REXYGEN Studio

User guide

REX Controls s.r.o.

Version 3.0.3
2024-11-03
Plzeň (Pilsen), Czech Republic

Contents

Chapter 1

Introduction

REXYGEN Studio is a graphical tool designed for the creation of real-time control algorithms with the support of a large function block library of the REXYGEN system [1]. A developer may utilize many function blocks from simple comparators and timers to advanced and specialized blocks designated for analog signal processing and regulation. There are various specialized regulators including PID regulators with automatic tuning of parameters. Any algorithm developed in the REXYGEN Studio may be instantly compiled and downloaded into arbitrary target device (Linux IPC, WAGO PFC 100/200, Raspberry Pi, TinkerBoard, RockPi, Pigeon PLC, Unipi Neuron/Axon/Patron, Insys MRX, etc.). After the algorithm is successfully compiled and downloaded to the target device, it is possible to switch REXYGEN Studio to a *Watch* mode in which parameters and variables of function blocks may be inspected or adjusted. A connection may be established locally or over the Internet using standard protocol IPv4 or IPv6. A secure connection over SSL protocol is supported.

Detailed documentation for the function block library may be opened at any time with the F1 key. The documentation is installed automatically in the default setup configuration and we advise not to uncheck it.

1.1 First Run

When you start REXYGEN Studio for the first time, you will be greeted by the dialog box (figure 1.1) where you have several options to start your work. If you choose the *Start with the Plain Project* option, you will be prompted to choose a folder where the project will be saved and a project will be created with minimal configuration that can be uploaded to your target device. The REXYGEN installation includes an extensive library of examples. We recommend starting the project by choosing an example, which you will then modify. The *Start from the Example Project* option is used for this. If you have an already prepared project available, you can open it using the *Open an Existing Project* option. The last option - *Load Project from the Target Device* will allow you to retrieve the currently saved project from the target device. Attention: The source files of the project must be stored on the device!

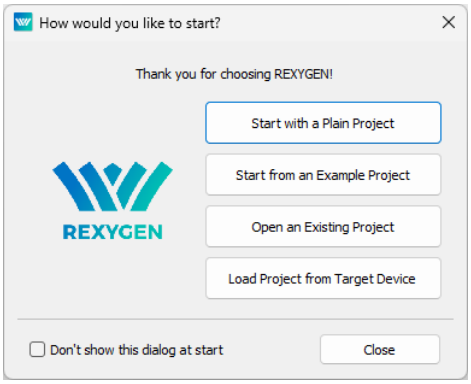

Figure 1.1: Start Menu

1.2 Application Modes

The application provides two modes: a *Development* mode and a *Watch* mode. An algorithm is developed in the *Development* mode by drawing a function block diagram on the canvas and by configuring function block parameters. The application may or may not be connected to a target device.

When connected to the target, the application may be switched to the *Watch* mode. In the *Watch* mode, the developed application can't be modified. Instead, the function block diagram serves as a diagnostic interface to the running algorithm. Users may inspect all signals, parameters, trends and detailed diagnostics of the target.

Chapter 2

Compile, Download and Diagnose Your Project

A project must be compiled and the resulting binary configuration has to be downloaded into the target device to put a control algorithm into operation. Both actions are performed directly from the REXYGEN Studio development tool.

2.1 Project Compilation

After an algorithm is designed[1], it is necessary to validate its structure, device configurations and block connections by *compilation*. The *compilation* results in a binary file called *binary configuration* which may be downloaded into a target device. The *binary configuration* is a platform/target independent file format with a `.rex` file name extension. The format of the *binary configuration* is optimized for fast processing on a target device and has several consistency check mechanisms.

Warnings and errors may occur during a compilation. All warnings, errors and other various informational messages are printed into the compile log. A warning does not cause a termination of the compile process. But it should be observed carefully as it may probably indicate a problem that may lead to unexpected behavior of the algorithm. All errors are treated as serious problems and cause a termination of the compiling process and no *binary configuration* is generated. All errors and warnings are indicated by their respective numbers and textual information message.

A *compilation* is started from the *Compiler* menu. A project is compiled by selecting *Compiler/Compile* from the menu, clicking on the icon on the toolbar or using the `F5` shortcut.

[1] For instructions on how to design the algorithm, read the Getting Started manual for your target device.

2.2 Downloading Project

A configuration is downloaded into the target device by selecting *Compiler/Compile and Download* from the menu, clicking on the icon on the toolbar or using the F6 shortcut.

