Model-Based Design Toolbox S32K3xx Series

Release Notes

Automatic Code Generation for the S32K3xx Family of Processors
Version 1.2.0

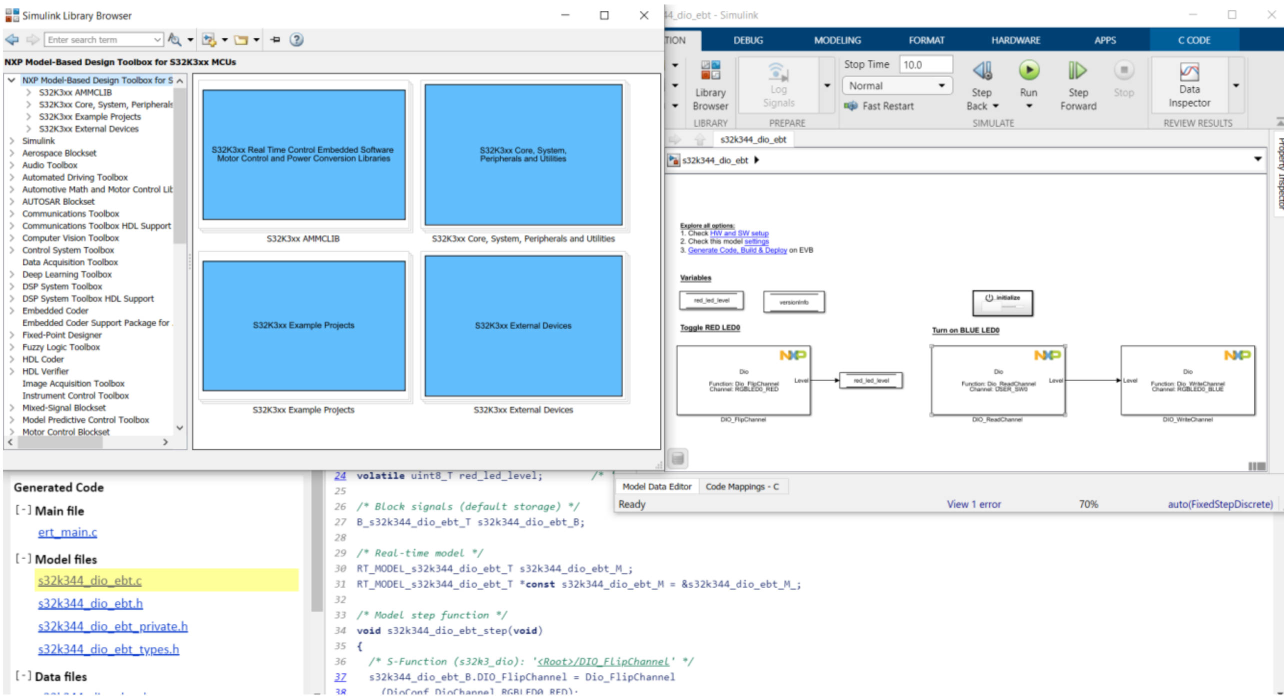
Target Based Automatic Code Generation Tools
For MATLAB™/Simulink™/Stateflow™ Models working with Simulink Coder ™ and Embedded Coder®

Summary

1 Main Features

The NXP's Model-Based Design Toolbox for S32K3xx Series version 1.2.0 is designed to support S32K312, S32K314, S32K322, S32K324, S32K341, S32K342 and S32K344 MCUs into MATLAB/Simulink environment, allowing users to:

- **Design** applications using Model-Based Design methodologies;
- **Simulate** and **Test** Simulink models for S32K3 MCU before deploying the models to the hardware targets;
- **Configure** the MCU peripherals from Simulink Blocks via external configuration tools: NXP S32 Configuration Tools or EB Tresos
- **Generate** the application code automatically without any needs for hand coding C/ASM
- **Deployment** of the application directly from MATLAB/Simulink to the NXP evaluation boards



The main features and functionalities supported by the RTM release v.1.2.0 are:

- Support for S32K312, S32K314, S32K322, S32K324, S32K341, S32K342 and S32K344 MCUs
- Integrated with S32Configuration Tool (v2021.R1.5) and EB Tresos (v27.1.0)
- Integrates the Real-Time Drivers (v2.0.0)
- Support for MC33775A, MC33772C BCC and MC33665 PHY
- Motor control examples (both for PMSM and BLDC)
- Compatible with MATLAB releases R2020a, R2020b, R2021a, R2021b, R2022a
- Fully integrated with Simulink Toolchain
- Includes extensive Simulink Library Blocks for S32K312, S32K314, S32K322, S32K324, S32K341, S32K342 and S32K344 MCUs, providing dedicated NXP blocks for peripheral components (e.g.: ADC, PWM, CAN, DIO, GPT, SPI, UART)
- Supports FreeMASTER v3.x
- Supports AMMCLib v1.1.29

- Supports AUTOSAR blockset (SW-C deployment)
- Supports download via JTAG and OpenSDA
- Includes an Example library with approximatively 130 examples that cover a wide range of topics like:
    - I/O control
    - Timers and scheduling
    - Communication
    - Motor control (3-shunt BLDC and PMSM)
    - Software-in-Loop, Processor-in-Loop, External mode
    - For more details about each of the topics highlighted above please refer to the following chapters.

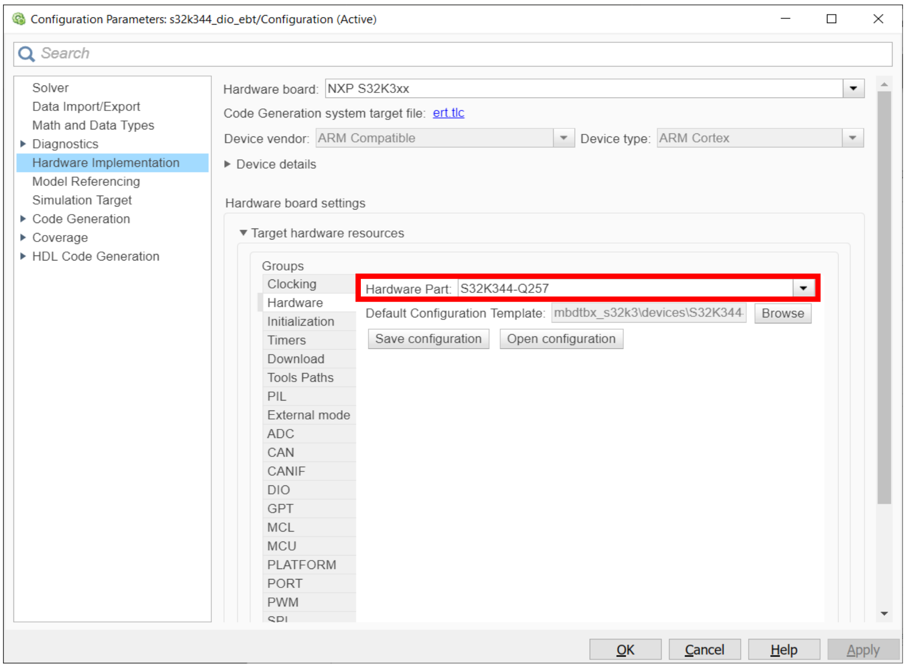
2 S32K3xx MCU Support

2.1 Packages & Derivatives

The NXP's Model-Based Design Toolbox for S32K3xx Series version 1.2.0 supports:

- S32K3xx MCU Packages:
    - S32K312-172MAXQFP
    - S32K314-172MAXQFP
    - S32K322-172MAXQFP
    - S32K324-172MAXQFP
    - S32K341-172MAXQFP
    - S32K342-172MAXQFP
    - S32K344-257MAPBGA
    - S32K344-172MAXQFP

The configurations can be easily changed for each Simulink model from the **Model Configuration Parameters** menu:

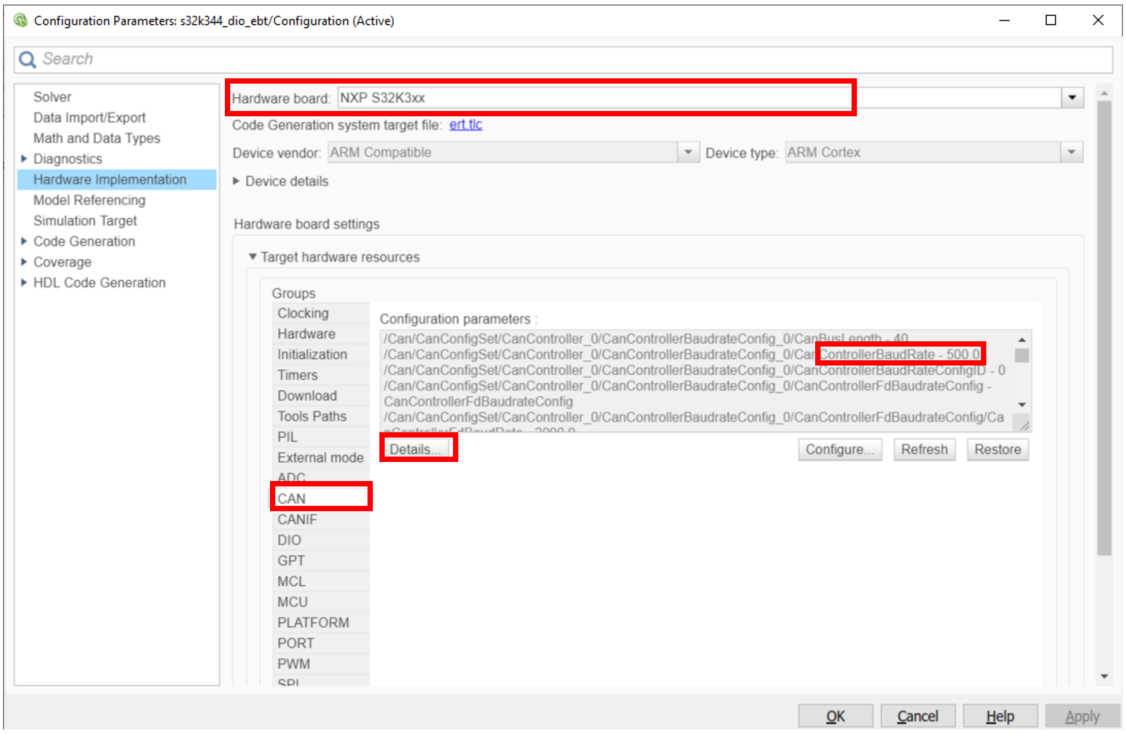


2.2 Peripherals and functions

The [NXP's Model-Based Design Toolbox for S32K3xx Series version 1.2.0] supports the following peripheral components and functions:

- ADC
- PWM
- DIO
- MCL
- CAN
- SPI
- UART
- GPT
- Memory read/write
- Register read/write
- Profiler

The default configuration supported by the toolbox for each peripheral is available inside the Target Hardware Resources panels:



From this panel, the user can access the hardware documentation (Reference Manual) using the **Details**… button. To update the model configuration, the user can click on the **Configure…** button to open the NXP S32 Configuration Tool / EB Tresos. The selected external configuration tool will be opened and the user will have access to the entire parameter list for pins / clock / peripherals.

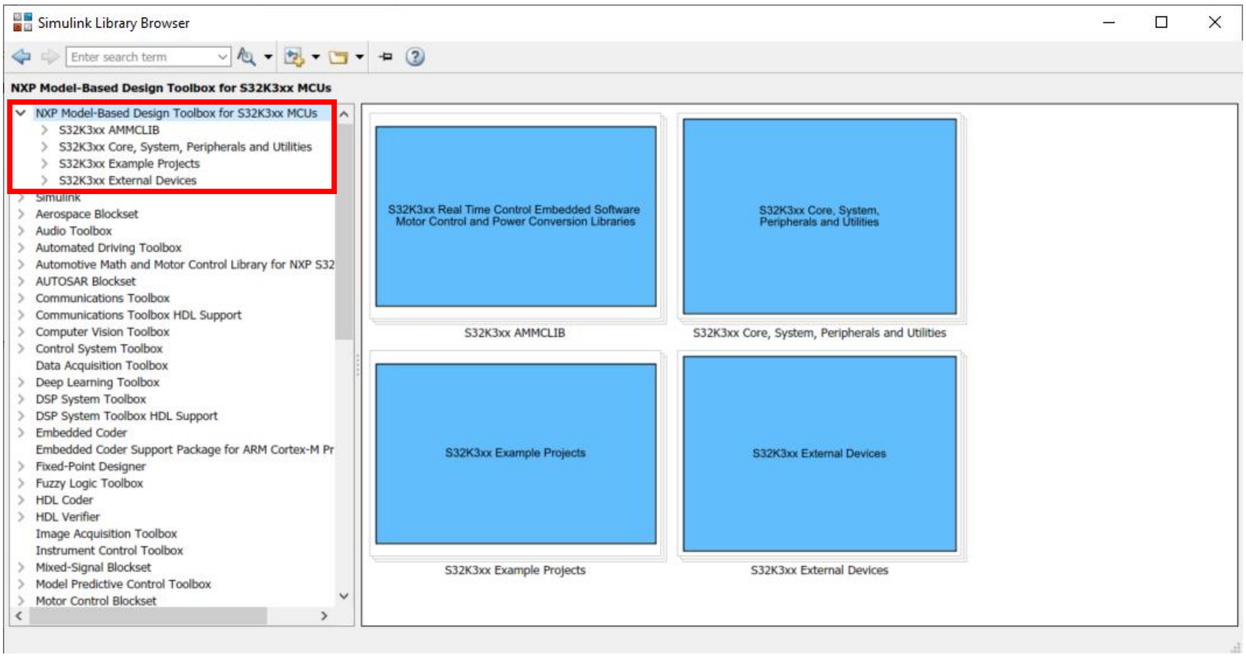
The [NXP's Model-Based Design Toolbox for S32K3xx Series version 1.2.0] has been tested using the official NXP Evaluation Boards for S32K3xx – S32K3X4EVB-Q257, XS32K3XXEVB-Q172, S32K312EVB-Q172 and XS32K3X2CVB-Q172.

