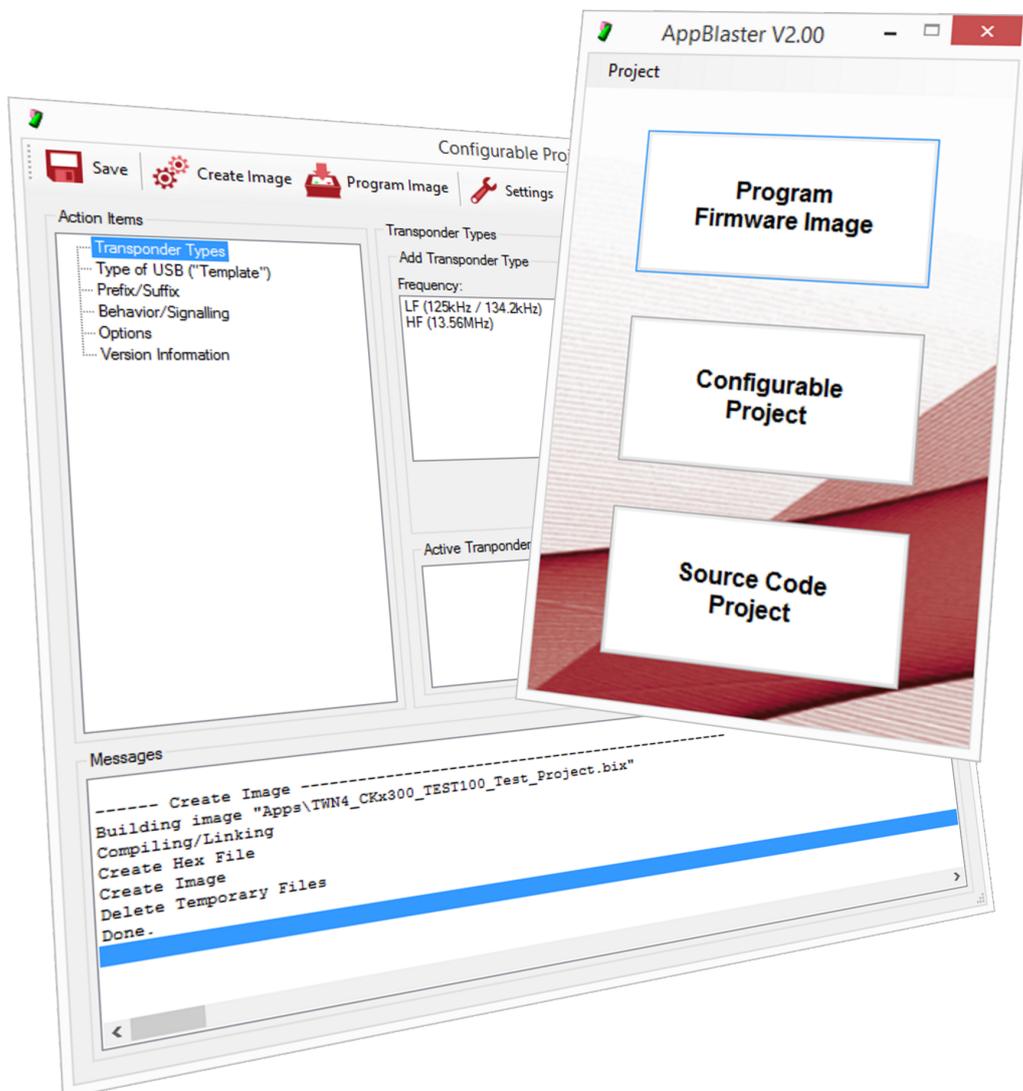


TWN4

AppBlaster User Guide

DocRev4, May 13, 2016



Elatec GmbH

Contents

1	Overview	4
2	System Requirements	5
3	Installation	6
4	Starting AppBlaster	7
5	Programming Firmware Image	8
6	Walk Through a Configurable Project	9
6.1	Step 1: Add Transponder Types	10
6.1.1	Remove Transponder Types	11
6.2	Step 1a: Specify Data Source	12
6.3	Step 1b: Specify Bit Manipulation	13
6.4	Step 1c: Specify Output Format	13
6.5	Step 2: Specify Type of USB	14
6.6	Step 3: Specify Prefix/Suffix (Optional)	15
6.7	Step 4: Signalling/Behaviour (Optional)	16
6.8	Step 5: Specify Options (Optional)	17
6.9	Step 6: Specify Version Information (Recommended)	18
6.10	Step 7: Create Image	19
6.11	Step 8: Program Image	20
6.12	Step 9: Save Project (Recommended)	21
7	Walk Through a Source Code Project	22
7.1	Step 1: Specify Type of USB	23
7.2	Step 2: Specify Source Code Module	23
7.3	Step 3: Specify Version Information (Recommended)	23
7.4	Step 4: Create Image	23
7.5	Step 5: Program Image	24
7.6	Step 6: Save Project (Recommended)	24
8	Creation of Apps with make	25
9	Setting Up AppBlaster	26
10	Disclaimer	27

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

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

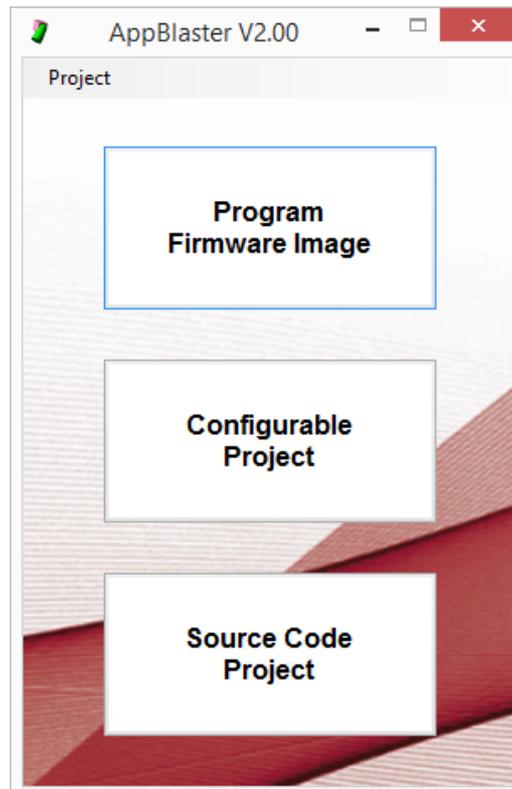
You received AppBlaster as part of the developer pack for TWN4, which is distributed as a zip file.

