imperix

BOOMBOX
QUICK START GUIDE
FOR PROGRAMMING WITH SIMULINK

Note

While every effort has been made to ensure accuracy in this publication, no responsibility can be accepted for errors or omissions. Data may change, as well as legislation, and the reader is strongly advised to obtain copies of the most recently issued regulations, standards, and guidelines.

INSTALLING THE SOFTWARE

Abstract — This section describes how to retrieve and install the software that is necessary to get started with the BoomBox. This includes Simulink libraries, imperix's BoomBox blockset, as well as BoomBox Control debug and monitoring software.

Keywords — *Simulink Blockset, BoomBox Simulink SDK, BoomBox Control, Install.*

Before programming the BoomBox, the necessary software development environment must be installed on a personal computer. The minimum software includes :

» *Matlab Simulink :* the graphical programming environment provided by MathWorks that allows to model and simulate the system before automatically generating the code for the BoomBox.

» *BoomBox Simulink SDK :* the software development kit (SDK) for Simulink that enables to build Simulink models that accurately simulate the behavior of the BoomBox. It also allows to automatically generate the corresponding run-time code.

» *BoomBox Control :* a utility software from imperix that provides a graphical user interface to monitor and control the BoomBox during run time. It is mostly useful for debugging and validating control code at the system level.

1.1 MATLAB SIMULINK

The first step consists in downloading and installing Matlab Simulink, if not already done. To activate your Matlab installation, a paid license issued by MathWorks is needed. This license is neither provided nor sold by imperix.

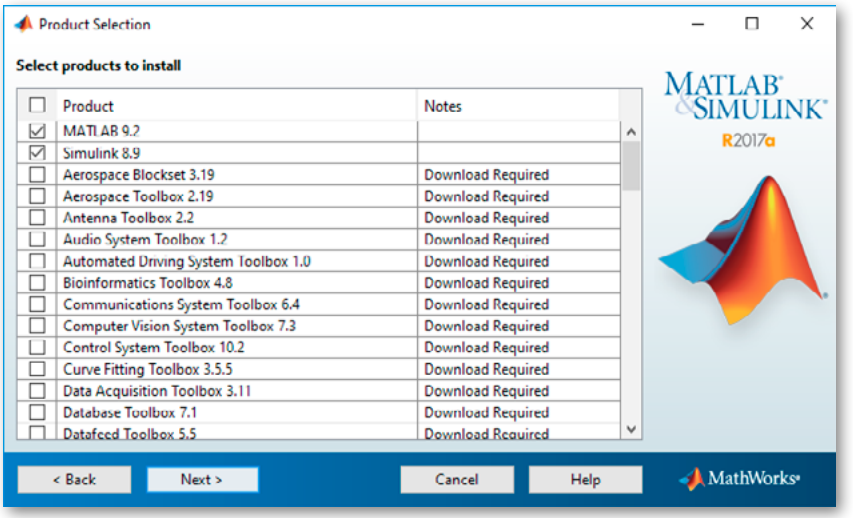
» Matlab and Simulink can be downloaded from *https://www.mathworks.com*.

» The imperix blockset is compatible with Matlab R2014a and newer, in both 32- and 64-bit versions.

» When prompted (figure below), select the following products during the installation :

- Matlab
- Simulink
- Embedded Coder
- Matlab Coder
- Simulink Coder

They are the only mandatory packages to start working.



1.2 BOOMBOX SIMULINK SDK

The Simulink SDK for the BoomBox enables the user to develop and debug run-time code for the BoomBox. The installer can be found on imperix's website. Registration is needed to access it.

» In order to download the installer from imperix's website, go to *https://www.imperix.ch/my-account* and log into your account.

» Click on the tab 'My downloads' and select 'My software'.

» Click on 'BoomBox Simulink SDK x.x.x' and the download button to get the latest version of 'SimulinkSDK_setup.exe'.

» Once the download is finished, execute the installer.

Note:

It is highly recommended to keep the default destination folder C:\imperix\BBSimulinkSDK\ and to avoid spaces or special characters in the path name.

» When prompted, select your licence file.

» If you have already requested a license, it should by available in the license section on imperix's website. To download it:

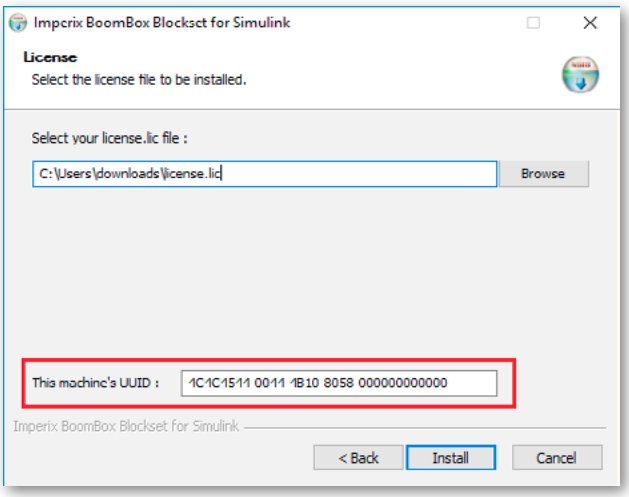
- Go to *https://www.imperix.ch/my-account* on log into your account.
- Click on the tab 'My downloads' and click 'My licenses'.
- Download the license file (.lic) that is meant for your computer.
- Use the downloaded license file for the ongoing install process.

» If you do not have a license file, you must request one using the following process:

- Go to *https://www.imperix.ch/request-license*.
- Fill in the requested information. The UUID of your computer is required. Provide the code displayed by the installer (see below).
- Your request will be validated and the license will be generated by a support engineer. You will receive the file at the e-mail address indicated in the request

form. The process usually takes less than one day if the request is sent during office opening times (CET).

- Use the provided license file in the install process.



» Once the licence file is selected, click on 'Install'. The install finishes shortly.