3 Model-Based Design Toolbox Features

The NXP's Model-Based Design Toolbox for S32K3xx Series version 1.2.0 is delivered with complete S32K3xx MCUs Simulink Block Library as shown below.

There are three main categories:

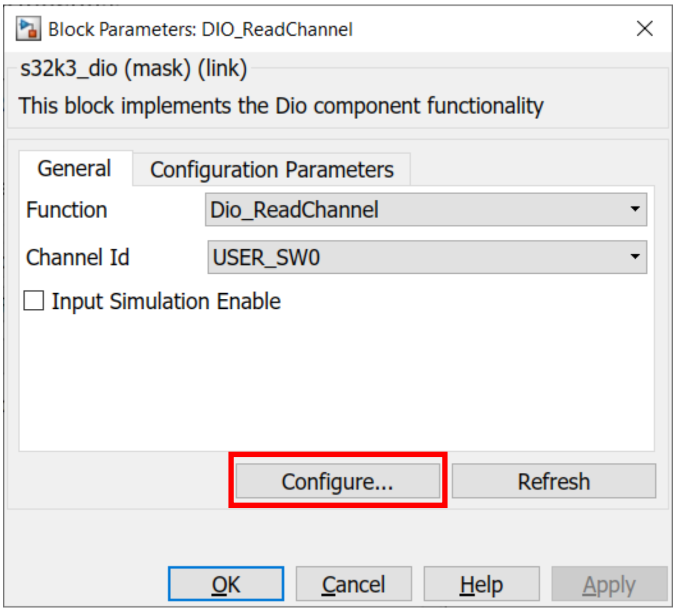
- S32K3xx AMMCLib
- S32K3xx Core, System, Peripherals and Utilities
- S32K3xx Example Projects
- S32K3xx External Devices



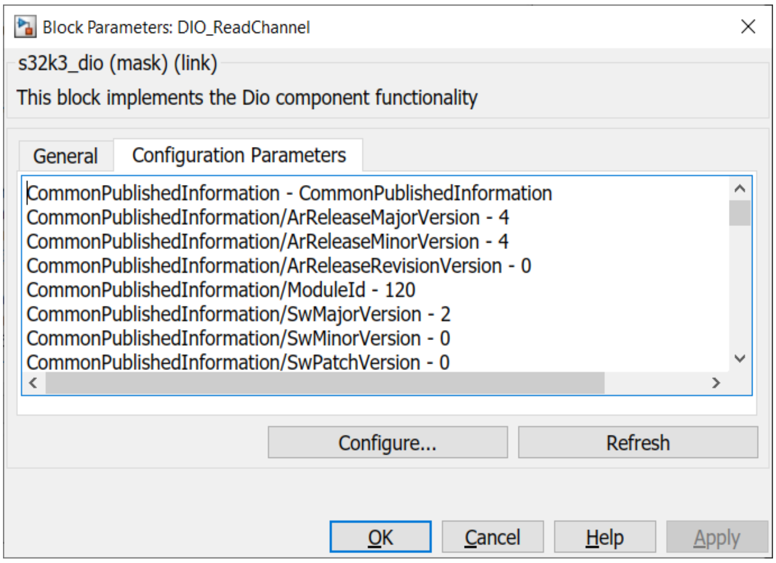
3.1 S32K3xx Simulink Library Blocks

All Simulink blocks supported by the NXP Toolbox are designed to offer the best out-of-the-box experience, providing a basic peripheral configuration that covers most of the hardware capabilities. Most of the Simulink Blocks contains just two tabs:

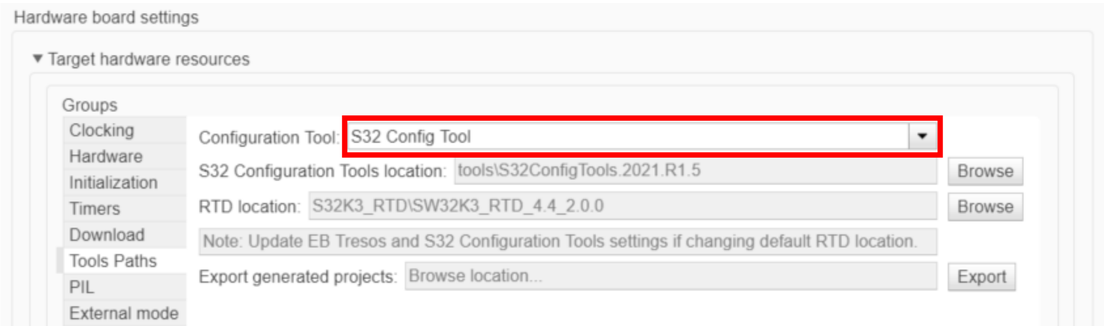
1. **General** Tab allows the user to select what functionality he wants the block to enable (usually this will be a list of MCAL functions). Besides that, the user can edit some parameters (this will be different, depending on the function selection). All peripheral components come with a working configuration (the toolbox default setup). These settings have been chosen to work out-of-the-box for the supported EVBs. The toolbox can be used as-is if no configuration parameters have to be changed. This is the BASIC mode of operation for the toolbox. In the ADVANCED mode, the user can click on the **Configure…** button, which will open an external configuration tool (NXP S32Configuration Tool / EB Tresos). There, the user has full access to all the parameters needed to set up the pins/clock/peripherals.

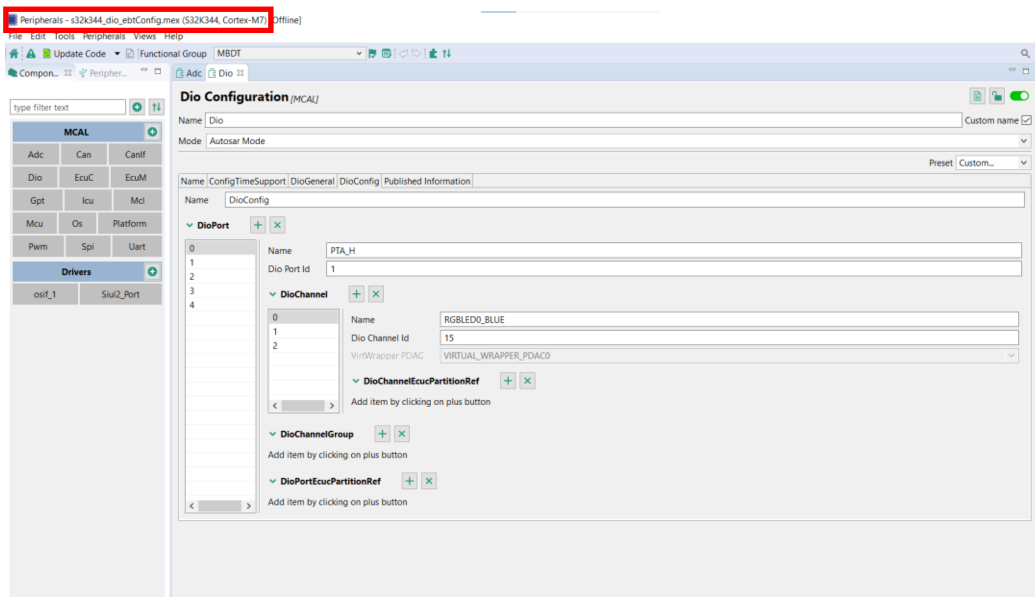


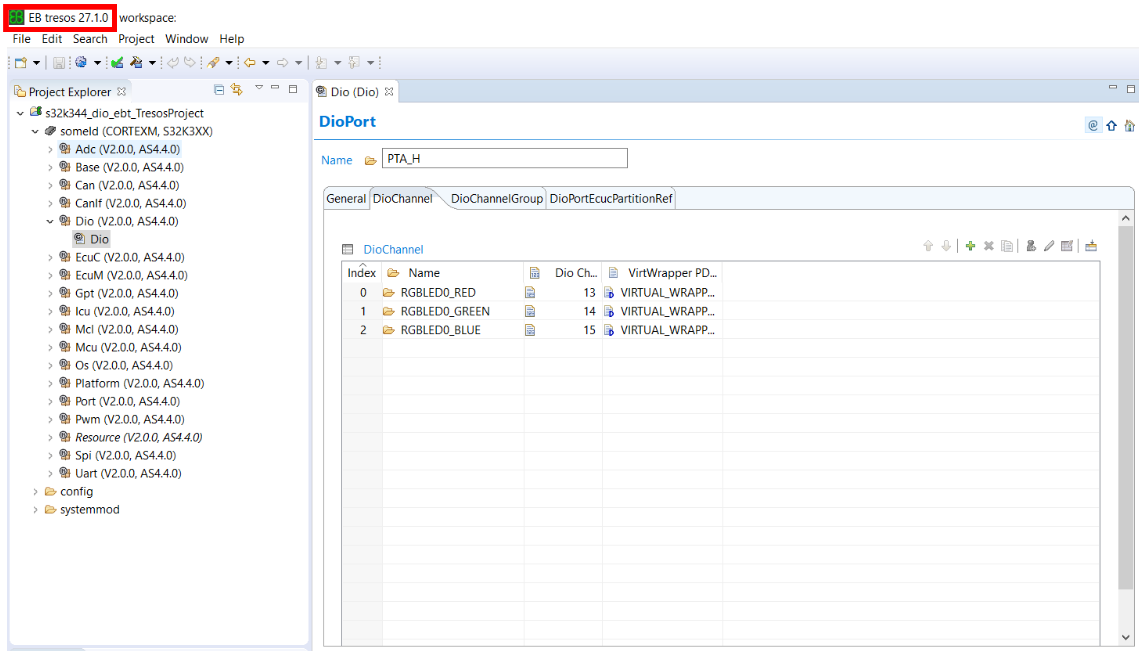
2. **Parameters** Tab which contains the detailed configuration available. This information is shown for verification purposes



From any of these blocks, by clicking on the **Configure…** button, the users can open the external configuration tool of choice (NXP S32Configuration Tools / EB Tresos) to alter the default configuration used by the Simulink model.