A compilation of a project into a *binary configuration* is always performed before it is downloaded into the target device to ensure consistency. If an error occurs during the compilation, a compiler window is shown and the download operation is not available. A user must fix all errors indicated by the compiler before proceeding with the download.

The download option *Store configuration permanently* is required to save the project configuration on the target device so that it gets loaded automatically when the target device boots up (e.g. after a restart, power cycle, power loss, etc.).

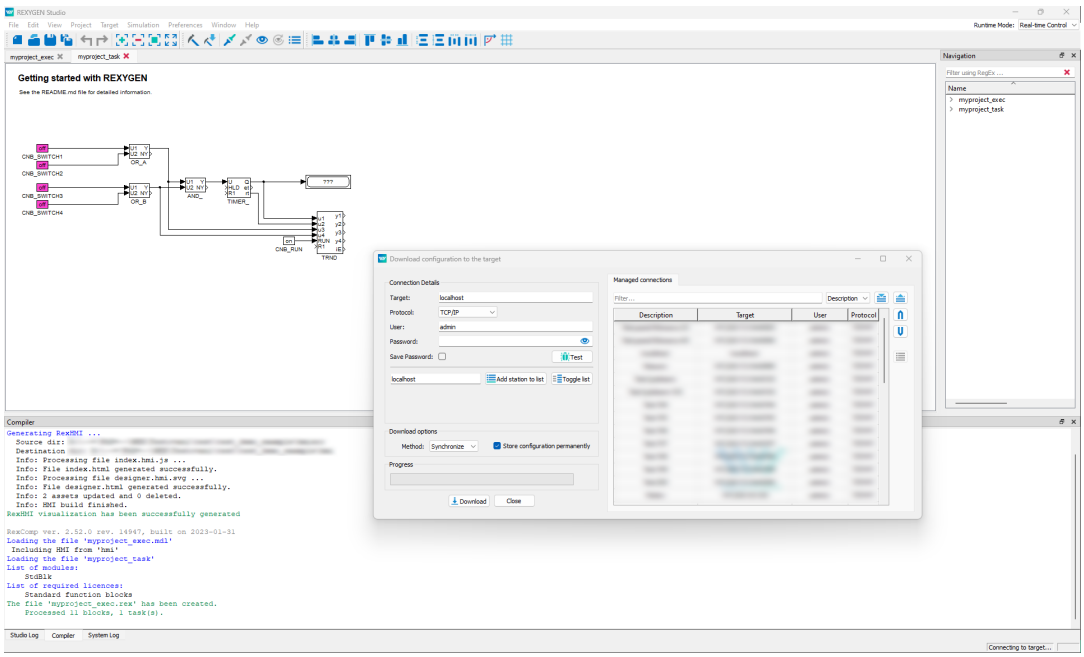

Figure 2.1: Project Compilation and Download Dialog

2.3 Licence Activation

Before a binary configuration may be downloaded and activated the target has to be properly licensed. For a permanent target operation, a full target licence can be obtained from www.rexygen.com/pricing. A DEMO licence is available for experiments and development. A DEMO licence may be obtained directly from the REXYGEN Studio application while connecting to an unlicensed target. A user may also get the DEMO licence from the web page of the REXYGEN.

An information dialog about licensing is shown while connecting to an unlicensed target (figure 2.2). A user has the following three options:

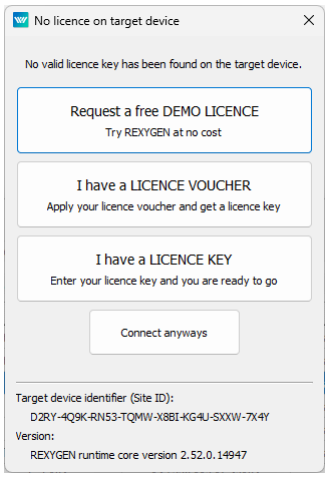

Figure 2.2: No Licence Dialog

Request a free DEMO LICENCE: Select this option if you want to obtain a DEMO licence directly by filling in the name and email address to which a licensing key is sent (figure 2.3).

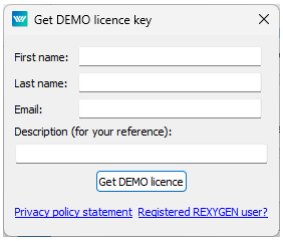

Figure 2.3: DEMO Licence Dialog

I have a LICENCE VOUCHER: Select this option if you already have a valid licence voucher and you want to apply it to the target device.