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:



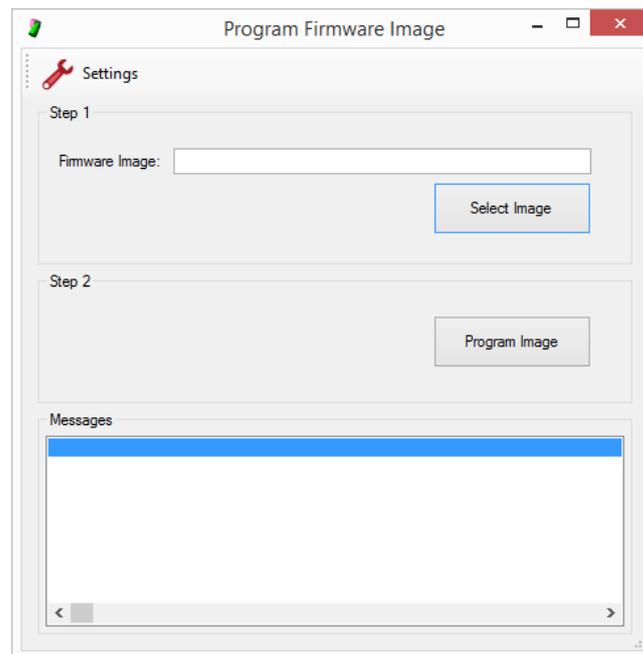
The start dialog offers three choices. In addition, an existing project can be loaded from disk via selection in the menu.

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:

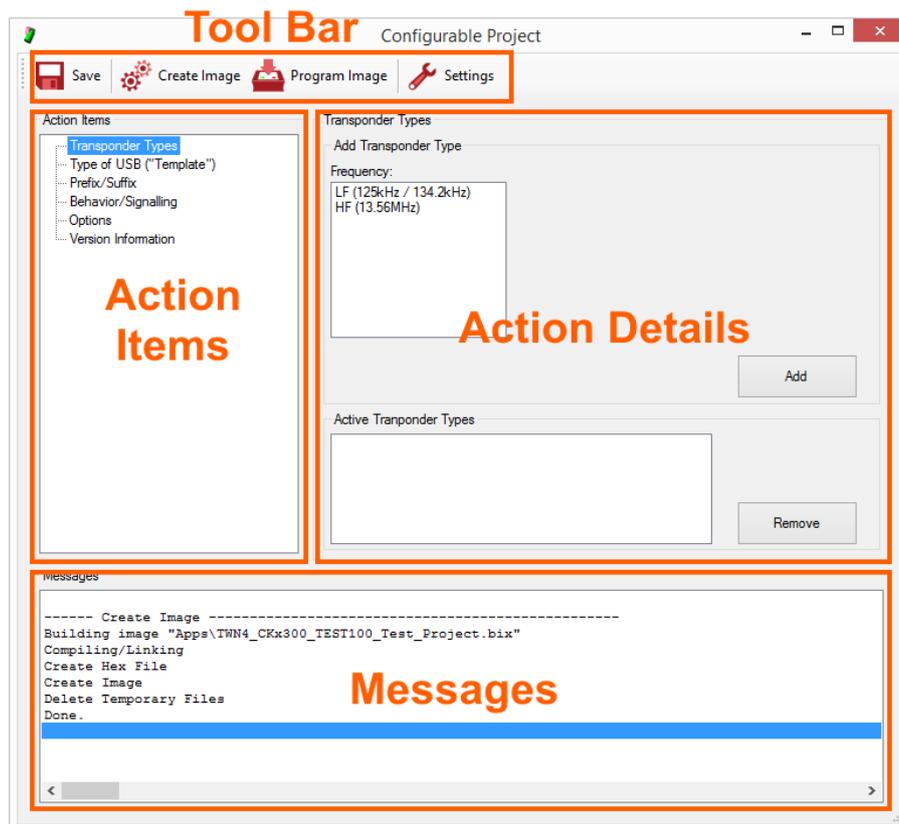
1. In "Step 1" choose the file of interest by clicking button "Select Image". Another way is to enter the known path and file name directly into the text box. The file extension for appropriate firmware images is ".bix".
2. In "Step 2" click button "Program Image". This will immediately start the programming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

6 Walk Through a 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:

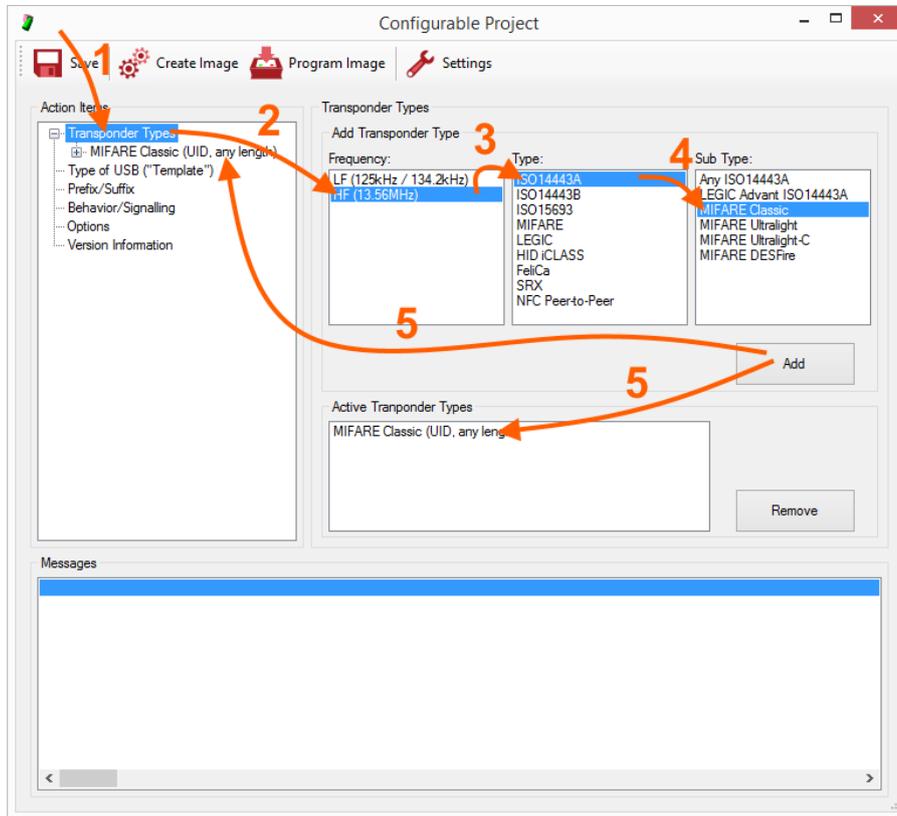
- Tool Bar: Buttons in the tool bar let you start specific actions at any time
- Action Items: There are several action items, which can or must be done to complete a project
- Action Details: For every action item, there are details which can or must be entered
- Messages: AppBlaster is giving feedback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