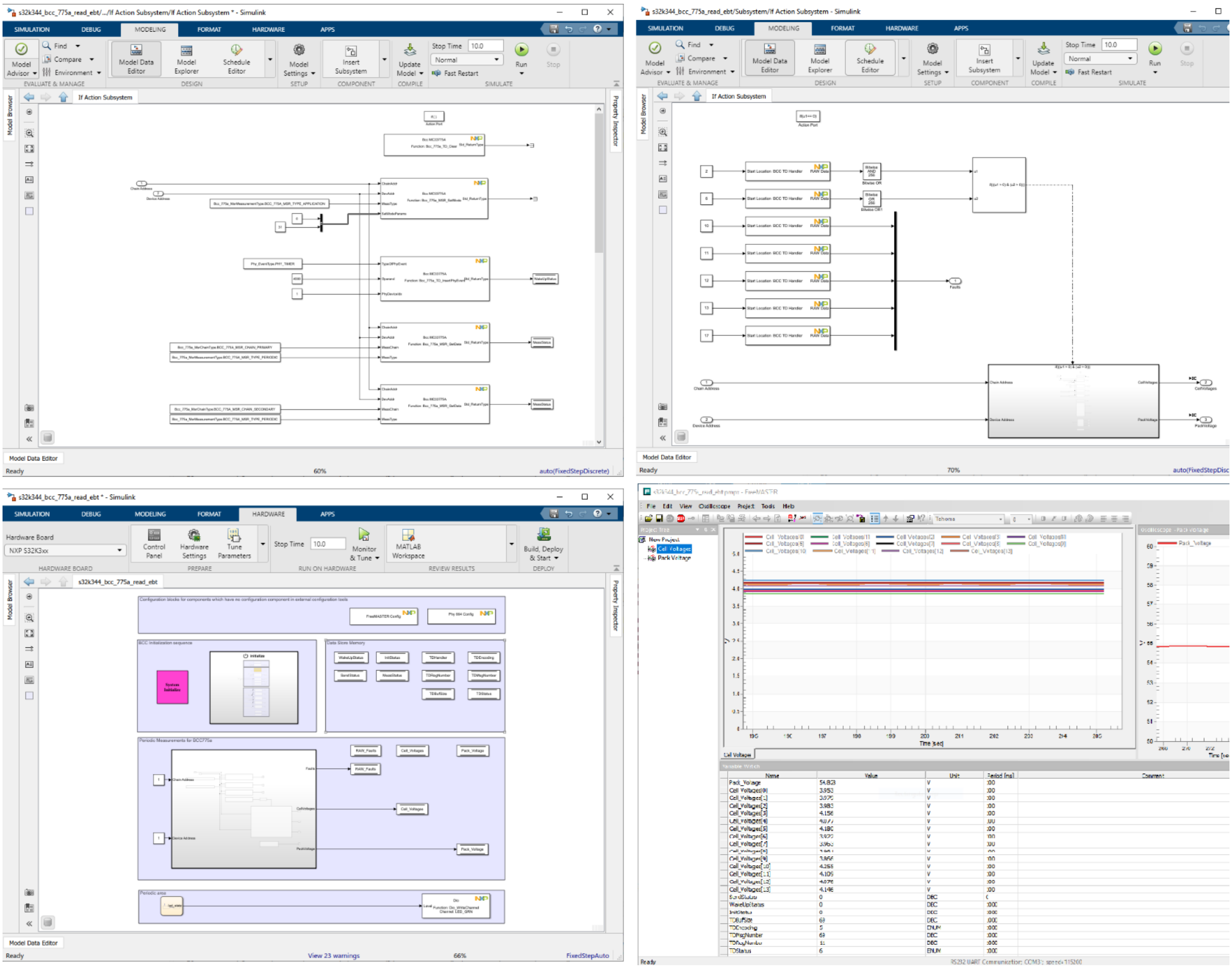




Using the external tools, the users can perform ADVANCED configurations and then use these new options in Simulink models. The validation of the configuration and peripheral code generation is done outside of Simulink.

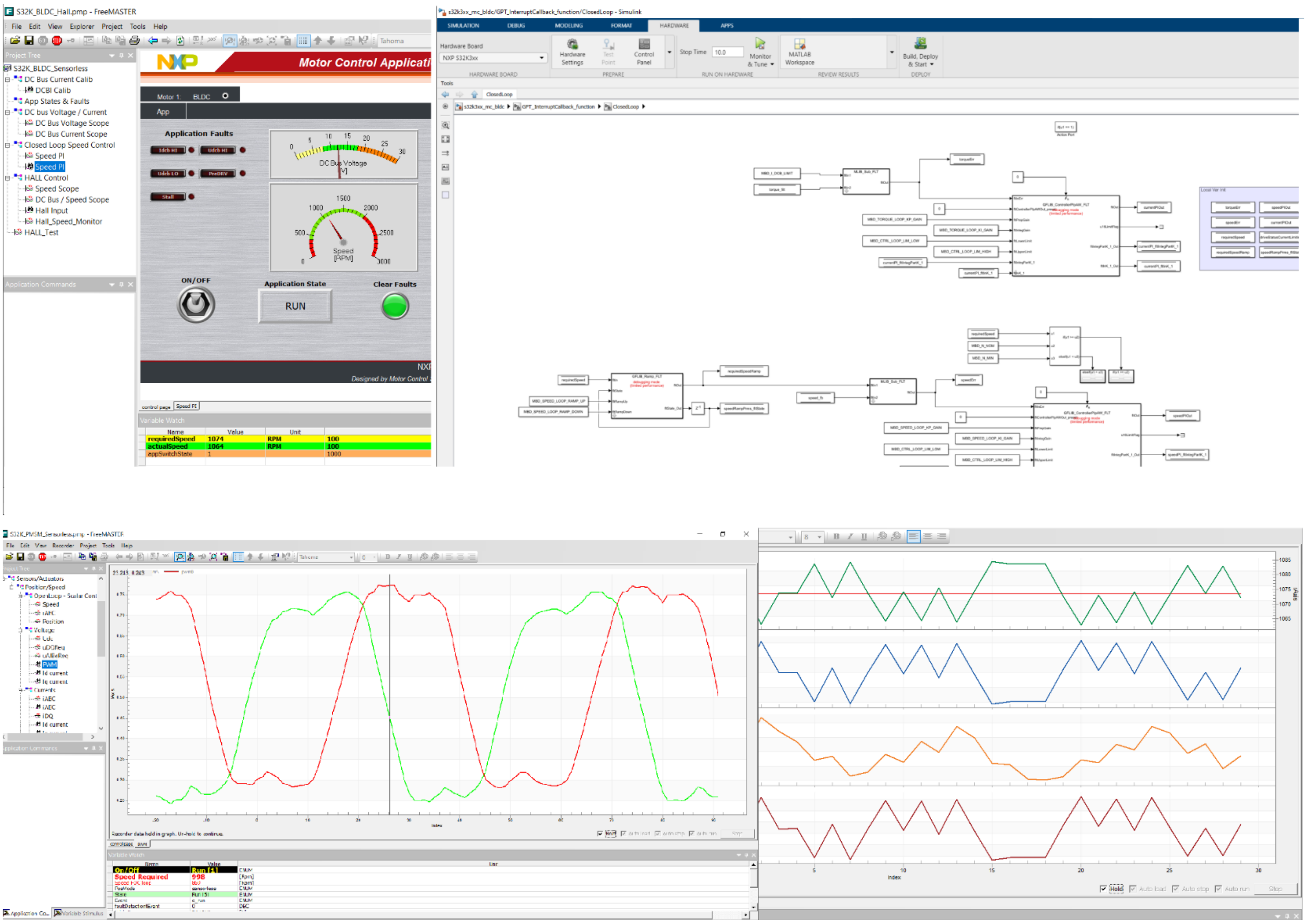
3.2 Support for battery cell controller for BMS applications (MC33775A, MC33772C & MC33665PHY)

The toolbox provides support for the MC33775A, MC33772C and MC33665. The MC33775A and MC33772C are lithium-ion battery cell controller ICs designed for automotive applications which perform ADC conversions of the differential cell voltages and battery temperatures, while the MC33665 is a transceiver physical layer transformer driver, designed to interface the microcontroller with the battery cell controllers through a high-speed isolated communication network. The ready-to-run examples provided with the MBDT for S32K3 show how to communicate between the S32K344 and the MC33775A and MC33772C via the MC33665 transceiver. For the MC33775A, the examples show how to configure the battery cell controller to perform Primary and Secondary chains conversion, read the cell voltages conversion results from the MC33775A, while for the MC33772C the examples show how to configure the Battery cell controller to read current. All the converted values are displayed to the user over the FreeMaster application.



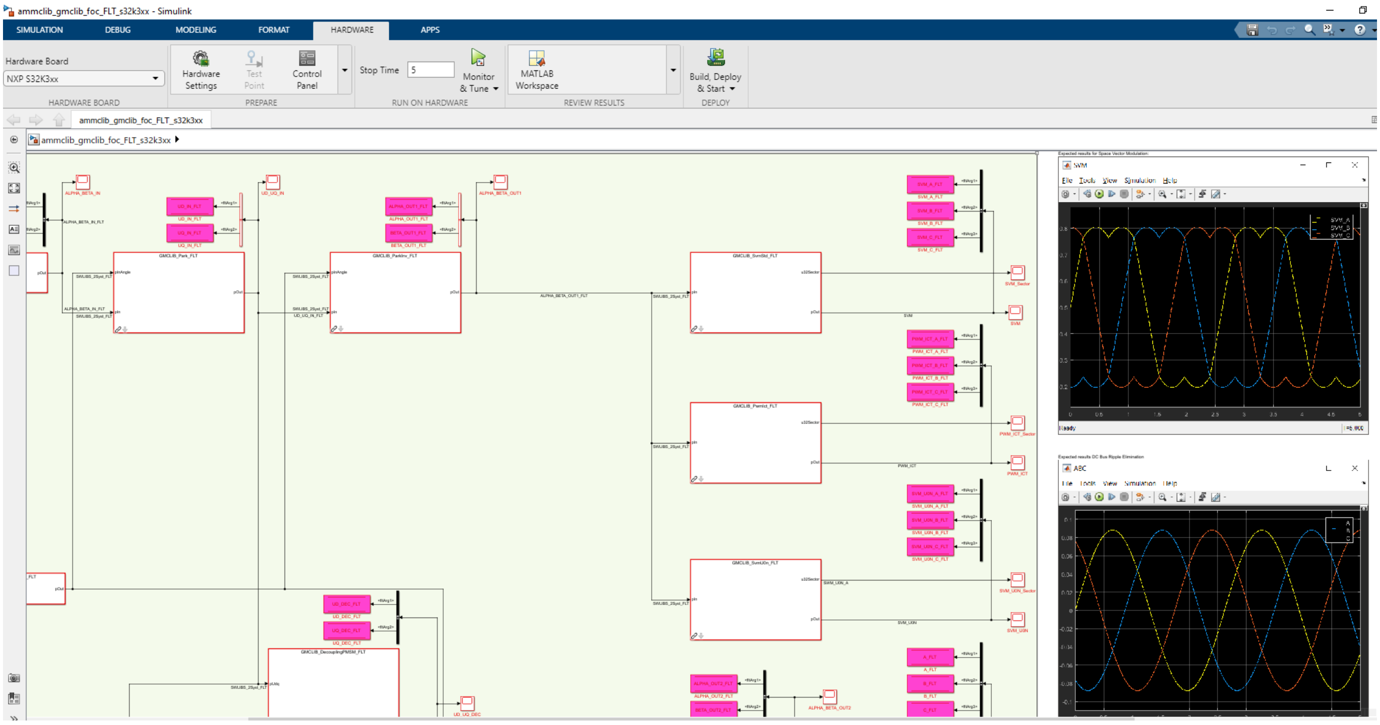
3.3 Motor control examples

The toolbox provides examples for both 3-shunt PMSM and BLDC motor control applications. Each of them has a detailed description of the hardware setup and an associated FreeMASTER project which can be used for control and data visualization.



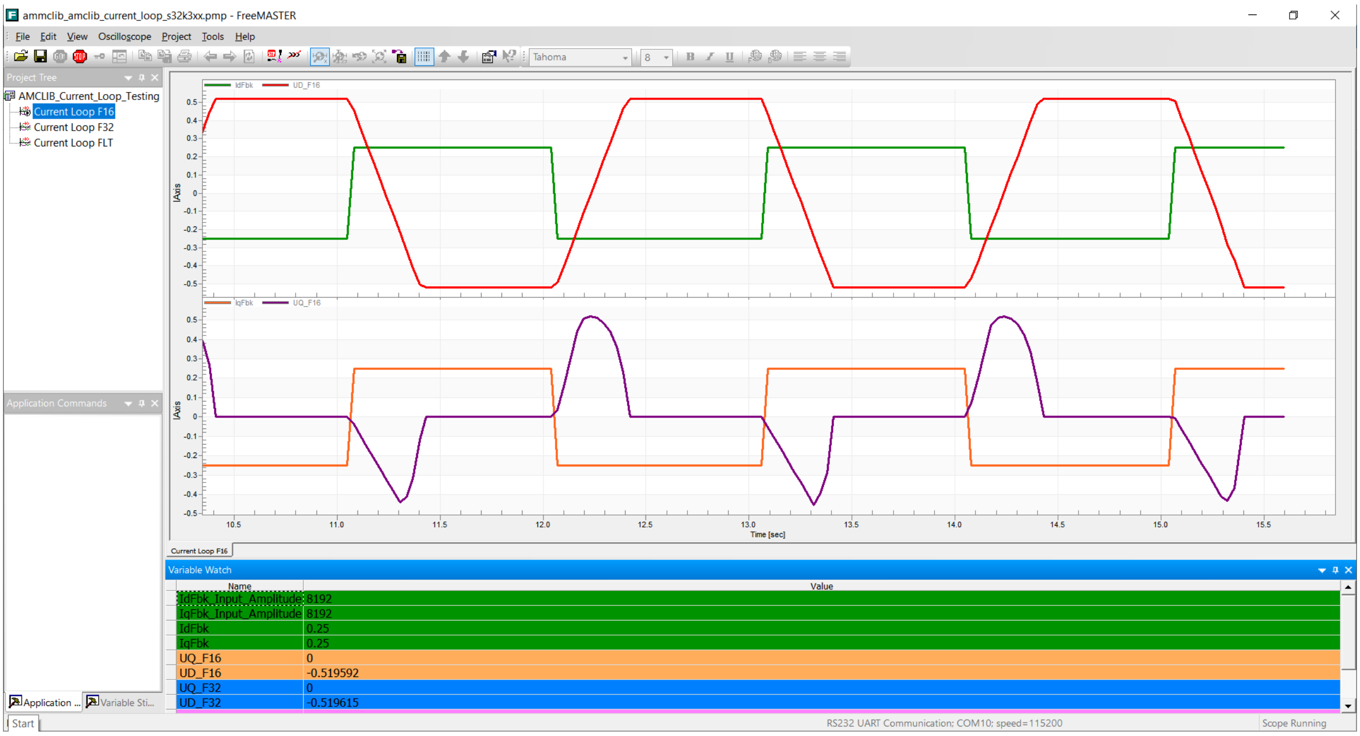
3.4 S32K3xx AMMCLib integration

This toolbox also ships the AMMCLib v1.1.29 blocks. These blocks are essential for building applications with high-performance arithmetic, trigonometric, digital signal processing and math functions.