1.3 BOOMBOX CONTROL

BoomBox Control is a graphical software that enables the user to monitor, control and save data from the BoomBox control platform.

» You will find the installer on the software download section of imperix's website:

- Go to *https://www.imperix.ch/my-account* and log into your account.
- Click on the tab 'My downloads' and then 'My software'.
- Download the latest version of 'BoomBox Control Utility'.

» Extract and execute 'BoomBoxControl_setup.exe'.

» Select an installation destination folder and click 'Install'.

» BoomBox Control has been installed on your computer!

1.4 ELECTRONIC CIRCUIT SIMULATOR

In many cases, it is recommended to run off-line simulations of your system before downloading and executing the associated runtime code. A third-party modeling software may be needed for that purpose. We recommend using Plexim's *PLECS Blockset* or MathWorks' *Simscape Power Systems*.

More details are provided in section 2.2.

4

WORKING WITH SIMULINK

Abstract — This section describes how to simulate a control implementation on Simulink using the imperix blockset and how to automatically generate code to be run on a BoomBox in real time. The simulation allows for a fine-tuning of the control before starting experimenting with the BoomBox and real power.

Keywords — *Template, plant model, simulation, automated code generation.*

2.1 RETRIEVING THE DEFAULT TEMPLATE

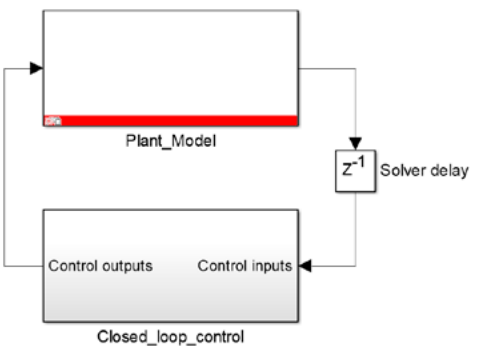
To start working with the BoomBox and Simulink, open the default template located in C:\imperix\BBSimulinkSDK\template\Bbox_template.slx. A copy should also be located in C:\Users\username\Documents\MATLAB. It is also fine to start with any code example provided by imperix.

It is highly recommended not to start with a Simulink blank model, since the template already contains the necessary configuration for the automated generation and flashing of the runtime executable.

The default template contains a basic skeleton to start working right away. The model contains two subsystems:

» *Plant_model* : contains the model of the system to be controlled. This is typically the model of the converter itself. More details are provided in section 2.2.

» *Closed_loop_control* : contains the control implementation that can be simulated or used to generate the control code for the BoomBox. It also contains a configuration block that performs the main configurations of the Simulink model. More details are provided in sections 2.4, 2.5 and 4.1.



2.2 PLANT MODEL

In order to run the control algorithm in simulation, Simulink needs a model of the converter hardware. As seen before, this model should be located inside the 'Plant_Model' block of the root view of your Simulink model.

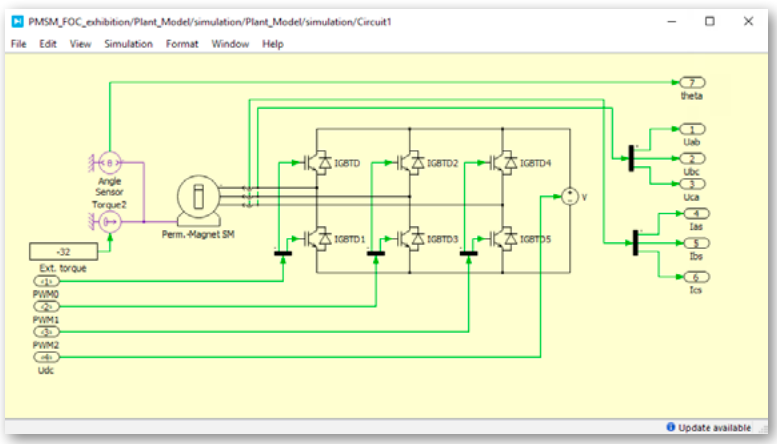
To model the converter hardware, it is recommended to use either *PLECS Blockset* or *Simscape Power Systems*.

2.2.1 PLECS BLOCKSET

PLECS is a simulation tool for power electronics provided by Plexim. The PLECS blockset for Simulink can be downloaded at *https://www.plexim.com/download/blockset*. With the installer, you can either install :

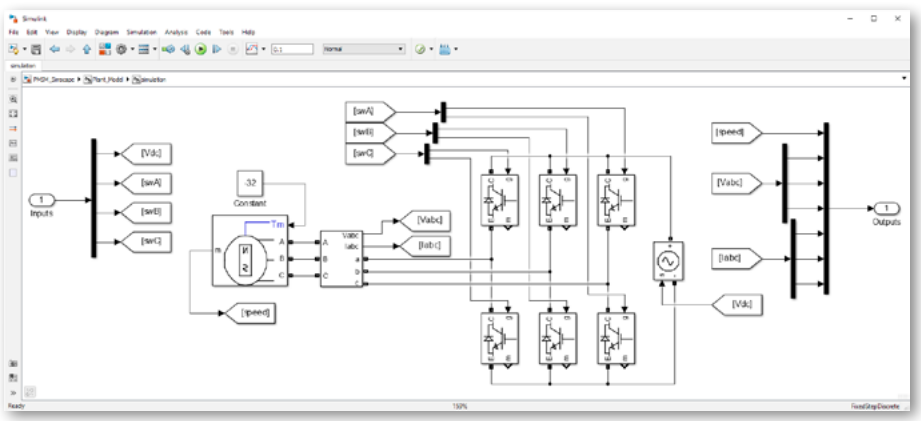
» *PLECS Blockset* : allows to create, edit, save and simulate a PLECS electrical circuit inside Simulink. It requires a paid license from Plexim. The license is neither provided nor sold by imperix.

» *PLECS Viewer* : allows only to view and simulate a PLECS electrical circuit that has been exported to work with PLECS Viewer. It doesn't require any license and is free of use.