6.1 Step 1: 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:



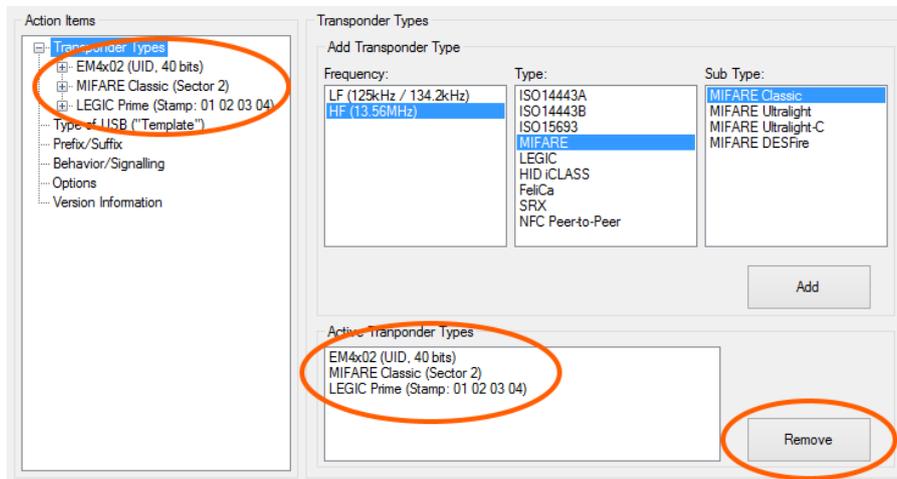
Following (sub-)steps:

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.

6.1.1 Remove Transponder Types

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



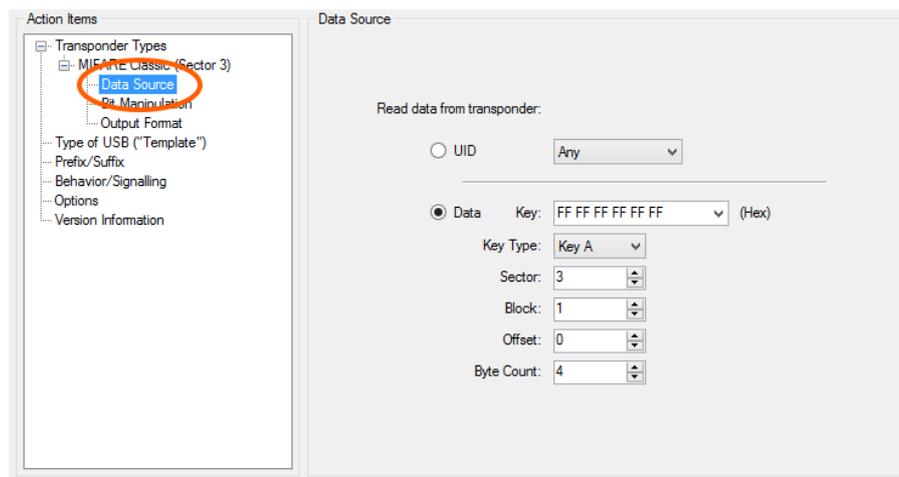
1. Select "Transponder Types" from "Action Items".
2. In the list box "Active Transponder Types" select transponder type to be removed.
3. Finally, click button "Remove" to remove the transponder type in question.

6.2 Step 1a: 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:

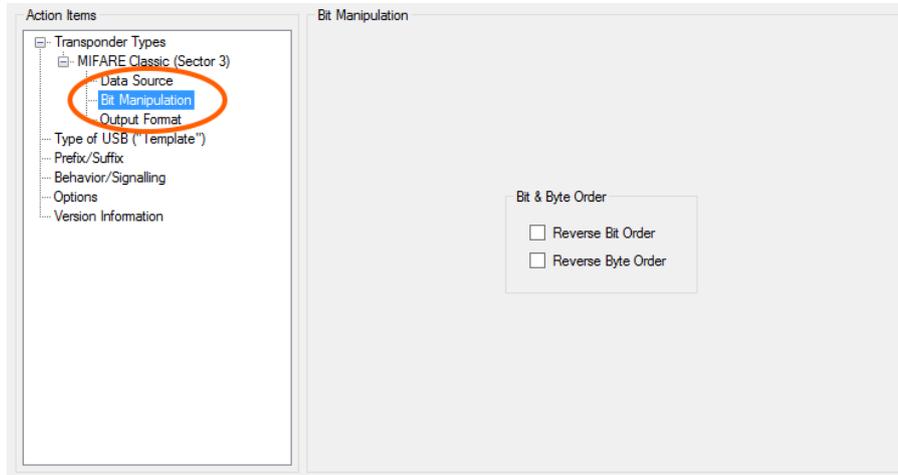


- Instead of reading UID choose "Data", which means read data from a specific block in a sector of MIFARE Classic.
- Specify the key, which gains access to the sector in question. There are some well-known default keys, which can be selected directly from the combo box.
- Select the sector number (0 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).