I have a LICENCE KEY: Select this option if you already have a valid licence key and you want to apply it to the target device.

Connect anyways: Select this option if you want to continue without licensing the target. But be aware that most of its functionality is not available without an active licence.

2.4 Watch mode and Diagnostics

A comprehensive diagnostic tool is integrated directly into the REXYGEN Studio. Before using the diagnostic tool, a binary configuration must be successfully downloaded into the target device. For information about downloading a binary configuration into the target device see the section 2.2.

A diagnostic tool in REXYGEN Studio may be used only when there is an active connection with the target device established. The connection is established by clicking on *Target/Connect* from the menu, clicking on the icon on the toolbar or using the `F7` shortcut. The connection may also be established after a binary configuration is downloaded into the target device and a dialog is shown in which a user is asked if the connection with the target should be preserved and the so-called *Watch mode* should be activated.

A *Watch* mode is indicated by a gray background of all windows with function block diagrams. It is not possible to add, delete or move function blocks or connections between blocks in the *Watch* mode. Instead, a user may inspect and adjust signals of block inputs, outputs and parameters and observe signal values in real-time with a refresh period that may be adjusted in the *Settings/Diagnostics Options* dialog and which is one second by default.

2.4.1 Monitoring signals

It would be inefficient and bandwidth-consuming if all signals were monitored immediately when the *Watch* mode is activated. For that reason, only the `DISPLAY` blocks are monitored automatically. A user has to switch other function blocks into the so-called *Monitoring State* if signals of a particular block should be monitored.

When a block is in the *Monitoring State*, all input and output values are displayed at the corresponding pin of the monitored block. A block is added to the *Monitoring State* by selecting *Target/Watch Selection* from the menu or using the `Ctrl+W` shortcut. A block is removed from the *Monitoring State* by selecting *Target/Exclude from Watch Selection* from the menu or using the `Ctrl+Shift+W` shortcut. There is also an option *Target/Exclude all from Watch Selection* in the menu to stop watching of all signals in the running project.

The values of parameters are shown in a floating yellow window when a user positions the mouse cursor over a function block. Parameters may also be inspected and adjusted from the online block properties dialog that is invoked by double-clicking or by pressing `Ctrl+E`. Be careful when modifying the values of parameters as changes are immediately committed to the target when the *OK* or *Apply* button is selected.

A dialog that allows transferring all changes to the parameters made in the *Watch* mode back to the drawing is displayed before exiting *Watch* mode. A user may accept or discard changes to all parameters individually. This simplifies an effort to keep a project synchronized with the running configuration.

A user should always observe the right part of the status bar where the state of the connection with the target device is indicated. A green color indicates that the

connection is active and the data is communicated periodically without any errors. A red color indicates that the connection with the target device has been unexpectedly closed or an error occurred during the data exchange.