2.2.2 SIMSCAPE POWER SYSTEMS

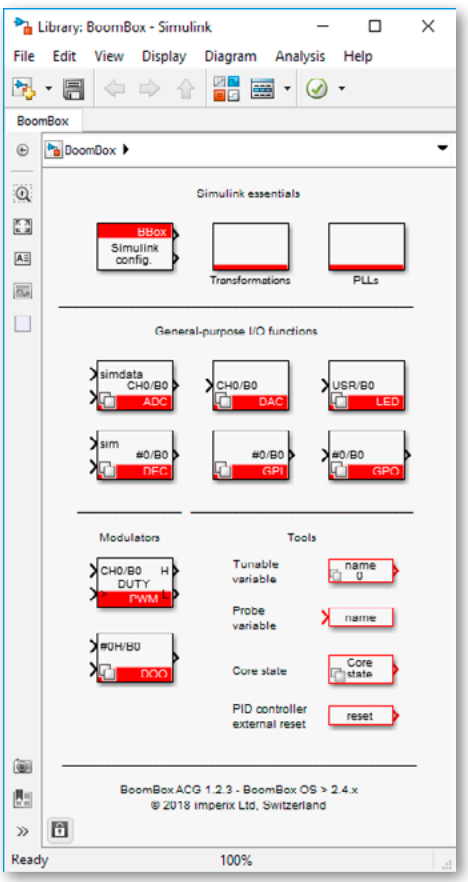
Simscape Power Systems is a MathWorks toolbox for Simulink that allows to model and simulate electrical power systems. It can be downloaded at *https://www.mathworks.com/products/simpower.html* and requires a paid license from MathWorks. The license is neither provided nor sold by imperix.



2.3 BOOMBOX BLOCKSET

Starting from the default template or an existing example, you can start implementing your own control, using the blocks provided in the BoomBox blockset, along with any standard Simulink block.

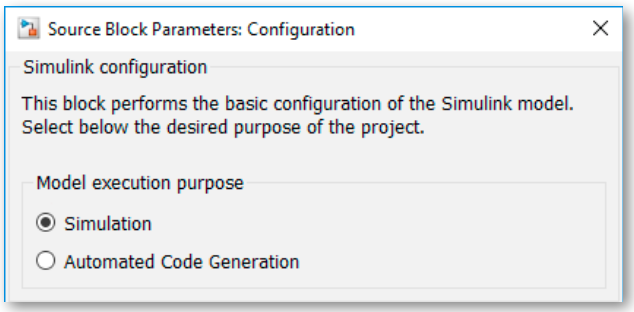
The blockset can be found in the library browser by clicking and browse to 'BoomBox'. Alternatively, you can open C:\imperix\BBSimulinkSDK\Lib\simulink\BoomBox.slx. It is shown below.



The main library blocks are described in section 4.1 and some recommendations regarding which blocks to use are given in section 4.3.

2.4 SIMULATION AND CODE GENERATION

Thanks to the power of Simulink and the way the blocks of the BoomBox library are implemented, you can do both simulation and Automated Code Generation (ACG) for the BoomBox from the same unique Simulink model. To switch from one configuration to the other, all you have to do is select the desired 'Model execution purpose' in the *Configuration* block of the BoomBox library, as shown below.



- In simulation, the model is simulated using the plant model drawn inside the *Plant_model* block. This block uses the simulation time step TSAMPLE to sample its input/output signals and solve the plant model. The signals of the closed-loop control are sampled at the control period CTRLPERIOD. Both those parameters can be specified in the *Configuration* block. More details are provided in section 4.1.
- In Automated Code Generation, C code is generated to be executed in the BoomBox. All the signals coming from and going to the *Plant_model* block are disregarded and are replaced by the analog inputs and the digital outputs of the BoomBox. The analog signals are sampled at the control period CTRLPERIOD and the digital PWM outputs switch at SWPERIOD. More details are provided in section 4.1.

2.4.1 RUNNING THE SIMULATION

To run the simulation, make sure that 'Simulation' is selected in the *Configuration* block of the model, and click the 'Run' button (Ctrl+T)  .

2.4.2 FLASHING THE BOOMBOX

To flash the code from Simulink, make sure that 'Automated Code Generation' is selected in the *Configuration* block of the model, and click the 'Build' button (Ctrl+B)  .

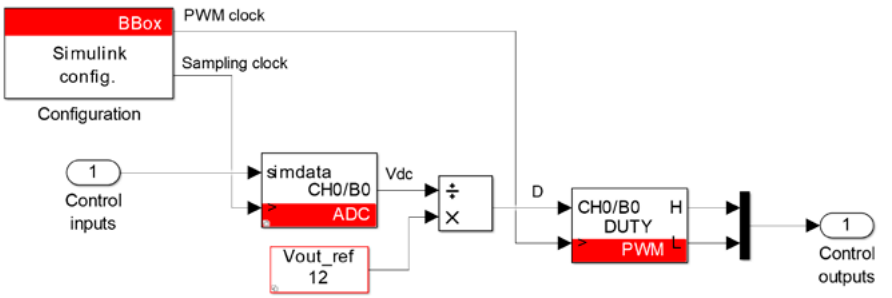
Note

This procedure will work, providing that a BoomBox is connected to the PC. To connect your BoomBox, follow the instructions of section 3.2.1.

With this approach, the program is directly loaded into the DSP's volatile memory and its execution is controlled using the emulator.

2.5 BASIC CONTROL EXAMPLE

For the sake of example, a very basic control algorithm of a buck converter is depicted below. It sets the output voltage at 12 V, regardless of the measured input voltage.