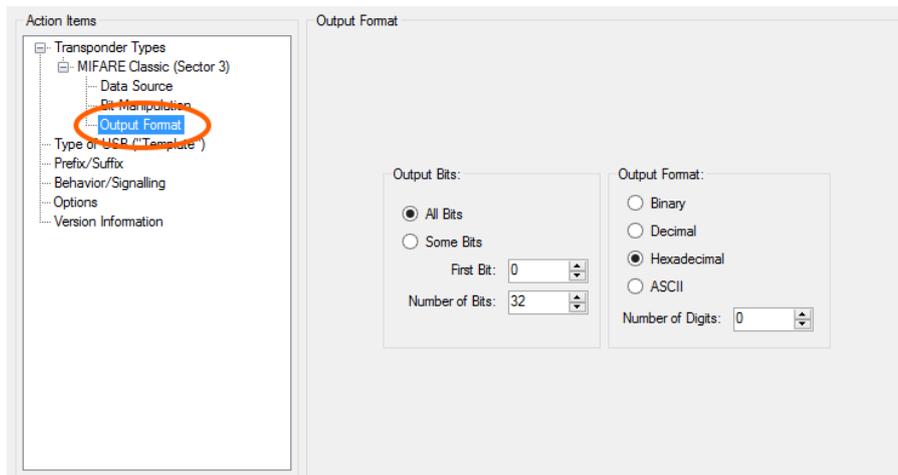
Notes:

- 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.

6.3 Step 1b: Specify Bit Manipulation

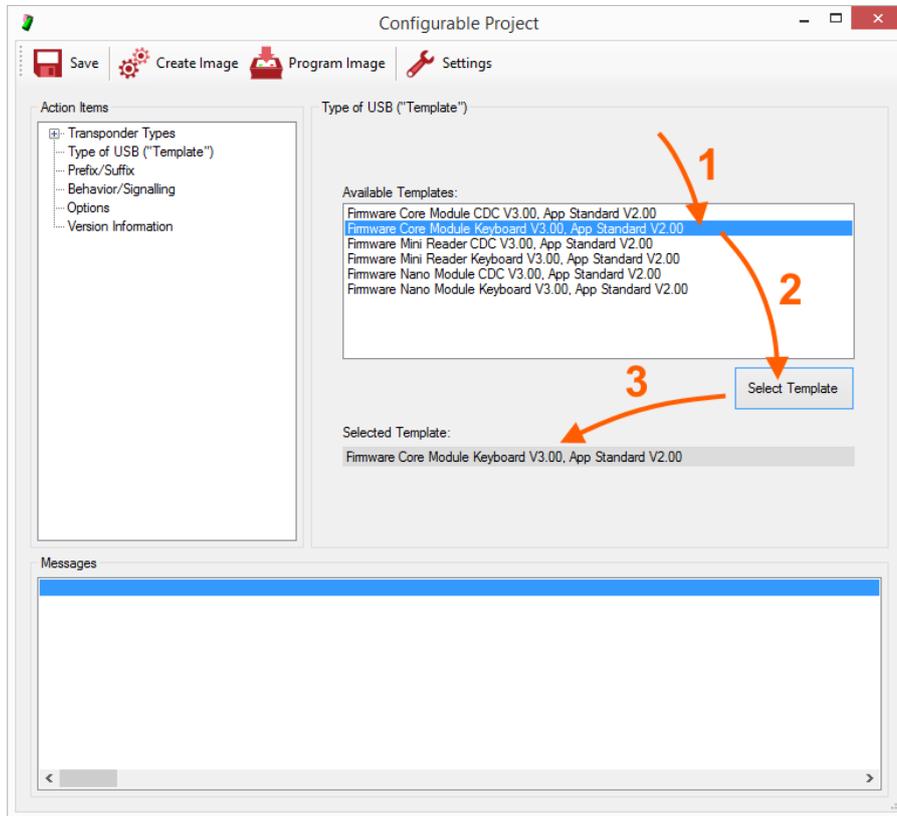


6.4 Step 1c: Specify Output Format



6.5 Step 2: Specify Type of USB

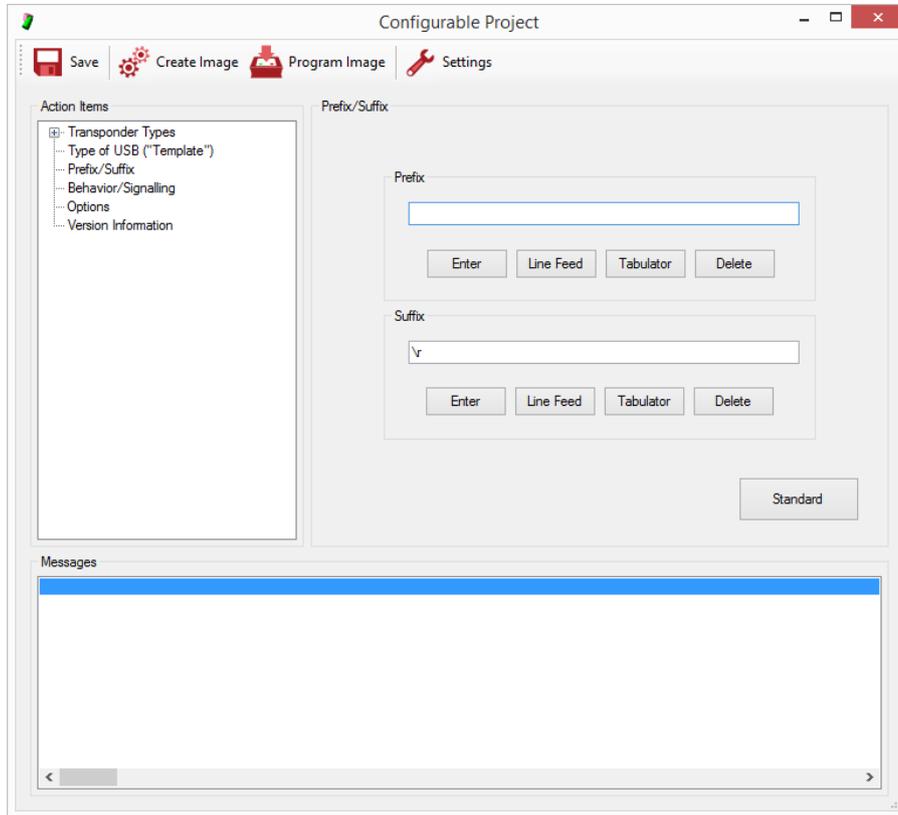
TWN4 can be programmed to act as several different type of USB. Select action item "Type of USB (Template)" in order to select appropriate operating mode.