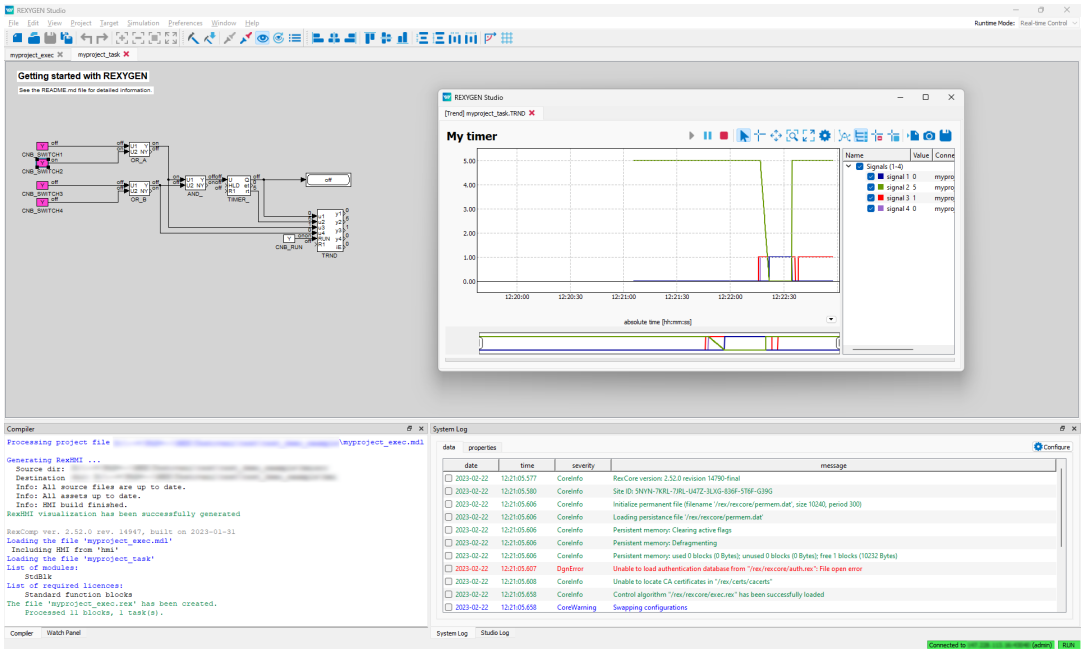

Figure 2.4: Monitoring and Diagnostics in the *Watch* Mode

2.4.2 Viewing trends

The blocks `TRND` and `TRNDV` are available in REXYGEN to monitor signals over a period of time. The `TRND` allows to store up to 4 signals and the `TRNDV` allows to store up to 64 signals in the RAM of the target device. The samples are stored synchronously during task execution. This allows us to store and observe very fast processes without losing a single sample.

The diagnostic tool of the REXYGEN Studio makes it possible to show all the data stored in the memory of the `TRND` block in the *Watch* mode. To show a window with all data signals shown in a graphical trend simply double-click on a `TRND` block while the REXYGEN Studio is in the *Watch* mode. A data may be zoomed, a scale and a range of both axes may be changed and the data may also be exported to a `CSV` file from the window.

2.4.3 Viewing REXYGEN System Log

A system log with errors, warnings and other info messages with timestamps are present on every target device. A user may display the content of the system log directly from

the REXYGEN Studio selecting *Target/Show System Log* from the menu or by clicking on the icon on the toolbar. A user may browse all messages, filter the content or export selected/all items to a `CSV` file.

The types and severities of messages that are written to the REXYGEN System log may be configured by selecting *Target/Configure System Log* from the menu. The *Safe print flags to the target* option must be selected if it is required to store the configuration of the system log permanently on the target device. A last saved configuration is loaded during the next system boot if the option remains unselected.

2.4.4 Diagnostics

A detailed diagnostics tool may be launched by clicking on *Target/Diagnostics* from the menu or by clicking on the icon on the toolbar. The diagnostic tool window is closed automatically when REXYGEN Studio disconnects from the target device.

Chapter 3

Creating a Library of Reusable Components

In REXYGEN, creating reusable components is based on using the so-called subsystems. A subsystem is a container for a group of function blocks and their connections, which then appear as a single block. Nesting of subsystems is allowed, i.e. a subsystem can include additional subsystems. Any subsystem can be turned into a reusable component.

A reusable component can be created in 3 steps:

1. Creation of a subsystem
2. Declaration of subsystem parameters
3. Moving the subsystem to a library

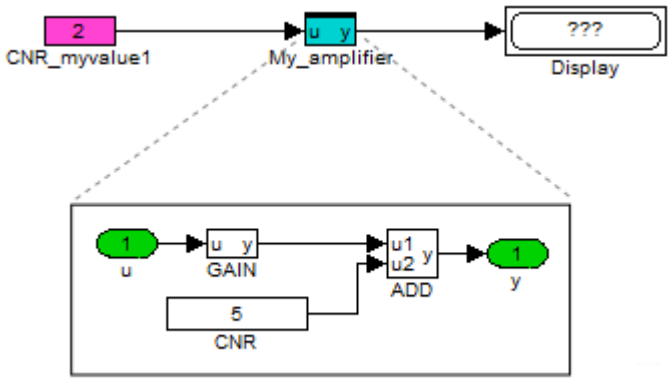

Figure 3.1: Subsystem – A container for a group of function blocks

3.1 Creation of a Subsystem

There are two possible ways of creating a subsystem in REXYGEN Studio:

- Copy the `SubSystem` block from the `INOUT` library to the given diagram (`.mdl` file). Enter the subsystem with a double-click. Duplicate `Inport` and `Outport` blocks as needed. Insert function blocks as needed. Route the signals from Inports through function blocks to Outports. Rename Inports and Outports as needed.
- Select a group of blocks and use the `Create subsystem` command (in menu *Edit→Create subsystem*). The selected blocks are then replaced by a subsystem block, which contains all the original blocks and `Inport` and `Outport` blocks for signals crossing the subsystem boundary.