It contains:

- A *Configuration* block to define the model parameters (more details in section 4.1)
- An *ADC* block to retreive the simulated DC bus voltage in simulation, and the analog input on channel 0 of the BoomBox in ACG (more details in section 4.1)
- A *Tunnable Parameter* block to define a variable that is accessible and modifiable in real time from the BoomBox Control software (more details in sections 4.1 and 3.2)
- A *PWM* modulator to generate PWM signals with a duty-cycle *D*. The PWM signals are wired to the plant model in simulation and directly output on the digital outputs of the BoomBox in ACG (more details in section 4.1)

USING THE BOOMBOX

Abstract — This section describes how to connect the BoomBox to a computer and how to use it once the code has been flashed. It includes instructions to configure its front-panel analog inputs and explains how to use BoomBox Control.

Keywords — *XDS100v2 emulator, flash, frontpanel configuration, monitoring, BoomBox Control.*

3.1 CONNECTING TO THE BOOMBOX

For most applications, two physical links should be established between the BoomBox and the computer :

» *Emulator :* Connect TI's XDS100v2 emulator (provided with the BoomBox) to the JTAG port on the back end of the BoomBox and to the PC using a USB-mini cable. This link is required to interact with the BoomBox from Simulink.

» *Virtual serial port :* Place another USB (type B) cable between the console port of the BoomBox (back end) and the PC. This link provides the virtual COM port that is used by BoomBox Control.

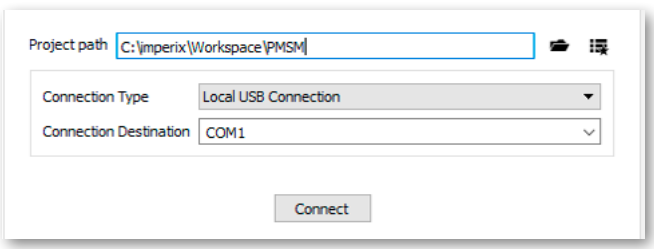
3.2 USING BOOMBOX CONTROL

3.2.1 ESTABLISHING A CONNECTION WITH THE BOOMBOX

Once the BoomBox is wired to the PC and the code has been flashed from Simulink, open BoomBox Control and proceed as follows :

» Select your working folder for code development (most likely somewhere inside your Simulink Workspace). This tells BoomBox Control where to look for code variables and other configuration files.

» Select 'Local USB Connection' and the COM port which corresponds to the BoomBox, usually the last one.



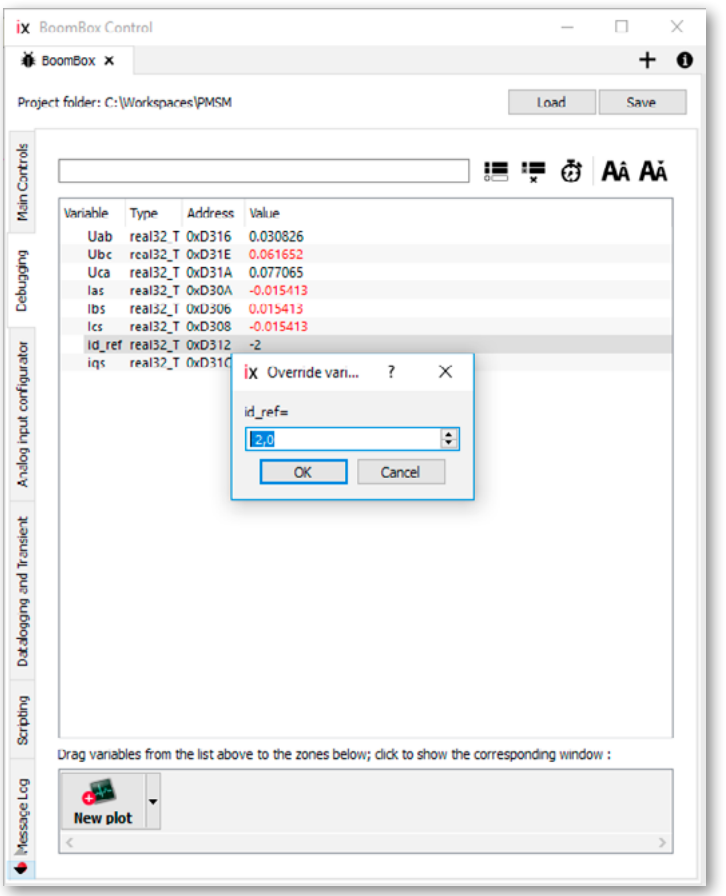
» Click connect to establish connection with the BoomBox.

Notes :

BoomBox Control communicates directly with the DSP. As such, a working code must have been downloaded and started inside the BoomBox. When the communication cannot be established, the most probable cause is that no code is running inside the BoomBox.

For a code to be running inside the BoomBox, it must have been previously flashed from Simulink, or be present inside the EEPROM.

3.2.2 WATCHING AND ALTERING VARIABLES



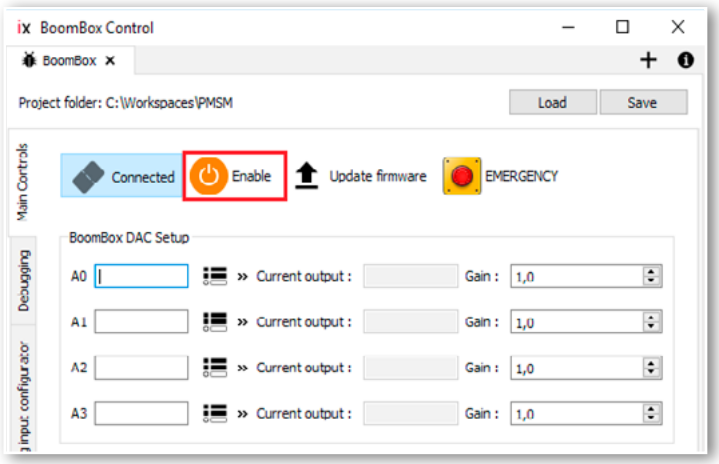
In the 'Debugging' tab, variables can be added to the watch list by typing their name in the top field and pressing Enter. Their current values can then be monitored in real-time in the list below. The user can also alter a variable's value by double-clicking on it (see figure above).