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.

6.6 Step 3: Specify Prefix/Suffix (Optional)

AppBlaster now allows to specify a prefix and/or 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 and "Enter" is sent as suffix.



6.7 Step 4: Signalling/Behaviour (Optional)

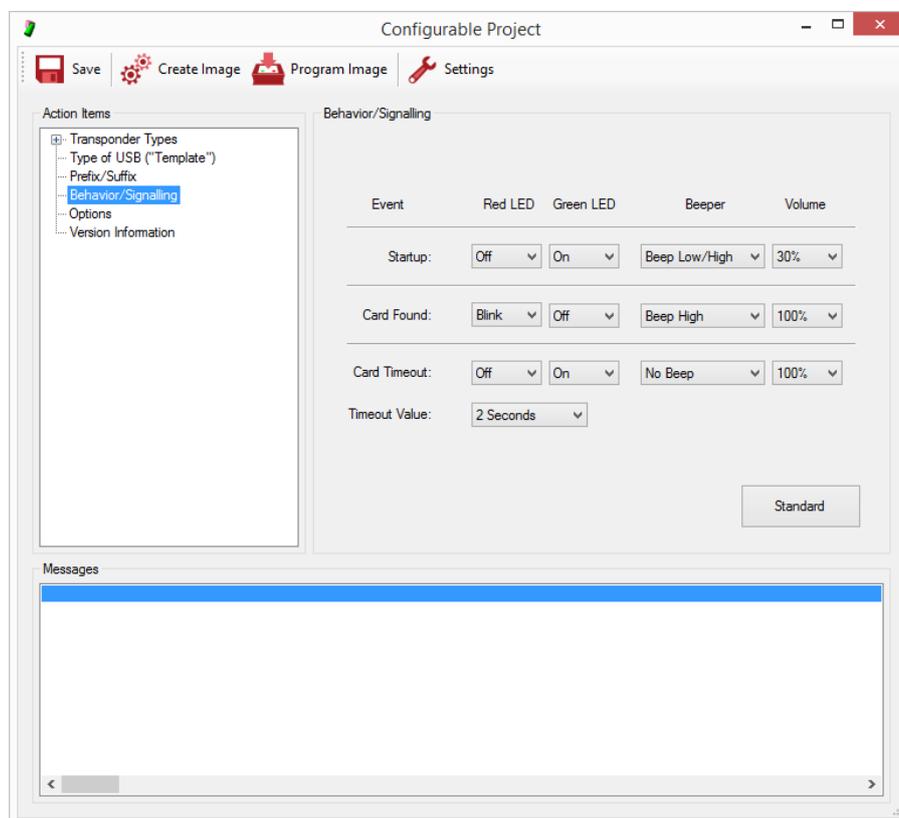
During operation of TWN4, there are several types of events, where action can take place. These events are

- 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:

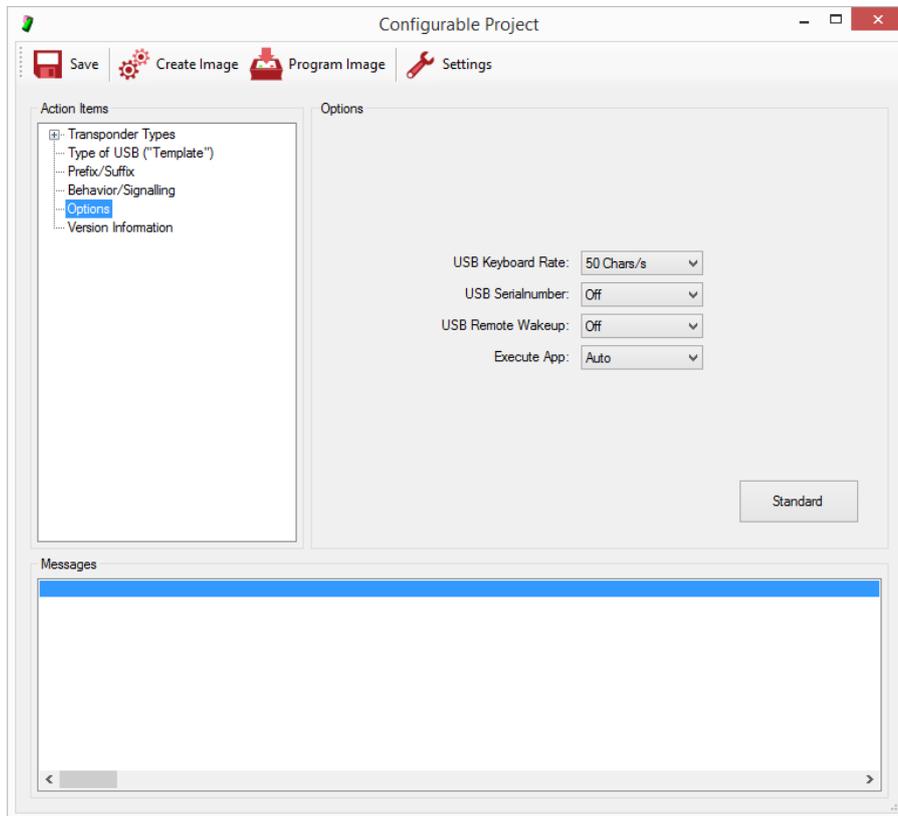
- Startup
Turn off red LED, turn on green LED, sound a low/high beep at 30% volume
- Found Card/Transponder
Let red LED blink, turn off green LED, sound a high beep at 100% volume
- Card/Transponder Timeout
Turn off red LED, turn on green LED. The timeout value is set to 2 seconds.

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.