The resulting subsystem contains function blocks and their connections as shown in Figure 3.1.

3.2 Declaration of Subsystem Parameters

A typical requirement for a subsystem is to have parameters just like a standard function block to allow the parametrization of its functionality.

For this purpose, the so-called *subsystem parameters* can be defined. The values of parameters can be used inside the subsystem. Each parameter also has a description of its meaning. To declare subsystem parameters, select one subsystem block and go to menu *Edit→Declaration of Parameters*. A dialog appears (see Figure 3.2) and you can declare parameters and the corresponding descriptions.

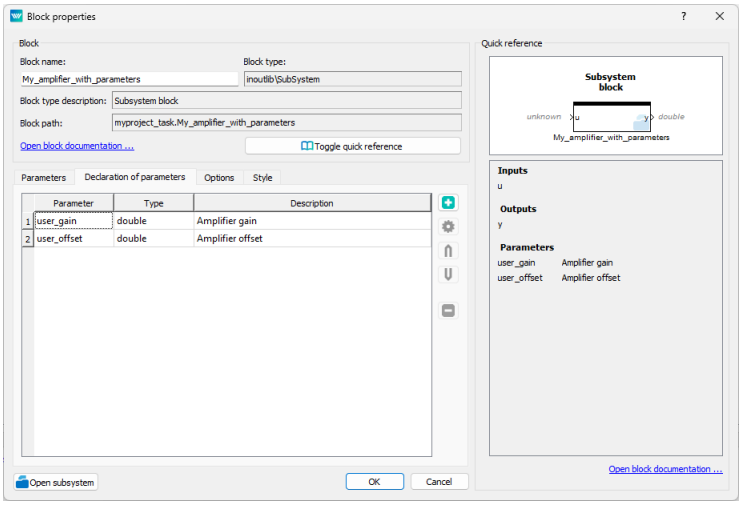

Figure 3.2: Subsystem – Declaration of parameters

Once the parameters are declared, the subsystem acts just like any other block – double-click it to open the *Block properties* dialog. You can see the dialog contains the parameters which were declared (Figure 3.3).

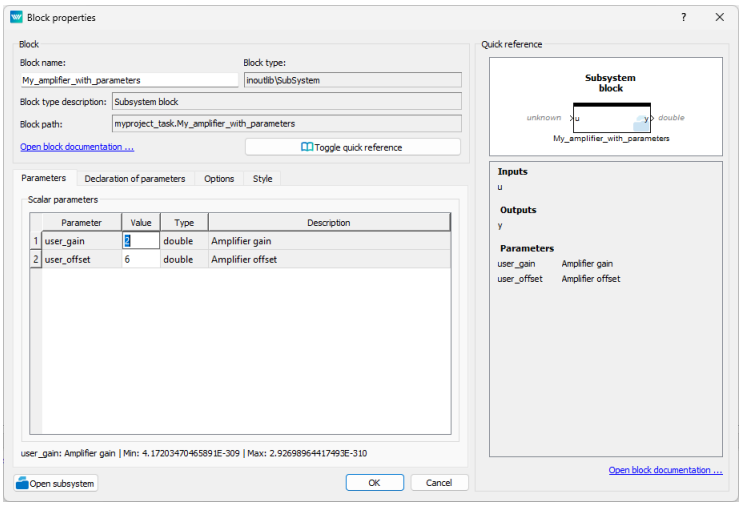

Figure 3.3: Subsystem – User-defined parameters

Now you need to edit the internals of a subsystem with parameters. Select it and go to menu *Edit→Open Subsystem*.

The function blocks inside a subsystem must be configured to accept the values of the declared parameters. To do so, open the Block properties dialog of individual function blocks, tick the `Inherit` checkbox and select the corresponding parameter (Figure 3.4).