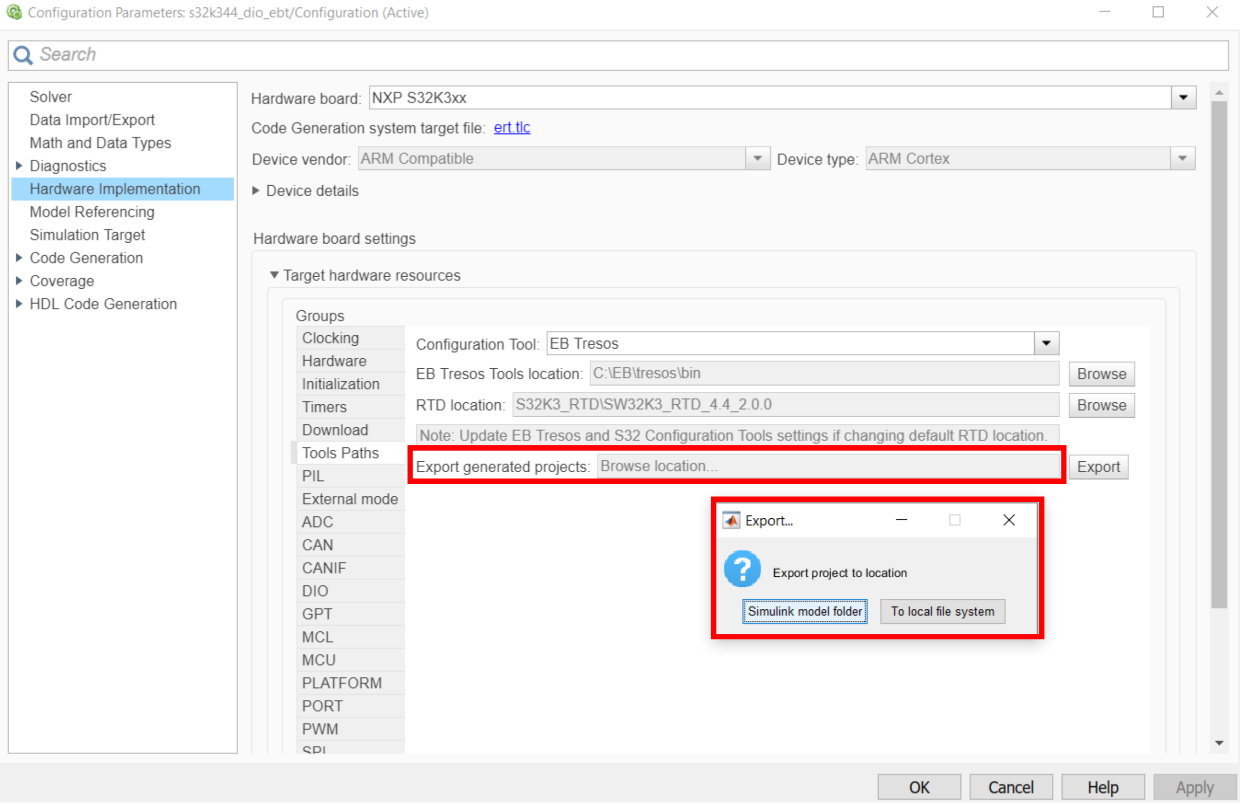
3.5 S32K3xx FreeMASTER integration

This toolbox has support for FreeMASTER 3.x – this is a user-friendly real-time debug monitor and data visualization tool that enables runtime configuration and tuning of embedded software applications. With this support, users can build models and monitor various variables, display them on multiple scopes and even add widgets (gauges, sliders, etc).



3.6 S32K3xx Design Studio integration

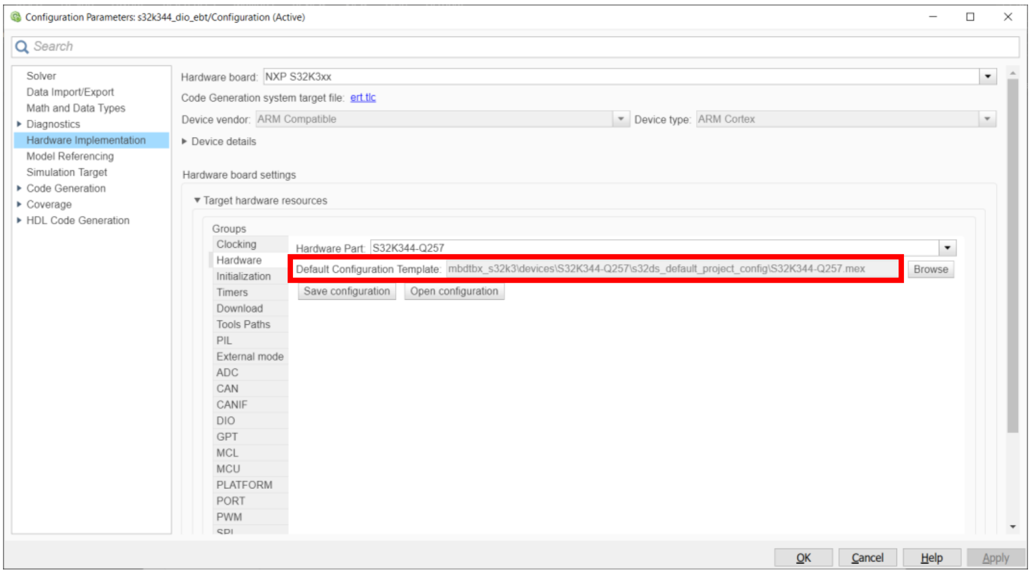
The toolbox provides users the opportunity to export the code generated from Simulink (using the S32K3xx toolbox) and easily import it into S32Design Studio. This functionality can be useful if the model is to be integrated into an already existing, larger project.



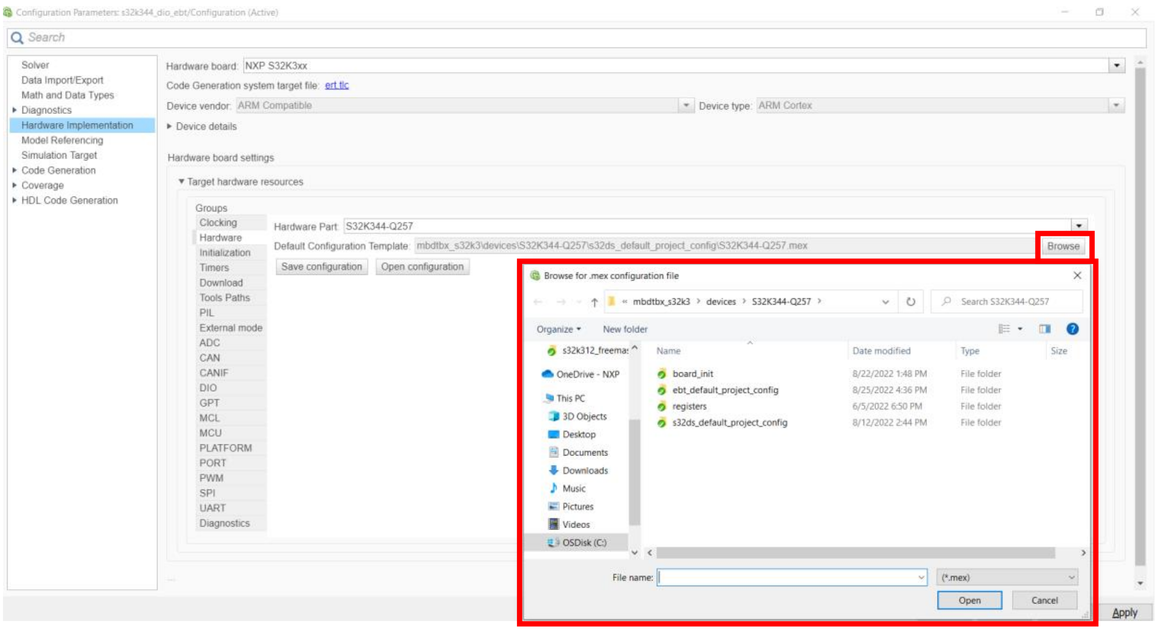
3.7 Support for custom default project

The toolbox provides support for users to create their own custom default project. This could be very useful when having a custom board design – the configuration for it needing to be created only once. After that configuration is saved as a custom default project, it can be used for every model that is being developed.

For the applications delivered with the toolbox and the new created models, the Default Configuration Template will be set automatically to the location of the default project the Model-Based Design Toolbox provides, according to the selected Hardware Part and Configuration Tool.

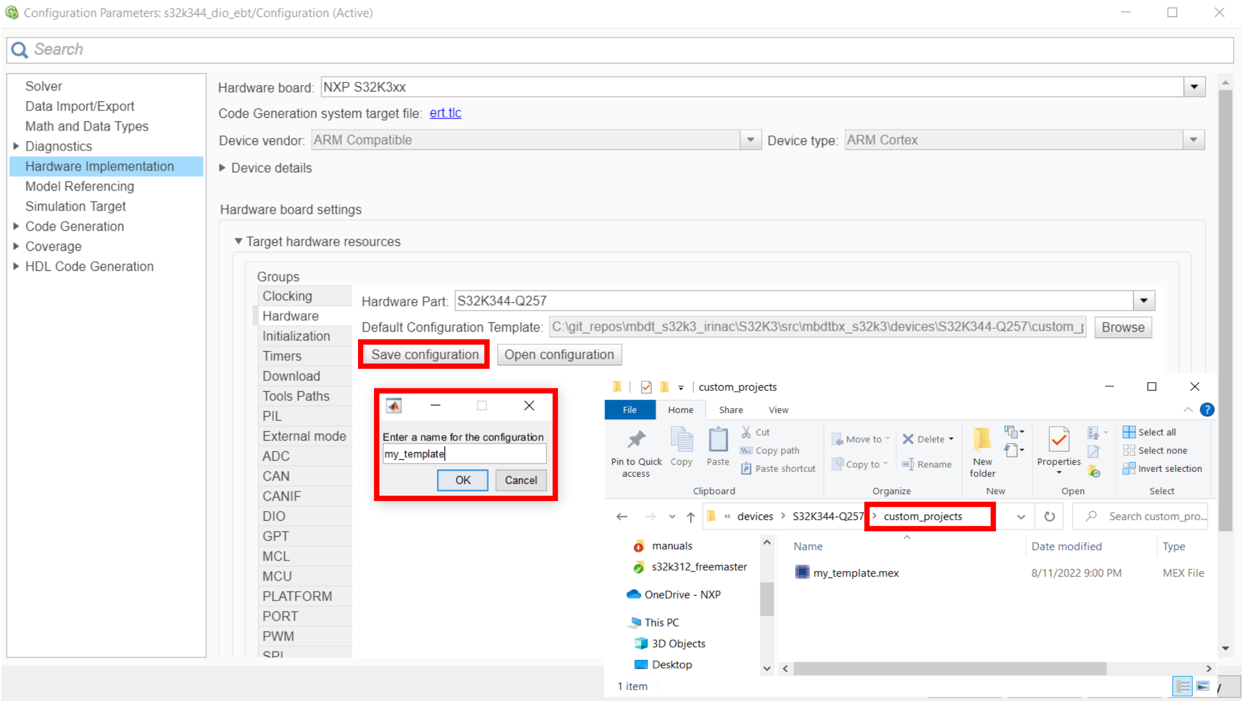


The **Browse** button will open the Windows Explorer allowing the selection of a different project to be used as the default one for the model.