6.8 Step 5: Specify Options (Optional)

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



6.9 Step 6: Specify Version Information (Recommended)

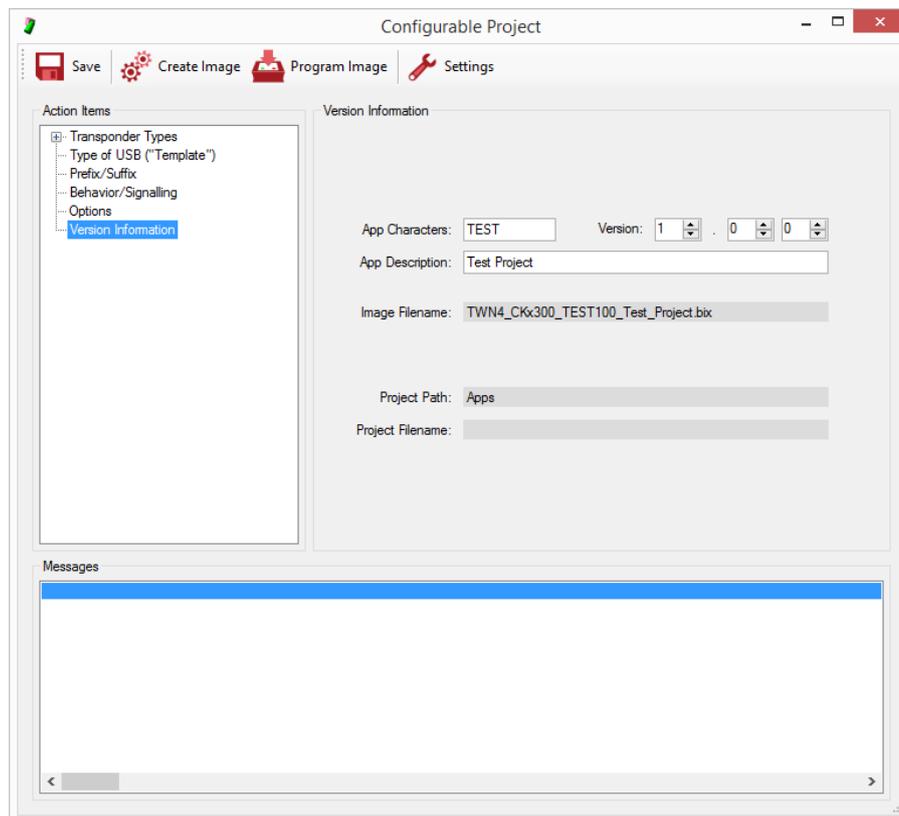
It is recommended to use a standard naming scheme for the firmware image. The information contained in the file name of the firmware image will be used at several places and allows recognition of correct version of a firmware image or TWN4.

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:

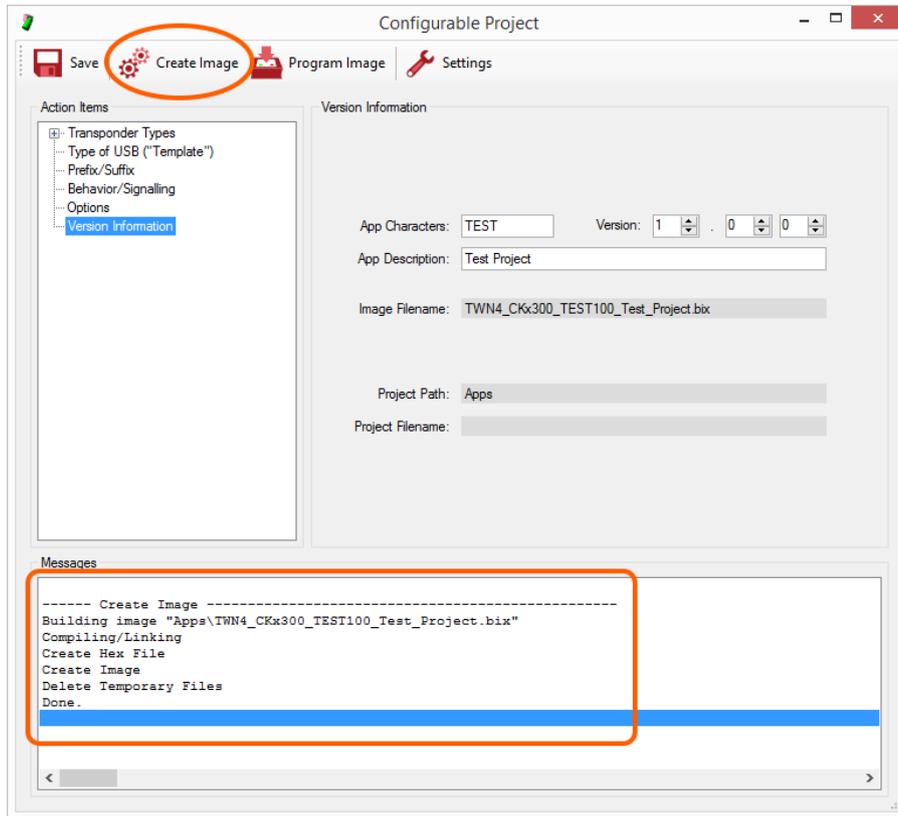
- App Characters: These characters are 1 to 4 letters ('A'-'Z') or digits ('0'-'9'). Please use at least 2 characters which identify the project.
- Version: The version always consists out of 3 digits, which can be set up here.
- App Description: You may do a short description of the project here.