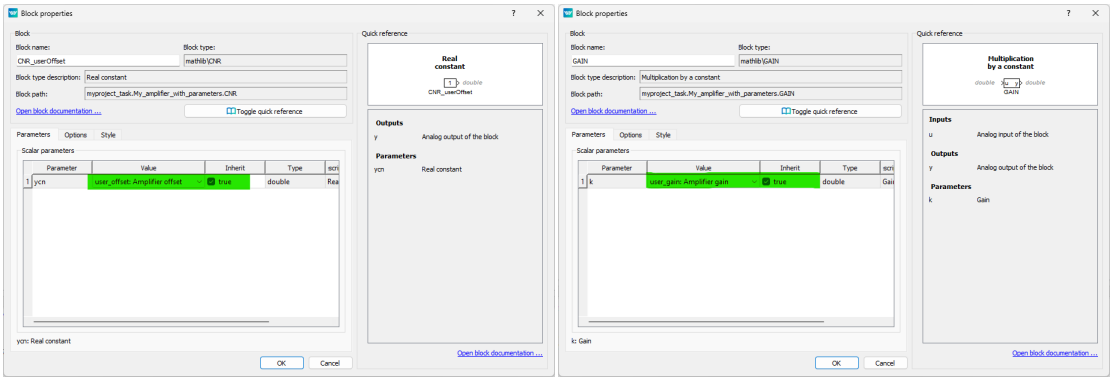

Figure 3.4: Block properties dialog – Accept subsystem parameter

The internals of the finished subsystem with parameters is shown in Figure 3.5.

In the end, you can use the subsystem just like any other function block and you'll find yourself configuring its behavior using the *Block properties* dialog. If everything works

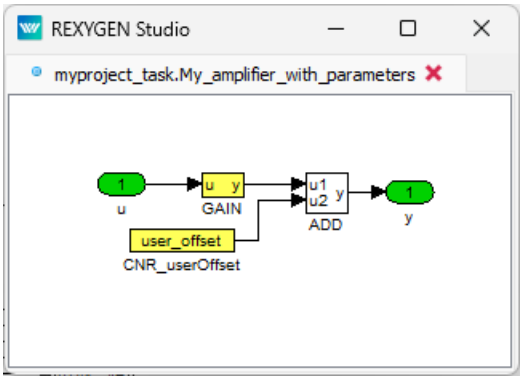

Figure 3.5: Subsystem with parameters – Internals of the finished subsystem

well, you might never go inside the subsystem again. If you need to debug your block or add new features/inputs/outputs, you can always *Open Subsystem* and do what's needed.

Please refer to an example project 0101-02 which demonstrates the use of subsystems. The example projects are a standard part of REXYGEN Studio.

3.3 Moving the Subsystem to a Library

Once you have your subsystem working, with or without parameters, chances are that you'll want to use it in multiple tasks or projects. For such a purpose, you'll create your Library of subsystems. Your very own problem-oriented library of reusable components.

To do so, create a new file in REXYGEN Studio and go to menu *File→Document properties* and change File Type to a *Library*.

Now you can fill your library with as many subsystems as you like. Just drag&drop subsystems into the library.

Afterward, save the library to the project folder. It is recommended to include 'library' in the filename so you can easily recognize it later. The extension is `.mdl`, just like in other REXYGEN Studio files.

Once you close the library and open it again, you'll notice its background is light blue and the library is in read-only mode. If you want to make some changes in the library, go to menu *File→Unlock Library*. This will make the library editable and you can do what is needed. Keep in mind that the changes you make will be applied to all files and projects referencing the library (we'll get to that in chapter 4).

Note 1: The subsystems in the library can be standalone or they can be built on top of other subsystems from the same or other user libraries.

Note 2: All files and projects referencing components in the library refer to the filename of the library. Renaming an existing library will lead to breaking references in all places. IT IS HIGHLY RECOMMENDED NOT TO RENAME A LIBRARY once you have started using its contents in your projects.

Chapter 4

Using User-Defined Libraries of Reusable Components

4.1 Including a Library in Your Project

When you have your library ready (or when you want to use a 3rd party library of function blocks), you must include the library in your project. There are three ways to do so:

1. Place the library (`.mdl` file) in the project folder.
2. Include the `PROJECT` block in the project main file and configure its `LibraryPath` to point to the folder with the library. The path can be absolute or relative to the project folder.
3. Define a global path to libraries in *Preferences→Advanced*. The path must be absolute.

4.2 Using Function Blocks From a Library

To use a function block from a library in your project, open the library like any other file in REXYGEN Studio and drag&drop function blocks to your project. This creates the so-called references to library blocks, which you can use just like all the native blocks of REXYGEN.

A library reference can be distinguished from a standard subsystem by the style of the upper border.