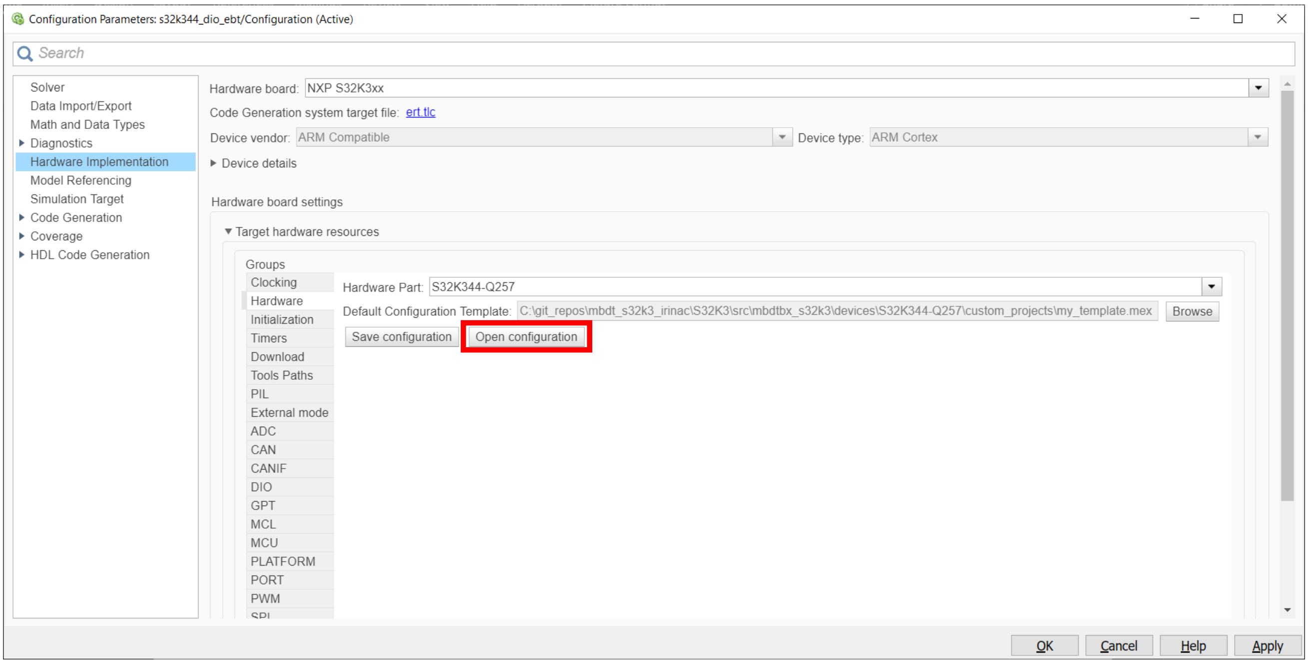


The **Save configuration** button allows storing the new selected template next to the already existing projects, under a name at the user's choice. The project will be stored inside the

custom_projects folder, created inside the location corresponding to the selected Hardware Part.

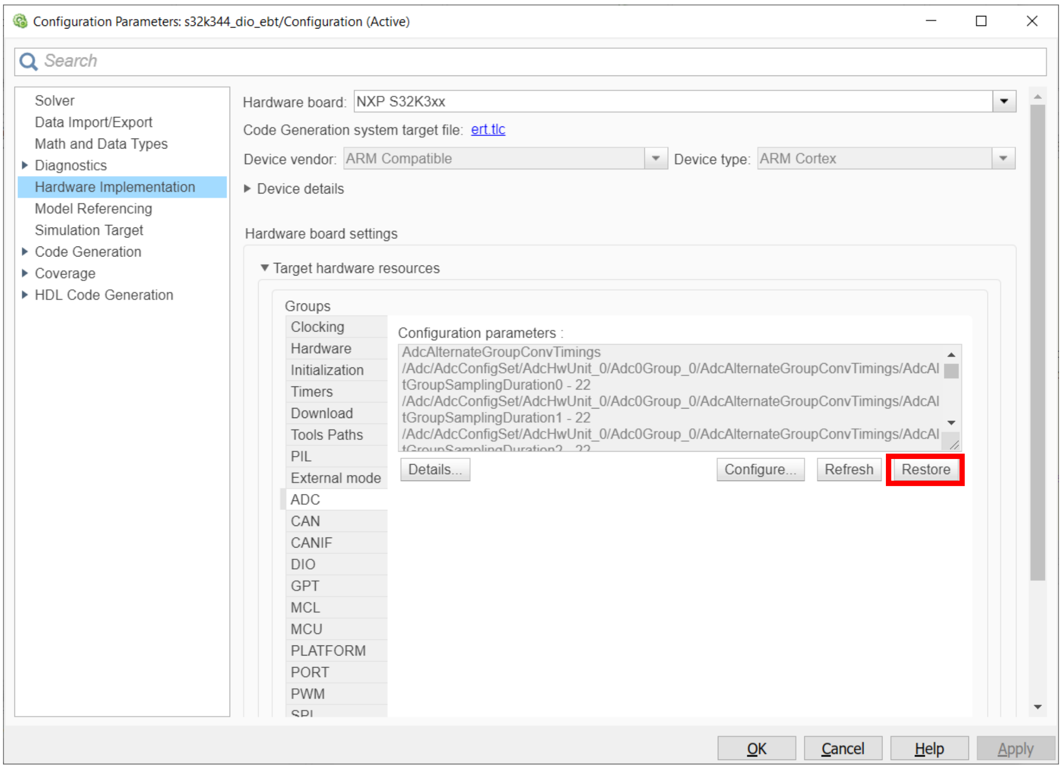


The **Open configuration** button allows opening the Default Configuration Template in the selected Configuration Tool, where it can be modified and saved.



3.8 Restore components configuration to default settings

The toolbox allows to **Restore** the configuration of a component (for models which use the EB Tresos configuration tool) to the settings corresponding to the Default Configuration Template the model uses. This allows to revert modifications (if made) to the default values.



3.9 S32K3xx Simulation modes

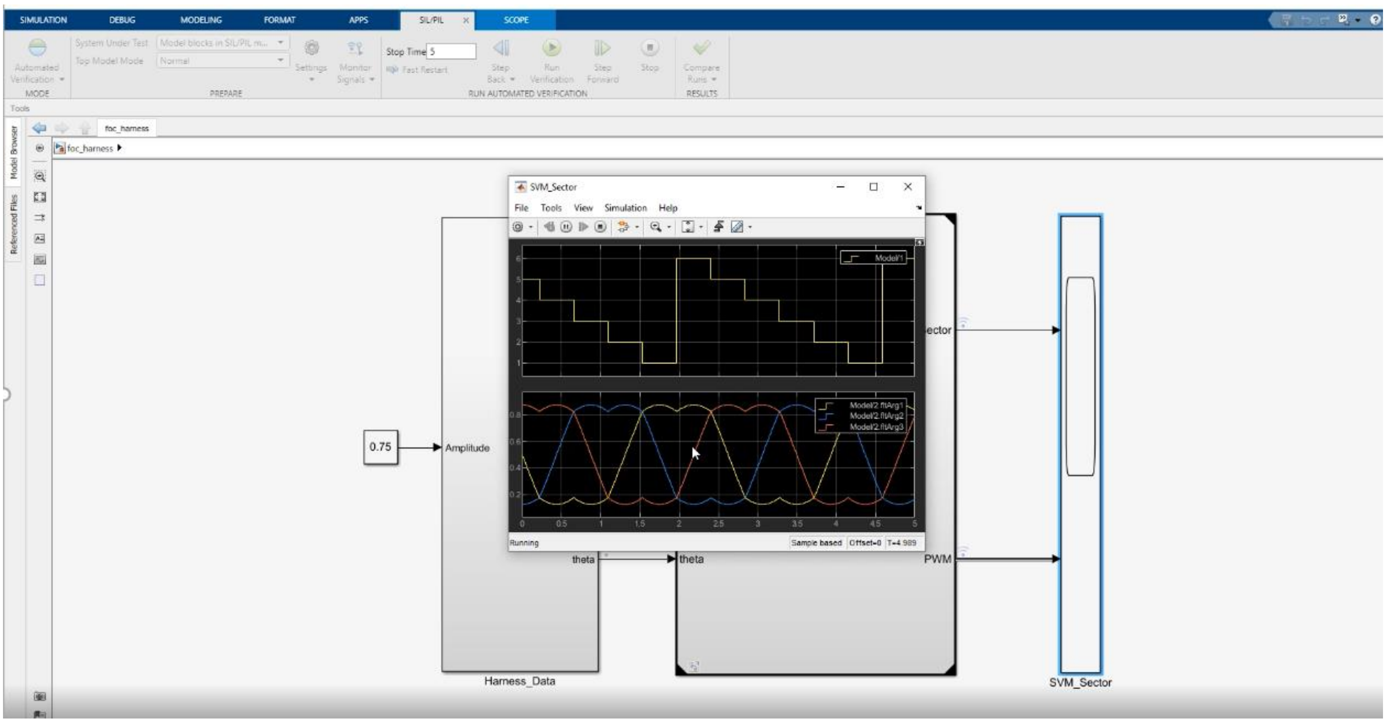
The toolbox provides support for the following Simulation modes:

- Software-in-Loop (SIL)
- Processor-in-Loop (PIL)
- External Mode

Software-in-Loop
A SIL simulation compiles and runs the generated code on the user's development computer. One can use such a simulation to detect early defects and fix them.

Processor-in-loop
In a PIL simulation, the generated code runs on the target hardware. The results of the PIL simulation are transferred to Simulink to verify the numerical equivalence of the simulation and the code generation results. The PIL verification process is a crucial part of the design cycle to ensure that the behavior of the deployment code matches the design.

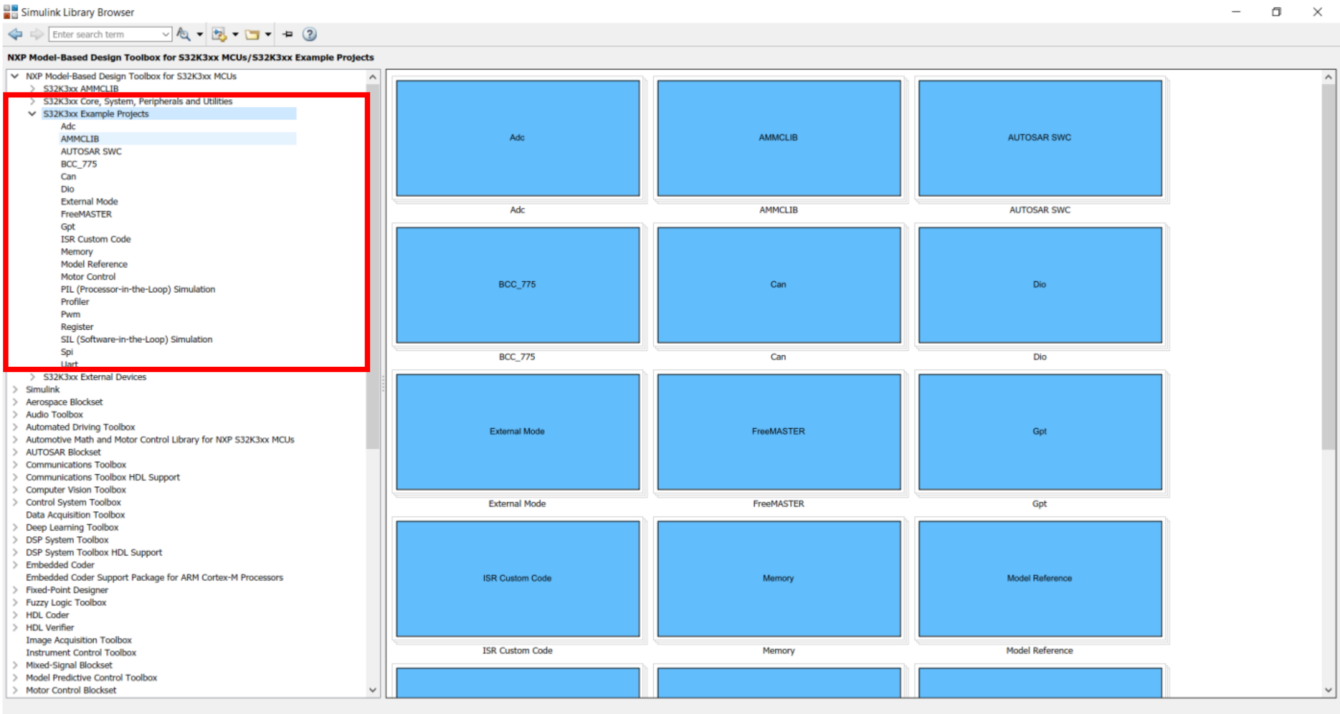
External mode
In this mode, the user can leverage Simulink to link algorithm code with the I/O driver code generated from the I/O blocks (from MATLAB). The resulting executable runs on the development computer and exchanges parameter data with Simulink via a shared memory interface.