6.10 Step 7: Create Image

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

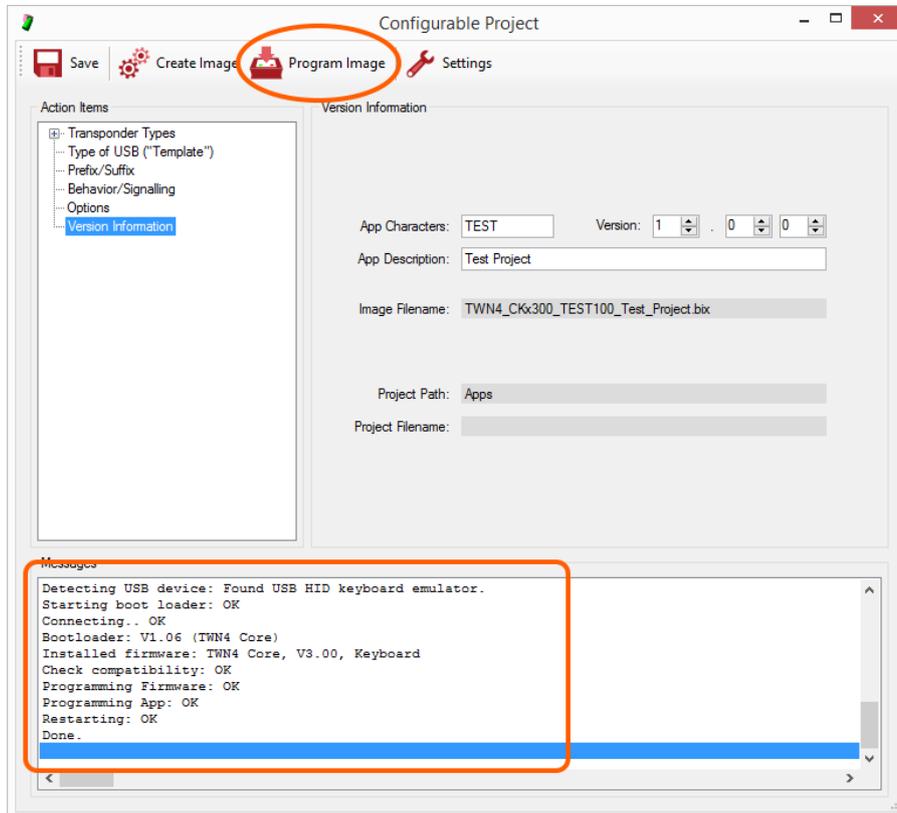
In order to create the firmware image, click button "Create Image". If the project never was saved before, the project will be created in the sub directory "Apps". Otherwise, it will be created, where the project file resides.



6.11 Step 8: Program Image

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

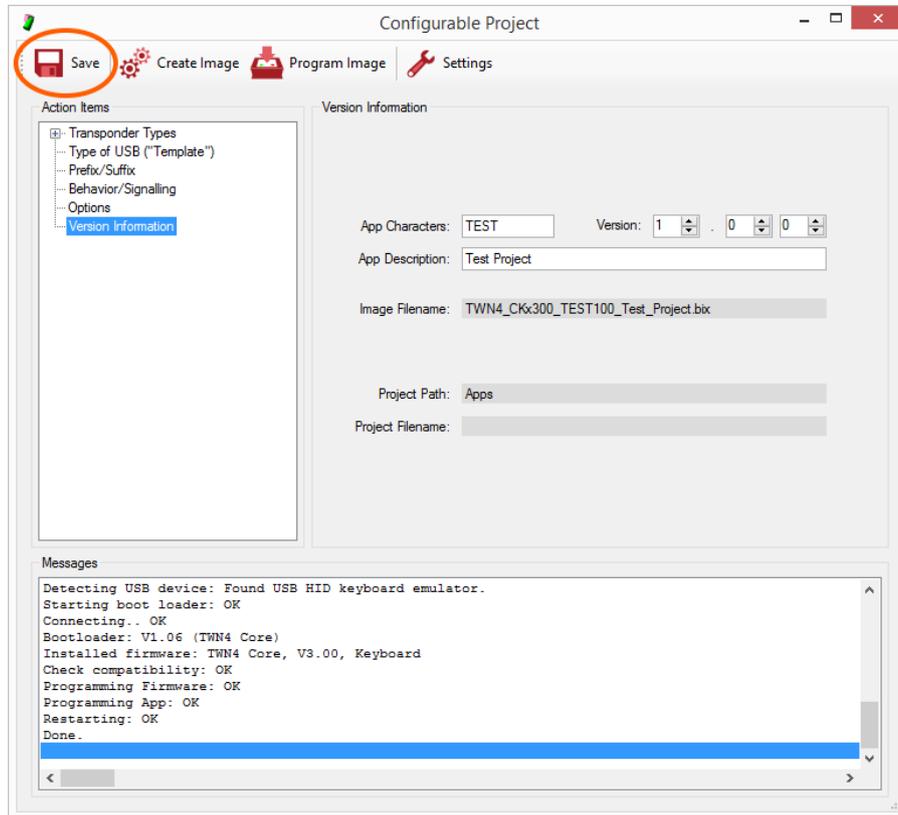
In order to program firmware image, click button "Program Image".



6.12 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.