The difference from a standard subsystem (and the greatest benefit and the main purpose of a library reference) is that the content of the subsystem is given by the library. Therefore if you or anyone else update the library, the changes will be distributed to all subsystems referencing the library. To apply the changes in a project, you'll need to open the project in REXYGEN Studio.

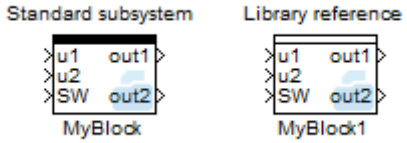

Figure 4.1: Standard subsystem vs. Library reference

Please refer to the example project 0101-03 which demonstrates the use of user-defined libraries. The example projects are a standard part of REXYGEN Studio.

Note: All files and projects referencing components in the library refer to the filename of the library. Renaming an existing library will lead to breaking references in all places. IT IS HIGHLY RECOMMENDED NOT TO RENAME A LIBRARY once you have started using its contents in your projects.

4.3 Detaching a Library Reference

In some cases, you might need to break the link between the library reference and its origin. WARNING! This process is IRREVERSIBLE. This will turn the library subsystem into a standard subsystem. To do so, select the block and go to menu *Edit→Break Library Link*. You'll immediately notice that the upper border of the block is now a thick full line, indicating that the subsystem is no longer dependent on the library.

Chapter 5

Block Options

Functional blocks can be partially influenced in their behavior during runtime. For example, it is possible to stop the execution of a specific block. This setting can be adjusted in the `Options` tab of each block.

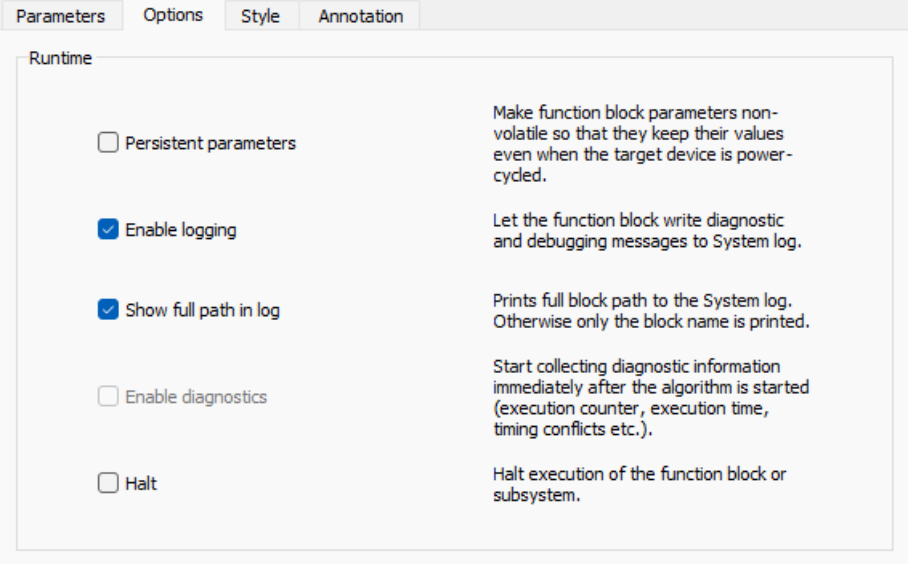

Figure 5.1: Block Options

5.1 Persistent parameters

REXYGEN supports permanent storage of function block parameters. Internal states of selected function blocks can be made persistent as well. Persistent memory performance is target platform specific. If the target platform does not have any supported persistent memory available, the permanent parameters will be stored to the `permem.dat` file by

default (to the same directory as exec.rex is stored - `/rex/rexcore` on Linux targets or `C:\ProgramData\REXControls\REX_<version>\RexCore\` on Windows targets).

In order to store the function block parameters, the checkbox `Persistent parameters` in block properties must be checked. All function block parameters and also internal states in selected blocks will be stored (excluding arrays).

The list of function blocks with persistent internal states follows:

- INTE
- SINT
- COUNT
- SHIFTOCT
- OSD
- SHLD
- DIF_
- TIMER_
- ABSROT
- E/ATMT

The default save period is 300 seconds, default persistent memory size is 10240 bytes.

The persistent memory can be reset or cleaned up using Download Dialog - see Fig. 2.1. Switch the Download option in Download Dialog to `Expert mode` and choose between `Reset persistent memory` or `Cleanup persistent memory`. `Reset persistent memory` will replace the entire persistent memory however `Cleanup persistent memory` will erase just the parameters which are not used anymore.