3.10 S32K3xx Example Library

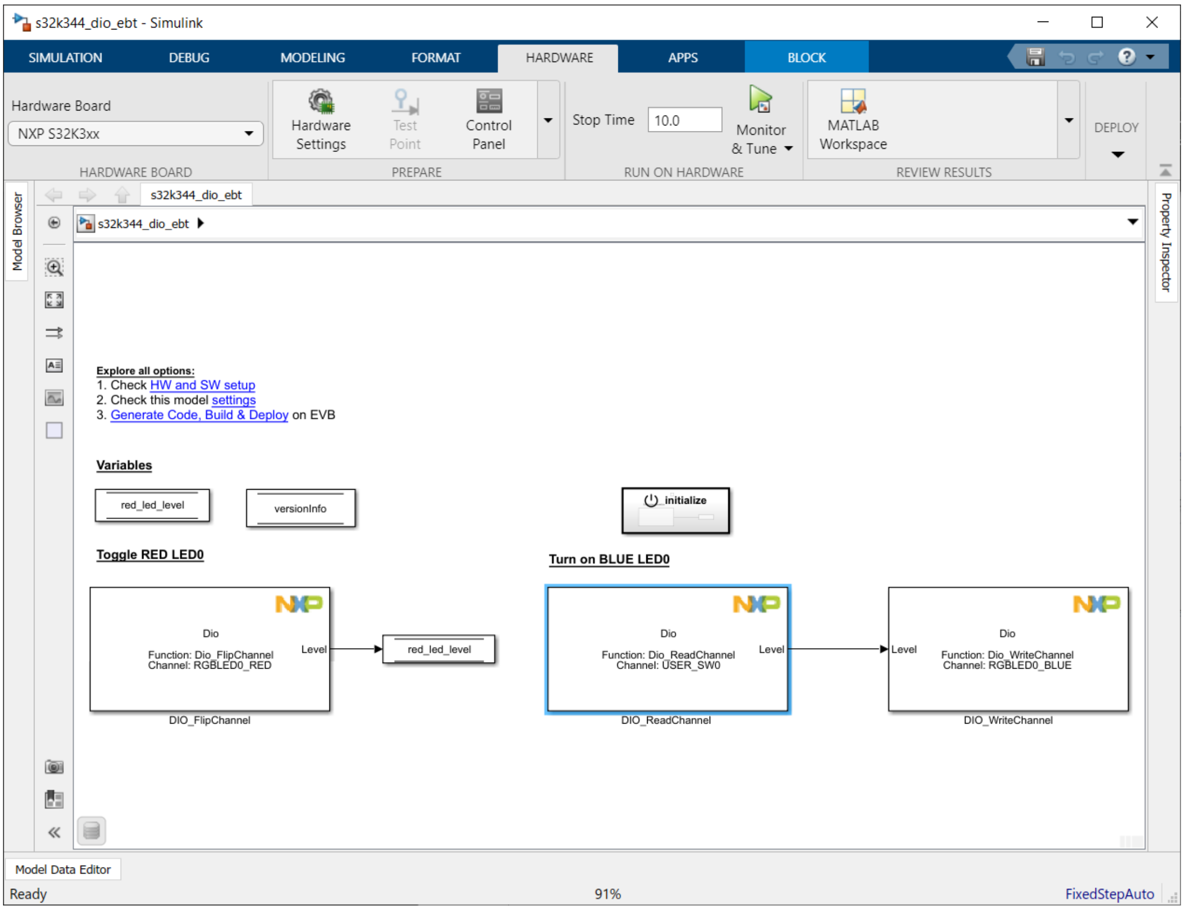
The Examples Library represents a collection of Simulink models that let you test different MCU on-chip modules and run complex applications. The toolbox provides examples targeting both S32 Configuration Tools and EB Tresos, and the supported S32K3 hardware derivatives.

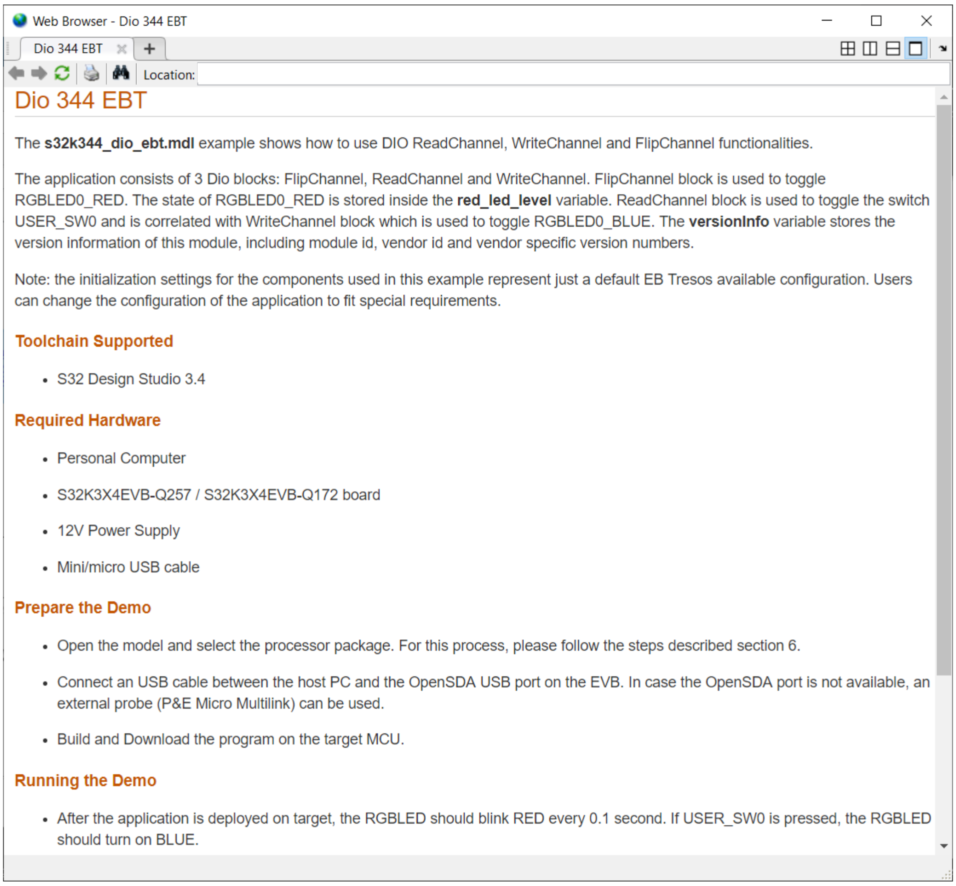
The examples are grouped in different layers that mimic a typical development flow: starting with basic building blocks that expose the MCU HW functionalities up to more complex applications that incorporate multiple building blocks.



The Simulink models shown as examples are enhanced with a comprehensive description to help users understand better the functionality that is exercised, hardware setup instructions whenever are necessary, and a result validation section.

The examples are also available from the MATLAB help page.





Web Browser - Dio 344 EBT

Dio 344 EBT

Location:

Dio 344 EBT

The **s32k344_dio_ebt.mdl** example shows how to use DIO ReadChannel, WriteChannel and FlipChannel functionalities.

The application consists of 3 Dio blocks: FlipChannel, ReadChannel and WriteChannel. FlipChannel block is used to toggle RGBLED0_RED. The state of RGBLED0_RED is stored inside the **red_led_level** variable. ReadChannel block is used to toggle the switch USER_SW0 and is correlated with WriteChannel block which is used to toggle RGBLED0_BLUE. The **versionInfo** variable stores the version information of this module, including module id, vendor id and vendor specific version numbers.

Note: the initialization settings for the components used in this example represent just a default EB Tresos available configuration. Users can change the configuration of the application to fit special requirements.

Toolchain Supported

- S32 Design Studio 3.4

Required Hardware

- Personal Computer
- S32K3X4EVB-Q257 / S32K3X4EVB-Q172 board
- 12V Power Supply
- Mini/micro USB cable

Prepare the Demo

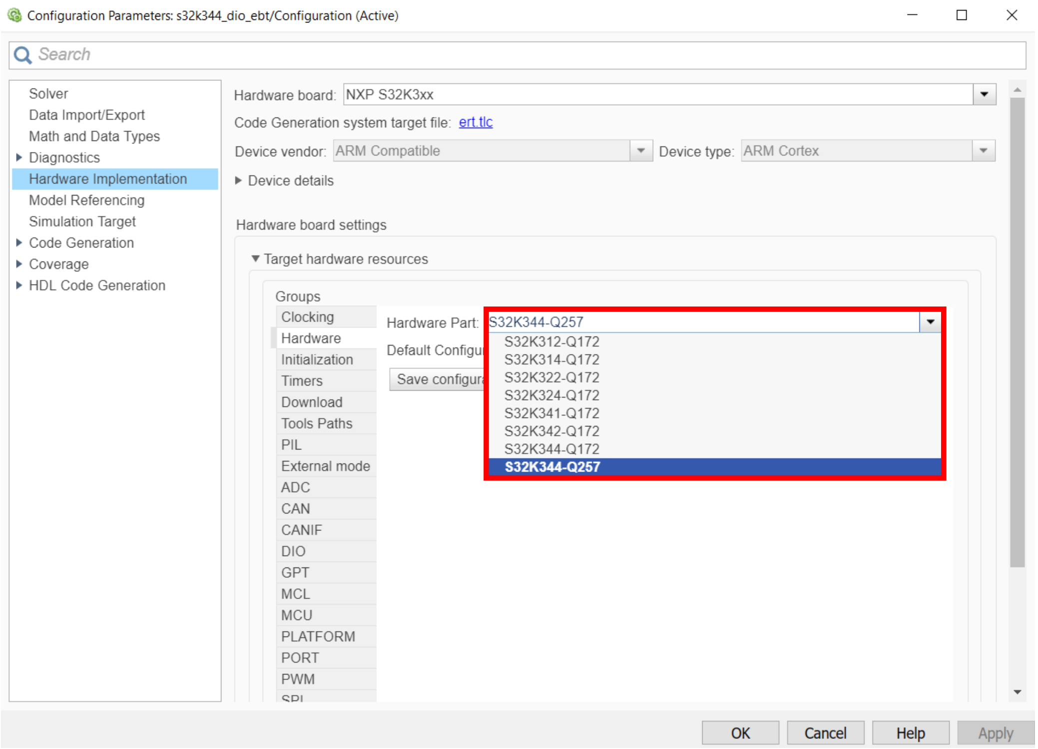
- Open the model and select the processor package. For this process, please follow the steps described section 6.
- Connect an USB cable between the host PC and the OpenSDA USB port on the EVB. In case the OpenSDA port is not available, an external probe (P&E Micro Multilink) can be used.
- Build and Download the program on the target MCU.

Running the Demo

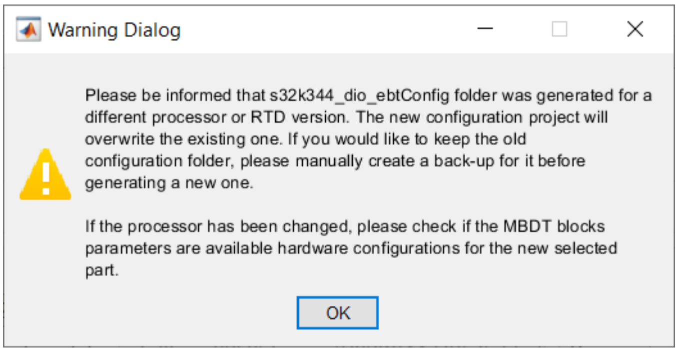
- After the application is deployed on target, the RGBLED should blink RED every 0.1 second. If USER_SW0 is pressed, the RGBLED should turn on BLUE.