Additionally, by dragging a variable from the watch list and dropping it over the plot below, the variable's evolution in time can be viewed.

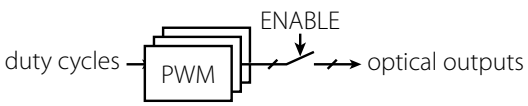
The maximum update rate of this plot is 10 Hz, so it is only suited to observe the evolution of slow-varying quantities. For fast phenomena and/or in order to retrieve consecutive samples, we recommend using the datalogging feature described in section 6.2.5 of the BoomBox User Manual.

3.2.3 ENABLING AND DISABLING GATING SIGNALS

Enabling and disabling the BoomBox's gating signals is as simple as clicking the 'Enable/Disable' button on the 'Main Controls' tab of BoomBox Control.



The enabling/disabling process is independent from the control algorithm running on the BoomBox and acts as a switch on all PWM signals :



3.3 CONFIGURING BOOMBOX FRONTPANEL

For each analog input, the following parameters can be configured:

- Gain (x2, x4 or x8)
- Low-impedance input (Yes or No)
- Filter (Yes or No) - If yes: filter frequency
- Low/high limit (between +/- 10 V)
- Disable safety (Yes or No)

These parameters can only by configured via the frontpanel of the BoomBox. This has been implemented on purpose to guarantee a complete independence between the control algorithm running on the DSP and the analog safety limits.

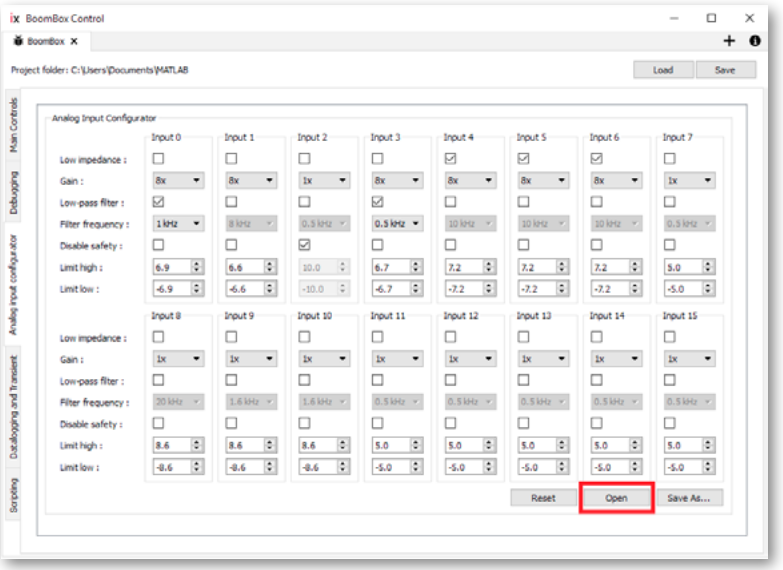
They are two ways of configuring the frontpanel: either directly on the frontpanel, using the rotary and push button, or via BoomBox Control and using a USB stick.

If the latter is chosen, proceed as follows:

a) Connect a USB stick to the USB port on the frontpanel of the BoomBox

b) Use the rotary and push button to select 'Backup config.' on the LCD screen. It will save the current analog configurations in a folder called 'imperix' on the root of the USB stick. The filename format is 'frontpanel#.bbox', where # is a number that gets incremented each time a new configuration is saved.

c) Connect the USB stick to your PC, open the generated configuration file inside the 'Analog input configurator' tab of BoomBox Control and perform the desired changes.



d) Save the new configuration file in the 'imperix' folder of the USB stick, keeping the filename in the form 'frontpanel#.bbox'.

e) Apply this new configuration by connecting back the USB stick to the BoomBox and selecting 'Restore config'. The BoomBox will read the configuration file in the folder 'imperix' with the highest number (#).

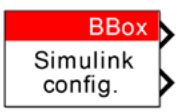
GOING FURTHER...

Abstract — This section aims to provide advanced material on the working principle of the BoomBox blockset. It also includes some guidelines and a list of common mistakes to avoid. Finally, it presents the steps to follow to flash the BoomBox in standalone.

Keywords — *Sampling time, interrupt period, library blocks, S-functions, standalone flash.*

4.1 MAIN LIBRARY BLOCKS

The main blocks of the BoomBox blockset (library) are described below.



» *Configuration* : defines the main configurations of the Simulink model, such as :

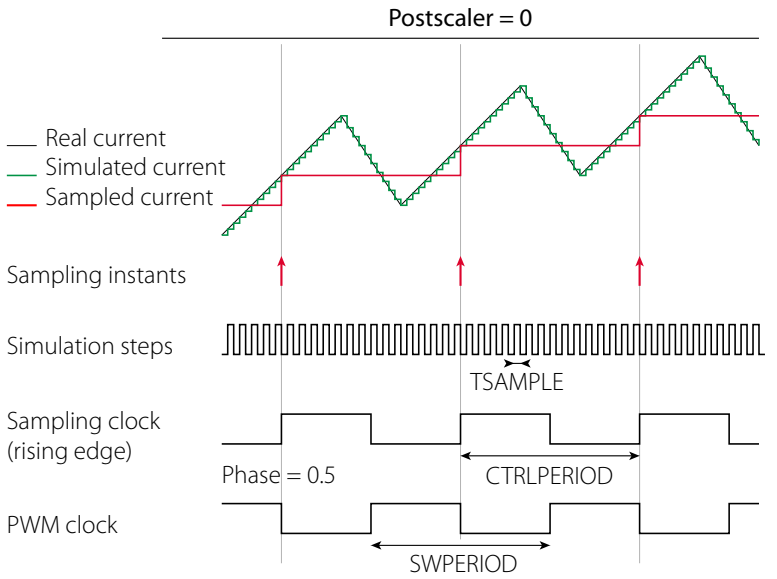
- The 'Model execution purpose' : the choice between 'Simulation' on Simulink and 'Automated Code Generation' (ACG) for the BoomBox.
- The 'Simulation step size' : the sampling time of the Simulink simulation. This parameter is only used in simulation and is disregarded in ACG. It can be accessed with the variable TSAMPLE.
- The 'Switching frequency' : the switching frequency of the PWM outputs. This is also the frequency of the main clock generator (ClockGen#0). Its period is accessible with the variable SWPERIOD.
- The 'Interrupt postscaler' : the clock divider between the main clock generator and the control interrupt. The control interrupt period is accessible with the variable CTRLPERIOD. This is also the sampling period of the ADC blocks.
  - *If postscaler = 0*, CTRLPERIOD = SWPERIOD.
  - *Otherwise*, CTRLPERIOD = SWPERIOD x (2 x postscaler).
- The 'Modulator arrangement' : the choice between using
  - 8 pairs of complementary PWM signals with configurable dead time or
  - 16 individual PWM signals without dead time.