Permanent storage file, its size, location and save period can be changed using configuration file `rexcore.cfg`. Find more details in the RexCore User Guide[2].

Be aware that the block parameters will be overwritten with the ones stored in persistent memory immediately after swapping executives if the `Persistent parameters` option is enabled for the function block.

5.2 Enable logging

Certain function blocks such as `ATMT`, `PIDU`, and others have the capability to write diagnostic and debugging messages to the System log. To enable this functionality, the `Enable logging` option must be activated. Moreover, you must select the appropriate logging level in `System Log Configuration -> Function block messages`. For the `SYSLOG` function block, this option has a special meaning. When it's not active, `SYSLOG` only writes the message from the input `msg`. However, when it's active, the message is prefixed with the `SYSLOG` name.

5.3 Show full path in log

Prints full block path to the System log. Otherwise only the block name is printed. This option only makes sense when `Enable logging` is checked.

5.4 Enable diagnostics

Start collecting diagnostic information immediately after the algorithm is started (execution counter, execution time, timing conflicts etc.). This option is intended for tasks, drivers and subsystems blocks.

5.5 Halt

Halt execution of the function block or subsystem.

Chapter 6

Backup and restore configuration

6.1 Backup

Target→Backup (Target → PC)

Backup offers a way of creating a backup of a control algorithm that is stored on a target device. A user specifies a local file name into which the backup is stored. These components of the control algorithm are stored in a backup: executive, human-machine interface and project source (if the project source is stored on the target). A text file is created beside the backup that contains detailed information about the content of the backup.

6.2 Restore

Target→Restore (PC → Target)

Restore offers a way of restoring a control algorithm from a backup file to the target device. The operation is very similar to a download operation but a user may specify a file name of a backup.

6.3 Sources on Target

To store the source files on the target device include the `PROJECT` block in the project main file and set its `SourcesOnTarget` parameter to True.

Chapter 7

Supported Operating Systems

Development tools of REXYGEN supports following operating systems:

- 64-bit Windows 10 and later
- Linux amd64, armhf, arm64 (Debian based)

Chapter 8

Keyboard Shortcuts

Keyboard shortcut	**Function (as presented in the menu)**
CTRL+N	New File
CTRL+O	Open File
CTRL+S	Save File
CTRL+P	Print
CTRL+Z	Undo
CTRL+Y	Redo
CTRL+A	Select All
CTRL+X	Cut
CTRL+C	Copy
CTRL+V	Paste
CTRL+R	Rotate Block Clockwise
CTRL+SHIFT+R	Rotate Block Counterclockwise
CTRL+F	Find Function Block
CTRL+E	Properties
CTRL+M	Declaration of Subsystem Parameters
CTRL+U	Open Subsystem
CTRL+G	Create Subsystem
CTRL+L	Block Library
CTRL+W	Watch Selection
CTRL+SHIFT+W	Exclude from Watch
F1	Help
F2	Zoom In
F3	Zoom Out
F4	Zoom Default
F5	Compile
F6	Compile and Download
F7	Connect
F8	Disconnect
F9	Activate/Deactivate *Watch* Mode

Table 8.1: Keyboard shortcuts in REXYGEN Studio

Chapter 9

Other Important Features and Tools

9.1 Human-Machine Interface

Each target device running the RexCore contains an integrated webserver. The web server and the webpages it provides may act as a Human-Machine Interface (HMI) for your application. The web interface with HMI may be opened from REXYGEN Studio by selecting *Target/Web Interface* in the menu.

Please refer to the documentation of the `HMI` block in [1] on including the HMI in your project. All types of HMI supported by REXYGEN are described in [3].

9.2 REXYGEN HMI Designer

One of the approaches to creating an HMI for your application is to use the REXYGEN HMI Designer.

The REXYGEN HMI Designer may be started directly from the REXYGEN Studio by clicking on *Tools/REXYGEN HMI Designer* from the menu or by clicking on the icon on the toolbar. See [4] and [3] for details on using the REXYGEN HMI Designer.

List of Figures

Bibliography

[1] REX Controls s.r.o.. *Function blocks of REXYGEN – reference manual*, 2020. →.

[2] REX Controls s.r.o.. *RexCore – User manual*, 2020. →.

[3] REX Controls s.r.o.. *REXYGEN HMI – User manual*, 2020. →.

[4] REX Controls s.r.o.. *Getting started with REXYGEN*, 2020. →.

Documentation reference number: 16843