3.11 Change the Hardware Part

In case you would like to change the application to target a different processor, you should go to Hardware Settings and set the corresponding hardware from the **Hardware Part** list, under the **Target hardware resources** pane.



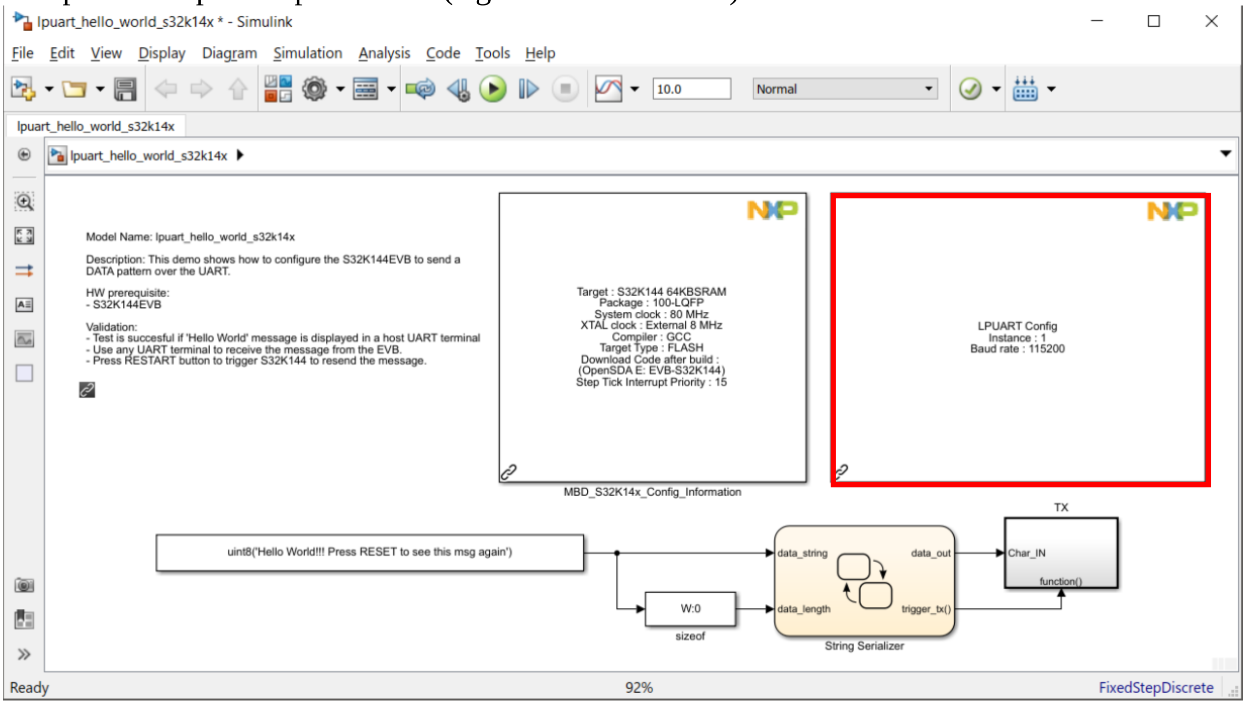
After selecting the new processor from the list, click Apply and a popup appears informing on the change of the configuration project, to fit the new selected processor. After clicking ok, the model can be built again and deployed on the new target.



4 Workflow and framework changes

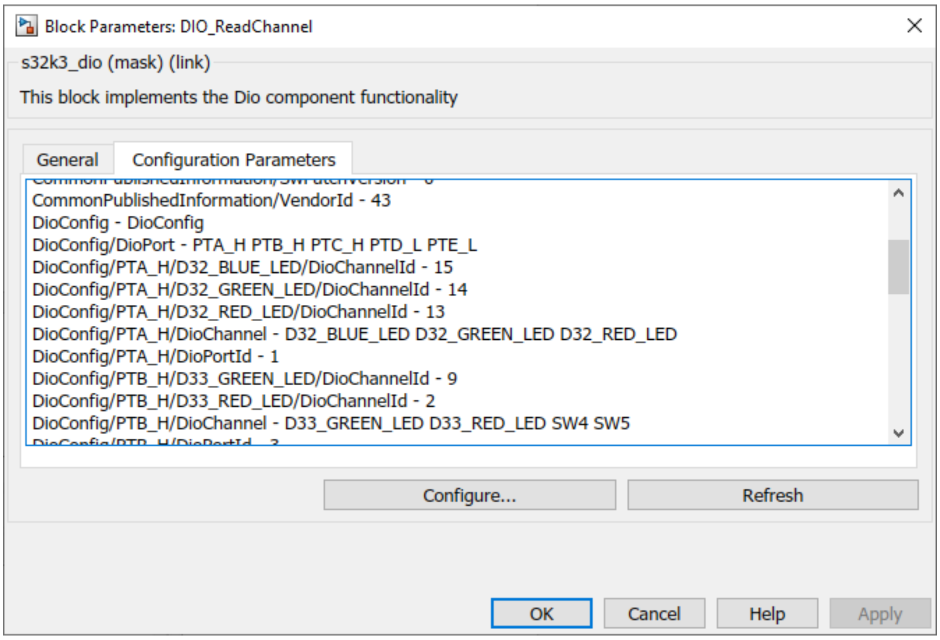
4.1 Using external configuration tools

One of the major differences between older toolboxes (e.g. for S32K1xx) and MBDT for S32K3xx is the introduction of a new workflow, where the modeling and configuration are split. In other toolboxes, every peripheral had associated a Configuration Block. There, the user could set up some important parameters (e.g.: UART baud rate):



4.1.1 Modes of operation – BASIC and ADVANCED

Starting with this release, all peripheral Configuration Blocks are no longer needed. Instead, we provide a default configuration for all the peripherals we support. Users can see the default parameters for every such block, going in the mask, searching for the 'Configuration Parameters' tab. This use case is what we call the BASIC mode of operation – where there is no additional configuration to be made (everything is based on the default one), so no interaction with an external configuration tool is needed.



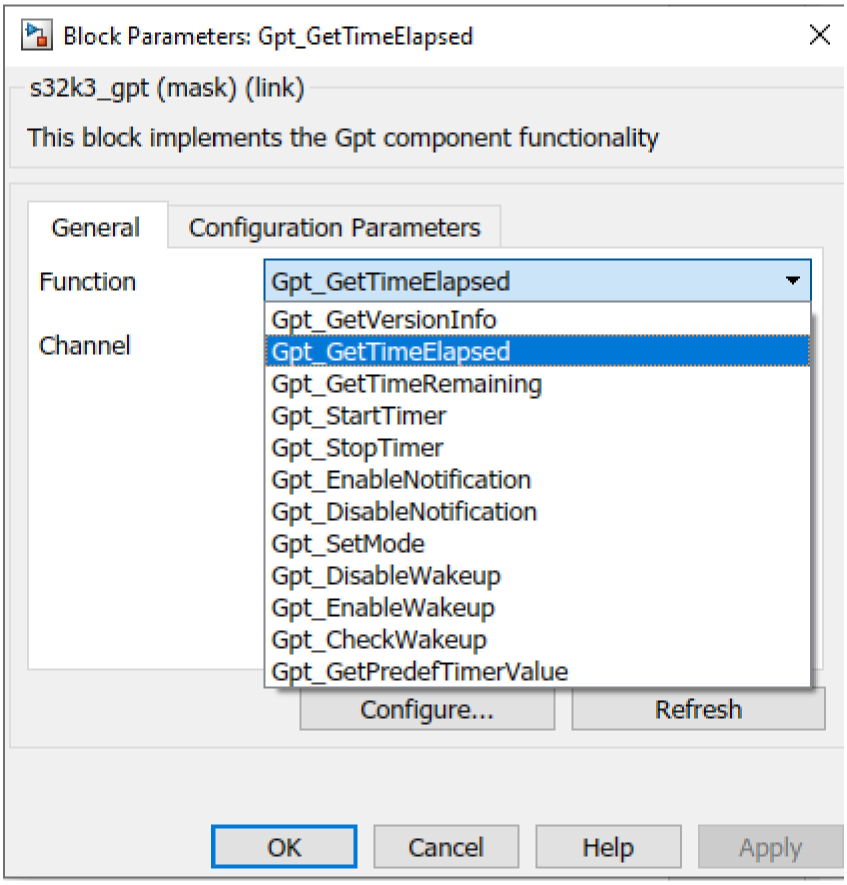
If something needs to be edited/added/removed, users can open an external configuration tool of choice (NXP S32Configuration Tools or EB Tresos). There, all the parameters of peripherals, pins and clock are available. The Simulink model is linked to the external configuration tools and is aware of changes made, so simply building the model one more time/clicking on the 'Refresh' button of the block fetches the new values of the parameters. This is the ADVANCED mode, where changes to the default configuration need to be made. Note that NXP S32Configuration Tools is shipped with our toolbox, so no extra installation process needed; EB Tresos needs to be installed and licensed (free of charge), following the steps in the MBDT Quick Start Guide.

4.1.2 Advantages for using external configuration tools

There are a couple of key advantages to using an external configuration tool instead of the peripheral configuration blocks. Firstly, using these dedicated tools, the user has access to all parameters for the peripherals. Additionally, full access to pins and clock configuration is provided (while the older framework has limited support for this). Secondly, splitting the modeling (algorithms, state machines, etc) from the configuration means they can be done in parallel. This can be an advantage when applications are developed in larger teams (one engineer might work on the configuration for all pins, clock, peripherals while another engineer can focus on modeling a complex algorithm in Simulink). Additionally, different configuration setups can be easily switched, without having to change anything inside the Simulink model.

4.2 Code generation based on RTD support (MCAL drivers)

In this release, the code generation is based on the Real Time Drivers. The peripheral support is different from other releases (e.g. for S32K1xx), not having a *per-peripheral* strategy, but a *per MCAL component/driver* one. For example, LPIT (low power interrupt timer) and STM (system timer module) support will not be done via separate blocks, but within one GPT (general purpose timer) block. The structure will be similar for all such blocks: a list of supported functions (MCAL standard functions and additional functions, added for optimization purposes) and a few parameters (depending on selected functionality).



5 Prerequisites

5.1 MATLAB Releases and OSes Supported

This toolbox is developed and tested to supports the following MATLAB releases:

- R2020a;
- R2020b;
- R2021a;
- R2021b;
- R2022a;

For a flowless development experience the minimum recommended PC platform is:

- *Windows® OS*: any x64 processor
- At least 4 GB of RAM
- At least 6 GB of free disk space.
- Internet connectivity for web downloads.

Operating System Supported

	SP Level	64-bit
Windows 7	SP1	X
Windows 10		X

5.2 Build Toolchain Support

The following compilers are supported:

Compiler Supported	Release Version
GCC for ARM Embedded Processors	V10.2.0

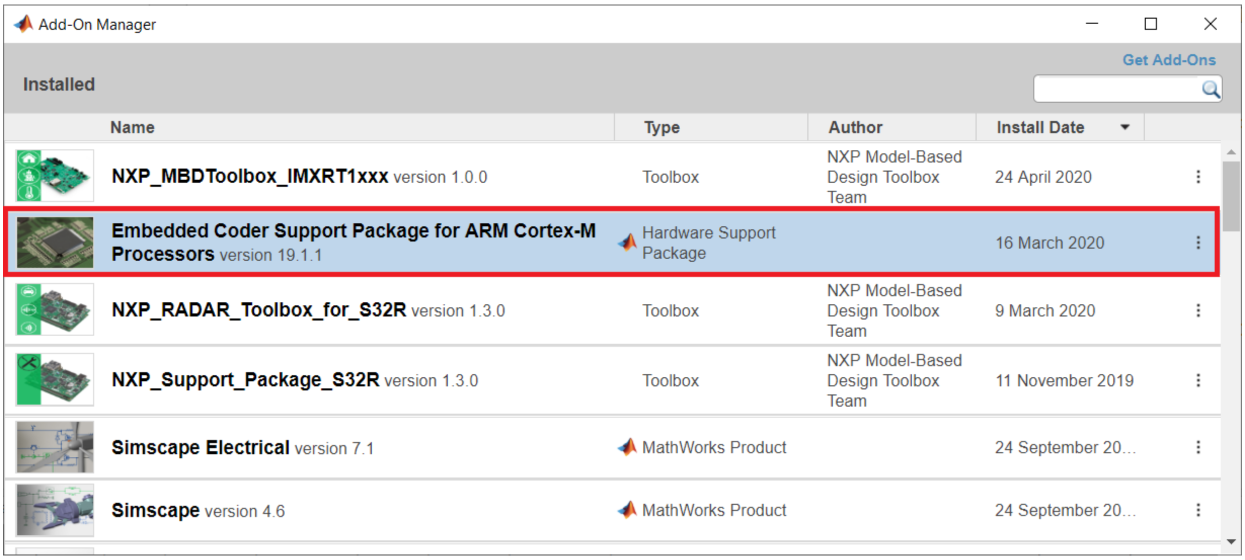
The target compiler for the Model-Based Design Toolbox needs to be configured.