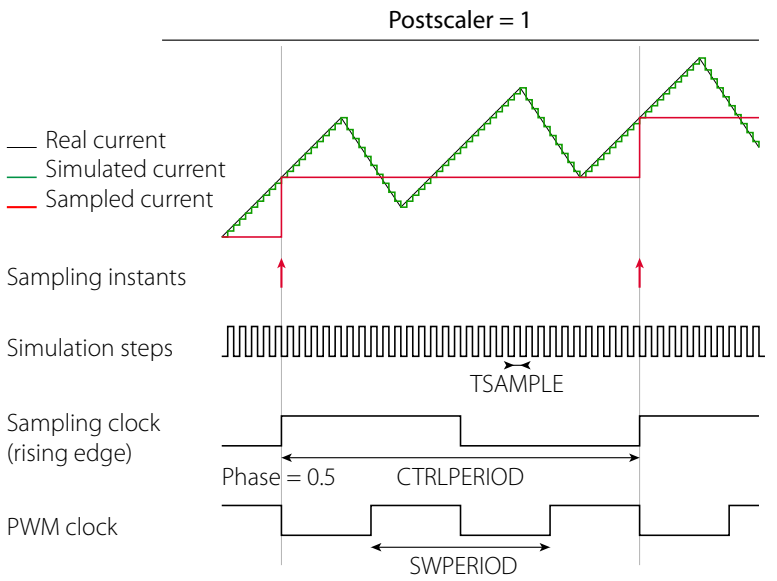
  If the latter is chosen, only the high-side PWM signals of the modulators are considered.

The outputs of the Configuration block are the PWM clock and the Sampling clock, to be wired to the PWM modulator blocks and the ADC blocks, respectively. These clock signals ensure that the sampling and the PWM generation in simulation are coherent with what happens in real-time on the BoomBox. Both clocks signals are disregarded in ACG.

Each Simulink model can contain one and only one configuration block.

The different clock signals and their influence on sampling are illustrated below for the cases postscaler = 0 and postscaler = 1.



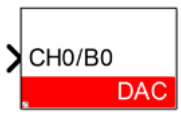


» *ADC* : the Analog to Digital Converter.

- In simulation, it acts as a sampler of the input signal 'simdata', which is typically a measurement signal coming from the simulated plant model. The lower input port must be connected to the Sampling clock signal of the *Configuration block*.
- In ACG, it retrieves the signal of the specified analog input channel as sampled by the BoomBox. The parameters to set are the 'Sensor sensitivity' and 'offset', and the 'Programmable gain value' of the corresponding analog channel, as set on the frontpanel of the corresponding BoomBox.

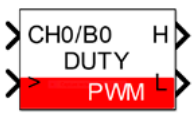
Note:

The programmable gain value of the 'Frontpanel configuration' is not automatically transferred to the frontpanel of the BoomBox and vice-versa. This parameter should correspond to the frontpanel configuration of the BoomBox that has been set manually, following the instructions of section 3.3.



» *DAC* : the Digital to Analog Converter.

- In simulation, this block is disregarded.
- In ACG mode, it allows to output any signal on one of the 4 analog outputs of the BoomBox (SMA connectors on the front end).

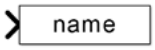


» *PWM* : the PWM modulator for a given PWM channel. The parameters are the 'Carrier type' and the 'Dead-time duration'. The inputs of the blocks are the wanted 'Duty-cycle' and/or the 'Relative phase'. Using the phase parameter as input allows typically to implement *phase shift modulation*.

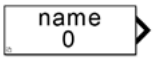
- In simulation, it outputs the PWM signals, which should be wired to the gates of the transistors of the simulated plant model. Dead-time is however not modelled in simulation. The lower input port must be connected to the PWM clock signal of the *Configuration block*.
- In ACG mode, the PWM signals are transferred to the optical outputs of the corresponding BoomBox PWM channel, providing the optical outputs have been 'Enabled' (see section 4.1). The outputs of the block are disregarded in ACG.

Warning

In ACG, special care should be taken of the 'Dead-time duration' parameter. If set too small, a large shoot-through current may appear, leading to damage to or destruction of the power switches. For safe operation, a value of 30 FPGA clock ticks (i.e. 1µs dead-time) is recommended.



» *Probe* : creates a global variable that is visible from BoomBox Control in ACG. This is typically used to plot any Simulink signal in real-time inside BoomBox Control.



» *Tunable parameter* : creates a global variable that is accessible and modifiable from BoomBox Control in ACG. This is typically used to change setpoints or controller gains in real-time from BoomBox Control. The variable can also be accessed from anywhere in the model by using a Simulink 'Data Store Read' block.

Further useful blocks are present, including coordinate transformations , a PLL, an incremental decoder, etc...

4.2 USER CUSTOM BLOCKS

In some situations, it is more convenient to write some code instead of using Simulink blocks only. For using C/C++ code, it is recommended to use *S-functions*. For MATLAB code, the *Matlab Function* block is more convenient. Both solutions are presented below.