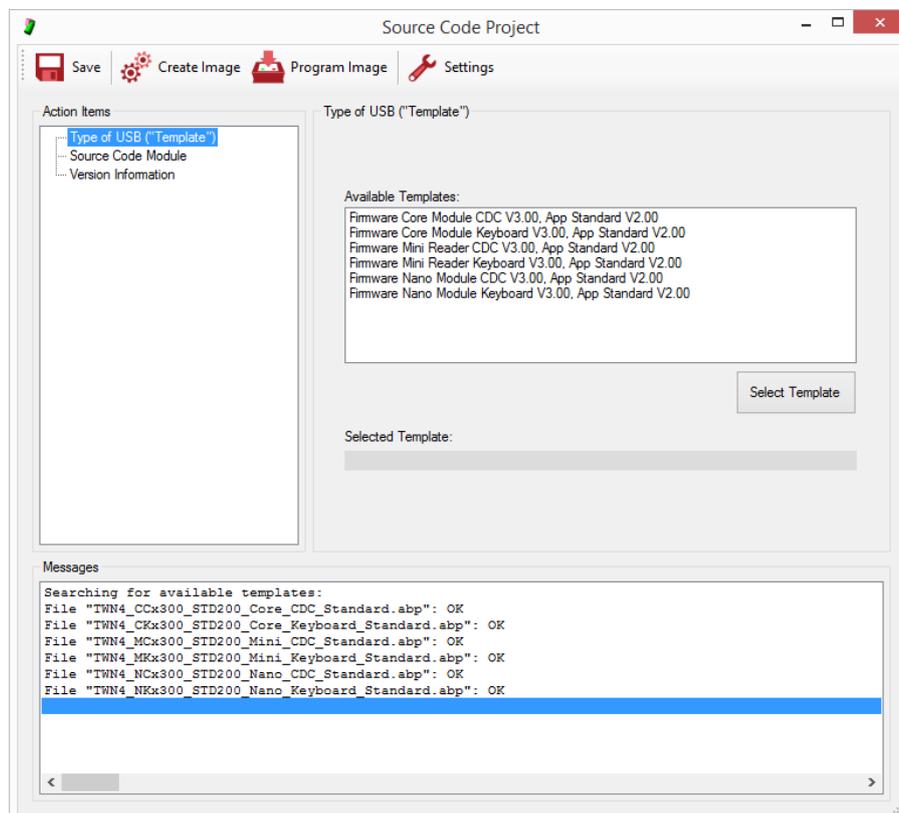
7 Walk Through a Source Code Project

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.

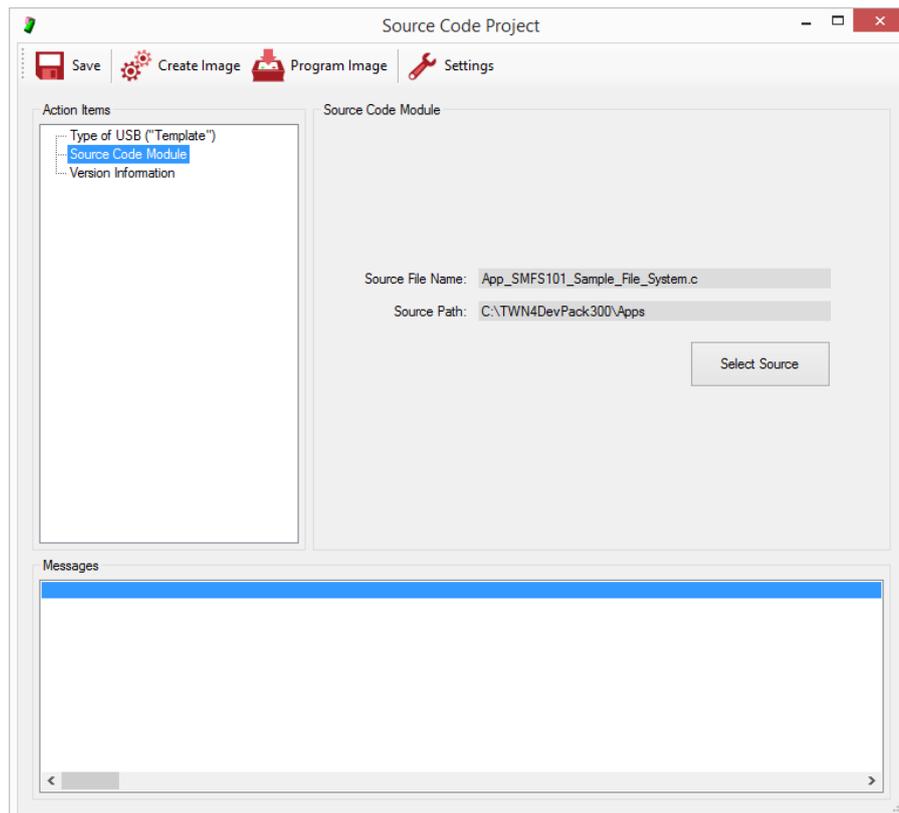


7.1 Step 1: Specify Type of USB

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.

7.2 Step 2: Specify Source Code Module

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



7.3 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.

7.4 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.

7.5 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".

7.6 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.

8 Creation of Apps with make

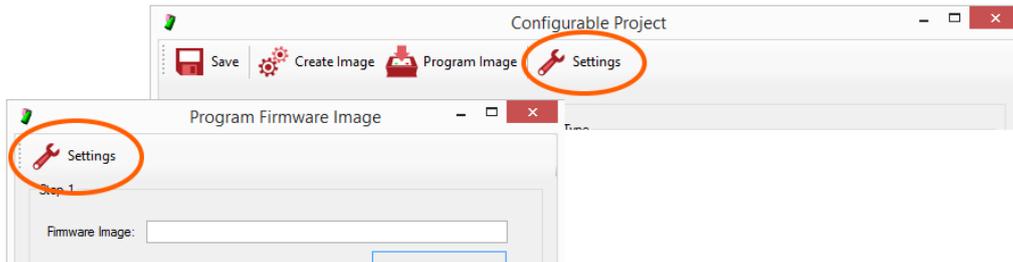
Within AppBlaster there are limitations, which might not be acceptable by advanced users, such as:

- 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.

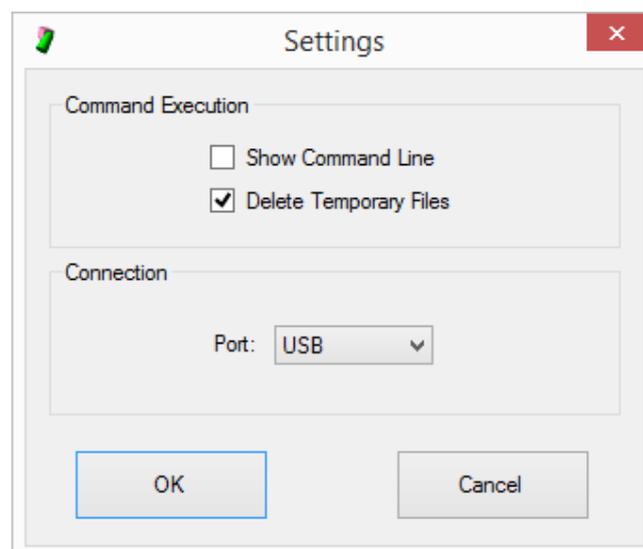
The tools for creation of an App from a command line are part of the TWN4 developer pack. AppBlaster is a graphical user interface (GUI) for the tool chain which resides in the subdirectory `Tools`. In order to become more convenient with how an App is built, there is an option to display the command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

9 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.



10 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.