The Model-Based Design Toolbox uses the Toolchain mechanism exposed by the Simulink to enable automatic code generation with Embedded and Simulink Coder toolbox. By default, the toolchain is configured for the MATLAB 2020a, 2020b, 2021a, R2021b and R2022a releases. For any other MATLAB release, the user needs to execute a toolbox m-script to generate the appropriate settings for his/her installation environment.

This is done by changing the MATLAB Current Directory to the toolbox installation directory (e.g.: `..\MATLAB\Add-Ons\Toolboxes\NXP_MBDToolbox_S32K3xx\`) and running the "`mbd_s32k3_path.m`" script.

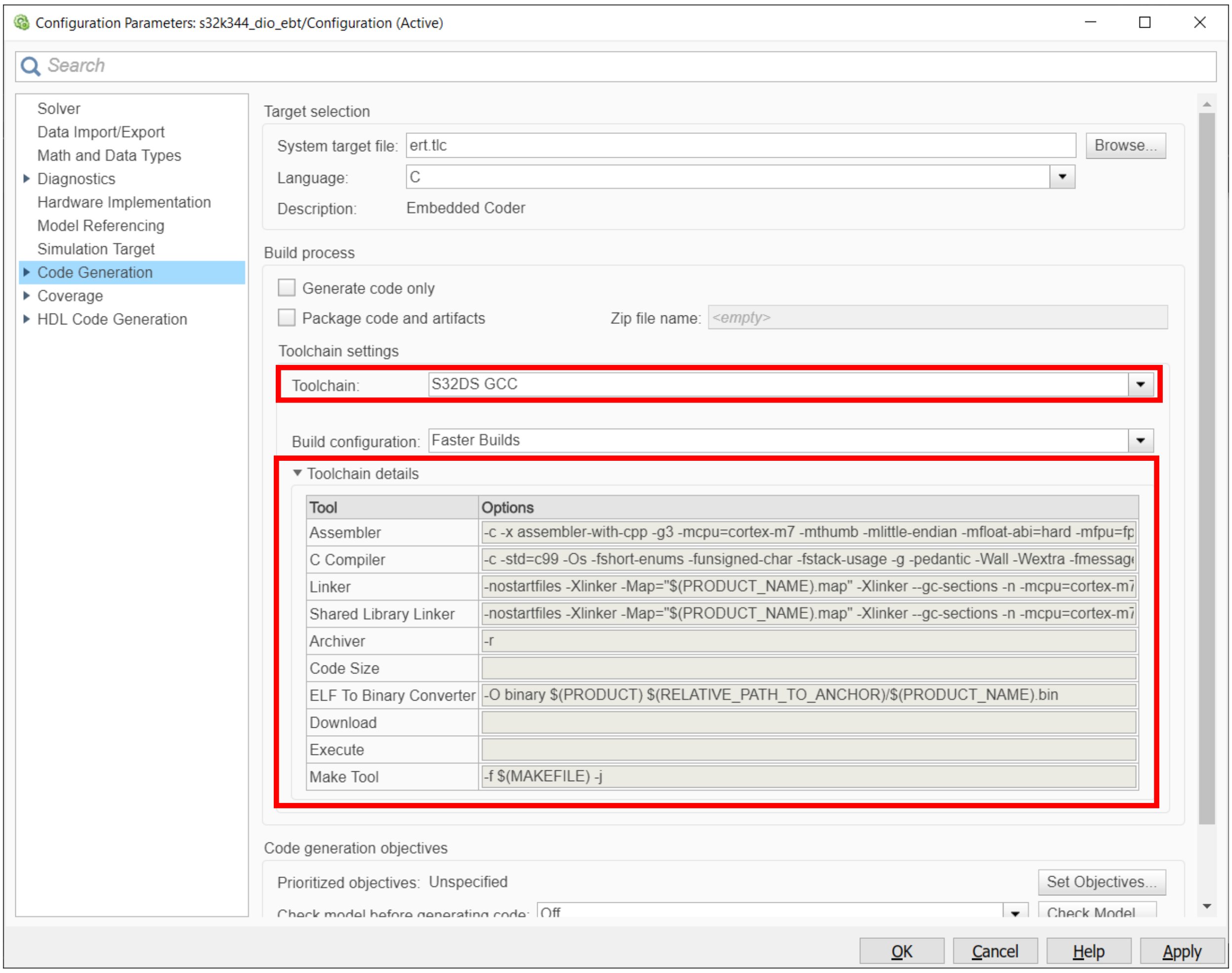
```
>> mbd_s32k3_path
Treating 'C:\Users\nxa14941\Documents\MATLAB\Add-Ons\Toolboxes\
\NXP_MBDToolbox_S32K3xx as MBD Toolbox installation root.
MBD Toolbox path prepended.
Registering the toolchain ...
Successful.
```

This mechanism requires users to install the [Embedded Coder Support Package for ARM Cortex-M Processor]() as a prerequisite.



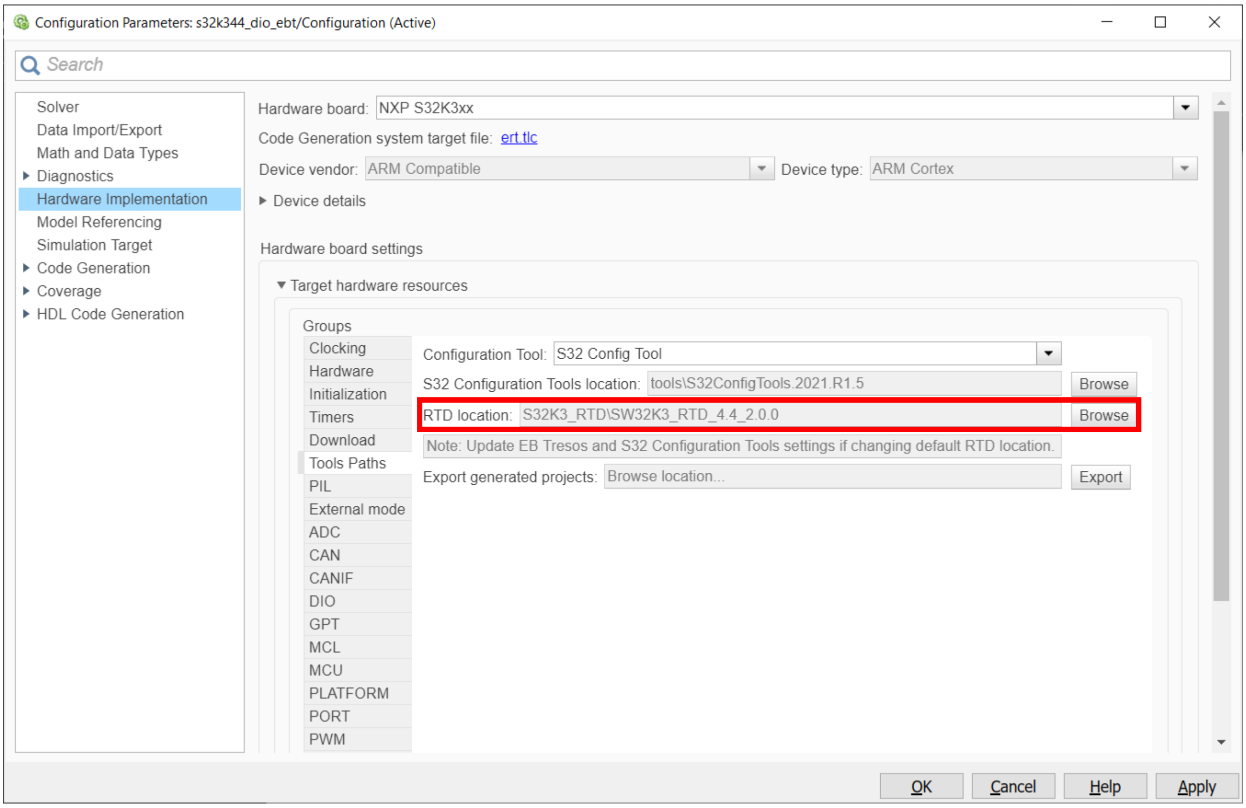
The “`mbd_s32k3_path.m`” script verifies the user setup dependencies and will issue instructions for a successful installation and configuration of the toolbox.

The toolchain can be further enhanced using the Simulink **Model Configuration Parameters** menu:



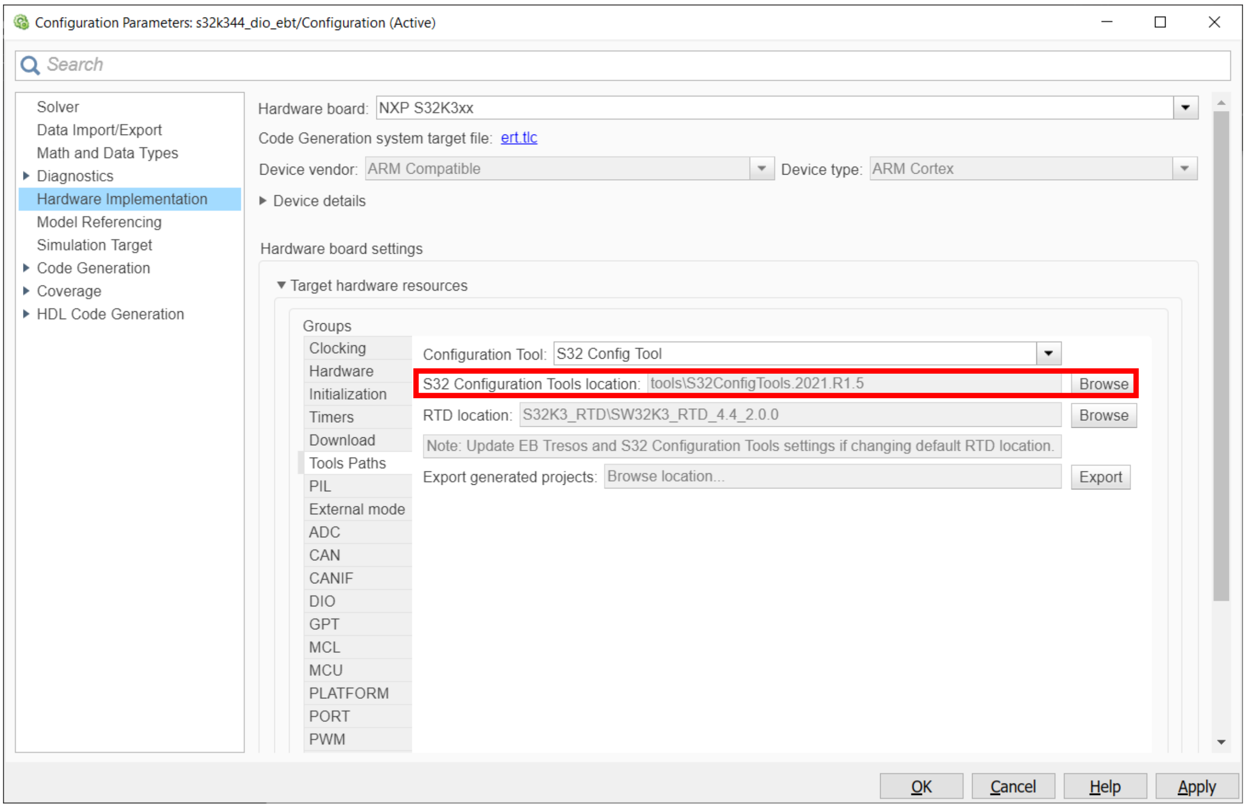
5.3 RTD Support

The toolbox is delivered with RTD 2.0.0 support. The user can change the RTD package from the Simulink **Model Configuration Parameters** menu



5.4 S32 Configuration Tool Support

The toolbox is delivered with Configuration Tools v2021.R1.5. The user can change the Configuration Tools version used by the Simulink from the Simulink **Model Configuration Parameters** menu



6 Known Limitations

The list of know limitations can be found the readme.txt file that is delivered with the toolbox and can be consulted in the MATLAB Add-on installation folder of the Model-Based Design Toolbox for S32K3xx Series.

7 Support Information

For technical support please sign on to the following NXP's Model-Based Design Toolbox Community: https://community.nxp.com/t5/NXP-Model-Based-Design-Tools/bd-p/mbdt

How to Reach Us:

Home Page:
www.nxp.com

Web Support:
www.nxp.com/support