4.2.1 S-FUNCTIONS

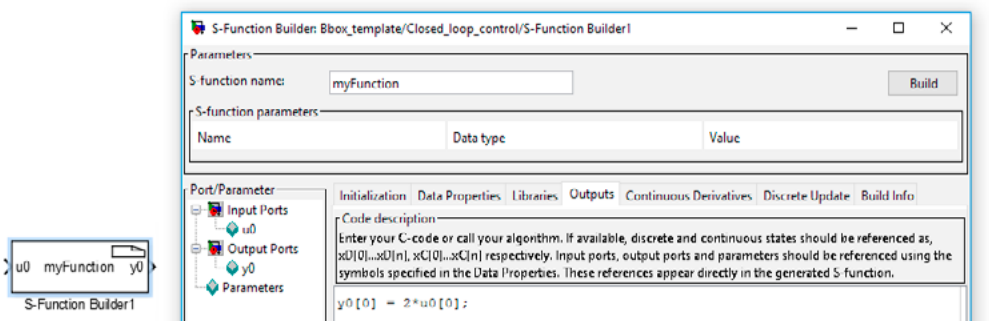
The S-functions allow to implement custom Simulink blocks written in C/C++ (among others) by compiling the code as a MEX file. They can be used in both simulation and ACG. To use them, proceed as follows:

» Use a 'S-function builder' block
  - In the 'Data properties' tab, define the input/output size, and the set their data type to *single*.
  - The C code has to be written in the 'Output' tab.

» Click the 'Build' button to build the code and generate the necessary files.

Note

This requires to have a compiler installed on the PC. You can find a list of the supported compilers at *https://ch.mathworks.com/support/sysreq/previous_releases.html*. The installed compiler can be displayed by taping the Matlab command `mex -setup`.



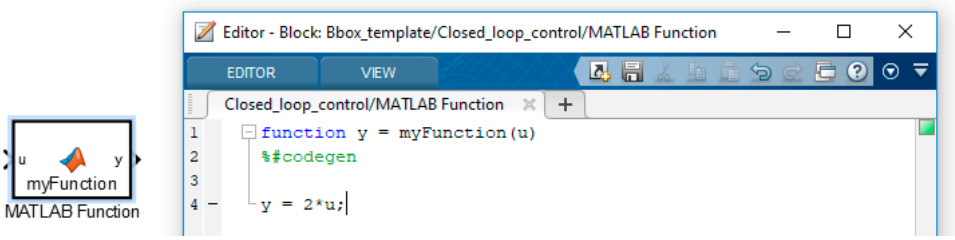
4.2.2 MATLAB FUNCTIONS

» For using Matlab code, the solution is to use the dedicated 'MATLAB Function' block. This will work in both simulation and ACG.

» A comprehensive list of the functions and objects supported for C/C++ code generation can be found at *https://ch.mathworks.com/help/simulink/ug/functions-supported-for-code-generation-alphabetical-list.html*.

Note

This requires to have a compiler installed on the PC. You can find a list of the supported compilers at *https://ch.mathworks.com/support/sysreq/previous_releases.html*. The installed compiler can be displayed by taping the Matlab command `mex -setup`.

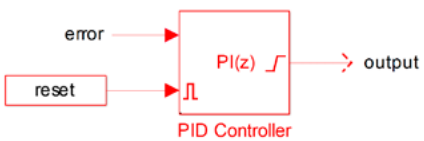


4.3 SOME GUIDELINES AND RECOMMENDATIONS

» You may want to use the standard *PID controller* block to implement your PID controllers. To use it, make sure to select 'Discrete-time' in the 'Time domain' field of the block configurations. In the 'Sample time' field, use the variable CTRLPERIOD.

In addition, to prevent the controller from integrating an error while the control is not yet running (i.e. when the PWM outputs are not yet enabled), the integral term must be reset when the PWM are disabled. This is only relevant when using 'Automated Code Generation', but it is good practice to implement this mechanism in simulation as well.

To do so, in the configuration of the PID controller, chose 'Level' in the 'External reset' field and connect the 'Reset' block provided in the BoomBox library to its reset port. The reset block outputs 1 while the PWM are disabled, and 0 otherwise. Your PID controller (respectively PI), should look like this:



» Any block requiring a sampling period (e.g. PI controller, integrator, delay, ...) should use the CTRLPERIOD variable. To avoid hazardous behaviors, no other value should be used.

» To verify that your code is running at the appropriate frequency, you can display the sampling frequency of each signal using colors. To do so, go to Display > Sample times > Colors and display the legend with Ctrl+J.

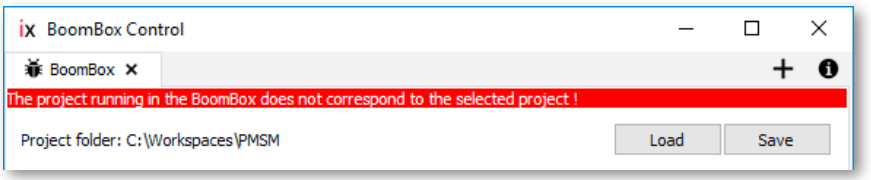
» It is recommended to use the *single* data type, instead of the default *double*. This guarantees the fastest code execution time on the DSP. When needed, the 'Data Type Conversion' can be used. A good practice when using constants is to define directly their data type as *single*, so that Simulink propagates properly the data types.

4.4 COMMON MISTAKES

Here are some common mistakes that should be avoided :

» If using multiple BoomBox on the same PC, note that only one JTAG connection can be used at a time. It is recommended to disconnect the JTAG that is not used. The support of multiple simultaneous connections is under development.

» When using BoomBox Control, if the selected folder does not contain the build files of the code that is currently running in the BoomBox, the following error message appears:



Rebuild the intended code or select the folder containing the appropriate build files to fix the problem.

» When loading the code in a BoomBox, if the LCD displays 'SOFTWARE FAULT', it can be that :

- The code you have flashed cannot be executed in real time. Reducing the interrupt frequency or simplifying the control algorithm can solve the problem.
- The FPGA firmware is not compatible with the Simulink library.
- There is an error in the interrupt configuration.
- The BoomBoxes have lost synchronisation.

If BoomBox Control is connected at that time, more details are displayed in the console.

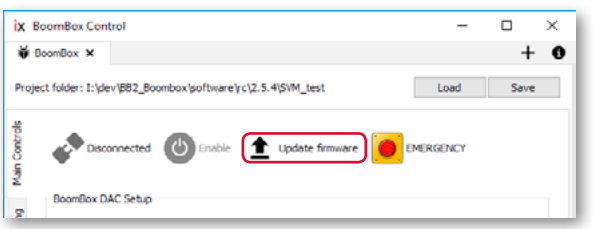
» If you cannot find the solution to your problem in this guide, do not hesitate to contact our support team at *support@imperix.ch*.

4.5 FLASHING THE BOOMBOX IN STANDALONE

With this approach, the program is flashed into the BoomBox startup EEPROM using the BoomBox Control software. The BoomBox will therefore automatically boot up on this code after every power cycle. This approach is useful when the BoomBox should later operate without a direct connection to a computer.

To do so, a connection to a BoomBox must to be established, using the procedure of section 3.2.1. Once the BoomBox is connected :

» Click the 'Update firmware' button on the 'Main Controls' tab.



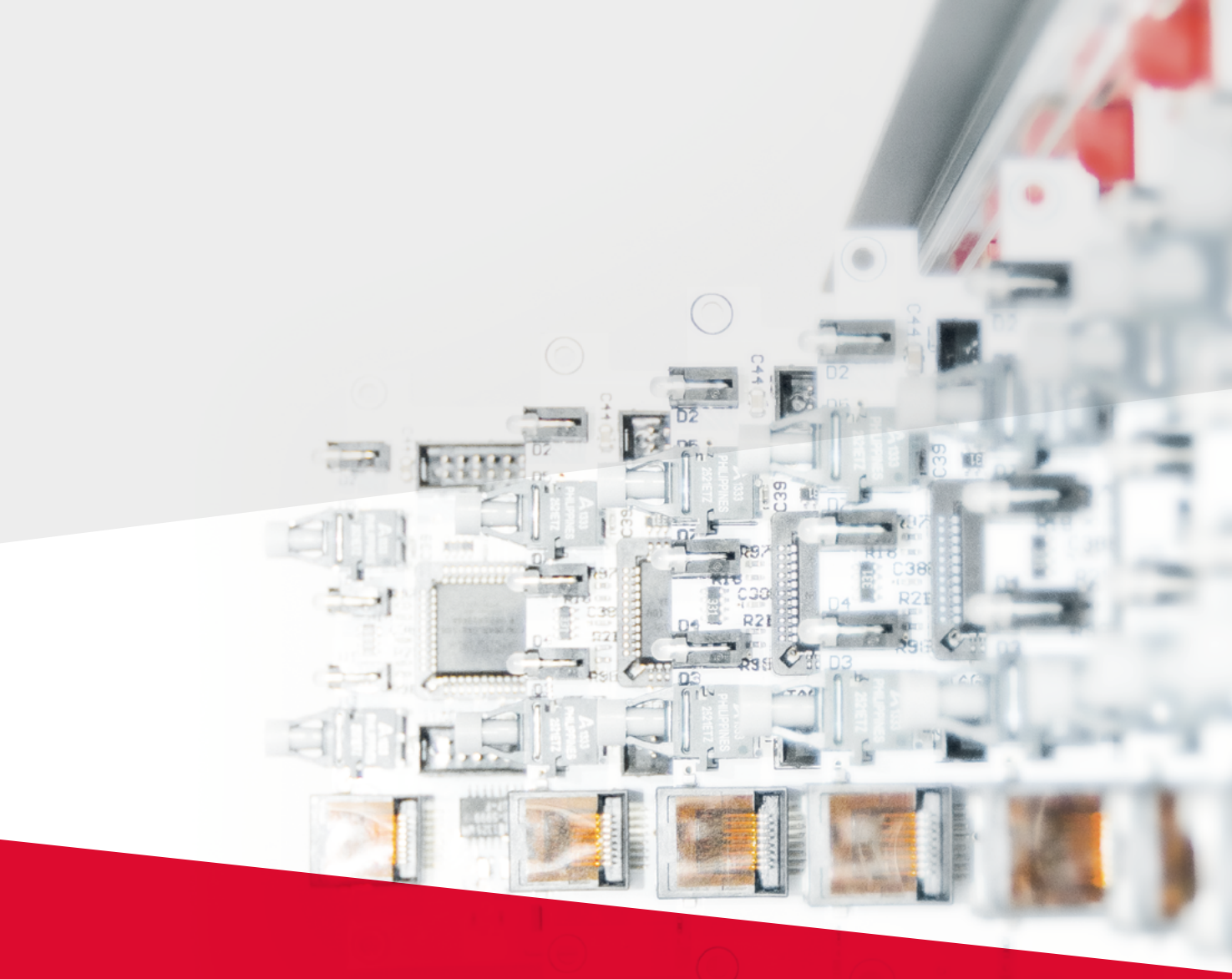
» Find the compiled output file in the Simulink workspace folder. It should be stored at the address : [Simulink Workspace folder]/[YourProjectName]/Debug_BoomBox/

» Select the '*.a00' file, and open it.

Note :

The '*.a00' file is generated by Simulink after a successful build. In consequence, this file is not produced until the compilation and linking processes have succeeded.

» A set of messages are displayed in the BoomBox Control console, confirming that the BoomBox has been programmed (its EEPROM is written) and rebooted.



imperix Ltd.

Rue de la Dixence 10
CH-1950 Sion
Switzerland

Phone +41 (0)27 552 06 60
Fax +41 (0)27 552 06 69

www.imperix.ch
support@imperix.ch

Find your closest distributor on imperix.ch/resellers